Decision Making for Computation Offloading in Mobile Cloud Computing

Anil Kumar
2015IPG-007

Aniruddha Shrikhande
2015IPG-008

Anand Deshmukh
2015IPG-020

Group no: 15
Mentors: Prof. Shashikala Tapaswi, Dr. Neetesh Kumar

Background:

Cloud computing simply refers to delivery of computation power using a network of remote servers. The main advantages of cloud computing [1] include on-demand computing resources, paying as per the use and simplified IT management. It comes into picture when the local processors are incapable of performing heavy tasks. Mostly, Internet is used to communicate with the cloud services. *Mobile cloud computing* extends the idea of cloud computing to handheld devices such as smartphones or tablets for increasing the computational efficiency. This is mainly achieved by handing all the resource-intensive tasks (like character recognition) of any mobile application to the remote server and then getting back the result. This is known as *computation offloading*. The remote servers comprise of powerful workstations which are much more efficient while dealing with heavy-compute tasks.

Motivation:

According to a recent survey [2], 95% people in this world use cellphones and 77% people use smartphones. This directly implies that demand for interactive apps like map navigators, optical character recognisers (OCRs), voice assistants etc. is also increasing. Interactive apps like these require more resources than other conventional apps like image viewer apps, file explorers. Resources such as CPU, RAM and GPU are utilised heavily and eventually this results into battery drainage or heating up of phone. The need for stronger and efficient processors like the ones used in PCs arise. This need can be fulfilled with the help of computation offloading. A lot of research and work has been done in this field. Still, there is no perfect method which can determine exactly when the computation must be offloaded to the cloud server. As described in the next section, we see that each and every method has its own limitations like slow computations resulting into high response time, high energy consumption, improper decisions regarding when to offload, incomplete parameter considerations and more. Overcoming the shortcomings of different methods can help us achieve a better way to deal with computation offloading and optimize the user experience for interactive apps.

In India, Reliance Jio Infocomm Limited has created a vast market in smartphones by providing 4G internet services at low cost. This makes computation offloading of interactive apps more feasible. This has created a demand for better Computation Offloading methods which can handle millions of users simultaneously.

Local processing is always better than the offloaded processing until and unless it does not bring too much load over our mobile devices. This brings us to the situation of deciding when to do local processing and when to offload it to the cloud server.

Literature Review:

Earlier approaches about Mobile Computation Offloading had least focused on the decision for when to offload a certain task. Although in recent years many researches have brought acceptable solutions for this decision making part of computation offloading.

In the paper by Mohammed A. Hassan et al [3], a prototype framework named POMAC was implemented on Android Dalvik virtual machine which was capable of dynamic and transparent computation offloading of mobile applications to clouds. Transparency was achieved by offloading computations without making any changes to the application source code. POMAC focused on improving time response of the mobile app.

In the paper by Ying-Dar Lin et al [4], a framework was developed, Ternary Decision Maker, which aimed at shortening response time and reducing consumption of energy simultaneously. Ternary decision making used three different execution environments: on-board CPU, on-board GPU and cloud server in accordance with the requirements of the application.

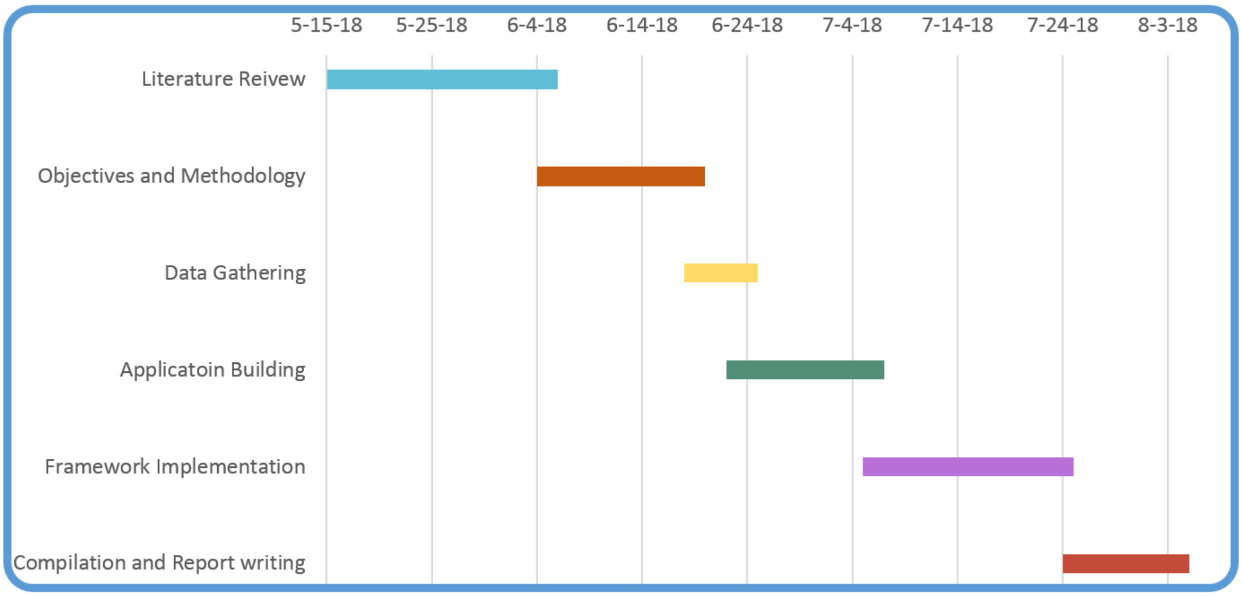
In the paper by Shanthi Al. and Dr V. Ramesh [5], previously implemented POMAC was modified and named M-POMAC. This version focused on designing a decentralized mechanism for application offloading and achieving deallocation of resources at the cloud server side for accommodating multiple users.

Previous works focused only on the general performance of mobile applications like response time and energy consumption. In the paper by Weiqing Liu et al [6], AppBooster, a mobile cloud platform, was developed. The focus was on optimizing both the general performance and quality of the application (ex: Recognition accuracy when the app is concerned with object recognition) by combining the results of constantly monitored environment conditions, history based knowledge and the information provided by the developers.

Problem Statement:

We aim to improve the performance of the currently available decision making methods for Computation offloading in Mobile Cloud Computing.

Timeline:



References

[1] Khadija Akherfi, Michael Gerndt and Hamid Harroud, "Mobile cloud computing for computation offloading: Issues and challenges", *Applied Computing and Informatics* vol. 14, issue 1, 2018.
[2] http://www.pewinternet.org/fact-sheet/mobile/
[3] M.A. Hassan, K. Bhattarai, Q. Wei and S. Chen, "Pomac: Properly offloading mobile applications to clouds", Energy (J), vol. 25, pp. 50, 2014.
[4] A. L. Shanthi and V. Ramesh, "Decentralized applications offloading to mobile clouds using M-POMAC," 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, 2017, pp. 1-6.
[5] Y. D. Lin, E. T. H. Chu, Y. C. Lai and T. J. Huang, "Time-and-Energy-Aware Computation Offloading in Handheld Devices to Coprocessors and Clouds," in IEEE Systems Journal, vol. 9, no. 2, pp. 393-405, June 2015.
[6] W. Liu, J. Cao, L. Yang, L. Xu, X. Qiu and J. Li, "AppBooster: Boosting the Performance of Interactive Mobile Applications with Computation Offloading and Parameter Tuning," in IEEE Transactions on Parallel and Distributed Systems, vol. 28, no. 6, pp. 1593-1606, June 1 2017.