Scoring and ranking serverless providers using multicriteria decision method

No Author Given

No Institute Given

Abstract. In the contemporary technological landscape, the process of developing and distributing software that fulfills the objectives of both creators and users while remaining economically viable and technically manageable presents significant challenges. This work examines the potential of cloud computing and the serverless paradigm as third-party infrastructure solutions to address these challenges. Specifically, it proposes a methodology for evaluating and selecting serverless platforms using a multicriteria decision-making approach. The criteria employed in this model were derived from two primary sources: serverless service providers and analyses of benchmarking reports on serverless providers. To assess the efficacy of this approach, experiments were conducted to evaluate both the accuracy and performance of the proposed solution. Additionally, a comparison was made with an existing implementation of the multicriteria method available in a software library. The results of these experiments demonstrated that both the proposed implementation and the existing library implementation of the decision-making multicriteria method achieved 100% accuracy in a controlled environment. However, the algorithm developed in this study exhibited superior performance in terms of runtime when applied to scenarios involving more than 500 serverless providers.

Keywords: Serverless; Multi-criteria decision analysis; AHP

1 Introduction

As technology advances, it becomes increasingly challenging to identify the best approach or method for developing software that meets the end user's pre-established end goals without becoming economically unviable and technically complex. The vast area of software development, including front-end, back-end, DevOps, infrastructure, among others, offers hundreds of different technologies. The difficulty in choosing the ideal technology is related, among other issues, to the various characteristics and criteria that can affect the software's performance in terms of cost, complexity, maintenance, etc.

Particularly, in the area of infrastructure, there are several technologies available, some of which are even redundant. The choice of the appropriate technology to develop and consequently to deliver to end users a software can be influenced by the availability of skilled labor, which, at times when the information technology job market is going through, can be scarce and therefore economically unviable.

The outsourcing of infrastructure services, known today as cloud computing, represents an effective strategy for tackling various computing challenges, influencing the development and distribution of software. In this cloud computing model, the dominant application development paradigm continues to be the web development model, where the application can be made available on web servers (running the Hyper Text Transfer Protocol (HTTP) and its associates) on virtualized servers. However, due to the need to control costs, new development paradigms are emerging, including the one known as serverless.

The serverless paradigm, or serverless computing, is a development paradigm for cloud computing that allows developers to create and run applications without the need to manage servers or backend infrastructure. This new development paradigm allows developers to focus exclusively on the business logic and code of the application, without the need to manage (and the cost of owning or operating) the virtualized servers in the cloud computing. In this paradigm, the execution of the computer program (often called a function) takes place on demand from the end user, which is characterized by payment for the use of the cloud provider's resources.

Thus, by opting to hire a company that offers serverless services, contracting companies (software developers and/or deployers, as well as other cloud software providers) can outsource the costs and responsibilities related to maintaining and operating computing resources (e.g. servers, network interconnection equipment, etc.), thus freeing up time and resources to focus on software development. This strategy not only saves money, but also promotes greater efficiency and quality in the end product, thanks to the internal team's ability to dedicate themselves fully to their work.

Recognizing the importance of opting for a third-party infrastructure solution and the use of the serverless paradigm, this work aims to propose a mechanism for selecting serverless platforms (providers) by means of the use of a decision-making multi-criteria method, considering different indicators related to the availability of software in this paradigm, associated with the providers in question. This analysis will help users of the serverless service (clients of the serverless providers) to compare several providers using a number of important criteria, simplifying the adoption of informed decisions on choosing the provider that best meets their specific needs.

Moreover, in order to validate the serverless providers selection approach developed, a comparison with the use of the same used decision-making multicriteria method available at a software library is performed. This comparison aims to evaluate the accuracy of the implementation performed (and consequently of the approach provided), assuming the software library is well implemented and correct.

The structure of this paper is organized as follows: Section 2 presents the theoretic referential needed to understand this paper. Section 3 highlights some of the work that has been done on the problem involving serverless providers. Next, Section 4 presents and discusses the proposal for selection through scoring and ranking, using the decision-making multicriteria technique implemented. Next, Section 5 discusses the results of experiments carried out to validate the proposal. Finally, Section 6 presents the final considerations of the work carried out.

2 Theoretical Reference

This section presents concepts and techniques considered essential to understanding the work.

2.1 The serverless paradigm

The serverless paradigm, or serverless computing, is a software development and execution paradigm that allows developers to create and run applications without the need to manage servers or backend infrastructure. Although physical and/or virtual servers are still used, they reside in the infrastructure of the cloud service provider. In this sense, the management of these resources is independent of the client of the serverless service, allowing them to focus exclusively on the business logic and code of the application [4, 14]. In this way, the cloud provider takes responsibility for provisioning, maintaining and scaling the infrastructure, offering a more efficient and cost-effective software development and deployment experience. According to [5], one of the main characteristics of serverless paradigm is scalability.

The serverless architecture guarantees fair and generous hosting costs as fine-grained resources are provisioned on demand and charges reflect only actual computing time [7]. This means that there are no charges for idle capacity, resulting in significant savings for companies. Applications on this paradigma are broadly called functions.

Currently, all the main cloud service providers offer serverless solutions, including AWS Lambda from Amazon, Azure Functions from Microsoft, Google Cloud Functions from Google and IBM Cloud Code Engine from IBM. These platforms allow developers to take advantage of the benefits of serverless computing, making it easier to create scalable and efficient cloud-native applications.

2.2 Multicriteria Decision Making

Multicriteria Decision Analysis (MCDA) is an approach for solving decision-making problems that has prominence in the field of complex decision-making. This approach is particularly useful in applications where there are several alternatives to solve the problem and it is not enough to consider just one criterion to decide which is the most suitable solution alternative, but rather multiple aspects that may even conflict with each other. The application of MCDA covers a wide range of areas, from engineering and management to the environment, public policy and health, demonstrating its versatility and practical usefulness.

The advantages of using MCDA are obvious. It allows multiple criteria to be taken into account, even qualitative or intangible ones, providing a transparent and auditable decision-making process. This facilitates communication between decision-makers and promotes coherent decision-making based on objective criteria.

In the field of MCDA, there is a wide variety of methodic approaches available, each one suitable for different types of problems. Therefore, careful selection of the most appropriate multi-criteria method for modeling and solving the problem becomes crucial. This work focuses specifically on the task of selecting, through scoring and ranking, serverless providers based on specific criteria. Based on this focus, the chosen method was the Analytic Hierarchy Process (AHP) [13].

Analytic Hierarchy Process (AHP) The Analytic Hierarchy Process (AHP), conceived by Thomas L. Saaty [13], has emerged as a methodology to help make complex decisions. This method, which stands out for its ability to break down intricate problems into manageable components through a well-defined hierarchy, offers a systematic approach to evaluating and comparing multiple criteria and problem solution alternatives. By employing pairwise comparisons, the AHP not only makes it easier to determine the relative importance of each criterion and alternative, but also calculates a weight or priority for each of them, helping to make the most appropriate choice.

As mentioned in [10], the AHP method is widely recognized for its simplicity and ease of use, which contributes significantly to its widespread use. One of its main strengths is its remarkable flexibility, allowing integration with other techniques such as Quality Function Deployment (QFD) [1] and the Strengths, Weaknesses, Opportunities, and Threats (SWOT) [11] matrix. The extensive academic research and practical application in various fields give the AHP high reliability and acceptance. A particularly valuable aspect, as cited in [8, 2, 10] is the ability to measure the internal consistency of experts' judgments on the problem being modeled, providing greater robustness to decisions. In addition, the method promotes productive interaction between analysts and decision-makers, facilitating a common understanding of the problem in question. This feature is especially useful in collaborative decision-making environments. Finally, the synthesis of results offered by the AHP allows for a clear comparison of the priorities and relative importance of each factor considered, providing valuable insights for the decision-making process.

In short, the Analytic Hierarchy Process represents a robust and structured approach to decision-making in environments characterized by the presence of multiple criteria. By providing a solid quantitative basis, facilitating comparative analysis and promoting inclusive participation, the AHP promotes more informed decisions based on hierarchical modeling between a problem's objective, available criteria and solution alternatives.

3 Related Works

Several studies in the literature address the correlation of *cloud* and serverless providers, providing valuable *insights* for selecting, scoring and ranking these services.

The work by [9] proposes a logical and mathematical scoring technique to classify and choose the most appropriate among several candidate cloud computing providers for the user. This method is based on the analysis of various criteria comprising performance indicator values required by the user and associated with each cloud computing provider that is able to meet the user's needs. Based on this study, it is possible to understand how to obtain and use performance indicators and how they are used in methods that are already known.

The research [6] examines the way in which each serverless platform (AWS Lambda, Azure Functions, and Google Cloud Functions) works with security and compliance issues. These platforms were chosen for the study because they are leaders in this field. The study aims to compare how each platform addresses important security issues. It was concluded that all platforms demonstrate high standards of security and compliance. AWS Lambda and Azure Functions have slight advantages in data residency controls and auditing capabilities.

The paper [3] evaluates the serverless computing platforms Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure, using a simulation of a network with thousands of sensors to test the platforms under stress conditions. The paper discusses and compares different data storage technologies, including relational, NoSQL, and time-series-optimized databases, which are essential for IoT applications.

4 Methodology & Development

This work is a documentary and quantitative study, involving an in-depth analysis of the information available on the official websites of the main serverless providers and in *benchmarking* reports. In the first stage, the study seeks to identify criteria/characteristics associated with the serverless service that help to model the problem of selecting the most suitable serverless service provider in view of the needs of its customers. In the second stage, once the criteria have been listed, the methodology is used to implement the proposed selection method, using the MCDM AHP. Associated with this, the work also compares the implementation of the AHP method developed, in terms of accuracy and execution time (evaluation metrics), with the implementation provided by the PyDecision library, as a mechanism for validating the implementation.

4.1 Selection Criteria

The ranking and consequent selection of serverless providers is carried out using the AHP method, which is based on a set of criteria or Performance Indicators (PIs). These indicators are fundamental as they serve as parameters for quantitative and qualitative evaluations, allowing the different alternatives to be compared and scored.

It is possible to classify PIs according to their function and usefulness [9]. In other words, how useful the PI is according to the final objective of an analysis. In this sense, the types classify IPs according to 3 possibilities: **1) HB (Higher is Better):** users and system managers prefer higher values for certain indicators. For example, system throughput, amount of resources (money, memory, material, etc.), system availability, etc, **2) LB (Lower is Better):** users and system managers prefer lower values for certain indicators. For example, response time, cost, latency, etc. **3) NB (Nominal is Best):** users and system managers prefer specific values, neither high nor low. For example, loading for system utilization, a high utilization of the system can generate a high response time, while a low utilization means that they are using the system little.

The values of the PIs are collected and stored with their classes/types since the AHP method uses them to build the judgment matrix, taking into account the value of the PI offered/available by the serverless provider and that requested by the client.

Defining the criteria to be used to select the most suitable serverless provider was done in two stages.

Firstly, the official websites of the main serverless providers (AWS Lambda, Google Cloud Functions and Microsoft Azure) were accessed directly, in order to see what characteristics they offered that could be taken as criteria for a customer's choice of provider. Likewise, the most recent values for these characteristics (PI's) were obtained from this access. The analysis was then supplemented with benchmark data for serverless providers from Rifai's [12] report, which provides a detailed assessment of various aspects relevant to choosing a serverless provider.

Thus, with the characteristics and values extracted from the official websites and the mentioned report, the following criteria were used to carry out the ranking and selection using the AHP method.

- **Computing time**: refers to the Execution time[1] (seconds) multiplied by the Resource consumption converted into GBs, the result is the free monthly amount per month that the user can use. Counted in GB/second. Type: HB.
- **1GB/additional second**: additional value for the 1GB/second that the user needs in the computing time that exceeds the free amount made available. Counted in R$. Type: LB.
- **Duration Rounding**: integer value closest to the duration of a function. Ex: function lasted 101ms, the provider's default billing value is 100ms, so the duration will be counted as having lasted 200ms. Counted in milliseconds. Type: LB.
- **Free request/month**: refers to the number of free http requests per month that the provider offers its client. Type: HB.

[1] Execution time is the result of the number of requests multiplied by the execution duration (seconds)

- **Additional request**: refers to the additional amount for every 1 million requests that exceed the number of free requests per month. Counted in R$. Type: LB.
- **Scalability**: refers to the provider's ability to automatically adjust to cope with an increase or decrease in the demand for execution of the function. Counted in binary. Type: NB
- **Concurrency**: refers to the ability to execute multiple instances of a function simultaneously. Counted in binary. Type: NB
- **Cold Start**: refers to the additional time to respond to a function access request on an inactive instance[2] Counted in seconds. Type: LB.
- **Memory**: refers to the amount of memory that the provider offers users for their application. Counted in MB. Type: HB.
- **Execution Time**: refers to the maximum time that a provider will run a function. Counted in minutes. Type: HB.

4.2 Customer parameters

The client of the serverless service provider presents its needs to the proposed method by parameterizing the criteria listed for selecting the provider, in the form of a request. Thus, the customer's request represents the formalization of their preferences and/or needs comprising the PI criteria, and consequently regarding to the choice of serverless provider. Through this, the AHP method receives the information it needs to build its internal decision matrices, and thus perform calculations and generate a score for each provider, in a way that reflects the user's hierarchy of preference between them.

Table 1 shows an example of a customer request for the proposed selection method. In it, the client indicates the desired values for each criterion/PI involved in the selection, as well as the relative weight of each criterion according to their needs.

Thus, the weights assigned to each criterion reflect its relative importance to the user's needs. These weights are expressed using the Saaty [13] scale, which assigns values from 1 to 9 to refer to the importance of a given criterion or value in relation to another, in a paired comparison. Thus, the value 1 indicates the lowest importance of the specific criterion in relation to the others, while the value 9 indicates the highest.

4.3 Set up

To develop the serverless provider selection approach, the AHP method was developed using the Python programming language. In parallel, experiments were also conducted using the implementation of the AHP method provided by the PyDecision library. The aim of this strategy was to assess the reliability of the implementation developed, assuming that the implementation of the PyDecision library is reliable and therefore a *baseline* for the experiments.

[2] Inactive instance is an instance that executed a function and was deallocated.

Table 1: Format of a request

Criteria or PI's	**Value**	**Weight**
Computing time (PI_1)	400,000 GB/s	1
Additional 1GB/second (PI_2)	0.25$	5
Rounding Duration (PI_3)	1 ms	1
Request (PI_4)	1000	9
Additional request (PI_5)	0.5$	7
Scalability (PI_6)	1	9
Competition (PI_7)	1	9
Cold Start (PI_8)	1s	9
Memory (PI_9)	512 MB	5
Runtime (PI_10)	30 min	9

In order to originally assess the accuracy of the different implementations, 5 hypothetical scenarios were created, each consisting of 1 client request and 5 serverless providers with predefined values and known rankings. This way, the ranking of the most to least suitable provider is known for each request and it is possible to evaluate scores assigned to each provider and consequently their ranking for each implementation of the AHP method. This approach allowed us to compare the accuracy of the implementations, resulting in both implementations being accurate.

Table 2 shows the set of data relating to the values of the criteria and the importance associated with the fictitious serverless providers for Scenario 1 of the experiments carried out.

Table 2: Providers with their respective values according to Scenario 1

Provider		***Criteria(PIs)***									
		PI_1	PI_2	PI_3	PI_4	PI_5	PI_6	PI_7	PI_8	PI_9	PI_{10}
P1	**Value**	400,000 GB/s	0.25$	1ms	200000	0.5$	1	1	1ms	512 MB	30 min
	Weight	1	1	1	9	1	1	1	9	1	9
P2	**Value**	400,000 GB/s	0.25$	1ms	200000	0.5$	1	1	2ms	512 MB	30 min
	Weight	1	1	1	9	1	1	1	1	1	9
P3	**Value**	400,000 GB/s	0.25$	1ms	200000	0.5$	1	1	2ms	512 MB	5 min
	Weight	1	1	1	9	1	1	1	1	1	1
P4	**Value**	400,000 GB/s	0.25$	1ms	100000	0.5$	1	1	2ms	512 MB	5 min
	Weight	1	1	1	1	1	1	1	1	1	1
P5	**Value**	400,000 GB/s	0.25$	1ms	100000	0.5$	1	1	2ms	512 MB	5 min
	Weight	1	1	1	1	1	1	1	1	1	1

5 Experiments and Results

In order to evaluate the proposed selection method, including the execution time and scalability of the different implementations of the AHP method, experiments, were developed. These experiments rely on a range of values for the criteria obtained from providers and the benchmark report from [12], as already mentioned. These values for the criteria form a database from which the limits are extracted for the automated generation of values for the criteria in the various simulated scenarios. To this end, a generator of fictitious requests and providers was developed. This generator uses the data available in the database as a basis, randomly selecting values for the criteria. With this approach, it was possible to simulate scenarios with a variable number of providers, from 5 to 1000, for each of the 100 randomly generated requests. This methodology made it possible to test the algorithms' ability to handle a large volume of data and to select, score and rank providers efficiently.

Figure 1 illustrates the increase in execution time for both implementations of the AHP method for ranking providers, as the number of providers grows.

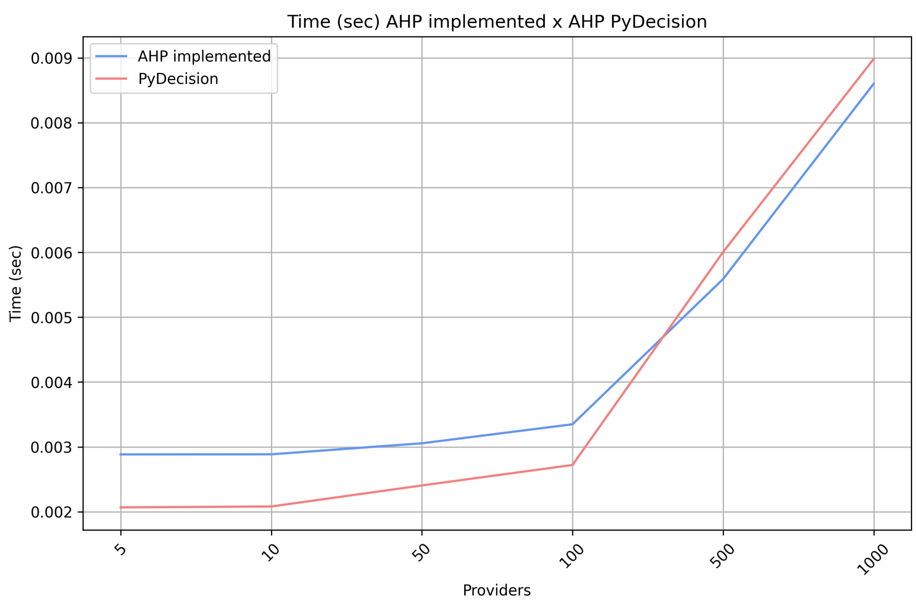

Fig. 1: Runtime x number of providers (scenarios)

The results of the scalability analysis indicate that, for data sets with more than 500 providers, the AHP algorithm implemented in this study shows superiority in terms of computing time. However, for smaller data sets (fewer providers), the PyDecision library, due to its optimizations, can perform more efficiently. This observation suggests that the choice of the ideal algorithm depends on the size of the dataset to be analyzed.

Figures 2 and 3 show the accuracy results (dispersion and variability, respectively) in scenarios with 5, 10, 50, 100, 500 and 1000 providers, whose correct classifications (ranking), comprising each one of the customer requests, were defined based on the PyDecision library ranking. Based on the data, it is possible to infer that the implemented AHP algorithm managed to obtain 100% accuracy for tests with 5 providers and between 60% and 100% accuracy for the remaining tests in the remaining scenarios with more providers.

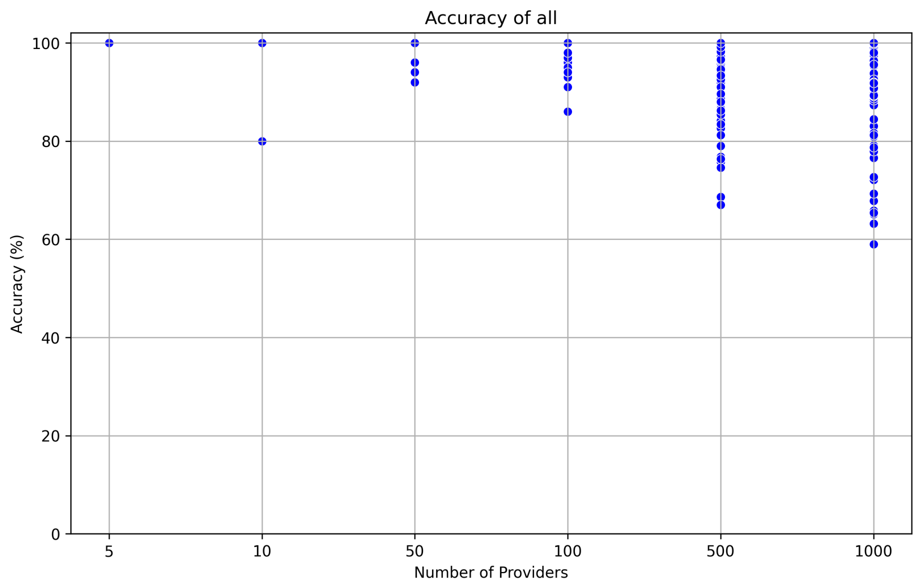

Fig. 2: Accuracy x request scatter plot for experiment scenarios

6 Final Considerations

This work proposed a method for selecting serverless providers by means of ranking, based on multi-criteria analysis for decision-making, using the AHP method. Multiple criteria were used to model the solution to the ranking selection problem. These criteria model the relationship between the offer of the serverless service and the needs of its customers, in terms of deploying software applications as a function in these environments. The method takes as input the data from the serverless providers and a client request, based on the criteria used and their importance to the requester's need.

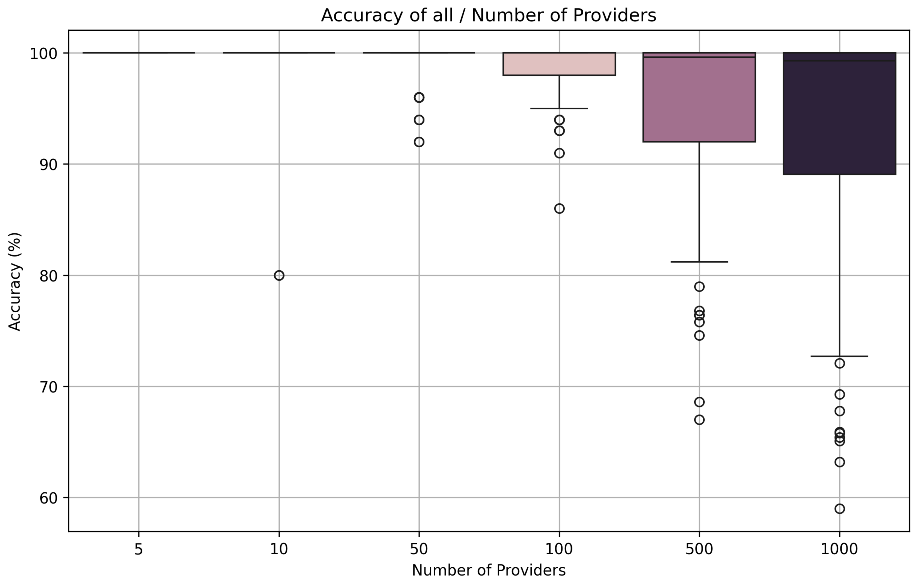

Fig. 3: Test accuracy of all x request boxplot graph

In addition to modeling the problem, this study implemented the AHP method using the Python programming language. It also makes use of an implementation of the AHP method provided by the PyDecision library, to compare results with the implementation carried out. Experiments were performed in order to measure the accuracy and performance (execution time) of the implementations.

Results of these experiments show that the AHP algorithm implemented performed better in terms of execution time when compared to the PyDecision library, especially in scenarios with 500 or more serverless providers. Both algorithms achieved 100% accuracy in classifying the best providers in a controlled environment (especially the first 3 providers). These results indicate that both methods are promising for selecting serverless providers in scenarios where the criteria are known a priori.

However, further research is needed to assess the robustness of these methods in more complex scenarios and with greater variability in the data. In this sense, future work proposes increasing the number of test cases, thereby generating even greater confidence in hypothetical scenarios with more than 5 providers, as well as making comparisons between other MCDMs to obtain greater reliability in the results obtained.

References

1. Akao, Y.: Quality Function Deployment: Integrating Customer Requirements into Product Design. Productivity Press, New York (Oct 2024)
2. Boucher, T.O., Gogus, O., Wicks, E.M.: A comparison between two multiattribute decision methodologies used in capital investment decision analysis. The Engineering Economist 42(3), 179–202 (1997)
3. Calderoni, L., Maio, D., Tullini, L.: Benchmarking cloud providers on serverless iot back-end infrastructures. IEEE Internet of Things Journal 9(16), 15255–15269 (2022)
4. Castro, P., Ishakian, V., Muthusamy, V., Slominski, A.: The rise of serverless computing. Communications of the ACM 62(12), 44–54 (2019)
5. Dey, N.S., Reddy, S.P.K., G, L.: Serverless computing: Architectural paradigms, challenges, and future directions in cloud technology. In: 2023 7th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC). pp. 406–414 (2023)
6. Khanal, D.D., Maharjan, S.: 1. comparative security and compliance analysis of serverless computing platforms: Aws lambda, azure functions, and google cloud functions. Indian Scientific Journal Of Research In Engineering And Management (2024)
7. Krishnamurthi, R., Kumar, A., Gill, S.S., Buyya, R. (eds.): Serverless Computing: Principles and Paradigms, Lecture Notes on Data Engineering and Communications Technologies, vol. 162. Springer International Publishing, Cham (2023)
8. Liberatore, M.J., Nydick, R.L.: Group decision making in higher education using the analytic hierarchy process. Research in Higher Education 38, 593–614 (1997)
9. de Moraes, L.B., Fiorese, A.: A scoring method based on criteria matching for cloud computing provider ranking and selection. In: Hammoudi, S., Śmiałek, M., Camp, O., Filipe, J. (eds.) Enterprise Information Systems. pp. 339–365. Springer International Publishing, Cham (2018)
10. Norris, G.A., Marshall, H.E.: Multiattribute decision analysis method for evaluating buildings and building systems. Tech. Rep. NIST IR 5663, National Institute of Standards and Technology, Gaithersburg, MD, USA (january 1995)
11. Puyt, R.W., Lie, F.B., Wilderom, C.P.M.: The origins of SWOT analysis. Long Range Planning 56(3) (Jun 2023)
12. Rifai, M.: Serverless showdown: Aws lambda vs azure functions vs google cloud functions. https://www.pluralsight.com/resources/blog/cloud/serverless-showdown-aws-lambda-vs-azure-functions-vs-google-cloud-functions (2023), accessed: 2024-12-08
13. Saaty, T.L.: How to make a decision: The analytic hierarchy process. European Journal of Operational Research 48, 9–26 (September 1990)
14. Wen, J., Chen, Z., Jin, X., Liu, X.: Rise of the planet of serverless computing: A systematic review. ACM Transactions on Software Engineering and Methodology 32(5), 1–61 (2023)