A Performance Analysis on Cloud based Mobile Augmentation in Mobile Cloud Computing

Sindhu K* and Dr. H.S.Guruprasad**
* Department of Information Science and Engineering, BMS College of Engineering, Bangalore, India
Sind19@gmail.com
** Department of Information Science and Engineering, BMS College of Engineering, Bangalore, India
drhsguru@gmail.com

Abstract: Mobile Cloud Computing involves three major components like mobile, cloud and wireless network wherein the data storage and processing will be done outside the mobile. This directly enhances the processing power, battery life and storage capacity as the resource intensive applications are moved onto the cloud. In this paper, we analyze the performance of the mobile by executing the task on the mobile, local server and cloud. Three different methods were used for execution and the program for generating prime numbers was used for the study. In the first method, the entire program was executed on mobile. In the second method, the input and output happens on the mobile and execution of generating prime numbers was done on local server. In the third method, the input and output happens on mobile and execution of generating prime numbers was done on cloud. The parameters considered for the measurement are the mobile energy consumption and execution time taken to execute the program. The results show that as the computation increases it is better to offload the task on the cloud or local server than executing on the mobile.

Keywords: Mobile cloud computing; Energy consumption; Execution Time; Cloud; Mobile.

Introduction

Cloud computing is a computing model where applications and services are provided over the internet. One of the prime advantages of cloud computing is to reduce the cost involved with servers, infrastructure costs and maintenance, thereby enabling user to access the information from anywhere anytime. Developing applications on the mobile devices imposes many constraints driven by their low processing speed, less memory and low battery power. Cloud computing enables to overcome these constraints by allowing mobile devices to perform resource intensive tasks on cloud and reducing the dependency on mobile devices.

Mobile Cloud Computing employs the services of cloud computing by allowing applications and data to reside over the cloud. In this scenario, the devices execute cloud based applications rather than native applications or it can be a combination of both. Few of the advantages of using mobile cloud computing are usage of minimal resources of the mobile, provides flexibility for using different application platforms and reduces the application development and execution time. Even though there are continuous technological advancements, mobile phones continue to face many limitations. Offloading resource intensive application to nearby servers or cloud increases the speed of execution, reduces battery consumption and other mobile resources. The entire application can be offloaded to the cloud or only the resource intensive tasks of an application can be offloaded onto the cloud based on the application.

Mobile cloud computing makes use of the resources from the cloud to increase the computational capabilities of mobile devices. Mobile Users can scale the available services to match their needs, customize apps and access cloud space from any location with an active internet connection. In this paper, an analysis on the execution time and energy consumption of the mobile is carried out. The program considered for analysis is generating prime numbers and the program is executed on the mobile, on local server and cloud. This paper is organized as follows. In Section II a discussion on the related work in Mobile cloud computing is done. Proposed work is discussed in Section III. Section IV provides the results and conclusion of the work is discussed in Section V.

Related Work

Liu et al [1] proposed a framework that uses Dynamic Programming table to find a solution for reducing the time and energy cost by calculating the shortest path for offloading decision. The framework produces the offloading decision before executing the application program by finding the functions of the application program which can be offloaded to cloud. Yue et al [2] presents a framework where energy consumption of mobile device is reduced by offloading to infrastructure based cloud server. An offline scheduler and three online scheduling algorithms are implemented. The offline scheduler makes optimum job partition selections. Online scheduling algorithms gives priority to tasks that are earlier generated at cloud, performs energy threshold test on job execution partition acceptance and tests the job partition feasibility. Hung [3] discusses a collaboration of thick and thin clients and also strategies to allocate resources to meet various service level agreements. Service to multiple thin clients is provided by using a broker to receive the requests and the broker in turn manages many thick clients each accessing different cloud providers. The architecture is divided into two layers. Layer 1 finds the number of virtual machine required to meet the SLA of thin client and categorize and allocate service images to virtual machine capacities. Layer 2 is used in finding how many thick clients are needed to satisfy quality of service, distribution of data to processors and combining data received to transfer to thin client. Xia et al [4] proposes a semiautomatic offloading system, the application is manually modified to run on cloud and also an offloading proxy is used to send and receive information from cloud. The energy consumption and execution time taken are tested on three applications. Sanaei et al [5] proposes an architecture which consists of three layers to increase quality of computing, decrease the communication latency and improve energy efficiency. The architecture provides multitier infrastructure layer from three sources Cloud, Mobile Network Operators (MNO's) and MNO's authorized dealers. Goudarzi et al [6] proposes the use of genetic algorithm to decide a suitable offloading location for the application to be executed inorder to reduce the cost. Energy consumption and execution time are considered to measure the performance of the system and the applications can be offloaded to various multisites like private or public cloud or cloudlet. Kakadia et al [7] presents a framework that uses the profiler module for collecting the statistics that is performance metrics and high level features like battery status, date and time, user location, etc for all user applications and the scheduler module for making offloading decisions. Mazza [8] discusses a partial offloading technique where mobile cloud computing is used to offload application to powerful server. Heterogeneous networks are used to use small and macro cells to provide steady connectivity and high speed. The time needed for running the application on mobile phone and time needed to offload entire application on cloud is computed. A function is derived to evaluate the percentage of offloading, which minimizes time and energy. Wu et al [9] proposes three intervals of execution that is either always offload, or don't offload or have a tradeoff. The proposed adaptive offloading model can improve performance when application is executed on cloud then on mobile device taking into consideration network bandwidth, amount of data to be transmitted and execution time on server. Kaya et al [10] presents an offloading technique where the objects are created using offloading factory instead of new keyword and a remote object, a local proxy for each object are created if the object needs to be offloaded. If the objects from the mobile and cloud needs to pass information then proxies are created at both mobile and cloud end, hence communication happens. The performance and energy consumption is evaluated by using OCR application. Liu et al [11] proposed an energy efficient way of adaptive resource discovery. A Central Resource Broker(CRB) communicates with all nodes in the area and through adhoc WLAN every node can communicate to each other. The two modes in the proposed architecture are centralized mode where resource directory is maintained by CRB. In this mode, the node first checks whether it has resources to allocate for its job and if resources are insufficient it sends a request to CRB via 3G network. CRB informs the node, the other node identifications where resources are available. The other mode is on demand flooding where a node sends a resource request to all nodes by broadcasting a packet through adhoc WLAN. Fekete et al [12] discusses an offloading technique which optimizes the code in development time. The framework decides which methods to be offloaded to the cloud based on method score. Silva et al [13] proposes a framework which performs load partitioning and offloading to improve the performance. The application was executed on cloudlets, the virtual machines were increased from one to four and it was observed that faster execution of applications and less energy consumption.

Proposed Approach

In the proposed approach three different ways of executing the program for generation of prime numbers were considered. In the first method, the program to generate prime numbers was executed on the mobile. In the second method, the input to the program was read from the mobile, the generation of prime numbers was done on the local server and the result was displayed back on the mobile. In the third method, the input from the user was read from the mobile and generation of prime number execution was done on cloud server and result was sent back to the mobile. The input to the program was varied from 10000 to 100000 in steps of 10000 and from 100000 to 500000 in steps of 50000. For each range of input the program was executed 10 times. The time taken to execute the program and energy consumed by the mobile was measured and average result was considered for comparison. The mobile energy consumed was measured using power tutor [14] considering processor energy usage. The procedure for all three different ways of executing the program is given below:

Method 1: Program execution on the mobile

The program for generation of prime numbers was written in Java and executed using Android SDK. The complete program execution was done on the mobile. Once the user entered the input, the prime numbers where generated based on the input value and displayed on the mobile. The execution time of the program and consumption of energy by the mobile was measured. The experiment was repeated 10 times and the average value of execution time and energy consumed was considered.

Algorithm
Step 1: Read the input from the user (N Value).
Step 2: Generate N prime numbers.
Step 3: Display the results on mobile.
Step 4: Calculate the time taken to execute the program and energy consumed.
The input value N was varied from 10000 to 100000 in steps of 10000. After noting that there is no significant difference in time taken to execute in steps of 10000 the input value N between 100000 to 500000 was varied in steps of 50000. For each input value the experiment was repeated 10 times and the average was considered. User interface was designed for asking the user to enter the input range and then the generated prime numbers was displayed on the mobile. The time taken to execute the program from entering of the input value to the end of displaying the result is calculated. Power tutor [14] was used to calculate the energy consumed on the mobile.

Method 2: Executing the program on the Local Server
Reading the input from the user and displaying the result was developed in Java and executed on Android SDK. The read input from the user is sent to the server. A program was written in PHP on the server to generate the prime numbers. The generated numbers are sent from the server to the mobile. The energy consumption on the mobile and the time taken to execute the program is measured.

Algorithm (Mobile Part)
Step 1: Internet permission to be added to Android manifest.
Step 2: Read the input from the user (N Value).
Step 3: Establish the connection to send and receive data over internet.
Step 4: Input value to the server need to be sent by setting the request method to POST, address of the server and input N.
Step 5: On receiving connection establishment success from the server, invoke the server program to perform generation of prime numbers by passing the server address and program name.
Step 6: The result from the server is read and displayed on the mobile.
Since the communication happens between the mobile and server, the internet permission had to be included in the Android manifest xml file. On giving the input by the user the PHP program on the server is invoked which stores the input value in the database. On successful connection between server and mobile the PHP program to generate prime number is invoked and the result sent from the server is displayed on the mobile screen.

Algorithm (Server Part)
Step 1: The input value (N) send from the mobile is obtained and stored on the database of the server.
Step 2: Generate prime numbers on the server for the input value given by the user.
Step 3: The result from the server is sent as JSON object to the mobile to be displayed on the display.
Using Power tutor the energy consumed by the mobile is measured and the time taken to execute the program is calculated from the time the input value is read from the user till the results are displayed on the mobile screen.
On the server-end, two programs were written in PHP. One program to establish the connection to the database and store the result of the input value read from the user. In the second program the input value is read from the database, the prime numbers are generated and the results are sent to the mobile to be displayed. The energy consumed and the time taken to execute the program from the time of reading the input to displaying the output is measured. The experiment is repeated 10 times and the average of energy consumed and time taken to execute is considered.

Method 3: Executing the program on the Cloud
A domain name was registered from the service provider and PHP was used at the server end. Reading the input from the user and displaying the result was developed in Java and executed on Android SDK. On reading the input from the user, the read value is sent to the cloud. A program was written in PHP on the cloud to generate the prime numbers. The generated numbers are sent from the server to the mobile. The energy consumption on the mobile and the time taken to execute the program is measured.
The algorithm at the mobile end and server end is same as given in Method 2. The server address needs to be changed from the local server address to the cloud domain name.
Devices used for Experimental Setup were:
Client - Mobile phone Sony Xperia M C1904 featuring dual core processor with 1 GHz speed and 1 GB RAM. Battery Capacity 1750 mAh Li-lon.
Local Server - Intel(R) Core(TM) i3 CPU M 370 @ 2.40 GHz x64 based processor, 4GB RAM 64-bit Operating System.

Results and Discussion

Fig. 1 gives the comparison of time taken to execute the Prime number generation program on mobile, local server and cloud for numbers between 10000 to 1 lakh varied in steps of 10000. Fig. 2 shows the comparison of time taken to execute the Prime number generation program on mobile, local server and cloud for number between 1 lakh to 5 lakhs varied in steps of 50000. Fig. 3 shows the comparison for time taken to execute the Prime number generation program on local server and cloud for number between 10000 to 5 lakhs.

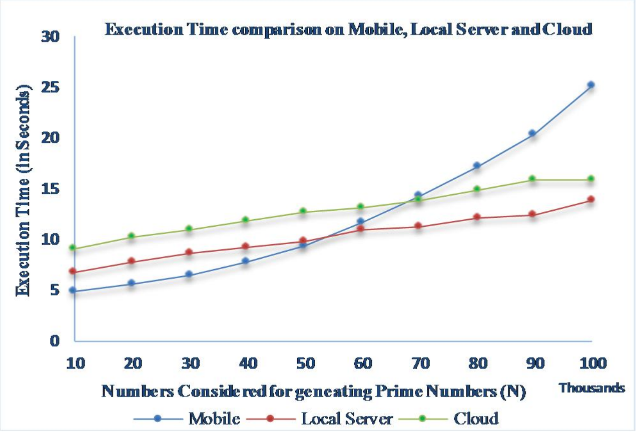

Fig. 1. Execution Time Comparison for numbers between 10000 to 1 Lakh

It is observed that for the input value N less than 50000 the execution time taken for execution of the program on mobile is faster compared to the servers. But as the input value N increases the performance on the Local server is better for numbers above 1 Lakh to 4 Lakhs. As input value is increased above 4 Lakhs the time taken to execute the program on the cloud is reducing comparatively as shown in Fig. 3.

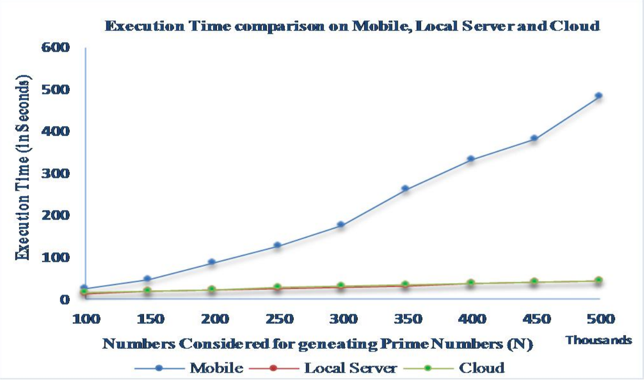

Fig. 2. Execution Time Comparison for numbers between 1 Lakh to 5 Lakh

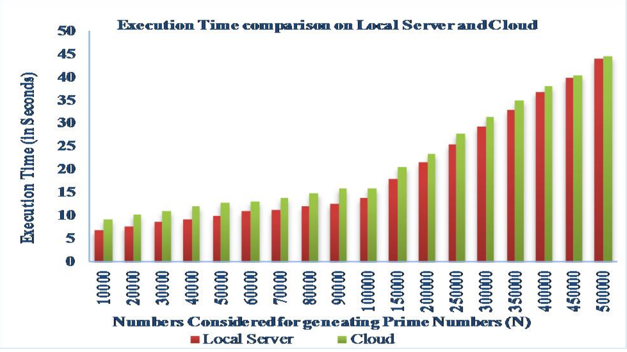

Fig. 3. Execution Time Comparisons between Local Server and Cloud

Fig. 4 gives the comparison of energy consumption for the Prime number generation program on mobile, local server and cloud for numbers between 10000 to 1 lakh varied in steps of 10000. Fig. 5 shows the comparison of energy consumption to execute the Prime number generation program on mobile, local server and cloud for number between 1 lakh to 5 lakh varied in steps of 50000. Fig. 6 shows the comparison for energy consumption to execute the Prime number generation program on local server and cloud for number between 10000 to 5 lakh.

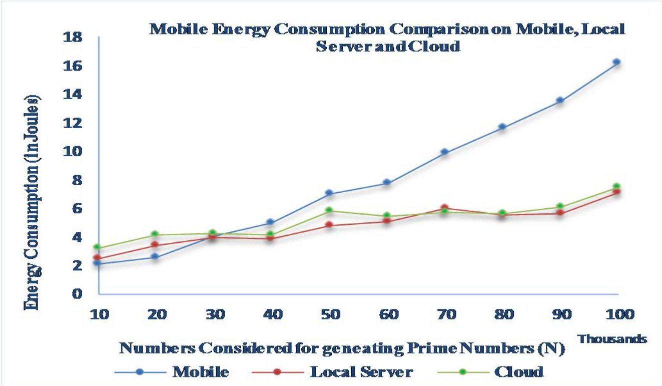

Fig. 4. Energy Consumption for numbers between 10000 to 1 Lakh

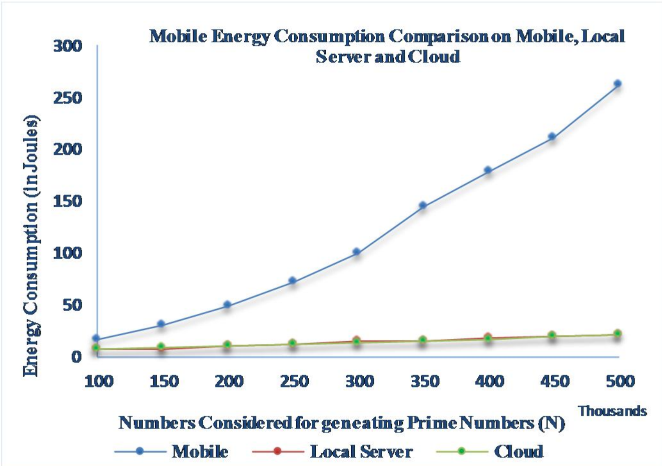

Fig. 5. Energy Consumption for numbers between 1 Lakh to 5 Lakh

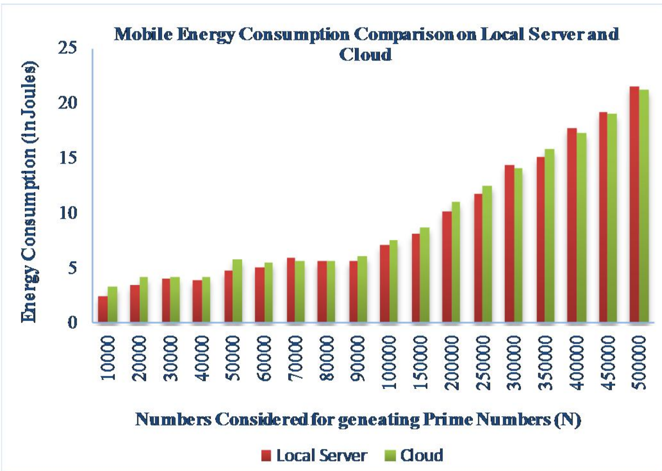

Fig. 6. Energy Consumption Comparisons between Local Server and Cloud

It is observed from the Fig. 4 that for the input value N less than 30000 the energy consumption for execution of the program on mobile is less compared to the servers. But as the input value N increases the mobile energy consumption increases gradually when the program is executed on the mobile. The energy consumption of mobile is slightly lesser when the program is executed on the local server when compared to execution on cloud for numbers upto 2.5 lakhs. For the numbers above 3 lakhs there is a slight difference in energy consumption when compared to execution on local server and cloud. The energy consumption of the mobile on cloud is marginally reduced when compared to local server execution of the program for numbers greater than 4 lakhs as shown in Fig. 6.

Conclusion

In this paper, an analysis on the energy consumption and execution time was done by executing the program for generating prime numbers on the mobile, local server and cloud. It was observed that when the input size is less, executing the program on the mobile is better compared to servers. As the input value increases, execution on the local server is better to a certain extent but when the input size is increased further it is observed that cloud facilitates faster performance. Energy consumption of the mobile is also less when the program is executed on both the servers compared to mobile execution.

References

[1] Liu, Yanchen, Myung J. Lee. "An effective dynamic programming offloading algorithm in mobile cloud computing system." In Wireless Communications and Networking Conference (WCNC), 2014 IEEE, pp. 1868-1873. IEEE, 2014.

[2] Yue, Jianting, Dongmei Zhao, Terence D. Todd. "Cloud server job selection and scheduling in mobile computation offloading." In Global Communications Conference (GLOBECOM), 2014 IEEE, pp. 4990-4995. IEEE, 2014.

[3] Hung, Pham Phuoc, Tuan-Anh Bui, Mauricio Alejandro Gómez Morales, Mui Van Nguyen, Eui-Nam Huh. "Optimal collaboration of thin–thick clients and resource allocation in cloud computing." Personal and ubiquitous computing 18, no. 3 (2014): 563-572.

[4] Xia, Feng, Fangwei Ding, Jie Li, Xiangjie Kong, Laurence T. Yang, Jianhua Ma. "Phone2Cloud: Exploiting computation offloading for energy saving on smartphones in mobile cloud computing." Information Systems Frontiers 16, no. 1 (2014): 95-111.

[5] Sanaei, Zohreh, Saeid Abolfazli, Abdullah Gani, Muhammad Shiraz. "SAMI: Service-based arbitrated multi-tier infrastructure for Mobile Cloud Computing." In Communications in China Workshops (ICCC), 2012 1st IEEE International Conference on, pp. 14-19. IEEE, 2012.

[6] Goudarzi, Mohammad, Zeinab Movahedi, Masoud Nazari. "Mobile cloud computing: a multisite computation offloading." In Telecommunications (IST), 2016 8th International Symposium on, pp. 660-665. IEEE, 2016.

[7] Kakadia, Dharmesh, Prasad Saripalli, Vasudeva Varma. "MECCA: mobile, efficient cloud computing workload adoption framework using scheduler customization and workload migration decisions." In Proceedings of the first international workshop on Mobile cloud computing & networking, pp. 41-46. ACM, 2013.

[8] Mazza, Daniela, Daniele Tarchi, Giovanni E. Corazza. "A partial offloading technique for wireless mobile cloud computing in smart cities." In Networks and Communications (EuCNC), 2014 European Conference on, pp. 1-5. IEEE, 2014.

[9] Wu, Huaming, Qiushi Wang, Katinka Wolter. "Tradeoff between performance improvement and energy saving in mobile cloud offloading systems." In Communications Workshops (ICC), 2013 IEEE International Conference on, pp. 728-732. IEEE, 2013.

[10] Kaya, Mahir, Altan Koçyigit, P. Erhan Eren. "A mobile computing framework based on adaptive mobile code offloading." In Software Engineering and Advanced Applications (SEAA), 2014 40th EUROMICRO Conference on, pp. 479-482. IEEE, 2014.

[11] Liu, Wei, Takayuki Nishio, Ryoichi Shinkuma, Tatsuro Takahashi. "Adaptive resource discovery in mobile cloud computing." Computer Communications 50 (2014): 119-129.

[12] Fekete, Krisztian, Adam Pelle, Kristof Csorba. "Energy efficient code optimization in mobile environment." In Telecommunications Energy Conference (INTELEC), 2014 IEEE 36th International, pp. 1-6. IEEE, 2014.

[13] Silva, Francisco Airton, Paulo Maciel, Rubens Matos. "SmartRank: a smart scheduling tool for mobile cloud computing." The Journal of Supercomputing 71, no. 8 (2015): 2985-3008.

[14] http://ziyang.eecs.umich.edu/projects/powertutor/

[15] Abolfazli, Saeid, Zohreh Sanaei, Abdullah Gani, Feng Xia, Wei-Ming Lin. "RMCC: Restful Mobile Cloud Computing Framework for Exploiting Adjacent Service-Based Mobile Cloudlets." In Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on, pp. 793-798. IEEE, 2014.

[16] S. Abolfazli, Z. Sanaei, M. Alizadeh, A. Gani, F. Xia, “An experimental analysis on cloud-based mobile augmentation in mobile cloud computing,” IEEE Transactions on ConsumerElectronics, vol. 60, no. 1, pp. 146–154, 2014.

[17] Suneel, K. S., H. S. Guruprasad. "An Approach for Server Consolidation in a Priority Based Cloud Architecture." BVICAM's International Journal of Information Technology 8, no. 1 (2016).

[18] Shiraz, Muhammad, Abdullah Gani, Azra Shamim, Suleman Khan, Raja Wasim Ahmad. "Energy efficient computational offloading framework for mobile cloud computing." Journal of Grid Computing 13, no. 1 (2015): 1-18.

[19] S. Abolfazli, Z. Sanaei, M. Alizadeh, A. Gani, F. Xia, “An experimental analysis on cloud-based mobile augmentation in mobile cloud computing,” IEEE Transactions on ConsumerElectronics, vol. 60, no. 1, pp. 146–154, 2014.

[20] Christensen, Jason H. "Using RESTful web-services and cloud computing to create next generation mobile applications." In Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications, pp. 627-634. ACM, 2009.