Lightweight Offloading System For Mobile Edge Computing

Hyuk-Jin Jeong
Department of Electrical and Computer Engineering
Seoul National University
Seoul, South Korea
jinevening@snu.ac.kr

***Abstract*—Rich mobile applications based on new technologies, such as deep neural network, rapidly increase computational demands in the mobile environment. A solution to run the computation-intensive applications on resource-constrained mobile devices is to offload computations to *edge servers*, computing servers located at the edge of the internet. Edge servers provide compute resources to mobile devices with a low end-to-end latency, but pose many research challenges on edge server customization and handling of the frequent disconnections of mobile devices. In this thesis, we propose a lightweight offloading system working on web-supported devices and techniques to offload mobile clients' DNN computations to edge servers. We design a web-based offloading system which exploits the portability of web platforms to migrate the execution of web applications to a remote server. We save the state of a web application as the form of web application code, named *snapshot*, so that the destination machine can restore the web application state by executing the snapshot. This significantly simplifies the migration process, allowing a web application to migrate within a few seconds without any pre-installation of application at the destination. Also, we propose incremental offloading of DNN, which simultaneously offloads DNN execution to an edge server while uploading a client's DNN model, to achieve performance benefits before uploading the whole DNN model. To adopt the incremental DNN offloading to realistic edge computing scenarios, we propose a multi-clients DNN partitioning algorithm for efficient utilization of edge server resources and proactive migration of DNN models for fast service handoff.**

I. Introduction

Emerging applications based on new technologies, such as real-time image processing with deep neural network (DNN), require intense computations, which cannot be accommodated with resource-constrained mobile devices. One solution to run the computation-intensive applications on mobile devices is to offload computations from mobile clients to remote servers. Existing central clouds are located in datacenters far from most mobile clients, leading to a problem for latency-sensitive applications which need servers in close proximity to meet the delay requirements. To resolve the issue, attention has focused on *edge servers*, computation servers dispersed at the edges of the network, e.g., computation servers on Wi-Fi APs, small cells, or a computing cluster made up of mobile devices.

Offloading with edge servers is different from offloading with central clouds. Edge servers are geographically widely distributed, and each of them covers a small region, compared to central clouds. Mobile clients can easily move across the boundary of edge servers, hence frequently being disconnected from an edge server and connecting to a new edge server. So, it is essential to rapidly establish the offloading state for user's seamless mobile experience. Another difference between edge and cloud servers is accessibility to the server. Cloud servers are accessible in any place where internet is supported, so mobile clients can save data in the cloud and use them in anytime. Otherwise, in edge computing, since a mobile client connects to a nearby edge server on demand, it is not plausible to save data in the edge server in advance. This difference makes it difficult to apply existing cloud computing techniques, e.g., computation partitioning, which assume application data is pre-installed at the server, to the edge server environment.

The purpose of our studies is to build an offloading system for edge computing, which enables fast migration of services without pre-installation of application (or the base system of the application) at the server. We exploit the portability of web applications, i.e., distributed as source code and runnable without installation, allowing a client to offload computations to any edge server equipped with web platforms. In addition, we focused on offloading of DNN computations in the edge server environment. We propose incremental offloading of DNN layers, which collaboratively executes DNN queries between a mobile client and an edge server while a client's DNN model is being uploaded to the server, to reduce the overhead of uploading a DNN model for offloading custom DNN computations.

II. Research Summary

This section summarizes our works both already published [1] [2] [3] and under construction [4] [5] [6].

A. Web-based Offloading

We propose an offloading system working on devices equipped with a *web platform*, which is an all-pervasive platform embracing mobile devices, personal computers, and servers. The system migrates the execution state of a web application from the client to the edge server before executing heavy computations, so that the edge server can execute the computations with its high-performance hardware. After execution, the edge server migrates the execution of the web application back to the client to continue execution at the client. When migrating application execution, we save the execution state of the web application in the form of executable web

978-1-5386-9151-9/19/$31.00 ©2019 IEEE

application code, called *snapshot*, allowing the destination machine can restore the execution state by simply executing the snapshot without any installation [3] [7]. The proposed system implemented on WebKit browser successfully offloaded web application execution to a remote server where no application-specific data is installed, and achieved a speedup up to 7.2x in real world web applications [3]. To reduce the time to migrate the execution state of a web application, we use a virtual DOM to migrate only the modified parts of the DOM tree state rather than generating the whole DOM tree from scratch [6]. Also, we adopted a pipeline scheme, which incrementally captures the snapshot of the execution state and sends the snapshot as soon as it is prepared, reducing the total migration time [6].

We applied the snapshot-based offloading to machine learning web applications performing image classification using convolutional neural networks (CNN) [1]. We addressed issues on pre-sending a DNN model to an edge server to reduce the time to upload a client's DNN model and conducting partial DNN inference at the client for better privacy [1]. Experimental result showed that our system can migrate the application execution much faster (up to ~2.5x) than a VM-based approach, which deploys a guest software encapsulated within a system VM [1]. Now, we are working on developing an offloading system that works based on a computation thread which can migrate between web-supported devices [4]. The system supports subsequent migration of service, allowing handoff and fallback in the edge computing environment. Also, we employed offloading of *webassembly*, a low-level binary format for web platforms, to reduce the performance gap between web-based offloading and offloading with native codes [4].

B. DNN Offloading

DNN is a de-facto standard solution for many complicated tasks, e.g., classification or regression. To offload a DNN computation to an edge server, a client's DNN model has to be prepared at the edge server before execution is offloaded to the server. However, in the edge server environment, it is not plausible to pre-install DNN models at the edge servers, because we cannot assure which edge servers will be used by the client on the move. Instead, it is more practical to upload a client's DNN model to an edge server on demand and then offload DNN execution to the edge server. The overhead of uploading the DNN model is not trivial, so we propose IONN (Incremental Offloading of Neural Network), which simultaneously offloads DNN execution while uploading a DNN model to start offloading before the whole DNN model is uploaded [2]. Also, we devised a linear time partitioning algorithm that decides which parts of the DNN model will be offloaded to the edge server, subject to minimizing the total DNN execution time [2].

We are extending IONN to support more realistic edge server environments. We considered a multi-client edge computing scenario where multiple mobile clients offload DNN computations to an edge server with finite resources. We devised a heuristic algorithm which decides the partitioning of DNN to minimize average DNN execution time of all clients while meeting a GPU resource constraint on the edge server [5]. Also, we predict the next edge server that a mobile client will visit and proactively migrate DNN layers to the server, so that the mobile client can start offloading DNN execution without uploading DNN layers to edge servers [5].

III. Conclusion

In this thesis, we propose an offloading system for web-supported devices based on the code-format snapshot, which allows lightweight and portable migration of application execution without pre-installation of applications on edge servers. Also, we propose DNN offloading techniques for edge computing, which allow the offloading of the execution of a mobile client's custom DNN model with small DNN uploading overhead in the realistic edge server environment (multi-clients and handoff situation). The future work would be the adaptation of advanced web features (e.g., webGL) to the web-based offloading system, and support of latest DNN structure (e.g., attention model, asynchronous reinforcement learning) in the edge computing. The issue of integration with existing VM/container-based edge computing is also an intriguing one which can be explored in further research.

Acknowledgment

I would like to thank my advisor Prof. Dr. Soo-Mook Moon for his support and valuable comments. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2017R1A2B2005562).

References

[1] H. Jeong, I. Jeong, H. Lee and S. Moon, "Computation Offloading for Machine Learning Web Apps in the Edge Server Environment," 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), Vienna, 2018, pp. 1492-1499.

[2] H. Jeong, H. Lee, C. Shin, and S. Moon. (2018). IONN: Incremental Offloading of Neural Network Computations from Mobile Devices to Edge Servers. In Proceedings of the ACM Symposium on Cloud Computing (SoCC '18). ACM, New York, NY, USA,

[3] H. Jeong and S. Moon. "Offloading of Web Application Computations: A Snapshot-Based Approach." 2015 IEEE 13th International Conference on Embedded and Ubiquitous Computing (2015): 90-97.

[4] H. Jeong, C. Shin, K. Shin, H. Lee, and S. Moon, "Mobile Web Worker: Migratable Thread Model for Web-based Mobile Edge Computing," 2018 (Under Review)

[5] H. Jeong, H. Lee, K. Shin, and S. Moon, " Offloading of Neural Network Computations from Multiple Mobile Clients to Edge Servers," 2018 (Under Review)

[6] H. Jeong, I. Jeong, and S. Moon. "Snapshot-based Offloading for Web Applications." 2018 (Under Construction)

[7] JinSeok Oh, Jin-woo Kwon, Hyukwoo Park, and Soo-Mook Moon. 2015. Migration of Web Applications with Seamless Execution. In Proceedings of the 11th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE '15). ACM, New York, NY, USA, 173-185.