Architectures and Distributed Algorithms for a Knowledge-Plane Supporting Edge Computing Applications

Alessandro Gaballo

EURECOM - Data Science and Engineering

Professional Thesis Report (confidential)

Research carried out at Saint Louis University under the supervision of Professor Flavio Esposito

January 2018

This thesis is dedicated to
someone
for some special reason

Acknowledgements

plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle.

Abstract

The spreading of smart mobile devices and the development of IoT has brought computation power in everyone's pocket, allowing people to perform simple task with their smartphones. There are some tasks however, which are more computationally expensive and therefore cannot be performed on a mobile device; for these type of tasks computation offloading represents a valid solution. Computation offloading is the process of transferring computation tasks to another platform for reasons such as limited computation resources and energy saving, which are evident in mobile and IoT devices. Typically, tasks are offloaded to powerful machines in the cloud, however for time sensitive applications the cloud could be too far, this is why the edge computing paradigm is spreading. Edge computing moves the processing from the core to the edge of the network, improving user experience by reducing the perceived latency. One way to reduce latency is to chose a proper path to the destination; paths advertised by the routing algorithms are not always the best because they don't take into account the network conditions. Our conjecture is that cooperative routing-based methods will steer edge traffic with (statistically) better quality, i.e., shorter end-to-end delays than reaction-based methods, such as Cartographer [3]. A way to perform path prediction is to use machine learning techniques, which in the last few years have spreaded and changed the way software is made [1]. Machine learning is the field of computer science that gives computers the ability to learn without being explicitly programmed [2], by extracting patterns from data. Recently, many goal have been achieved with the use of machine learning in the field of computer vision, gaming, text and speech recognition, but not much has been done in networking: this work is a first approach to machine learning in networking for path prediction. In this project, we present a system architecture to manage the offloading process of a task at the edge of the network: this architecture has a modular design so that the offloading policies could be easily plugged into the system. As part of the offloading

policies we implement a path prediction system based on a deep neural network, with the aim of reducing the latency with respect to the standard routing policies.

Abstract

The spreading of smart mobile devices and the development of IoT has brought computation power in everyone's pocket, allowing people to perform simple task with their smartphones. There are some tasks however, which are more computationally expensive and therefore cannot be performed on a mobile device; for these type of tasks computation offloading represents a valid solution. Computation offloading is the process of transferring computation tasks to another platform for reasons such as limited computation resources and energy saving, which are evident in mobile and IoT devices. Typically, tasks are offloaded to powerful machines in the cloud, however for time sensitive applications the cloud could be too far, this is why the edge computing paradigm is spreading. Edge computing moves the processing from the core to the edge of the network, improving user experience by reducing the perceived latency. One way to reduce latency is to chose a proper path to the destination; paths advertised by the routing algorithms are not always the best because they don't take into account the network conditions. Our conjecture is that cooperative routing-based methods will steer edge traffic with (statistically) better quality, i.e., shorter end-to-end delays than reaction-based methods, such as Cartographer [3]. A way to perform path prediction is to use machine learning techniques, which in the last few years have spreaded and changed the way software is made [1]. Machine learning is the field of computer science that gives computers the ability to learn without being explicitly programmed [2], by extracting patterns from data. Recently, many goal have been achieved with the use of machine learning in the field of computer vision, gaming, text and speech recognition, but not much has been done in networking: this work is a first approach to machine learning in networking for path prediction. In this project, we present a system architecture to manage the offloading process of a task at the edge of the network: this architecture has a modular design so that the offloading policies could be easily plugged into the system. As part of the offloading

policies we implement a path prediction system based on a deep neural network, with the aim of reducing the latency with respect to the standard routing policies.

Contents

List of Figures

Chapter 1

Introduction

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Einstein's paper: [**?**]

Chapter 2

Background

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Chapter 3

Conclusions and Results

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Appendix A

Sample Title

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Appendix B

Sample Title

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Bibliography

[1] Andrej Karpathy. Software 2.0. https://medium.com/@karpathy/software-2-0-a64152b37c35.

[2] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, July 1959.

[3] Patrick Shuff. Building a billion user load balancer. Dublin, 2015. USENIX Association.