UNIVERSITY OF AMSTERDAM

PROJECTPLAN

A simulation framework for virtualized networking infrastructure

April 6, 2021

Lecturer:
dr. Chrysa Papagianni

Student:
Tom J. Wassing
12386715

Course:
Afstudeerproject bachelor Informatica

Course code:
5062ABI18Y

1 Introduction

In the modern cloud, the combination of virtualization and Software Defined Networks (SDN) forms the basis for the current cloud architecture. The dynamic and programmable properties of SDN bring many benefits to large-scale networks. Virtualization of network services removes the need for specialized hardware and offers much flexibility. Combining these two principles makes it possible to virtualize and host networks within a virtualized environment completely.

Service Function Chaining (SFC) creates a chain of network functions. Chaining is achieved by dynamically routing traffic through virtualized network functions (e.g., firewalls, NAT traversal, packet inspection) to form a service using the capabilities of SDN. This approach is radically different than how services were created in the past, in which specialized hardware was placed in a particular order, and a lot of manual configuration was needed. This also introduced the need for planning capacity where resources needed to be over-provisioned to ensure the service requirements in case of a surge in usage are met, leading to many wasted resources when standard traffic patterns are observed. By virtualizing the network functions, the cloud becomes more flexible, cheaper, and most importantly scalable. The virtualized nature of the cloud creates new opportunities to manage the resources more efficiently, making it possible to save cost and reduce energy waste.

1.1 Problem

Managing resources efficiently for a virtualized environment boils down to the allocation of computational and transport resources to one ore more SFCs, optimized with regards to resources utilization and cost. However, this is constrained by the service requirements that have to be met in order for the SFC to function properly. The allocation problem is often referred to as *SFC embedding.* Numerous approaches have addressed the resource allocation problem as a traditional optimization problem [6]. More recently machine learning (ML) has been used as a new approach to the optimization problem, because of the unpredictable nature of network environments and traffic patterns. In particular reinforcement learning such as the Q-Learning algorithm have been suggested [5, 3].

1.2 Research question

A Java-based simulator called *CVI-Sim* has been used in literature by Papagianni et al. to serve as an experimentation environment in order to evaluate the analytically oriented solution for the resource allocation problem. The framework does not have a interface to explore new machine learning methods to tackle the SFC embedding problem. In this project we want to research the following question:

"How can we extend the *CVI-Sim* with an interface to provide for integration of open-source machine learning libraries using an asynchronous messaging system, to allow for experimentation of AI/ML to tackle the SFC embedding problem"

The research project produces an extension of the original simulator source, along with documentation about the newly added features and description of the source code in the form of a UML diagram and text. Furthermore, a small study will be done about the inner workings of the simulator, to determine which machine learning library and asynchronous messaging system will be chosen to be integrated.

1.3 Related work

Simulators to emulate cloud environments have been build before, apart from the simulator created in the research of Papagianni et al. (2013). *CloudSim* [2] is a popular simulator that allows for the evaluation of resource provisioning algorithms. *CloudSim* provides a framework to simulate higher level cloud resources in contrast to the *CVI-Sim* [6], which is more focused on low-level data center resources allocation. *CloudSim* uses an internal message queue system in which state changes and resource allocations are communicated through a message broker to the underlying framework. The internal message system could serve as a design guideline for the implementation of the asynchronous messaging system of the extension.

With regards to the use of ML used as means to address the SFC embedding problem. Indicatively, in the paper of Mijumbi et al. (2014) a decentralised learning system is proposed. The approach uses the Q-Learning reinforcement learning algorithm in a multi-agent system setup to achieve decentralization. The model performs actions individually on each node but uses the complete network state to act upon. The complete network state consist of the utilisation of the nodes but also the utilisation of the links. The results how that the learning approach can not make the right decisions in the beginning, since it has not gather sufficient information. When enough time and packets have passed the learning approach catches up with the static approach. The paper shows that a multi-agent system can be used to tackle the SFC embedding problem. Due to the multi-agent system setup, the extension of the *CVI-Sim* must be able to support multi-agent systems with the possibility for the agent to performs actions on a single node. Also nodes and links in the simulator should expose their utilisation for the agents to act upon.

The paper of Fu et al. published in 2019 proposed a similar reinforcement learning solution as Mijumbi et al.. The main difference between the papers is that a single agent system is used in contrast to the multi-agent system. This approach also shows a significant benefit over static resource allocation. The implications of the single-agent system approach in contrast to the multi-agent system approach [5] is that the *CVI-Sim* extension must support both multi-agent and single-agent setups. To support single-agent setups, the extension must provide a way to manipulate the network as a whole, as opposed to the multi-agent setup on which actions must be taken a the individual node level. The paper also provides us with some insight into the simulation process, where they used the machine learning library TensorFlow [1] in combination with the graph library NetworkX [4] to simulate and evaluate their SFC embedding solution.

2 Method

The research can be divided into roughly three phases: exploration, implementation and, evaluation. In the first phase of the project, all components (simulator, machine learning library, and messaging system) must be examined. The simulator must be understood on a functional and code level, whereby careful attention must be paid to how to communicate with the machine learning agent at a later stage. Then it must be investigated how a machine learning library can be used to perform experiments. For the two components to work together, it must be investigated how an asynchronous message system can be used for communication. All this together will result in a textual explanation with necessary UML diagrams.

The second phase of the research mainly consists of the implementation. An interface will be built in the simulator where communication will take place with the machine learning agent, using an asynchronous message queue. Using the communication channel, the simulator can take actions based on the incoming responses from the ML agent, but also messages are sent to the machine learning agent when events take place in the simulator. Subsequently, the same kind of interface will be build on the machine learning agent's side, whereby the algorithm must be able to determine the state in the simulator and perform actions on it. The result of this phase will mainly be code and a deployment configuration, with documentation and diagrams of the adjustments.

In the final phase of the project, the correctness of the solution must be tested. The simulator will be verified by comparing the results of a reference native java-based Q-learning implementation with the proposed one, which will be documented and conclude the research question.

3 Planning

The project consists of approximately 480-500 hours. The planning presented in a Gannt chart in figure 3, following the method's tasks, shows a rough estimation of the tasks that have to be completed.

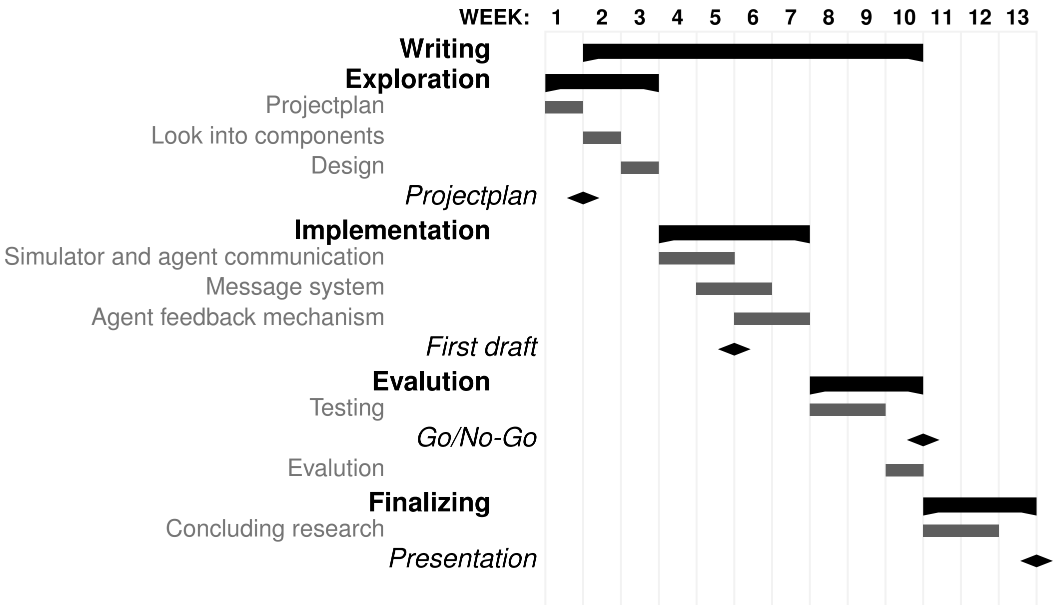

Figure 1: Gantt chart

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, Savannah, GA, November 2016. USENIX Association. ISBN 978-1-931971-33-1. URL `https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi`.

[2] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, César A. F. De Rose, and Rajkumar Buyya. Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Softw. Pract. Exper.*, 41(1):23–50, January 2011. ISSN 0038-0644. doi: 10.1002/spe.995. URL `https://doi.org/10.1002/spe.995`.

[3] X. Fu, F. R. Yu, J. Wang, Q. Qi, and J. Liao. Service function chain embedding for nfv-enabled iot based on deep reinforcement learning. *IEEE Communications Magazine*, 57(11):102–108, 2019. doi: 10.1109/MCOM.001.1900097.

[4] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. In Gaël Varoquaux, Travis Vaught, and Jarrod Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

[5] R. Mijumbi, J. Gorricho, J. Serrat, M. Claeys, F. De Turck, and S. Latré. Design and evaluation of learning algorithms for dynamic resource management in virtual networks. In *2014 IEEE Network Operations and Management Symposium (NOMS)*, pages 1–9, 2014. doi: 10.1109/NOMS.2014.6838258.

[6] C. Papagianni, A. Leivadeas, S. Papavassiliou, V. Maglaris, C. Cervelló-Pastor, and Á. Monje. On the optimal allocation of virtual resources in cloud computing networks. *IEEE Transactions on Computers*, 62(6):1060–1071, 2013. doi: 10.1109/TC.2013.31.