Machine Learning based Runtime Scheduler with Online-Training Mechanism for Mobile Offloading Framework

Heungsik Eom, Renato Figueiredo
Advanced Computing and Information Systems Laboratory
Electrical and Computer Engineering
University of Florida, Gainesville, Florida, USA
$\{$*hseom, renato*$\}$*@acis.ufl.edu*

Huaqian Cai, Gang Huang
Operating System and Middleware Laboratory
School of Electronics Engineering and Computer Science
Peking University, Beijing, China
$\{$*caihq12, hg*$\}$*@pku.edu.cn*

***Abstract*—**

***Keywords*-Mobile platform, computation offloading, machine learning, runtime scheduler, online training**

I. Introduction

Over the last decade, mobile offloading techniques have emerged as intelligent means to overcome the constraints of the limited resources from mobile platforms, smartphones and tabletPCs, so that these types of devices delegate computationally intensive computing tasks to more powerful external resources such as personal workstations or cloud servers. Initially, most of research interests on mobile offloading techniques have focused on core mechanisms in which *what to offload* and *how to offload* have been primarily considered. The research community has studied various approaches to implement mobile offloading frameworks which fall in the following categories: application partitioning [1]–[3], thread migration [4], [5], application migration [6], and distributed offloading frameworks [7]–[9].

However, one important fact from an offloading performance standpoint of view is that benefits from offloading computation-intensive portions of mobile applications can be influenced by various internal and extenal factors such as application requirements, network conditions, and computing capabilities of mobile or external devices. Thus, *whether to offload or execute locally* needs to be decided periodically by monitoring aforementioned dynamic features on runtime. Otherwise, incorrect offloading decisions may cause the performance degradation or worse energy consumption. For that reason, research focuses have been naturally shifted into dynamic scheduling or decision making problems for mobile offloading framework. For example, Kwon et al. [**?**] consider a simple rule-based scheduler in which the framework decides to offload the mobile computation only when the data transfer size is greater than a certain threshold. MAUI [2] utilizes a linear regression model among predefined features to make offloading decisions.

Even though these studies on the runtime scheduler for mobile offloading system take dynamic features such as data transfer size or network conditions into account to make offloading decisions, it is impractical for these approaches to build a globally well-defined offloading decision policy while considering all possible cases against dynamic mobile environments. Furthermore, it is difficult to generalize the above efforts for various mobile application use case scenario, since different applications need to have different offloading decision policies due to different application requirements or characteristics. Therefore, in practice, it is necessary for the scheduler to learn from self-observation of the previous decision correctness and to dynamically adapt the decision policy on runtime, thereby an identical decision model can be *generally* applied to various mobile applications without any predefined decision policies.

In this paper, we aim to develop a general framework for a runtime adaptive scheduler for mobile offloading framework by employing various types of machine learning techniques By applying machine learning techniques to scheduling problems for mobile offloading framework, the machine learning classifier can make decisions on whether mobile computations should be offloaded to external resources or executed locally. To this end, we modularized our previous work on the machine learning based runtime scheduler [**?**] in which any appropriate machine learning classifiers can be utilized for the runtime scheduler for mobile offloading framework. This work can be plugged and played with any types of mobile offloading frameworks in conjunction with the well-defined APIs to monitor and acquire dynamic features such as data size, network conditions, or the status of external devices. Furthermore, our work supports an online training mechanism for the machine learning based runtime scheduler such that it learns from observation on the previous offloading decision performance, and dynamically adapts its decision policy on runtime.

As part of the modularization effort, we integrated the modularized machine learning based runtime scheduler into the Java based mobile offloading system called *Dpartner* [**?**]. Originally, Dpartner depends on the web based user interface to offload or execute the offloadable tasks(i.e. classes) in local, so the user schedules them manually by manipulating the UI in a drag-and-drop way. By combining the machine learning based runtime scheduler with Dpartner, the user can be free of the scheduling task. Based on this integration, we evaluated the cost and performance for three machine learning algorithms, instance based learning, perceptron, and naïve bayes with respect to classifier build time, classification time, and scheduling accuracy. Even though there have been prior related studies which suggest utilizing machine learning techniques for mobile computing environments, to the best of our knowledge, our work is the first to consider the online training mechanism for the machine learning based runtime scheduler for mobile offloading framework and demonstarate the performance and cost of various machine learning algorithms for the use of the runtime scheduler of mobile offloading framework.

The rest of the paper is organized as follows. In Section II, we overview prior research efforts on offloading decision problems in mobile offloading framework as well as the use of machine learning techniques for scheduling problems from various domains. Section III summarizes our previous work which proposed the machine learning based runtime scheduler for mobile offloading framework. Section IV motives the concept of the online machine learning based runtime scheduler. In Section IV and V, we explain and evaluate our implementation of the online ML scheduler. Also, Section VI describes the current and potential applications for our work. Finally, we conclude the paper in Section VII.

II. Related Works

III. Adaptive Scheduler for Mobile Offloading System

In this section, we summarize our previous work on the machine learning based runtime scheduler for mobile offloading frameowork. In our previous work, we noticed the offloading performance dependency on various dynamic features. Then, we suggested applying machine learning techniques to the runtime scheduler for mobile offloading framework by showing the performance difference among various machine learning algorithms.

A. *Offloading Performance*

In our previous work [**?**], we demonstrated the performance dependancy

B. *Machine Learning based Runtime Scheduler*

IV. Challenge on Self-Training ML based Runtime Scheduler for Mobile Offloading System

A. *Offline vs. Online*

B. *Requirement for Self-Training ML based Runtime Scheduler for Mobile Offloading System*

V. Implementation of Self-Training ML based Runtime Scheduler

A. *Integration with Java based Mobile Offloading System*

B. *Self-Training Mechanism*

VI. Evaluation

A. *Training Cost*

B. *Offloading Decision Performance*

VII. Use Case

VIII. Conclusion and Future Work

References

[1] J. Flinn, S. Park, and M. Satyanarayanan, "Balancing performance, energy, and quality in pervasive computing." in *In proceeding of International Conference on Distributed Computing Systems(ICDCS)*, 2002, pp. 217–226.

[2] E. Cuervo, A. Balasubramanian, D. ki Cho, A. Wolman, S. Saroiu, R. Ch, and P. Bahl, "Maui: making smartphones last longer with code offload." in *In proceeding of International Conference on Mobile Systems, Applications and Services(MobiSys)*. ACM, 2010, pp. 49–62.

[3] R. Kemp, N. Palmer, T. Kielmann, and H. E. Bal, "Cuckoo: A computation offloading framework for smartphones." in *In proceeding of International Conference on Mobile Computing, Applications and Services(MobiCASE)*, vol. 76. Springer, 2010, pp. 59–79.

[4] B. G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: elastic execution between mobile device and cloud." in *In proceeding of the European Conference on Computer Systems(EuroSys)*, 2011, pp. 301–314.

[5] M. S. Gordon, D. A. Jamshidi, S. Mahlke, Z. M. Mao, and X. Chen, "Comet: code offload by migrating execution transparently," in *In proceeding of of 10th USENIX conference on Operating Systems Design and Implementation(OSDI)*. ACM, 2012, pp. 93–106.

[6] S. H. Hung, C. S. Shih, J. P. Shieh, C. P. Lee, and Y. H. Huang, "Executing mobile applications on the cloud: Framework and issues." *Computers and Mathematics with Applications*, vol. 63, no. 2, pp. 573–587, 2012.

[7] M. A. Hassan and S. Chen, "Mobile mapreduce: Minimizing response time of computing intensive mobile applications." in *In proceeding of International Conference on Mobile Computing, Applications and Services(MobiCASE)*, vol. 95. Springer, 2011, pp. 41–59.

[8] F. Yang, Z. Qian, X. Chen, I. Beschastnikh, L. Zhuang, L. Zhou, and G. Shen, “Sonora: A platform for continuous mobile-cloud computing.” in *In Technical Report, Microsoft Research Asia*, 2012.

[9] C. Shi, V. Lakafosis, M. H. Ammar, and E. W. Zegura, “Serendipity: enabling remote computing among intermittently connected mobile devices.” in *In proceeding of ACM International Symposium on Mobile Ad Hoc Networking and Computing(MobiHoc)*. ACM, 2012, pp. 145–154.