Volume : 4 | Issue : 6 | June 2015 • ISSN No 2277 - 8179

Research Paper

Engineering

KEYWORDS : MCC, Cloud computing, Code Offloading

A Study on Code Offloading Techniques in Mobile Cloud Computing

Kirti Vijayvargia | IIPS, DAVV

ABSTRACT

In today's environment smart phones have become essential for everybody's life. Smart phones are being used for all those applications, which could be run previously on personal computer and laptops only. But smart phones have still some limitations like processing power of CPU, small memory and battery life. To overcome these limitations mobile cloud computing has become popular so that mobile applications can make use of cloud computing capabilities for data storage as well as its computations. But improper use of code offloading algorithm can create performance issues. This paper presents an overview of different code offloading issues and available remedies for these issues.

INTRODUCTION

In today's era mobile devices have become an indispensable part of everybody's life because of its communication capabilities immaterial of place and time. But these mobile devices are also constrained with limited battery, short storage and processing power. To overcome these limitations mobiles can make use of another emerging technology that is cloud computing. Cloud computing is known as delivery of data, applications and infrastructure to the user on demand and on pay per use policy. When cloud computing is integrated in mobile environment it is called mobile cloud computing.[1]

Mobile cloud computing can be defined as a technique where both the data storage and the data processing occur outside of the mobile device. Mobile cloud applications move the computing and data storage from mobile phones to cloud. It is applicable not for only smart phones but applies to other mobile devices as well.[3]

Mobile cloud computing has many benefits like extended battery life, improved storage and processing power, reliability, multitenacy, scalability but on other side of the coin it has many challenges and issues like code offloading, low bandwidth, security, pricing and standards establishment etc.[2]

The main aim of this paper is to discuss issues and challenges related to code offloading in mobile cloud computing. Code offloading is the process of migration of computation from mobile devices to the cloud. Either whole computing or part of it can be migrated.

Static or dynamic offloading

When the program is partitioned at the time of program development it is called static offloading. It is simpler but has a disadvantage that it is useful only when all the variables can be predicted previously. But in dynamic offloading decision is taken at execution time based on the actual conditions.[4]

There are various code offloading techniques but choosing one of them is decided by following factors as shown in figure 1.[4]

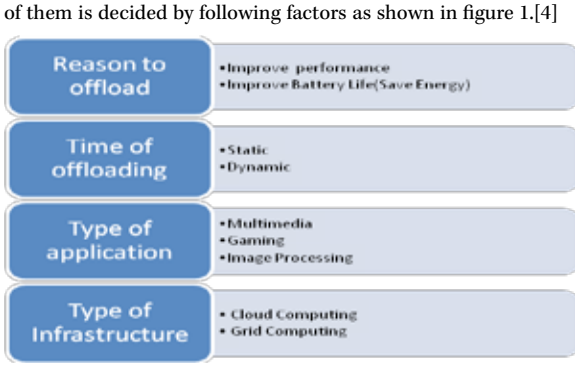

Figure 1: Factors for selecting code offloading technique

Rest of the paper is as follows: section 2 discusses related works of offloading techniques of mobile Cloud computing. In Section 3, conclusion and our contribution is presented.

BACKGROUND

A large body of literature is available on offloading techniques in mobile cloud computing. Various research groups are exploring the ways to offload computation to make efficient use of resources in the clouds and gaining maximum from it.

Literature on code offloading in mobile cloud computing:

Zhang et. al. [5], [6] developed a reference framework for partitioning a single application into elastic components with dynamic configuration of execution. The components called weblets are platform independent and can be executed transparently on different computing infrastructures including mobile devices or IaaS (Infrastructure as a Service) cloud providers. The application is split down to a UI component, weblets, and a manifest describing the application. An elasticity manager component decides on migration, instantiation and migration of the weblets. These processes are transparent to the running application.

Eduardo Cuervo et. al. presented MAUI, a system that enables fine-grained energy-aware offload of mobile code to the infrastructure. MAUI uses the benefits of a managed code environment to offer the best of both worlds: it supports fine-grained code offload to maximize energy savings with minimal burden on the programmer. MAUI decides at runtime which methods should be remotely executed, driven by an optimization engine that achieves the best energy savings possible under the mobile device's current connectivity constrains. In their evaluation, they showed that MAUI enables: 1) a resource-intensive face recognition application that consumes an order of magnitude less energy, 2) a latency-sensitive arcade game application that doubles its refresh rate, and 3) a voice-based language translation application that bypasses the limitations of the smart phone environment by executing unsupported components remotely. [7]

Byung-Gon Chun et al. presented the design and implementation of CloneCloud, a system that automatically transforms mobile applications to benefit from the cloud. The system is a flexible application partitioner and execution runtime that enables unmodified mobile applications running in an application-level virtual machine to seamlessly off-load part of their execution from mobile devices onto device clones operating in a computational cloud[8]

Mark S. Gordon et. al. introduced a runtime system to allow unmodified multi-threaded applications to use multiple machines. The system allowed threads to migrate freely between machines depending on the workload. Their prototype, COMET (Code Offload by Migrating Execution Transparently), is a realization of this design built on top of the Dalvik Virtual Machine. COMET leverages the underlying memory model of runtime to imple-

ment distributed shared memory (DSM) with as few interactions between machines as possible.[9]

Aki Saarinen et. al. proposed SmartDiet, a toolkit to identify the constraints that reduce offloading opportunities and to calculate the energy-saving potential of offloading communication-related tasks. SmartDiet traces the method-level application execution and estimates the allocation of communication energy cost from traffic traces.[10]

Roelof Kemp et. al. proposed the Cuckoo framework, which simplified the development of smart phone applications that benefit from computation offloading and provides a dynamic runtime system that can, at runtime, decide whether a part of an application will be executed locally or remotely.[11]

Sokol Kosta et al. .in their paper proposed ThinkAir, a framework that made it simple for developers to migrate their smart phone applications to the cloud. ThinkAir exploits the concept of smart phone virtualization in the cloud and provides method-level computation offloading. It focuses on the elasticity and scalability of the cloud and enhanced the power of mobile cloud computing by parallelizing method execution using multiple virtual machine(VM) images. They implemented ThinkAir and valuated it with a range of benchmarks. They showed that the execution time and energy consumption decrease two orders of magnitude for a N-queens puzzle application and one order of magnitude for a face detection and a virus scan application. They then showed that a parallelizable application could invoke multiple VMs to execute in the cloud in a seamless and on-demand manner such as to achieve greater reduction on execution time and energy consumption. 12]

Seungjun Yang et. al. in their paper, introduced novel techniques based on compiler code analysis that effectively reduced the transferred data size by transferring only the essential heap objects. They exhibited through experiments that the reduced size positively influences not only the transfer time itself but also the overall effectiveness of execution offloading, and ultimately, improved the performance of mobile cloud computing significantly in terms of execution time and power consumption [13].

Lei Yange et. al. studied how to optimize the computation partitioning of a data stream application between mobile and cloud to achieve maximum speed/throughput in processing the streaming data. They placed optimization on achieving high throughput of processing the streaming data rather than minimizing the make span of executions as in other applications. They proposed a framework to provide runtime support for the dynamic computation partitioning and execution of the applications. The framework not only allowed the dynamic partitioning for a single user but also supports the sharing of computation instances among multiple users in the cloud to achieve efficient utilization of the underlying cloud resources. Based on the framework, they designed a genetic algorithm for optimal computation partition. They showed through experiments and numerical analysis that the partitioned application could achieve at least two times better performance in terms of throughput than the application without partitioning. [14]

Flores, Huber, and Satish Srirama proposed a fuzzy decision engine for code offloading, that considered both mobile and cloud variables. The cloud parameters and rules were introduced asynchronously to the mobile using notification services. The paper also proposed a strategy to enrich the offloading decision process with evidence-based learning methods, by exploiting cloud-processing capabilities over code offloading traces. They presented that offloading is not a decision process that happens just in the device, but offloading is a learning process that involves a global understanding of the complete mobile cloud infrastructure [15]

Muhammad Shiraz et.al. presented that in MCC, application offloading is ascertained as a software level solution for augmenting application processing capabilities of smart mobile devices . They proposed thematic taxonomy of the existing Distributed Application Processing Frameworks , reviewed that offloading frameworks by using thematic taxonomy and analyzed the implications and critical aspects of existing offloading frameworks on the basis of significant parameters like offloading scope, migration granularity, partitioning approach, and migration pattern.[16]

Dejan Kovachev et al. studied several mobile cloud approaches and concluded that Native (offline) and Web (online) applications are the two extremes of mobile applications and full potential of mobile cloud applications lies in between these two extremes, while dynamically shifting the responsibilities between mobile device and cloud. Several researchers have shown how to achieve that by replicating whole device software image or offloading parts of the application. The offloading can happen to some remote data center, nearby computer or cluster of computers, or even to nearby mobile devices. To simplify the development a convenient, but effective, programming abstraction is required.[17]

CONCLUSION

Code offloading in an important aspect of mobile cloud computing .In this paper several approaches of code offloading are discussed to give an overview of existing techniques and the issues and challenges associated with them. Discussed problems and their solutions in this paper are the key research topics that can be further explored.

REFERENCE

[1] http://en.wikipedia.org/wiki/Cloud_computing | | [2]http://en.wikipedia.org/wiki/mobile_cloud _computing | [3] http://www.mobilecloud-computingforum.com/ | | [4] Kumar, Karthik, et al. "A survey of computation offloading for mobile systems." Mobile Networks and Applications 18.1 (2013): 129-140. | [5] X. Zhang, J. Schiffman, S. Gibbs, A. Kunjithapatham, and S. Jeong,"Securing Elastic Applications on Mobile Devices for Cloud Computing,"in CCSW '09: Proceedings of the 2009 ACM Workshop on Cloud Computing Security. Chicago, IL, USA: ACM, Nov. 2009, pp. 127–134. | [6] X. Zhang, S. Jeong, A. Kunjithapatham, and Simon Gibbs, "Towards an Elastic Application Model for Augmenting Computing Capabilities of Mobile Platforms," in The Third International ICST Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications,Chicago, IL, USA, 2010. | [7] Cuervo, Eduardo, et al. "MAUI: making smartphones last longer with code offload." Proceedings of the 8th international conference on Mobile systems, applications, and services. ACM, 2010. | | [8]Chun, Byung-Gon, et al. "Clonecloud: elastic execution between mobile device and cloud." Proceedings of the sixth conference on Computer systems. ACM, 2011. | [9]Gordon, Mark S., et al. "COMET: Code Offload by Migrating Execution Transparently." OSDI. 2012. | [10] Saarinen, Aki, et al. "SmartDiet: offloading popular apps to save energy." ACM SIGCOMM Computer Communication Review 42.4 (2012): 297-298. | [11]Kemp, Roelof, et al. "Cuckoo: a computation offloading framework for smartphones." Mobile Computing, Applications, and Services. Springer Berlin Heidelberg, 2012. 59-79. | [12] Kosta, Sokol, et al. "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading." INFOCOM, 2012 Proceedings IEEE. IEEE, 2012. | | [13] Yang, Seungjun, et al. "Fast dynamic execution offloading for efficient mobile cloud computing." Pervasive Computing and Communications (PerCom), 2013 IEEE International Conference on. IEEE, 2013. | | [14] Yang, Lei, et al. "A framework for partitioning and execution of data stream applications in mobile cloud computing." ACM SIGMETRICS Performance Evaluation Review 40.4 (2013): 23-32. | [15]Flores, Huber, and Satish Srirama. "Adaptive code offloading for mobile cloud applications: Exploiting fuzzy sets and evidence-based learning." Proceeding of the fourth ACM workshop on Mobile cloud computing and services. ACM, 2013. | [16]Shiraz, Muhammad, et al. "A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing." Communications Surveys & Tutorials, IEEE 15.3 (2013): 1294-1313. | [17]Kovachev, Dejan, Yiwei Cao, and Ralf Klamma. "Mobile cloud computing: a comparison of application models." arXiv preprint arXiv:1107.4940 (2011). |