POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

OSCAR-P: a framework for automating application profiling in the computing continuum

Tesi di Laurea Magistrale in
Computer Science and Engineering - Ingegneria Informatica

Enrico Galimberti, 10573044

Advisor:
Prof. Danilo Ardagna

Co-advisors:
Federica Filippini,
Hamta Sedghani

Academic year:
2021-2022

Abstract: This thesis proposes an auto-profiling tool for OSCAR, an open-source platform able to support serverless computing for scientific data-processing. The tool, named OSCAR-P, is designed to automatically test a specified application workflow on different hardware and node combinations, obtaining relevant information on the timing of the execution of the individual components. It then uses the collected data to build performance models using machine learning, making it possible to predict the performance of the application on unseen configurations. OSCAR-P has been tested on clusters with different architectures (x86 and ARM64) and with different workloads. The use case considered for the test is a mask detection application that can be executed in a smart city to detect how many people are not wearing a face mask in a certain area.
OSCAR-P proved its efficiency, greatly reducing the time needed to set up OSCAR, collect the logs and process them manually. The preliminary results obtained on the performance models accuracy are also promising, showing a mean absolute percentage error lower than 20% in all the considered scenarios.

Key-words: edge computing, computing continuum, performance profiling, machine learning

1. Introduction

The client-server paradigm has been for years the standard computing model for industrial-level distributed systems. That model evolved over time and nowadays we have edge systems, where data produced by devices at the edge of the network has to be moved to a centralized data-center, processed, and then shipped back to their origin point to be used. This is not efficient, as it increases both the bandwidth usage and the latency, introducing a round-trip time delay for every access to the central servers. As the number of those edge devices kept on growing [1], so did the amount of data they were capable of producing, and the combinations of this two factors inevitably put an higher level of stress both on the network and on the remote servers [2]. However as time passed those devices also became more technologically advanced, with improvements both to their processing power and their storing capacity. The natural evolution of the system was therefore moving a part of the computation on those machines at the edge of the network, close to where the data is produced [3].

The aim of this novel distributed computing paradigm is not to completely remove centralized data centers, but to create a compute continuum that seamlessly integrates the edge devices with the remote servers, usually running in the cloud, splitting the workload among them. The load sharing is not restricted to only two distinct

entities either: the computation can happen on multiple layers, with the cloud on one side, IoT (e.g. AI-enabled edge sensors) on the other, and edge servers in between, such as smartphones, Raspberry Pis or just PC and laptops.
This approach offers several advantages: i) lower latency: by moving part of the computation where the data resides we remove the round-trip-time delays needed to access the remote cloud data centers, resulting in faster response times and better performance overall; ii) reduced bandwidth usage: local processing at the edge removes the need to send huge amounts of data through the network, saving bandwidth, avoiding bottlenecks and leaving more room for expansion; iii) improved privacy: the possibility to process some of the data on edge devices means that said data can be "anonymized" on the spot before being communicated, ensuring that potentially sensitive information are never moved and saved on the central server; iv) better scalability: edge devices are usually cheap, and adding more to the network to help with data processing is both easy to implement and economical.

Another novelty in cloud computing is represented by the introduction and quick rise in popularity [4][5] of a new type of model called Functions as a Service, or FaaS for short, that breaks down complex applications in workflows that run on reusable services. The FaaS model allows the execution of code, usually both reduced in its scope and short-lived, inside of stateless containers activated by an event, e.g. the upload of a file. Data processing systems can greatly benefit from this approach, as they can trigger the execution of resource-intensive applications on demand just by uploading data inside a storage bucket, and obtain the results as soon as the processing is done from another storage bucket. The use of containers instead of full virtual machines (VMs) both reduces the development and deployment complexity and the resource usage, and given their stateless nature they can be reused to serve another event as soon as the previous computation completes, without having to be discarded and then recreated. The FaaS model is also more flexible, as the containers can be created or destroyed dynamically in response to an increase or decrease of the workload, and therefore it works well in scenarios with a workload that is on average low, interleaved with peaks of activity. For industries that rely on public clouds this approach can turn out to be more economical, as they can now pay per seconds of actual usage instead of paying a fixed rate at all times, regardless of how many resources are actually consumed [6].

This thesis is developed under the AI-SPRINT project[1], that aims at executing AI applications in secure privacy-preserving computing continuum supported by the OSCAR[2] framework, a state-of-the-art runtime environment for edge applications, built to support FaaS efficiently and on-premises. OSCAR is a porting of the SCAR framework to an on-premises scenario, and it aims at creating an highly-parallel event-driven file-processing serverless environment to execute general-purpose file-processing computing applications.

Achieving low latency and high throughput is one of the main drivers behind edge computing, however evaluating performance of a complex application, whose components may be allocated on different levels of the computing continuum, is a problem without an easy solution. There is a need for automated tools that can help with the profiling of those components, able to predict execution times with a varying amount of resources available (such as cores number). The end goal of such a tool is to guarantee upper bounds on applications execution, and find the optimal hardware structure for deployment.

The goal of the thesis is therefore twofold:

- implement *OSCAR-P*, an auto-profiling framework built around OSCAR, able to automatically test a specified application workflow on different hardware and node combinations, obtaining relevant information on the timing of the execution of the individual components
- use the resulting data to build performance models using machine learning, making it possible to predict the performance of the application on unseen configurations

This thesis is organized as follows. Section 2 presents relevant literature works related to the same area as the subject of the thesis. Section 3 gives an overview of the OSCAR framework and its architecture. Section 4 explains in details the *OSCAR-P* goals and its components, while Section 5 focuses on the experimental scenarios considered to validate the tool. Finally, Section 6 summarizes the achievements of the thesis and introduces future work.

[1] `https://www.ai-sprint-project.eu`
[2] `https://docs.oscar.grycap.net`

6. Conclusions and future work

During this thesis we achieved our goal of developing an auto-profiling framework for OSCAR, named *OSCAR-P*, that can automatically test an application workflow on different hardware combinations and generate machine learning performance models from the collected data.
The framework proved its efficiency, greatly reducing the time needed to set up OSCAR, collect the logs and process them manually. With *OSCAR-P* the whole procedure is automatic and the user has just to compile the configuration file and launch the tool. It also works on cluster with different architectures, having been tested both on VM (x86) and Raspberry Pis (ARM64).
The results obtained from the experimental campaigns, even though only preliminary, were still good, with the performance models having a mean absolute percentage error lower than 10% on interpolation and lower than 20% on extrapolation for the first testing campaign. The results were a bit worse on the second VM campaign but the data collected was also more noisy due to the shorter execution times.
Future works include expanding *OSCAR-P* capabilities to enable the profiling of heterogeneous clusters, and testing it on industrial-scale clusters.

References

[1] P. Middleton, T. Tsai, M. Yamaji, A. Gupta, and D. Rueb, "Forecast: Internet of things–endpoints and associated services," *Worldwide.[(accessed on 5 May 2018)]*, 2017.

[2] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[3] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[4] A. Das, S. Patterson, and M. Wittie, "Edgebench: Benchmarking edge computing platforms," in *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*, pp. 175–180, IEEE, 2018.

[5] I. Baldini, P. Castro, K. Chang, P. Cheng, S. Fink, V. Ishakian, N. Mitchell, V. Muthusamy, R. Rabbah, A. Slominski, *et al.*, "Serverless computing: Current trends and open problems," in *Research advances in cloud computing*, pp. 1–20, Springer, 2017.

[6] L. F. Albuquerque Jr, F. S. Ferraz, R. Oliveira, and S. Galdino, "Function-as-a-service x platform-as-a-service: Towards a comparative study on faas and paas," in *ICSEA*, pp. 206–212, 2017.

[7] M. Sewak and S. Singh, "Winning in the era of serverless computing and function as a service," in *2018 3rd International Conference for Convergence in Technology (I2CT)*, pp. 1–5, IEEE, 2018.

[8] S. Risco, G. Moltó, D. M. Naranjo, and I. Blanquer, "Serverless workflows for containerised applications in the cloud continuum," *Journal of Grid Computing*, vol. 19, no. 3, pp. 1–18, 2021.

[9] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019.

[10] X. Wang, Y. Han, V. C. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 869–904, 2020.

[11] E. Li, L. Zeng, Z. Zhou, and X. Chen, "Edge ai: On-demand accelerating deep neural network inference via edge computing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 447–457, 2019.

[12] S. Disabato, M. Roveri, and C. Alippi, "Distributed deep convolutional neural networks for the internet-of-things," *IEEE Transactions on Computers*, vol. 70, no. 8, pp. 1239–1252, 2021.

[13] Y. Kang, J. Hauswald, C. Gao, A. Rovinski, T. Mudge, J. Mars, and L. Tang, "Neurosurgeon: Collaborative intelligence between the cloud and mobile edge," *ACM SIGARCH Computer Architecture News*, vol. 45, no. 1, pp. 615–629, 2017.

[14] W. Jia, K. A. Shaw, and M. Martonosi, "Stargazer: Automated regression-based gpu design space exploration," in *2012 IEEE International Symposium on Performance Analysis of Systems & Software*, pp. 2–13, IEEE, 2012.

[15] U. Gupta, M. Babu, R. Ayoub, M. Kishinevsky, F. Paterna, S. Gumussoy, and U. Y. Ogras, "An online learning methodology for performance modeling of graphics processors," *IEEE Transactions on Computers*, vol. 67, no. 12, pp. 1677–1691, 2018.

[16] J. Grohmann, M. Straesser, A. Chalbani, S. Eismann, Y. Arian, N. Herbst, N. Peretz, and S. Kounev, "Suanming: Explainable prediction of performance degradations in microservice applications," in *Proceedings of the ACM/SPEC International Conference on Performance Engineering*, pp. 165–176, 2021.

[17] N. Mahmoudi and H. Khazaei, "Temporal performance modelling of serverless computing platforms," in *Proceedings of the 2020 Sixth International Workshop on Serverless Computing*, pp. 1–6, 2020.

[18] N. Mahmoudi and H. Khazaei, "Performance modeling of serverless computing platforms," *IEEE Transactions on Cloud Computing*, 2020.

[19] M. Copik, G. Kwasniewski, M. Besta, M. Podstawski, and T. Hoefler, "Sebs: A serverless benchmark suite for function-as-a-service computing," in *Proceedings of the 22nd International Middleware Conference*, pp. 64–78, 2021.

[20] A. Das, S. Imai, S. Patterson, and M. P. Wittie, "Performance optimization for edge-cloud serverless platforms via dynamic task placement," in *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*, pp. 41–50, IEEE, 2020.

[21] J. McChesney, N. Wang, A. Tanwer, E. de Lara, and B. Varghese, "Defog: fog computing benchmarks," in *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, pp. 47–58, 2019.

[22] T. Bures, V. Matena, R. Mirandola, L. Pagliari, and C. Trubiani, "Performance modelling of smart cyber-physical systems," in *Companion of the 2018 ACM/SPEC International Conference on Performance Engineering*, pp. 37–40, 2018.

[23] U. Tadakamalla and D. A. Menasce, "Autonomic resource management for fog computing," *IEEE Transactions on Cloud Computing*, 2021.

[24] F. J. Ferri, P. Pudil, M. Hatef, and J. Kittler, "Comparative study of techniques for large-scale feature selection," in *Machine Intelligence and Pattern Recognition*, vol. 16, pp. 403–413, Elsevier, 1994.

[25] E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik., *Quantitative system performance: computer system analysis using queueing network models.* Prentice-Hall, 1984.

[26] M. Lattuada, E. Gianniti, D. Ardagna, and L. Zhang, "Performance prediction of deep learning applications training in gpu as a service systems," *Cluster Computing*, pp. 1–24, 2022.

[27] F. Filippini, M. Lattuada, A. Jahani, M. Ciavotta, D. Ardagna, and E. Amaldi, "Hierarchical scheduling in on-demand gpu-as-a-service systems," in *2020 22nd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pp. 125–132, IEEE, 2020.

[28] A. Maros, F. Murai, A. P. C. da Silva, J. M. Almeida, M. Lattuada, E. Gianniti, M. Hosseini, and D. Ardagna, "Machine learning for performance prediction of spark cloud applications," in *2019 IEEE 12th International Conference on Cloud Computing (CLOUD)*, pp. 99–106, IEEE, 2019.

[29] E. Ataie, A. Evangelinou, E. Gianniti, and D. Ardagna, "A hybrid machine learning approach for performance modeling of cloud-based big data applications," *The Computer Journal*, 2021.