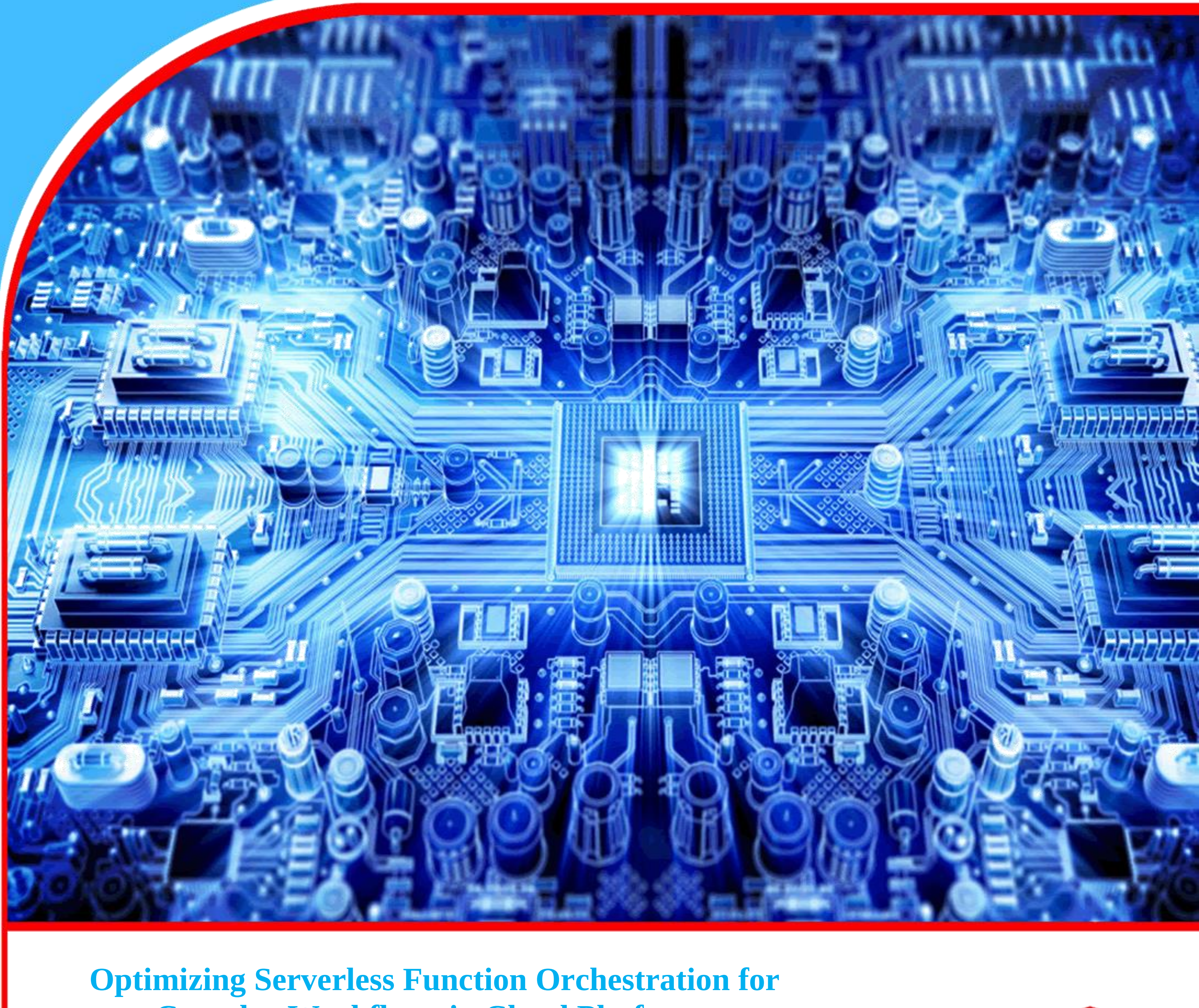
American Journal of
Computing and Engineering
(AJCE)



Optimizing Serverless Function Orchestration for Complex Workflows in Cloud Platforms

Anaswara Thekkan Rajan

American Journal of Computing and Engineering
ISSN 2790-5586 (Online)
Vol.7, Issue 5, pp 1 – 11, 2024
www.ajpojournals.org

Optimizing Serverless Function Orchestration for Complex Workflows in Cloud Platforms

Anaswara Thekkan Rajan[1*]

Article history

Submitted 10.08.2024 Revised Version Received 01.09.2024 Accepted 01.10.2024

Abstract

Purpose: This paper aims to investigate ways of increasing the efficiency of various workloads of serverless function orchestration in cloud systems while decreasing latency and optimizing resources.

Materials and Methods: The report suggests adaptive learning systems and a feedback loop framework for managing the orchestrating serverless environment.

Findings: It has also been affirmed that case studies help validate the framework rather than presenting the practical implications of the proposed techniques.

Implications to Theory, Practice and Policy: The paper recommends focusing on managing performance, dependencies, and expenses in serverless function orchestration by employing adaptive learning systems and feedback loop frameworks.

Keywords: *Serverless Computing (C61), Function Orchestration (C63), Cloud Platforms (L86), Workflow Optimization (C61), Adaptive Learning Systems (D83), Feedback Loops (D83)*

1.0 INTRODUCTION

Serverless computing is a recent innovation that has brought a paradigm shift in how cloud computing services are delivered to application developers, enabling them to execute applications while simplifying the management complexities. This paradigm, known as Function-as-a-Service (FaaS), allows for automated scaling of functions directly handled by the cloud providers [1]. The serverless approach aligns well with microservices and event-based systems, where small, stateless functions are triggered by specific events [1]. However, the stateless nature of serverless functions poses challenges such as execution time limitations, lack of synchronization, and difficulty in coordinating complex workflows. These constraints often lead to inefficient coordination, resulting in higher service costs, increased latency, and resource contention.

The increasing adoption of cloud solutions across organizations of various sizes has highlighted the need for effective workflow management principles. In traditional unmanaged cloud computing models, developers were responsible for managing servers, which involved configuring physical host machines, setting up kernels, and deploying virtual machines [2]. Serverless computing abstracts these tasks, allowing developers to focus on building applications rather than managing infrastructure. However, the shift introduces new challenges because serverless applications are built from numerous tiny, loosely coupled functions that lack a clear mechanism for synchronizing their actions towards achieving a specific goal. This paper addresses these issues by proposing a framework for optimizing serverless function orchestration in cloud platforms. The proposed framework is based on adaptive learning systems and feedback loops that aim to enhance performance, resource utilization, and cost efficiency. This approach is then validated through qualitative research and live case examples.

Gaps Addressed by the Study

This study addresses the limitations of current serverless orchestration frameworks by focusing on the core issues of statelessness, execution time constraints, and the lack of efficient synchronization mechanisms. These gaps hinder the seamless integration and management of complex, large-scale workflows in serverless environments.

Beneficiaries of the Study

The findings and proposed framework are particularly beneficial for cloud service providers, application developers, and organizations utilizing serverless computing. By implementing the suggested solutions, cloud providers can offer more robust orchestration services, developers can optimize resource usage, and organizations can achieve better cost efficiency and reduced latency in their applications

Theoretical Foundations

Serverless computing entails thinking of computing as a vertically integrated service delivery model whereby one has to hide and address several layers of issues away from the developers who only concentrate on actual logic and code running [3]. The serverless model is based on the execution of functions on specific events, for example, HTTP requests, changes in the database, or in the timeframe [4]. However, it entails the following gains: The ability of this model to scale automatically, the ability to be billed based on use, and lower operational overheads.

The serverless model differs from the IaaS and PaaS hosting models, where developers are forced to administer or, at the least, provision the lowest amount of infrastructure assets, such as storage and network [5]. In serverless architectures, all these tasks are delegated to the cloud provider, which provides and manages resources. This abstraction helps reduce the market time for an application to a greater extent, as developers and system architects do not need to bother about infrastructural constraints while deploying or scaling their applications.

However, as serverless functions are stateless, there is no internal state maintained between the two consecutive executions, which makes it a little more challenging to handle the stateful operations in the workflow [6]. Stateful workflows are found in complex business applications requiring user session identifiers, tracking massive processes, or saving partial results between function calls. The short lifetime of serverless functions, along with their execution time restrictions, makes the management of such processes challenging [6]. Therefore, developers have no choice but to create their applications with such constraints in mind. So, they utilize external state services such as databases or object storage.

Challenges in Orchestrating Serverless Functions

Coordinating activities in serverless functions requires handling several things, such as dependencies, ordering tasks, and managing the state [7]. This is an obvious concern as it directly addresses the problem of guaranteeing that functions are executed in the correct order with the right input. The main problem with mechanistic structures is that they have many interconnecting tasks in a workflow, and these dependencies have to be appropriately managed to ensure accuracy. A notable example is the outage experienced by Disney+ during its launch in 2019. Inefficient serverless orchestration led to excessive function calls and resource contention, which caused significant performance issues, resulting in high latency and service disruptions for users [8].

The other major problem is the issue of efficiency in the use of resources and fighting costs. As pointed out earlier, the serverless model is already cost-optimized given the usage-based billing model, but inefficient coordination results in more function calls, unbounded executions, and wasted resources [8]. This causes inefficiency that raises costs and impairs performance, particularly where a high frequency of function calls in the workflow is involved [8]. Essentially, it shows that the capability integration of third-party services to the serverless workflow brings sophistication. Serverless functions traditionally depend on external APIs, databases, or another computing service [9]. These dependencies must be well coordinated so that they become logical and do not negatively impact the efficiency of the functions, especially when external services fail. This means one must incorporate harnesses for failure on external services such as circuit breakers, fallback mechanisms, and timeouts.

Case Management and Adaptive Systems

Implementing case management principles into serverless orchestration allows for finer-grained control of the adaptive execution [10]. A more dynamic and flexible manner is adopted to orchestrate individual cases, and case management is used for tracking and managing cases. Thus, it is possible to design a coherent workflow between the case management and adaptive learning systems. Workflow performance within adaptive systems is controlled, and feedback loops are used to make real-time corrective adjustments. These systems can thus alter the temporal parameters of execution sequences and the distribution of resources, as well as define strategies for managing errors derived from the amounts of performance observed [11].

This adaptive approach ensures that the three levels' orchestration is consistent and does not deteriorate due to changes in the environment [11]. Additionally, using cost adaptivity, the systems can also enhance costs by considering the efficiency of functionality and resource consumption. For example, the system might conclude that it is cheaper to schedule a specific function to the higher tier because although the cost of invoking it is higher, the number of invocations and the overall running time of the function will be lower due to the increased performance.

Feedback Loops in Cloud Systems

There must always be feedback loops when working on complex systems, including serverless function orchestration [12]. It is similar to the standard feedback loop practiced in cloud environments, where the execution times, resource usage, or error rates are gathered and used to make real-time changes. Using feedback loops in serverless cases enables the orchestration engine to gather data on past executions and improve the strategies to take [12]. For instance, if a given function has a high latency value, the system can change its order of operation or allocate more power to the function. This continuous improvement process is essential in ensuring that some complex workflow processes are efficient and reliable. All the feedback loops mentioned above can be applied to serverless architecture at various tiers, from high to low.

The micro functions on an individual level can, for example, gather and report on the performance of a specific function and how it is executed in the enterprise's actual time. The entire workflow process may be supervised at the macro level, and emerging patterns can be used to determine moments that indicate that the system may be slowing down. Thus, it becomes possible to implement a set of feedback loops that will produce an integrated picture of the work at both levels and bring more detailed insights into the performance question.

Real-Time Memory Tuning

For example, if a feedback loop reports that a certain AWS Lambda function is slower than usual because it doesn't have adequate memory allocation, then the system may alter the next invocation of such a function's memory setting. Consider an image processing function that may be needed in order to resize images for a web application. If the function is run with minimal memory, it may lead to slower processing times or even timeouts in periods of heavy usage.

It can analyze execution metrics, for example, how long it takes, on average, to process an image or memory consumption patterns. If it finds that the function has consistent approaches to memory limits and high execution times, then it may recommend increasing the allocated memory, in turn reducing execution time for later requests. It not only enhances the user experience as the processing speed of images increases but also optimizes cost because AWS Lambda derives its pricing model based on allocated memory and execution duration.

Elimination of Redundant Functions

It might be a very rarely called Google Cloud Function, and the feedback loop may identify this behavior and recommend either to remove the function or to merge its functionality with more frequently used functions. Thus, a cloud function intended for notification to users is rarely triggered a few times per month. A feedback loop can analyze invocation history which might suggest that it costs without adding significant value.

From this, it may suggest to change the functionality for it to be more useful by merging it with another functionality that deals with the user interface or even remove it so not to use resources, which appear to be consuming unutilized. Being proactive only ensures keeping the significant functionalities in the cloud environment while still saving on operational costs.

Enhancing Workflow Productivity

Feedback loops allow the orchestration engine to optimize entire workflows based on dependencies and interactions among several serverless functions. A real-life example can be given in terms of an AWS Lambda-based multi-step pipeline of data processing, where if a feedback loop detects a certain function that aggregates data is consistently causing its downstream functions to run behind schedule, then that particular feedback loop will immediately trigger the corresponding adjustments.

Suppose that the aggregation function interacts with several microservices to fetch data and, for some reason, takes longer than expected to complete its task. The feedback loop will track the execution times of all the functions in the workflow; when it identifies this as a bottleneck, it will automatically suggest either the scaling of such a function or the configuration of a caching layer that temporarily holds the results of sub-operations. This should reduce the number of calls to the aggregation function, improving overall performance for the whole pipeline.

Dynamic Scaling and Load Balancing

Another important application for feedback loops is dynamic scaling and load balancing for serverless functions. Think of an AWS Lambda processing transactions in the context of peak events such as online sales or promotions. Here, the feedback loop continuously monitors the performance metrics of the function, with invocation rates and error responses being part of them.

If the feedback loop identifies that the function is approaching its concurrency limits or increasingly error-prone, it can trigger the orchestration engine to scale the function automatically, adding more instances to handle the increased load. Such dynamic scaling capability ensures the smooth processing of user transactions without any delays, considerably increasing customer experiences.

2.0 MATERIALS AND METHODS

This work uses an exploratory qualitative research design to investigate the improvement of serverless function orchestration. The research method is based on case studies, professional interviews, and the study of workflow performance data. The research process follows these steps:

Defining the Research Problem

The paper starts by stating the problems related to serverless function orchestration and the necessity to improve the practices in the case of complicated processes.

Selection of Cases

Some of the case studies used by researchers on serverless computing in cloud platforms. These are chosen to cover a variety of applications, such as data processing pipelines, microservices, and IoT.

Data Collection

Information is accumulated with the help of conversations with cloud architects and developers, evaluation of the performance of the cloud work, and documentation.

Data Analysis

Quantitative data is analyzed to uncover issues and opportunities for increasing the efficiency and effectiveness of adaptive systems and feedback mechanisms in improving performance.

Conclusion

The data gleaned from the study are then integrated to make conclusions regarding the practical strategies for enhancing serverless function orchestration. This is why the chosen approach was based on qualitative research, which is considered more appropriate for studying the phenomenon when expressing it in measurable parameters takes work. Involving case studies in the research helped establish practice-oriented serverless orchestration solutions and the issues that companies in this field face in different sectors. This approach also enabled the researchers to identify other practices and efficient strategies that other systems could adopt.

The most helpful information in the given case was received during the interviews with industry representatives, as they provided historical rather than theoretical background and possible insights concerning the difficulties of serverless orchestration. These cloud architects, developers, and IT managers understand serverless best because they have implemented the concept in their organizations. This helped to give credibility to the author's identified conclusions from the case studies and further suggestions for improving the organization of serverless working.

Detailed Justification

Qualitative research is suited best for exploring the intricate, multifaceted aspects of serverless orchestration, especially with regard to things not measurable, such as adaptive system integration and developer experiences. This type of study will draw out the following advantages:

Depth Understanding of Complex Phenomena

The serverless orchestration is, indeed, rife with a multitude of complex interactions among a wide array of components and stakeholders-whether it's developers, architects, or even the cloud service providers themselves. Qualitative research can penetrate the complexities and depth with subjective experiences, challenges, and perspectives of individuals engaged with serverless computing. While quantitative approaches often serve the purpose of generalization across a huge population, qualitative research provides rich, detailed, and minute insights into the subtleties of developers' experiences and the complexities of adaptive system integration.

Contextual Insights

Serverless architectures may be quite different for different organizations and applications. By the very nature of qualitative studies in particular, case studies and interviews are used to capture the contextual factors guiding the orchestration of serverless applications in given environments. It would mean studying specific instances of the orchestration of serverless functions-ways of processing data pipelines and IoT applications-to understand how contextual elements outline these orchestration processes with regard to organisational culture, team dynamics, and industry-specific challenges. This level of knowledge is crucial in formulating customized solutions that satisfy the different needs of different organizations.

Exploring Developer Experiences

Developer experiences are integral to effective serverless orchestration. Qualitative research enables researchers to explore the actual experiences of developers and cloud architects and better understand views, pain points, and workflows that prevail when adopting a serverless approach. To grasp the concrete consequences of serverless orchestration in the real world, especially critical pain points that perhaps cannot be inferred only by numerical metrics, interviews and focus groups will be conducted. Such insights are important to guide the design of more effective and user-friendly orchestration strategies.

Identification of Adaptive Strategies

Serverless function orchestration requires adaptability to the changing workloads and evolving business requirements. Qualitative research is best suited for establishing and analyzing the adaptive strategies organizations put to practice in the process of enhancing their architectures, which are serverless. By access and interaction with industry professionals, the related innovative practices and strategies used will be ascertained through researching successful serverless orchestration. This applies to the fluidity within the world of cloud computing; hence, an organization must be continually fit and prepared.

Theory and Practice: Bridging

Qualitative research builds the bridge between theoretical frameworks and their practical applications. Using case studies, one is in a position to validate theory-based concepts on serverless orchestration against reality, hence ensuring that findings are practically grounded. Insights from industry representatives through interviews provided historical context and the nature of practical challenges organizations were facing. It is this interface of theory and practice that lends credibility to the research findings and provides practical recommendations for practitioners.

Flexibility and Iterative Learning

Qualitative research techniques are intrinsically flexible. Researchers could adapt their approaches based on new insights that may emerge during the data collection process. This iterative learning process is particularly helpful in a field of knowledge like serverless computing, wherein new technologies and practices continually take shape. Researchers, while being flexible, could explore emerging trends, sharpen their research questions, and unlock insights that were not anticipated at the outset of the study.

Conclusion

Serverless function orchestration can be justified to be explored through qualitative research methods. Qualitative research forms a logical choice to probe issues of the phenomenon that are not quantifiable or whose complexity cannot be solved using numbers. Through case studies, professional interviews, and performance data analysis, qualitative research provides a comprehensive understanding of the challenges and opportunities inherent in serverless orchestration. This approach, therefore, enriches the academic discourse on serverless computing and gives proper insight with pragmatic strategies to organizations aiming to optimize their serverless architecture.

3.0 FINDINGS

Organisation and Areas of Concern

The organizations of interest in this study focused on the financial and healthcare sectors. These organizations are Beth Israel Deaconess Medical Center (BIDMC) in Boston and Capital One. Each organization has adopted serverless computing as a part of its cloud environment, primarily by utilizing the AWS Lambda and Google Cloud functions. The critical areas of concern identified across these organizations include:

1) Performance Bottlenecks: Several organizations are experiencing difficulties in terms of performance through serverless, and many have pointed out problems with dependencies and executing functions in particular [13].
2) Cost Management: Serverless is a pay-per-use model that, by definition, is inexpensive, but in practice, companies often encounter the problem of rapidly rising costs associated with poor coordination and resource allocation [14].
3) Error Handling and Recovery: Managements also needed help in establishing mechanisms to control errors; some executions failed, and some workflow was only half accomplished (15).

The Studies' Purview

The study aimed to validate the use of integrated forms of adaptive systems and feedback loops to orchestrate serverless functions. Two case studies were conducted to explore these issues in detail:

Case 1: Financial Services Workflow Optimization

The first case was about a financial services organization that employed serverless computing to manage transaction processing patterns [16]. The company encountered difficulties in managing several related activities. The practice of adaptive learning in this workflow helped the company find the most effective ways of function execution order and resource usage. Additionally, the feedback loops ensured that the performance was constantly checked, and improvements on the orchestration strategy that would help to minimize the latency and maximize the throughput were affected.

The adaptive learning system employed machine learning to determine the peak transaction period to increase resource use. This preventive action slowed the latency, and the company began to process the transactions with higher dependability. Moreover, every transaction carried out by the system provided a feedback loop that allowed for the observation of transaction errors that occurred in real time, and such errors could be retried or rerouted to an appropriate function to continue the workflow despite unfavorable circumstances.

Case 2: Healthcare Data Processing Pipeline

The second case study was based on a healthcare organization that adopted serverless computing for the data processing of many patients [17]. In the case of the organization, there was a concern with getting data from different sources, cleaning, and loading it to a central repository [17]. The main issue observed was the coordination of data extraction, transformation, and loading functions. The organization utilized a case management system wherein every data processing has a unique instance identifier and an adaptive system to determine and allocate the best sequences and resources. Workflow feedback was utilized to assess performance and fine-tune the processing in real time, thus enhancing the speed and accuracy of data processing.

One of the most successful aspects of the adaptive system within this context was the latter's ability to adapt to and manage fluctuating data loads, which have been known to choke conventional systems. Based on real-time feedback, the flow of data processing tasks was optimized dynamically to further reduce the overall processing time. Moreover, through the case management approach, it was possible to monitor data instances at a much finer level and make better and more targeted changes regarding errors.

Discussion

Comparison with Existing Literature and Alternative Solutions

Optimizations in serverless orchestration based on different frameworks and methodologies have recently been attempted by quite a few. Current solutions often focus on static resource allocation and predetermined workflow, which can lead to inefficient workloads, particularly if the working environment is dynamic due to changing workloads. For instance, in traditional serverless architectures, performance bottlenecks and increased operational costs have been reported by many in literature resulting from failure to dynamically respond to changeful demands.

Contrarily, case studies developed in this research project reflect the employment of adaptive learning systems and the mechanism of feedback loops toward a more dynamic orchestrating technique for serverless applications. The approach offers some distinctive advantages over the prevailing solution, such as:

Optimization of the Financial Services Workflow

In previous research projects, it was observed that fixed resource allocation is recommended and leads to excess charges at peak times and underutilization at low-traffic times. However, our case study demonstrates how adaptive systems effectively identified the latency problems through dynamic resequencing and dynamic resource allocation. This leads to a tremendous reduction in total latency. The results reflect a more responsive orchestration mechanism than static models, where the method adapts itself dynamically by predicting the behavior of historical as well as expected traffic coming into the system.

Healthcare Data Processing Pipeline Current literature has focused on increasing the accuracy and error correction of health care data processing by providing more human oversight. Our results demonstrate how adaptive systems with feedback mechanisms not only minimize the processing time but also improve error handling and recovery. The feed forward nature of the identified feedback loops provides early data quality issues and preventative errors from occurring systematically-a common oversight in traditional methods.

Contributions to the Field

This work contributes to the field since it shows that even simple feedback loop integration into serverless orchestration frameworks may greatly improve performance as well as cost efficiency. Unlike most solutions, ours focuses on aspects of orchestration as a step forward for providing support through an adaptive holistic view concerning workflow management, resource allocation, and error handling. This makes our study a worthwhile reference point for academia and industry, positioning itself to foster the adoption of adaptive methodologies in serverless architectures.

Conclusion

Hence, putting our work into its context with the literature sets the spotlight on the novelty of our contributions and spurs thoughts towards realising these benefits in practical use of organisations in the service sector, like finance and healthcare, especially in ever-changing environments. The kind of adaptive systems and feedback mechanisms proposed here make them a very strong alternative to traditional orchestration methods in serverless strategies for building highly responsive, and thus efficient, computing solutions.

4.0 CONCLUSION AND RECOMMENDATIONS

This paper has evaluated the threats and possibilities of serverless function orchestration in cloud platforms. More so, applying adaptation within learning systems and feedback mechanisms results in higher performance and even optimization of serverless processes within organizations. Some of the benefits of such techniques, as demonstrated in the case studies of this paper, are low latency, optimal resource utilization, and low cost. The findings indicate that the approach to serverless orchestration should be reactive and adaptive due to the variability of the executing workflow instance. This should be adopted as serverless computing is expected to advance since it will help order the cloud platforms.

REFERENCES

[1] Lannurien, V., D'orazio, L., Barais, O., & Boukhobza, J. (2023). Serverless Cloud Computing: State of the Art and Challenges. Serverless Computing: Principles and Paradigms, 275-316. https://ensta-bretagne.hal.science/hal-04114984/document

[2] Al Qassem, L. M. (2022). Microservice architecture and efficiency model for cloud computing services.

[3] Arjona, A., López, P. G., Sampé, J., Slominski, A., & Villard, L. (2021). Triggerflow: Trigger-based orchestration of serverless workflows. Future Generation Computer Systems, 124, 215-229. https://www.sciencedirect.com/science/article/pii/S0167739X21001989

[4] Mahmoudi, N. (2022). Performance Modelling and Optimization of Serverless Computing Platforms.

[5] Mampage, A., Karunasekera, S., & Buyya, R. (2022). A holistic view on resource management in serverless computing environments: Taxonomy and future directions. ACM Computing Surveys (CSUR), 54(11s), 1-36.

[6] Burckhardt, S., Chandramouli, B., Gillum, C., Justo, D., Kallas, K., McMahon, C., ... & Zhu, X. (2022). Netherite: Efficient execution of serverless workflows. Proceedings of the VLDB Endowment, 15(8), 1591-1604. https://www.microsoft.com/en-us/research/uploads/prod/2022/07/p1591-burckhardt.pdf

[7] Baresi, L., Hu, D. Y. X., Quattrocchi, G., & Terracciano, L. (2024). NEPTUNE: a comprehensive framework for managing serverless functions at the edge. ACM Transactions on Autonomous and Adaptive Systems, 19(1), 1-32. https://dl.acm.org/doi/pdf/10.1145/3634750

[8] Deng, S., Zhao, H., Huang, B., Zhang, C., Chen, F., Deng, Y., ... & Zomaya, A. Y. (2024). Cloud-native computing: A survey from the perspective of services. Proceedings of the IEEE.https://arxiv.org/pdf/2306.14402

[9] Ivan, C., Vasile, R., & Dadarlat, V. (2019). Serverless computing: An investigation of deployment environments for web apis. Computers, 8(2), 50. https://www.mdpi.com/2073-431X/8/2/50

[10] Sabbioni, A. (2023). Serverless middlewares to integrate heterogeneous and distributed services in cloud continuum environments.https://amsdottorato.unibo.it/10893/1/Serverless%20Middlewares%20to%20Integrate.pdf

[11] Kumari, A., & Sahoo, B. (2022). Serverless architecture for healthcare management systems. In Handbook of research on mathematical modeling for smart healthcare systems (pp. 203-227). IGI Global. https://www.researchgate.net/profile/Anisha-Kumari-3/publication/361432646_Serverless_Architecture_for_Healthcare_Management_Systems/links/62eca645505511283e8f9a9f/Serverless-Architecture-for-Healthcare-Management-Systems.pdf

[12] Arjona, A., López, P. G., Sampé, J., Slominski, A., & Villard, L. (2021). Triggerflow: Trigger-based orchestration of serverless workflows. Future Generation Computer Systems, 124, 215-229. https://www.sciencedirect.com/science/article/pii/S0167739X21001989

[13] Taibi, D., El Ioini, N., Pahl, C., & Niederkofler, J. R. S. (2020, May). Serverless cloud computing (function-as-a-service) patterns: A multivocal literature review. In Proceedings of the 10th International Conference on Cloud Computing and Services Science (CLOSER'20). https://www.researchgate.net/profile/Davide-Taibi/publication/340121613_Patterns_for_Serverless_Functions_Function-as-a-Service_A_Multivocal_Literature_Review/links/5e79f9fb92851c3091392bd4/Patterns-for-Serverless-Functions-Function-as-a-Service-A-Multivocal-Literature-Review.pdf

[14] D'Amore, A. (2020). Implementation of a serverless application (Doctoral dissertation, Politecnico di Torino). https://webthesis.biblio.polito.it/secure/15911/1/tesi.pdf

[15] Datta, P., Kumar, P., Morris, T., Grace, M., Rahmati, A., & Bates, A. (2020, April). Valve: Securing function workflows on serverless computing platforms. In Proceedings of The Web Conference 2020 (pp. 939-950). https://par.nsf.gov/servlets/purl/10146532

[16] Christine ET AL (2024, February 19). Virtual Visit with Spatial Computing on AWS | Amazon Web Services. Amazon Web Services. https://aws.amazon.com/blogs/industries/virtual-visit-with-spatial-computing-on-aws/

[17] Mao, G. (2023). AWS Lambda & Serverless Excellence | Capital One. Capital One. https://www.capitalone.com/tech/cloud/aws-lamba-serverless-architecture/