2015 2nd ACM International Conference on Mobile Software Engineering and Systems

Extending Mobile Device's Battery Life by Offloading Computation to Cloud

Hao Qian[1], Daniel Andresen[2]
Department of Computing and Information Sciences
Kansas State University
Manhattan, KS, USA
[1]hqianm@gmail.com [2]dan@ksu.edu

***Abstract* — The need for increased performance of mobile device directly conflicts with the desire for longer battery life. Offloading computation to resourceful servers is an effective method to reduce energy consumption and enhance performance for mobile applications. Android provides mechanisms for creating mobile applications but lacks a native scheduling system for determining where code should be executed. This paper presents Jade, a system that adds sophisticated energy-aware computation offloading capabilities to Android applications. Jade monitors device and application status and automatically decides where code should be executed. Jade dynamically adjusts offloading strategy by adapting to workload variation, communication costs, and device status.**

***Keywords* — code offloading; energy management; mobile computing; cloud computing; scheduling**

1. Introduction

Battery life has become one of the biggest obstacles for mobile device advancements. Performance demanded by smartphones and tablets is increasing at a much faster rate than technological improvements in battery capacity. The need for increased performance of mobile devices directly conflicts with the desire for longer battery life.

One popular technique to reduce energy consumption of mobile devices is computation offloading in which an application reduces energy consumption by delegating code execution to other devices. In this paper, we present Jade, an energy-aware computation offloading system for mobile devices (Figure 1). The work of this paper builds on our previous research of computation offloading between ad-hoc net-worked mobile devices [1]. Jade, built for mobile devices running Android operating system, minimizes energy consumption of mobile devices through fine-grained computation offloading to the cloud.

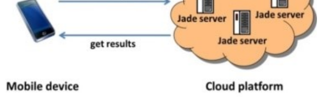

Figure 1: Jade reduces energy consumption of mobile device by offloading energy consuming computation to Jade servers running on the cloud.

2. System Design

In this section, we present the high-level design of Jade in order to demonstrate how they integrate into one system, thereby supporting distributed execution of mobile applications.

For mobile applications with heavy computation needs, computation offloading is an effective method to reduce energy consumption and enhance performance. However, it requires additional efforts and skills to develop applications with computation offloading ability and, unfortunately, no mature frameworks or tools exist for mobile application developers. We designed Jade to minimize the workload of developing applications with computation offloading ability by:

- Offering the Jade runtime engine (Figure 2) which provides services for wireless communication, device profiling, program profiling and computation offloading. Conceptually, the Jade runtime engine automatically transforms application execution on single mobile device into a distributed execution optimized for wireless connection, power usage, and server capabilities (Figure 3).
- Providing an easy-to-use programming model for developers to build mobile applications that support energy-aware computation offloading.

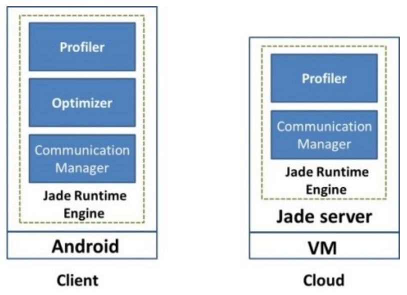

Figure 2: High-level design of the Jade runtime engine.

3. Multi-level Data Storage

Wireless network interface accounts for a big portion of mobile device's energy consumption. For example, the Wi-

Fi interface of a smart phone consumes almost 1500mW when downloading data. Because computation offloading system needs to transfer data between client and server frequently, a lot of energy of mobile device is consumed by network interface.

Jade provides a Multi-level Data Storage Service (MDSS) that optimizes the energy cost of data transfer when computation offloading occurs. MDSS allows developers to save application data on the cloud without writing any backend code. Application data is also saved locally on mobile devices allowing applications to work when devices are offline. MDSS automatically synchronizes data between local device and the cloud, so developers can focus on creating applications instead of having to worry about building backend solution to handle data storage and synchronization.

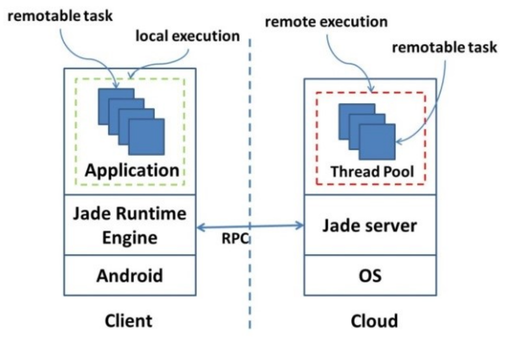

Figure 3: Jade enables computation offloading for mobile applications.

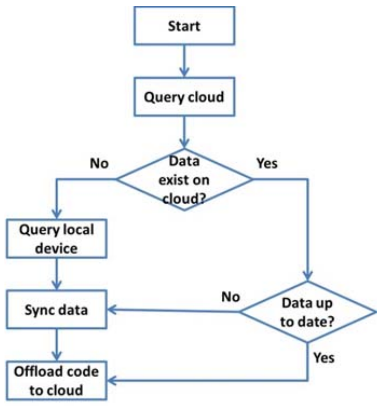

Figure 4: MDSS reduces the amount of data transferred in the network if the cloud already has the latest data.

In a computation offloading system, a remotable task contains two elements: 1) *app data* (e.g., images, texts, numbers) and 2) *task code* that performs execution on app data (e.g., sorting algorithm, image processing). App data and task code are bundled and transferred when a remotable task is offloaded to the server. In most cases, the size of app data is much bigger than the size of task code (e.g., size of an image could be a few MB, whereas size of task code performing complex computation could be a few KB). By introducing MDSS, a remotable task's app data and task code are separated. In Jade, a remotable task i contains only task code, the app data accessed by i is stored separately and referenced by URI. When i is offloaded to the cloud, if the cloud already has the most recent copy of app data that i needs to access, Jade only offloads task code to the cloud in order to reduce the amount of data transferred. MDSS effectively helps reducing the energy consumption for mobile device's network interface by avoiding transferring app data every time when i is offloaded (Figure 4).

4. Related Work

Jade was built upon previous research regarding program partitioning, code offloading, and remote execution. In this section, we provide an overview of proposals by these researches and how they relate to Jade.

Cuervo et al. proposed MAUI [2], a system that enables energy-aware offloading of mobile code to the infrastructure. MAUI enables developers to produce an initial partitioning of their applications by annotating methods and/or classes as remotable. At runtime, the MAUI solver decides which remotable methods should execute locally and which should execute remotely. Unlike MAUI, Jade provides a sophisticated programming model with a full set of APIs, so developers have total control on: how application is partitioned, where code is offloaded and how remotable code interacts with local code. In Jade, dependencies do not exist between remotable tasks, the profiler and optimizer do not need to analyze the whole program, thereby, energy cost of program profiling and cost model calculation is lower than MAUI.

Chun et al. proposed CloneCloud [3], an application partitioner and execution runtime that enables unmodified mobile applications running in an application-level virtual machine to seamlessly offload part of their execution from mobile devices onto device clones operating in a computational cloud. In CloneCloud, threads must be paused, all states of the threads must be transferred to the server, and then threads resume on the server in order to offload computation. Offloading is expensive, especially when the client is resource constraint mobile device. In contrast, code offloading in Jade is lightweight. Remotable objects are serialized, transferred, and deserialized, resulting in much lower overhead compared to thread migration.

References

[1] H. Qian and D. Andresen. Jade: An Efficient Energy-aware Computation Offloading System with Heterogeneous Network Interface Bonding for Ad-hoc Networked Mobile Devices. In Proceedings of the 15th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2014.

[2] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl. MAUI: Making smartphones last longer with code offload. In MobiSys, 2010.

[3] B. Chun, S. Ihm, P. Maniatis, M. Naik, A. Patti. CloneCloud: Elastic Execution between Mobile Device and Cloud. In ACM EuroSys, 2011.