From Cloud to Serverless: MOO in the new Cloud epoch

Michail Georgoulakis Misegiannis
School of ECE, National Technical University of Athens
Athens, Greeece
el16061@mail.ntua.gr

Laurent d'Orazio*
Univ. Rennes, CNRS, IRISA
Lannion, France
laurent.dorazio@univ-rennes1.fr

Verena Kantere
School of ECE, National Technical University of Athens
Athens, Greece
verena@dblab.ece.ntua.gr

ABSTRACT

During the last 10 years, the volume of global data has risen more than tenfold. The commercial rise of cloud computing eased the process of storing, processing and managing big data. Recently, the cloud evolved with the emergence of serverless computing platforms that offer an even more abstracted service model. The elasticity of cloud computing creates significant optimization problems, which can be tackled either with a single objective, or as multi-objective opimization problems (MOO). When it comes to data management, the two main MOO problems in a cloud computing environment are query optimization and task scheduling. Some of the techniques used for solving MOO problems in the cloud are the weighted sum model, mathematical-programming based algorithms and genetic algorithms.

We propose the presentation of a tutorial that will underline the main MOO problems of a cloud computing environment in regards to data management, and evaluate the use of serverless computing for such problems. The tutorial will offer the audience a better understanding of current MOO challenges and applications in the cloud, while also giving them an overview of different solutions to such problems and the techniques that can be used for solving them.

1 DURATION

The tutorial will be presented in one and a half hour. It will consist of three parts, lasting half an hour each. The topics to be discussed in each part will be presented in the following sections.

2 TUTORIAL DESCRIPTION

2.1 From Cloud Computing...

Cloud computing refers to the on demand delivery of computing services to different clients. These services are provided by cloud vendors on three levels of abstraction, each one represented by a service model:

Infrastructure as a service (IaaS) In IaaS platforms, users can rent and manage data center infrastructure like compute and storage services in the form of virtual machines.

Platform as a service (PaaS) PaaS provides hardware and software tools for the user to build and manage a customized application.

Software as a service (SaaS) SaaS delivers web applications to its users, which are fully managed by the SaaS vendor.

Cloud computing comes with significant advantages, enabling users to make savings in cost through a pay for use pricing scheme, and providing elasticity in terms of resource management. The significance of these benefits led to the emergence of a new cloud service model:

Function as a service (FaaS) Serverless computing consists of two service models, FaaS and BaaS (backend as a service). Serverless however most often refers to FaaS platforms, which are code execution environments where developers can write applications in the form of independent functions and execute them using cloud resources. FaaS has similarities with IaaS and PaaS. It differs from PaaS in terms of scaling, which is done automatically in FaaS, and its difference from IaaS is that apart from scaling, resource allocation and management are also automatic in FaaS. FaaS also has a pay for use charging policy, meaning that a FaaS user will never sustain and pay for idle resources.

Despite the advantages, the increased abstraction of FaaS comes with some significant limitations, not allowing it to reach its full potential [11, 14, 28]. Some of them are the following: (1) the functions short lifespan; (2) the lack of direct communication between functions; (3) the startup delay when invoking multiple functions (cold start); (4) the lack of resource heterogeneity in today's FaaS offerings; (5) security vulnerabilities.

Pricing and elasticity make the cloud a suitable environment to run data-intensive applications. As a result, large volumes of data are stored and processed in the cloud, and data management in such applications is critical. An important part of data management is query processing, which, combined with cloud resources' management are two procedures containing significant optimization problems. The focus of this tutorial are IaaS platforms, where the different resource allocation options mean that one combination of resources will be better than another, depending on user objectives, which leaves room for optimization problems.

Query processing consists of four stages: parsing, translation, optimization and evaluation, with query optimization being the most challenging one. During this phase, a logical plan is received by a query optimizer and an optimal physical query plan is returned, including the physical operators that will be used for query execution. The space of alternative query plans can be very big for complex queries, complicating query optimization.

When it comes to resource management, IaaS platforms include a pool of available resources which can be rented by their users to execute their tasks. The resource configuration a user chooses for an application will determine how efficiently the application will be executed. Furthermore, the task of scheduling tasks to available resources is a difficult optimization problem, due to the infinite space of alternative schedules. These two are the two main resource optimization problems in an IaaS environment.

Optimization problems such as query optimization and task scheduling can be tackled as multi-objective optimization problems (MOO). In MOO, the goal is to find a set of Pareto-optimal

*The supervisors' names are included in alphabetical order, both contributed equally.

solutions, which are solutions that cannot be improved in any of the objectives without degrading at least one of the other objectives. The different optimization goals considered for the aforementioned data and resource management problems can be the following: (1) execution time, the primal optimization objective in related MOO works (2) monetary cost, another important factor in cloud computing [10, 13, 18, 24] (3) energy consumption, closely related to monetary cost [23, 26] (4) result precision,which can be critical in cases where decision making has to happen quickly [1, 27].

Multi-objective query (MOQO) and resource optimization have been the basis of numerous research works, which will be analysed in the following sections. Most of of these works are based on IaaS platforms where decision making on resources complicates optimization problems.

2.2 ...To Serverless Computing

As FaaS built on some of the cloud's existing advantages, the same optimization problems occur there, offering new opportunities to benefit from when solving them:

Even more cost, energy savings By using a FaaS platform to solve MOO problems, users will pay less in many cases, due to its event-driven charging policy. In the same manner, cloud providers will save energy by not sustaining idle resources and will also have economic benefits.

Handling of massively parallel tasks Massively parallel applications are a good fit for serverless platforms [11], as they exploit their autoscaling feature, and requests need not communicate with each other. When considering data management and querying in the cloud, data-intensive database queries requiring processing over a distributed data set can be a good serverless fit, as their tasks can be parellelized and almost no data shipping between serverless functions will be required.

Response to interactive analytics Serverless has also proven to be a good fit for other data-intensive applications like interactive analytics, and subsequently is suitable for optimizing interactive queries [19].

However, the aforementioned limitations of serverless mean that the serverless paradigm also comes with significant obstacles concerning MOO in data management scenarios:

Data Shuffling Data shuffling is the most commonly used communication pattern in distributed query processing to transfer data across stages. In serverless computing, functions are stateless and do not communicate. Consequently, intermediate data between stages needs to be stored in persistent storage systems like Amazon S3 [19, 21], and accessing it each time comes with significant overhead. Slow data shuffling is a big obstacle for distributed query processing, having a negative impact on query execution time.

Lack of control over resources From a user perspective, the fact that there is zero control over computing resources and task scheduling means that the optimization challenges they offer are also abstracted by the users and rely on FaaS vendors' methods, which makes the need for quality solutions critical.

Short term tasks only Serverless functions have a short lifespan. In AWS Lambda, this limit is 15 minutes [17]. As a result, they cannot be used for long-running tasks like complex analytical queries, limiting even more the queries that can leverage the serverless paradigm.

Security Issues Security is also an open issue in serverless. Existing platforms present some vulnerabilities due to increased co-residency, as well as higher leak probability because of increased network communication.

Serverless might be gaining ground, however at the moment the areas where FaaS performs better and overcomes traditional, VM-based IaaS are still limited [2]. MOQO and resource optimization problems in IaaS have been a hot research topic with numerous contributions.

2.3 Techniques and Motivations

Solving MOO problems comes with a set of challenges most of which concern the generation of Pareto optimal solutions. An MOO problem can be efficiently tackled by producing a good number of Pareto optimal solutions which are diverse enough to provide a good coverage of the Pareto front

In order to respond to this challenge, a lot of MOO techniques have been used. Multi-objective optimization methods can be divided into four classes depending on the decision making involved.

No preference methods The goal is to find a relevant set of Pareto optimal solutions providing different trade-offs, and no decision making is involved.

A priori methods The decision maker has provided his/her weighted objectives and then a single solution best satisfying these preferences is returned.

A posteriori methods The two aforementioned techniques are combined, as a representative set of Pareto optimal solutions is first found, and then it is up to the decision maker to select the most suitable for him.

Interactive methods The decision maker iteratively searches for his/her preferred solutions, each time providing the system with information to specify and improve the quality of proposed trade-offs and solutions. It is up to the decision maker to stop the search and pick a solution.

When optimization happens a priori, MOO is usually tackled with a scalarized method, which translates the MOO problem to a SOO problem with Pareto optimal solutions. When decision making happens in the end, an MOO problem is tackled either with a mathematical-programming based method, or with an evolutionary method that involves a genetic algorithm.

Scalarized Methods The most typical scalarized method is linear scalarization, also known as the weighted sum model. With the use of a weighted sum model (WS), a MOO problem is transformed into a single objective one, by distributing a number of preference weights into the objectives [8, 10, 25]. It calculates the optimal plan for a number of weight distributions, and returns a Pareto plan set (PPS). WS can find every point on the Pareto-optimal front for a convex problem, however this is not the case for non-convex problems, where the WS approach can miss some Pareto-optimal points. A slight fixup method to this is the e-constraint method which is similar to the weighted sum, but it repeatedly adds constraints to all but one objective, forcing this way the generation of more diverse solutions [5].

Mathematical-programming based Methods A common class of a posteriori MOO methods are mathematical programming based ones where an algorithm is implemented and run repeatedly. After each loop, a Pareto optimal solution is produced. This algorithm can be exhaustive [16, 27], linear programming-based [15], probabilistic [16], greedy [16], or incremental [26]. Those algorithms usually solve the multi-objective optimization problem by constructing several scalarizations.

Evolutionary Methods The other class of a posteriori MOO methods are evolutionary ones, where one run of a genetic algorithm produces a set of Pareto optimal solutions. Non-dominated Sorting Genetic Algorithm-II (NSGA-II) is probably the most popular choice [4, 23] or basis for new algorithms [18, 20], with interesting alternatives having considered SPEA2 [4], particle swarm optimization [3], techniques based on simulated annealing [7], and ant colony optimization [22]. Genetic algorithms are a good fit for tackling MOO problems, as they generate sets of solutions with a parallel search, they can apply techniques like niching in order to ensure the divergence of solutions, and finally the parallelism achieved through evolutionary algorithms' operators can provide a better search speed. However, in some cases the convergence of the algorithm (and consequently the Pareto optimality of the solution set) cannot be guaranteed and measured, and might depend on the initial solutions provided. [6].

Most of the works mentioned aim to optimize query execution time alongside monetary cost, however other MOO works aim to reduce energy consumption alongside latency [23, 26], and others including result precision in their use case scenarios [27], or having error percentage constraints alongside time constraints [1]. Most works involved two or three objectives, with the exception of some more generic solutions having used up to 9 optimization metrics

The target environment of most research works is IaaS platforms. However, there have also emerged works studying scheduling in FaaS environments [4, 12], and more are expected to appear in the future, as serverless steadily gains ground in the cloud landscape [25].

2.4 Usecases and Applications

In this section, we present some exemplary systems using the presented techniques as well as the use cases that they support:

Weighted Sum Model Normalized Weighted Sum Model (NWSM) [9, 10] is a user-interactive MOQO strategy based on a modified weighted sum model and is used to achieve MOO in a mobile-cloud database environment, able to take into account 3 cost objectives: query latency, monetary cost, and mobile device energy consumption. By using NWSA, the user assigns weights of importance to each of the 3 objectives, which are multiplied with the different metric costs to produce the final cost estimation of a QEP. In the end, the QEP with the lowest score is selected and executed. More recently, Kllapi et al. [15] tackled dataflow optimization with index interleaving algorithms aiming to utilise idle slots in the dataflow execution schedule and build indexes in parallel, without extra delay or cost. In order to include both of the objectives in the optimization process, the weighted sum method is used.

MOO Algorithms Multi-objective parametric query optimization [27] takes a different approach to query optimization, which happens with the use of an exhaustive DP algorithm executing before query runtime, and models queries as functions of parameters. When a query is submitted its parameters are specified, and optimization is completed a posteriori. Real life applications of such technique involve querying cloud databases with a time-money trade-off. In another work concerning dataflow schedule optimization in the cloud [16], several greedy, probabilistic, and exhaustive optimization algorithms are used to explore the space of alternative schedules, before choosing an optimal one based on a time-money trade-off.

Evolutionary Algorithms Christoforou et al. [4] proposed a new resource management approach in a FaaS platform, based on intelligent techniques and genetic algorithms. Three different algorithms were used to find a set of near-optimal solutions to the MOO problem, and their performance is compared. In another MOO work aiming to reduce energy consumption and makespan [23], the DVFS technique is used to reduce energy consumption, which enables VMs to run at different blends of frequencies and voltages. The MOO is solved with the help of NSGA-II for energy consumption and latency. Finally, Artificial Neural Networks were also used in the genetic algorithms optimization procedure to predict the available resources based on task characteristics.

Other An additional contribution, concerning MOQO came from Karampaglis et al. [13], who introduced the first proposal for a bi-objective query cost model suitable for optimization of queries executed over a multi-cloud environment. It successfully provides estimates of both the expected running time and the monetary cost associated with a query.

3 OUTLINE

The tutorial will start with a brief introduction covering and comparing cloud and serverless computing platforms, and evaluating optimization problems in such environmental settings. Then, the focus will be shifted to different techniques applied for solving MOO problems in cloud computing, and then use cases and applications using each technique will be presented and discussed, allowing a comparison study to be made with regards to different techniques. Finally, the tutorial will lead to a discussion on MOO problems in a serverless environment, evaluating the opportunities and limitations the serverless paradigm gives for solving them.

4 GOALS AND OBJECTIVES

The tutorial aims to familiarize the audience with the main MOO challenges the cloud offers, as well as possible techniques to tackle them. The audience will be presented with some successful works on this domain, and their advantages and disadvantages will be evaluated. Finally, the tutorial will encourage a discussion on the future of the cloud with regards to serverless computing.

5 INTENDED AUDIENCE

The tutorial targets a wide audience. People with only basic knowledge regarding the optimization challenges in a cloud landscape will receive all the necessary feedback to understand the current situation with its challenges and opportunities. More familiar members of the audience will also have the chance to participate as common techniques for solving MOO problems will be discussed. A further discussion about their advantages and disadvantages, alongside ideas about the future of the cloud and serverless will enable even experienced users and researchers to gain a good insight on the topic.

6 SHORT BIOGRAPHIES

Michail Georgoulakis Misegiannis is a graduate of the School of Electrical and Computer Engineering in the National Technical University of Athens (NTUA). His research interests include query processing, query optimization and cost modeling which were the basis of his diploma thesis. He is also interested in trends and open problems in cloud computing, as well as data management systems for modern hardware.

Verena Kantere is an Assistant Professor at the School of ECE of the NTUA. Before she was a Maître d'Enseignement et de Recherche at the Centre Universitaire d' Informatique of the University of Geneva. Previously, she was a tenure-track junior assistant professor at the Department of Electrical Engineering and Information Technology at the Cyprus University of Technology. She has received a Diploma and a Ph.D. from the NTUA and a M.Sc. from the Department of Computer Science at the University of Toronto.

Laurent d'Orazio has been a Professor at Univ Rennes, CNRS, IRISA since 2016. He received his PhD degree in computer science from Grenoble National Polytechnic Institute in 2007. He was an Associate Professor at Blaise Pascal University and LIMOS CNRS, Clermont-Ferrand from 2008 to 2016. His research interests include (big) data algorithms and architectures, distributed and parallel databases.

REFERENCES

[1] Sameer Agarwal, Aurojit Panda, Barzan Mozafari, Anand P. Iyer, Samuel Madden, and Ion Stoica. 2012. Blink and it's done: Interactive queries on very large data. *Proceedings of the VLDB Endowment* 5, 12 (2012), 1902–1905. https://doi.org/10.14778/2367502.2367533

[2] Istemi Ekin Akkus, Ruichuan Chen, Ivica Rimac, Manuel Stein Klaus Satzke, Andre Beck, Paarijaat Aditya, and Volker Hilt. 2020. SAND: Towards high-performance serverless computing. *Proceedings of the 2018 USENIX Annual Technical Conference, USENIX ATC 2018* (2020), 923–935.

[3] Fatemeh Azimzadeh and Fatemeh Biabani. 2017. Multi-objective job scheduling algorithm in cloud computing based on reliability and time. *2017 3rd International Conference on Web Research, ICWR 2017* (2017), 96–101. https://doi.org/10.1109/ICWR.2017.7959312

[4] Andreas Christoforou and Andreas S. Andreou. 2019. An effective resource management approach in a FaaS environment. *CEUR Workshop Proceedings* 2330 (2019), 2–8.

[5] Indraneel Das and J. Dennis. 1997. A Closer Look at Drawbacks of Minimizing Weighted Sums of Objectives for Pareto Set Generation in Multicriteria Optimization Problems. *Structural Optimization* 14 (01 1997), 63–69. https://doi.org/10.1007/BF01197559

[6] Michael T.M. Emmerich and André H. Deutz. 2018. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural Computing* 17, 3 (2018), 585–609. https://doi.org/10.1007/s11047-018-9685-y

[7] Gan Guo-ning, Huang Ting-lei, and Gao Shuai. 2010. Genetic simulated annealing algorithm for task scheduling based on cloud computing environment. In *ICISS2010*. https://doi.org/10.1109/ICISS.2010.5655013

[8] Florian Helff, Le Gruenwald, and Laurent D'Orazio. 2016. Weighted sum model for multi-objective query optimization for mobile-cloud database environments. *CEUR Workshop Proceedings* 1558 (2016).

[9] Florian Helff, Le Gruenwald, and Laurent d'Orazio. 2016. Weighted Sum Model for Multi-Objective Query Optimization for Mobile-Cloud Database Environments. In *EDBT/ICDT Workshops*.

[10] Florian Helff, Chenxiao Wang, Le Gruenwald, and Laurent d'Orazio. 2016. Interactive Multi-Objective Query Optimization in Mobile-Cloud Database Environments based on a Weighted Sum Model. (2016).

[11] Joseph M. Hellerstein, Jose Faleiro, Joseph E. Gonzalez, Johann Schleier-Smith, Vikram Sreekanti, Alexey Tumanov, and Chenggang Wu. 2018. Serverless Computing: One Step Forward, Two Steps Back. *arXiv* 3 (2018). arXiv:1812.03651

[12] Mohammadreza Hoseinyfarahabady, Javid Taheri, Zahir Tari, and Albert Y. Zomaya. 2017. A Dynamic Resource Controller for a Lambda Architecture. *Proceedings of the International Conference on Parallel Processing* (2017), 332–341. https://doi.org/10.1109/ICPP.2017.42

[13] Zisis Karampaglis, Anastasios Gounaris, and Yannis Manolopoulos. 2014. A Bi-objective cost model for database queries in a multi-cloud environment. *MEDES 2014 - 6th International Conference on Management of Emergent Digital EcoSystems, Proceedings* (2014), 109–116. https://doi.org/10.1145/2668260.2668271

[14] Anurag Khandelwal, Arun Kejariwal, and Karthikeyan Ramasamy. 2020. Le Taureau: Deconstructing the Serverless Landscape A Look Forward. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2020), 2641–2650. https://doi.org/10.1145/3318464.3383130

[15] Herald Kllapi, Ilia Pietri, Verena Kantere, and Yannis Ioannidis. 2020. Automated management of indexes for dataflow processing engines in IaaS clouds. *Advances in Database Technology - EDBT* 2020-March (2020), 169–180. https://doi.org/10.5441/002/edbt.2020.16

[16] Herald Kllapi, Eva Sitaridi, Manolis M. Tsangaris, and Yannis Ioannidis. 2011. Schedule optimization for data processing flows on the cloud. In *SIGMOD*. https://doi.org/10.1145/1989323.1989355

[17] Lambda 2022. *Amazon Lambda*. https://aws.amazon.com/lambda/

[18] Trung Dung Le, Verena Kantere, and Laurent D'Orazio. 2020. Dynamic Estimation and Grid Partitioning Approach for Multi-objective Optimization Problems in Medical Cloud Federations. *Lecture Notes in Computer Science* 12410 LNCS (2020), 32–66. https://doi.org/10.1007/978-3-662-62386-2_2

[19] Ingo Müller, Renato Marroquín, and Gustavo Alonso. 2019. Lambada: Interactive data analytics on cold data using serverless cloud infrastructure. *arXiv* (2019), 115–130.

[20] Guang Peng. 2019. Multi-objective optimization research and applied in cloud computing. *Proceedings - 2019 IEEE 30th International Symposium on Software Reliability Engineering Workshops, ISSREW 2019* (2019), 97–99. https://doi.org/10.1109/ISSREW.2019.00051

[21] Matthew Perron, Raul Castro Fernandez, David DeWitt, and Samuel Madden. 2019. Starling: A scalable query engine on cloud function services. *arXiv* (2019), 131–141.

[22] A.A. Priyanto, Kang Adiwijaya, and Warih Maharani. 2009. Implementation of ant colony optimization algorithm on the project resource scheduling problem. In *ICTel*.

[23] A. Sathya Sofia and P. GaneshKumar. 2018. Multi-objective Task Scheduling to Minimize Energy Consumption and Makespan of Cloud Computing Using NSGA-II. *Journal of Network and Systems Management* 26, 2 (2018), 463–485. https://doi.org/10.1007/s10922-017-9425-0

[24] Fei Song, Khaled Zaouk, Chenghao Lyu, Arnab Sinha, Qi Fan, Yanlei Diao, and Prashant Shenoy. 2021. Spark-based Cloud Data Analytics using Multi-Objective Optimization. (2021), 396–407. https://doi.org/10.1109/icde51399.2021.00041 arXiv:arXiv:2005.03314v1

[25] Immanuel Trummer and Christoph Koch. 2014. Approximation schemes for many-objective query optimization. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2014), 1299–1310. https://doi.org/10.1145/2588555.2610527 arXiv:1404.0046

[26] Immanuel Trummer and Christoph Koch. 2015. An incremental anytime algorithm for multi-objective query optimization. *Proceedings of the ACM SIGMOD International Conference on Management of Data* 2015-May (2015), 1941–1953. https://doi.org/10.1145/2723372.2746484

[27] Immanuel Trummer and Christoph Koch. 2017. Multi-objective parametric query optimization. *Commun. ACM* 60, 10 (2017), 81–89. https://doi.org/10.1145/3068612

[28] Michal Wawrzoniak and Gustavo Alonso. 2021. Boxer: Data Analytics on Network-enabled Serverless Platforms. *Cidr* (2021). https://doi.org/10.3929/ethz-b-000456492