iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** IV **Month of publication:** April 2024

DOI: https://doi.org/10.22214/ijraset.2024.60566

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue IV Apr 2024- Available at www.ijraset.com

Achieving Agility through Auto-Scaling: Strategies for Dynamic Resource Allocation in Cloud Computing

Preetham Vemasani[1], Sai Mahesh Vuppalapati[2], Suraj Modi[3], Sivakumar Ponnusamy[4]
[1, 3]*Uber Technologies Inc, USA*
[2]*Tubi Inc, USA*
[4]*Independent Researcher, USA*

Abstract: ***Auto-scaling is a crucial aspect of cloud computing, allowing for the efficient allocation of computational resources in response to immediate demand. This article delves into the concept of auto-scaling, its key components, and the strategies used to effectively manage resources in cloud environments. This study emphasizes the importance of auto-scaling in the cloud computing landscape by exploring its benefits, including cost efficiency, performance optimization, high availability, and scalability [1]. The article explores the various factors to consider when implementing scaling policies, such as selecting the right approach for scaling, whether it be predictive or reactive and the availability of auto-scaling services provided by major cloud platforms like Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure [2, 3]. In addition, the paper addresses the challenges and complexities related to configuring auto-scaling systems, cost management, and latency in resource provisioning [4]. The article also showcases case studies that illustrate the successful implementation of auto-scaling in different industries, along with valuable insights and recommended approaches [5]. Lastly, this paper delves into future trends and research directions in auto-scaling techniques, integration with emerging technologies, and potential research areas [6].***
Keywords: Auto-scaling, Cloud computing, Resource management, Scalability, Performance optimization

I. INTRODUCTION

Cloud computing has completely transformed the deployment and management of applications, providing unmatched flexibility, scalability, and cost-effectiveness. With the growing trend of businesses moving their workloads to the cloud, it has become essential to carefully manage resource allocation for optimal performance.

Auto-scaling, a crucial concept in cloud computing, tackles this requirement by dynamically adjusting the computational resources according to the application's real-time demand [7]. Cloud computing offers organizations the flexibility to adjust their resources based on workload changes, optimizing performance, and reducing costs [8]. Auto-scaling enhances elasticity by automating resource allocation, eliminating the need for manual intervention, and enabling applications to effortlessly handle fluctuating levels of traffic [9]. The importance of auto-scaling in the cloud computing landscape cannot be overemphasized enough. Through the dynamic provisioning of resources, auto-scaling enables organizations to achieve cost efficiency by only paying for the resources they consume [10]. In addition, auto-scaling guarantees that applications maintain peak performance even during unexpected increases in demand, improving the user experience and minimizing the chance of downtime [11].

This article provides a thorough examination of auto-scaling in cloud computing, covering its fundamental principles, advantages, implementation approaches, and obstacles. Through an analysis of real-world case studies and industry best practices, we will explore the practical aspects of utilizing auto-scaling to maximize resource utilization in cloud environments. In addition, future trends and research directions in auto-scaling will be discussed, emphasizing the potential for integration with emerging technologies and the opportunities for further advancements in this field [12].

II. OVERVIEW OF AUTO-SCALING

Definition and Concept

Auto-scaling is a cloud computing technique that dynamically adjusts the number of computational resources allocated to an application based on its real-time performance requirements [13]. The main objective of auto-scaling is to guarantee that applications have adequate resources to handle fluctuating workloads while maximizing cost efficiency. Through the dynamic adjustment of resources based on demand fluctuations, auto-scaling allows applications to consistently achieve optimal performance and availability.

Key Components of Auto-scaling Systems

Auto-scaling systems usually involve multiple components that collaborate to monitor application performance, initiate scaling actions, and modify resource allocation accordingly.

A. Monitoring

Monitoring plays a vital role in auto-scaling systems, as it continuously tracks different performance metrics of the application, including CPU utilization, memory usage, network bandwidth, and request latency [14]. Auto-scaling systems depend on monitoring data to determine when to initiate scaling actions based on predefined thresholds. Some commonly used monitoring tools are Amazon CloudWatch, Google Stackdriver, and Prometheus [15].

B. Triggers and Policies

Triggers and policies establish the guidelines and conditions for initiating auto-scaling actions [16]. The monitored metrics serve as triggers and set the thresholds that, when exceeded, cause the auto-scaling system to take action [17]. On the other hand, policies dictate the specific actions that need to be taken to scale, such as adding or removing instances, and the intervals of time between scaling events.

C. Scaling Actions

Auto-scaling systems utilize two primary types of scaling actions: horizontal scaling and vertical scaling [18].

1) Horizontal Scaling (Scaling Out/In)

Horizontal scaling, also referred to as scaling out or in, entails the addition or removal of application instances to accommodate fluctuations in workload. When demand increases, the auto-scaling system deploys extra instances to evenly distribute the load, ensuring optimal performance. On the other hand, when the demand decreases, the system terminates unnecessary instances to reduce costs [19]. Horizontal scaling is highly efficient for stateless applications that can effortlessly distribute incoming requests across multiple instances.

2) Vertical Scaling (Scaling Up/Down)

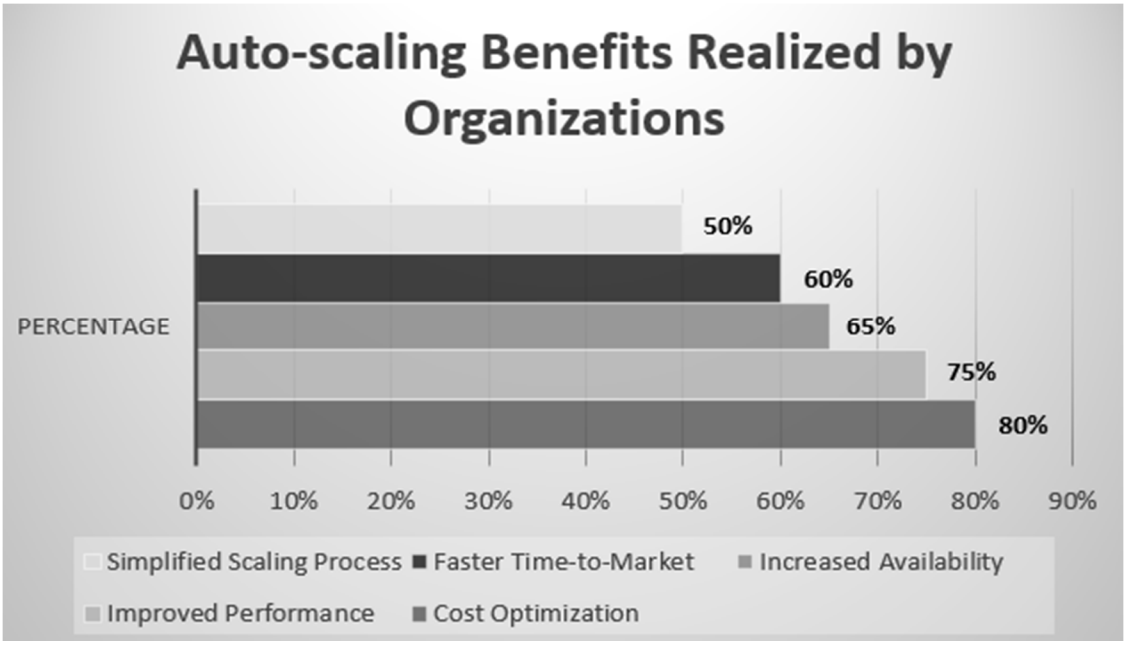
Vertical scaling, also known as scaling up or down, involves adjusting the resources allocated to individual instances, such as CPU, memory, or storage [20]. Vertical scaling adjusts the capacity of existing instances to effectively handle changes in workload [21]. Vertical scaling is a good option for applications that need consistent instance configurations and have licensing constraints [22].

3) Load Balancing

Load balancing plays a crucial role in auto-scaling systems, ensuring that incoming traffic is evenly distributed among multiple instances of an application [23]. Load balancing plays a crucial role in maintaining optimal performance and availability by distributing requests evenly across multiple instances [24]. Auto-scaling systems typically operate alongside load balancers, like Amazon Elastic Load Balancer or Google Cloud Load Balancer, to efficiently direct traffic [25].

D. Benefits of Auto-scaling

The following graph represents benefits of auto-scaling



Graph 1: Benefits of Auto-scaling: Percentage of Organizations Realizing Specific Advantages

Auto-scaling provides a multitude of benefits for organizations that deploy applications in cloud environments. Through the dynamic adjustment of resources in response to demand, auto-scaling guarantees that applications are capable of managing fluctuating workloads, all while maximizing cost efficiency, performance, availability, and scalability.

Cost Efficiency

Here is a table that provides a clear cost comparison for auto-scaling:

	5	10
	t3.medium	t3.medium
	$0.0416	$0.0416
	120	240
	$4.99	$9.98
	$149.76	$299.52

Table 1: Auto-scaling Cost Comparison

One of the main advantages of auto-scaling is its cost optimization feature, which automatically adjusts resource provisioning based on the application's needs. Auto-scaling allows organizations to only pay for the resources they consume, avoiding the need to overprovision and maintain idle resources when demand is low. By scaling down resources during periods of low workload, auto-scaling helps to minimize waste and decrease operational costs [26]. The pay-per-use model allows organizations to better manage their IT expenses by aligning them with their actual resource consumption. This can result in substantial cost savings.

Performance Optimization

Auto-scaling is essential for optimizing application performance, as it ensures that there are enough resources to handle different levels of workload. Through the monitoring of key performance metrics and the automatic adjustment of resources based on predefined thresholds, auto-scaling ensures optimal performance is maintained, even in the face of unexpected traffic increases. By implementing dynamic resource allocation, applications can swiftly respond to user requests, resulting in reduced latency and an enhanced user experience. Auto-scaling ensures optimal performance by evenly distributing the workload across multiple instances, preventing bottlenecks and resource contention.

High Availability

Auto-scaling improves application availability by automatically replacing failed instances and maintaining a desired number of healthy instances [27]. When an instance becomes unresponsive or fails, auto-scaling detects the failure and launches a new instance to replace it, ensuring continuous accessibility for users [28]. By distributing the workload across multiple instances and availability zones, auto-scaling minimizes the impact of individual instance failures on the overall application availability [29]. The ability of auto-scaling to heal itself ensures minimal downtime and improves application resilience in the event of hardware or software failures.

Scalability

Auto-scaling allows applications to effortlessly adjust their capacity according to the workload, ensuring excellent scalability. With the rise in demand, auto-scaling effortlessly allocates extra resources to handle the growing traffic, guaranteeing optimal performance even under heavy loads. Similarly, when demand decreases, auto-scaling removes unnecessary resources, allowing the application to scale down and avoid over-provisioning. This elastic scalability enables applications to adapt to changing workloads dynamically, making it easier for organizations to handle unpredictable traffic patterns and maintain optimal performance [30].

E. Implementing Auto-scaling Strategies

Implementing efficient auto-scaling strategies requires careful consideration of scaling policies and utilizing the auto-scaling services offered by leading cloud platforms.

Determining Scaling Policies

Scaling policies establish the guidelines and criteria that prompt auto-scaling actions. It is essential to carefully consider the appropriate scaling policies to achieve the best possible performance and cost efficiency.

F. Selecting Appropriate Metrics and Thresholds

Here is a table that provides a clear understanding of auto-scaling metrics and thresholds [32]:

	> 80% for 5 minutes	< 30% for 15 minutes
	> 75% for 10 minutes	< 40% for 20 minutes
	> 1000 RPS for 3 minutes	< 200 RPS for 10 minutes

Table 2: Auto-scaling Metrics and Thresholds

Choosing the right metrics and thresholds is crucial for developing impactful scaling policies. Common metrics commonly used for auto-scaling include CPU utilization, memory usage, network bandwidth, and request latency [31]. Thresholds establish the boundaries for these metrics, prompting scaling actions when exceeded [32]. Choosing metrics that accurately reflect the application's performance and setting thresholds that balance performance and cost is crucial.

G. Predictive vs. Reactive Scaling Approaches

Auto-scaling strategies can be broadly classified into predictive and reactive approaches [33]. Anticipating future workload patterns and proactively scaling resources, predictive scaling leverages historical data and machine learning algorithms. This method effectively prevents any potential performance issues by proactively allocating resources in advance. Reactive scaling, on the other hand, relies on real-time monitoring data to trigger scaling actions based on predefined thresholds. Reactive scaling is easier to put into action but could lead to a small delay in resource provisioning. The decision between predictive and reactive scaling relies on the specific needs of the application and the presence of past data [34].

Auto-scaling Services in Major Cloud Platforms

Cloud platforms provide convenient auto-scaling services that streamline the implementation and management of auto-scaling strategies.

H. Amazon Web Services (AWS)

Amazon Web Services offers Amazon EC2 Auto Scaling, a convenient service that automatically adjusts the number of EC2 instances according to predefined scaling policies. AWS Auto Scaling is capable of supporting both reactive and predictive scaling approaches. This solution seamlessly integrates with Amazon CloudWatch for monitoring and provides users with the flexibility to define scaling policies based on a wide range of metrics [35]. Additionally, AWS provides Target Tracking Scaling, an automated feature that adjusts resources to uphold a specific metric at a desired value [36].

I. Google Cloud Platform (GCP)

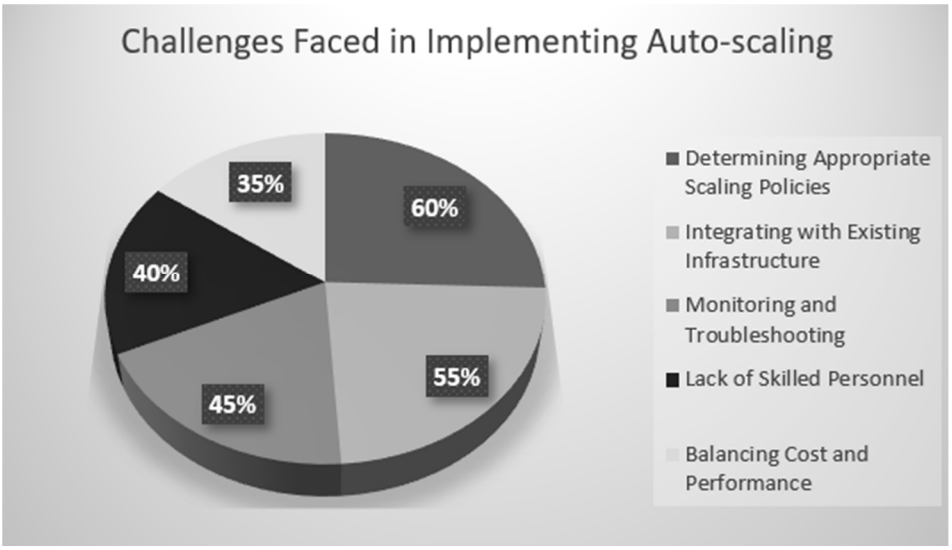
Google Cloud Platform provides Autoscaling, a convenient service that adjusts the number of instances in a managed instance group according to CPU utilization, HTTP load balancing serving capacity, or Stackdriver monitoring metrics [37]. GCP Autoscaling supports both reactive and predictive scaling approaches [38]. The interface for defining scaling policies is straightforward and user-friendly. It seamlessly integrates with other GCP services, including Stackdriver Monitoring and Logging.

J. Microsoft Azure

Microsoft Azure offers Azure Autoscale, a convenient service that automatically adjusts resource scaling according to predefined rules and schedules. Azure Autoscale offers support for both reactive and predictive scaling methods. Users can define scaling policies based on metrics like CPU usage, memory usage, and custom application metrics. Azure Autoscale seamlessly integrates with Azure Monitor for comprehensive monitoring and enables effortless scaling across a variety of services, such as virtual machine scale sets, cloud services, and app service plans.

III. CHALLENGES AND CONSIDERATIONS

Here is a graph illustrating the challenges organizations encounter when implementing auto-scaling [39].



Graph 2: Overcoming Hurdles: Challenges Faced by Organizations in Implementing Auto-scaling

Although auto-scaling has its advantages, there are various challenges and factors to consider when implementing and managing auto-scaling systems.

Complexity in Configuring Auto-scaling Systems

Configuring auto-scaling systems can be a challenging task, as it necessitates a thorough comprehension of the application's behavior and performance characteristics [39]. Choosing the right metrics, establishing appropriate thresholds, and crafting effective scaling policies necessitate thorough analysis and meticulous adjustments. Improperly configuring auto-scaling parameters can result in less-than-optimal performance or unnecessary expenses. Continuous monitoring and adjustment of the auto-scaling configuration is crucial to ensuring it stays aligned with the application's requirements and workload patterns [40].

Cost Management and Optimization

Optimizing costs through dynamic resource adjustments based on demand is a key benefit of auto-scaling. However, it is important to diligently manage and monitor the associated costs to ensure efficiency.

Scaling policies that are not optimized or thresholds that are set incorrectly can lead to the unnecessary expenditure of resources. Striking a balance between performance and cost involves setting appropriate scaling policies and closely monitoring the cost impact of auto-scaling actions. By implementing cost optimization techniques like reserved instances or spot instances, along with auto-scaling, you can enhance cost efficiency.

Latency in Resource Provisioning

Auto-scaling systems may encounter delays in resource provisioning, particularly when there is a sudden increase in workload. The time needed to provision and configure new instances may differ based on factors like instance type, operating system, and application setup. The latency can have a significant impact on the application's responsiveness and overall user experience, especially during periods of high demand. To address this, it is crucial to take into account the time it takes to provision resources when establishing scaling policies. Additionally, employing methods such as pre-warming or keeping a reserve of pre-configured instances can be beneficial. By closely monitoring performance metrics and implementing suitable cooldown periods between scaling actions, one can effectively avoid unnecessary scaling oscillations.

IV. CASE STUDIES

Studying real-world case studies of auto-scaling implementations in the industry offers valuable insights into the effective adoption of auto-scaling strategies and the valuable lessons derived from these experiences.

Successful Implementation of Auto-scaling in Industry

An excellent example of a successful auto-scaling implementation is Netflix, a top-tier video streaming platform. Netflix relies heavily on auto-scaling to effectively manage the high demand for its services, which can vary greatly due to factors such as the time of day, new content releases, and user behavior. Through the use of AWS Auto Scaling, Netflix can effortlessly adapt the number of instances to match metrics such as CPU utilization and request latency. This guarantees a smooth streaming experience for users.

One interesting case study to consider is Airbnb, an online marketplace for lodging and experiences. Airbnb utilizes auto-scaling to efficiently manage the fluctuating workload caused by its extensive user base across the globe. Through the use of both reactive and predictive scaling methods, Airbnb effectively maximizes resource utilization and ensures consistent availability, even during periods of high traffic. The company's auto-scaling strategy has played a crucial role in facilitating its rapid growth and broadening its market presence.

Lessons Learned and Best Practices

Companies like Netflix and Airbnb provide valuable insights and best practices for implementing auto-scaling in the industry. A crucial lesson to learn is the significance of thorough monitoring and data collection [41]. Efficient auto-scaling depends on precise and timely metrics to make well-informed scaling decisions. It is highly recommended for organizations to invest in comprehensive monitoring solutions that offer insights into application performance, resource utilization, and user behavior.

It is recommended to embrace a data-driven approach when it comes to auto-scaling. Examining historical data and utilizing machine learning techniques can assist organizations in creating more precise scaling policies and predicting future workload patterns. This proactive approach allows for resource provisioning in advance, minimizing the chance of performance issues and ensuring an enhanced user experience.

Continuously testing and optimizing auto-scaling configurations is crucial. Regular load testing and performance benchmarking can assist in identifying inefficiencies and optimizing scaling policies. Organizations need to establish a feedback loop that includes monitoring data, user feedback, and performance metrics to continuously enhance their auto-scaling strategies.

In addition, effective collaboration between development and operations teams is crucial for the successful implementation of auto-scaling [42]. DevOps practices, like infrastructure as code and continuous deployment, empower organizations to automate the provisioning and configuration of auto-scaling systems. This approach promotes collaboration to ensure that auto-scaling policies are in line with application requirements and performance goals.

V. FUTURE TRENDS AND RESEARCH DIRECTIONS

With the continuous evolution of cloud computing, there is an expectation for the advancement of auto-scaling techniques and strategies. These advancements will integrate with emerging technologies, creating new research opportunities.

Advancements in Auto-scaling Techniques

One of the key advancements in auto-scaling techniques is the increasing adoption of machine learning and artificial intelligence.

Machine learning algorithms can analyze historical data, identify patterns, and predict future workload demands more accurately, enabling proactive and intelligent auto-scaling decisions. Reinforcement learning, a subfield of machine learning, has shown promise in optimizing auto-scaling policies by learning from the environment and adapting to changing conditions [43].

Another growing trend involves the utilization of serverless computing and function-as-a-service (FaaS) platforms, which naturally facilitate auto-scaling. Serverless architectures abstract the underlying infrastructure, enabling developers to concentrate on writing code while the platform automatically scales resources based on incoming requests [44]. The integration of auto-scaling with serverless computing provides a highly scalable and cost-effective solution for event-driven and compute-intensive applications.

Integration with Emerging Technologies

The integration of auto-scaling with emerging technologies like edge computing and the Internet of Things (IoT) brings forth fresh possibilities and obstacles. Edge computing brings computation closer to the data source, reducing latency and enabling real-time processing. Auto-scaling at the edge is essential for managing the dynamic workloads generated by IoT devices and ensuring quick response times [45]. Scientists are investigating innovative auto-scaling techniques that take into account the limitations and diversity of edge devices.

Another aspect worth exploring is the integration of blockchain technology, specifically in the realm of decentralized applications (dApps). Efficient resource management is crucial for handling the computational demands of consensus mechanisms and smart contract execution in blockchain-based systems. Auto-scaling techniques can be utilized to optimize resource allocation in blockchain networks, ensuring enhanced scalability and performance while upholding the decentralized nature of the system.

Potential Research Areas

Multiple research areas in the field of auto-scaling deserve further exploration. An important area of focus is the advancement of advanced auto-scaling algorithms that take into account various objectives, including cost optimization, energy efficiency, and quality of service [46]. Utilizing multi-objective optimization techniques can assist in identifying the optimal trade-offs between various goals and offer more comprehensive solutions for auto-scaling.

Another area of research involves studying auto-scaling in multi-cloud and hybrid-cloud environments. With the growing adoption of multi-cloud strategies, organizations can avoid vendor lock-in and take advantage of the unique benefits offered by various cloud providers. However, effectively implementing auto-scaling across multiple clouds can be a complex undertaking. Research efforts can be directed towards the development of auto-scaling frameworks that can smoothly operate across various cloud platforms and efficiently allocate resources in a multi-cloud environment.

In addition, applying auto-scaling techniques to specific domains like big data processing, machine learning workloads, and scientific computing brings about distinct challenges and opportunities [47]. Every domain has unique requirements and characteristics that must be taken into account when developing auto-scaling strategies. Researchers can explore domain-specific auto-scaling approaches that consider the specific resource requirements, data interdependencies, and performance measurements associated with these domains.

VI. CONCLUSION

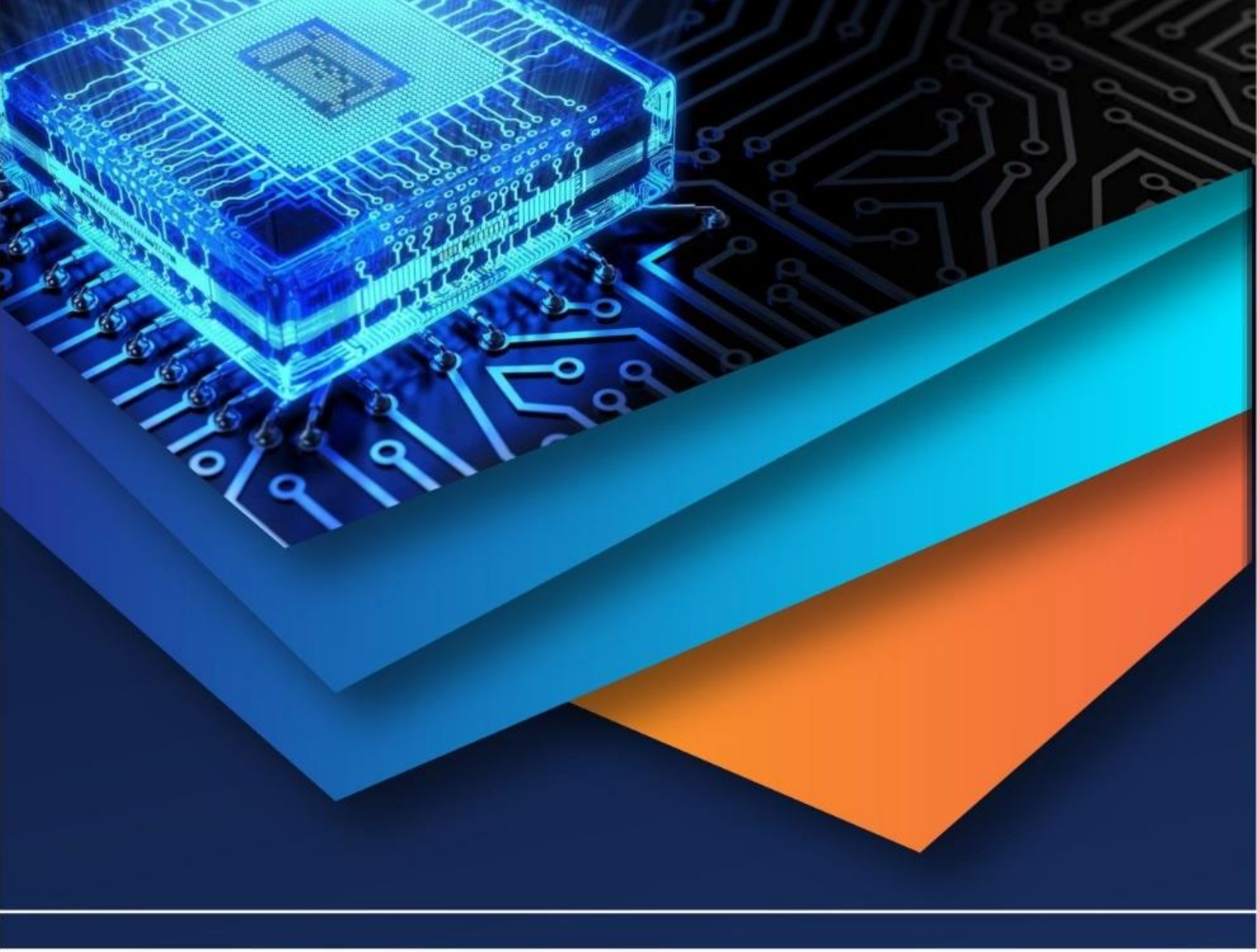
Ultimately, auto-scaling has become a vital component of cloud computing, allowing organizations to flexibly allocate resources, enhance application performance, and achieve cost-effectiveness. This article offers a thorough examination of auto-scaling, encompassing its fundamental principles, advantages, deployment tactics, obstacles, and practical examples. The importance of auto-scaling goes beyond its technical capabilities, allowing businesses to prioritize their main goals by taking advantage of the cloud's elasticity and flexibility. Cloud computing is constantly evolving, and auto-scaling is becoming increasingly important. Future trends and research directions show great potential for innovation and advancement in this area. Having a solid grasp of auto-scaling strategies is crucial for organizations that are embracing the cloud. This allows them to easily adjust to market changes, handle varying demands, and provide users with smooth experiences. Staying up-to-date with the latest advancements and best practices in auto-scaling allows businesses to fully utilize cloud computing and propel their digital transformation initiatives.

REFERENCES

[1] Lorido-Botran, T., Miguel-Alonso, J., & Lozano, J. A. (2014). A review of auto-scaling techniques for elastic applications in cloud environments. Journal of Grid Computing, 12(4), 559-592. https://doi.org/10.1007/s10723-014-9314-7
[2] Amazon Web Services. (2021). AWS Auto Scaling. https://aws.amazon.com/autoscaling/
[3] Google Cloud. (2021). Autoscaling groups of instances. https://cloud.google.com/compute/docs/autoscaler/
[4] Qu, C., Calheiros, R. N., & Buyya, R. (2018). Auto-scaling web applications in clouds: A taxonomy and survey. ACM Computing Surveys (CSUR), 51(4), 1-33. https://doi.org/10.1145/3148149

[5] Netto, M. A., Cardonha, C., Cunha, R. L., & Assunção, M. D. (2014). Evaluating auto-scaling strategies for cloud computing environments. In 2014 IEEE 22nd International Symposium on Modelling, Analysis & Simulation of Computer and Telecommunication Systems (pp. 187-196). https://doi.org/10.1109/MASCOTS.2014.32
[6] Al-Dhuraibi, Y., Paraiso, F., Djarallah, N., & Merle, P. (2018). Elasticity in cloud computing: state of the art and research challenges. IEEE Transactions on Services Computing, 11(2), 430-447. https://doi.org/10.1109/TSC.2017.2711009
[7] Mell, P., & Grance, T. (2011). The NIST definition of cloud computing. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-145
[8] Herbst, N. R., Kounev, S., & Reussner, R. (2013). Elasticity in cloud computing: What it is, and what it is not. In Proceedings of the 10th International Conference on Autonomic Computing (ICAC 13) (pp. 23-27). https://doi.org/10.1145/2465529.2465530
[9] Galante, G., & De Bona, L. C. E. (2012). A survey on cloud computing elasticity. In 2012 IEEE Fifth International Conference on Utility and Cloud Computing (pp. 263-270). https://doi.org/10.1109/UCC.2012.30
[10] Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J., & Ghalsasi, A. (2011). Cloud computing—The business perspective. Decision support systems, 51(1), 176-189. https://doi.org/10.1016/j.dss.2010.12.006
[11] Vaquero, L. M., Rodero-Merino, L., & Buyya, R. (2011). Dynamically scaling applications in the cloud. ACM SIGCOMM Computer Communication Review, 41(1), 45-52. https://doi.org/10.1145/1925861.1925869
[12] Coutinho, E. F., de Carvalho Sousa, F. R., Rego, P. A. L., Gomes, D. G., & de Souza, J. N. (2015). Elasticity in cloud computing: a survey. annals of telecommunications-annales des télécommunications, 70(7-8), 289-309. https://doi.org/10.1007/s12243-014-0450-7
[13] Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computer Systems, 25(6), 599-616. https://doi.org/10.1016/j.future.2008.12.001
[14] Aceto, G., Botta, A., De Donato, W., & Pescapè, A. (2013). Cloud monitoring: A survey. Computer Networks, 57(9), 2093-2115. https://doi.org/10.1016/j.comnet.2013.04.001
[15] Kanagasundaram, R., & Majumdar, S. (2017). A comparison of open source cloud monitoring tools. In 2017 IEEE International Conference on Computer and Information Technology (CIT) (pp. 193-198). https://doi.org/10.1109/CIT.2017.37
[16] Naskos, A., Gounaris, A., & Sioutas, S. (2019). Cloud elasticity: A survey. In Algorithmic Aspects of Cloud Computing (pp. 151-167). Springer, Cham. https://doi.org/10.1007/978-3-030-19759-9_9
[17] Copil, G., Moldovan, D., Truong, H. L., & Dustdar, S. (2013). Multi-level elasticity control of cloud services. In International Conference on Service-Oriented Computing (pp. 429-436). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-45005-1_30
[18] Dutta, S., Gera, S., Verma, A., & Viswanathan, B. (2012). SmartScale: Automatic application scaling in enterprise clouds. In 2012 IEEE Fifth International Conference on Cloud Computing (pp. 221-228). https://doi.org/10.1109/CLOUD.2012.12
[19] Ilyushkin, A., Ali-Eldin, A., Herbst, N., Bauer, A., Papadopoulos, A. V., Epema, D., & Iosup, A. (2018). An experimental performance evaluation of autoscalers for complex workflows. ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS), 3(2), 1-32. https://doi.org/10.1145/3164537
[20] Moldovan, D., Copil, G., Truong, H. L., & Dustdar, S. (2013). On analyzing elasticity relationships of cloud services. In 2013 IEEE Sixth International Conference on Cloud Computing (pp. 447-454). https://doi.org/10.1109/CLOUD.2013.41
[21] Mao, M., & Humphrey, M. (2012). A performance study on the vm startup time in the cloud. In 2012 IEEE Fifth International Conference on Cloud Computing (pp. 423-430). https://doi.org/10.1109/CLOUD.2012.103
[22] Dawoud, W., Takouna, I., & Meinel, C. (2012). Elastic virtual machine for fine-grained cloud resource provisioning. In Global Trends in Computing and Communication Systems (pp. 11-25). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-29219-4_2
[23] Ranjan, R., Benatallah, B., Dustdar, S., & Papazoglou, M. P. (2015). Cloud resource orchestration programming: overview, issues, and directions. IEEE Internet Computing, 19(5), 46-56. https://doi.org/10.1109/MIC.2015.20
[24] Xu, G., Pang, J., & Fu, X. (2013). A load balancing model based on cloud partitioning for the public cloud. Tsinghua Science and Technology, 18(1), 34-39. https://doi.org/10.1109/TST.2013.6449405
[25] Amazon Web Services. (2021). Elastic Load Balancing. https://aws.amazon.com/elasticloadbalancing/
[26] Grozev, N., & Buyya, R. (2014). Multi-cloud provisioning and load distribution for three-tier applications. ACM Transactions on Autonomous and Adaptive Systems (TAAS), 9(3), 1-21. https://doi.org/10.1145/2662112
[27] Han, R., Guo, L., Ghanem, M. M., & Guo, Y. (2012). Lightweight resource scaling for cloud applications. In 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid) (pp. 644-651). https://doi.org/10.1109/CCGrid.2012.52
[28] Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, 41(1), 23-50. https://doi.org/10.1002/spe.995
[29] Gong, Z., Gu, X., & Wilkes, J. (2010). PRESS: Predictive elastic resource scaling for cloud systems. In 2010 International Conference on Network and Service Management (pp. 9-16). https://doi.org/10.1109/CNSM.2010.5691343
[30] Jayasinghe, D., Pu, C., Eilam, T., Steinder, M., Whally, I., & Snible, E. (2011). Improving performance and availability of services hosted on IaaS clouds with structural constraint-aware virtual machine placement. In 2011 IEEE International Conference on Services Computing (pp. 72-79). https://doi.org/10.1109/SCC.2011.28
[31] Aceto, G., Botta, A., De Donato, W., & Pescapè, A. (2013). Cloud monitoring: A survey. Computer Networks, 57(9), 2093-2115. https://doi.org/10.1016/j.comnet.2013.04.001
[32] Naskos, A., Gounaris, A., & Sioutas, S. (2019). Cloud elasticity: A survey. In Algorithmic Aspects of Cloud Computing (pp. 151-167). Springer, Cham. https://doi.org/10.1007/978-3-030-19759-9_9
[33] Nikravesh, A. Y., Ajila, S. A., & Lung, C. H. (2015). Towards an autonomic auto-scaling prediction system for cloud resource provisioning. In Proceedings of the 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (pp. 35-45). https://doi.org/10.1109/SEAMS.2015.22
[34] Gambi, A., Toffetti, G., Pautasso, C., & Pezze, M. (2013). Kriging controllers for cloud applications. IEEE Internet Computing, 17(4), 40-47. https://doi.org/10.1109/MIC.2012.142

[35] Amazon Web Services. (2021). Predictive Scaling for EC2, Powered by Machine Learning. https://aws.amazon.com/blogs/aws/new-predictive-scaling-for-ec2-powered-by-machine-learning/
[36] Amazon Web Services. (2021). Dynamic Scaling. https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scale-based-on-demand.html
[37] Amazon Web Services. (2021). Target Tracking Scaling Policies. https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scaling-target-tracking.html
[38] Google Cloud. (2021). Autoscaling Groups of Instances. https://cloud.google.com/compute/docs/autoscaler
[39] Jindal, A., Podolskiy, V., & Gerndt, M. (2019). Performance modeling for cloud microservice applications. In Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering (pp. 25-32). https://doi.org/10.1145/3297663.3310309
[40] Kumrai, T., Ota, K., Dong, M., Kishigami, J., & Sung, D. K. (2017). Multiobjective optimization in cloud brokering systems for connected Internet of Things. IEEE Internet of Things Journal, 4(2), 404-413. https://doi.org/10.1109/JIOT.2016.2646375
[41] Schroeder, B., & Harchol-Balter, M. (2006). Web servers under overload: How scheduling can help. ACM Transactions on Internet Technology (TOIT), 6(1), 20-52. https://doi.org/10.1145/1125274.1125278
[42] Cito, J., Leitner, P., Fritz, T., & Gall, H. C. (2015). The making of cloud applications: An empirical study on software development for the cloud. In Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering (pp. 393-403). https://doi.org/10.1145/2786805.2786826
[43] Arabnejad, H., Pahl, C., Jamshidi, P., & Estrada, G. (2017). A comparison of reinforcement learning techniques for fuzzy cloud auto-scaling. In 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID) (pp. 64-73). https://doi.org/10.1109/CCGRID.2017.16
[44] Baldini, I., Castro, P., Chang, K., Cheng, P., Fink, S., Ishakian, V., ... & Suter, P. (2017). Serverless computing: Current trends and open problems. In Research Advances in Cloud Computing (pp. 1-20). Springer, Singapore. https://doi.org/10.1007/978-981-10-5026-8_1
[45] Yousefpour, A., Fung, C., Nguyen, T., Kadiyala, K., Jalali, F., Niakanlahiji, A., ... & Jue, J. P. (2019). All one needs to know about fog computing and related edge computing paradigms: A complete survey. Journal of Systems Architecture, 98, 289-330. https://doi.org/10.1016/j.sysarc.2019.02.009
[46] Guerrero, C., Lera, I., & Juiz, C. (2018). A lightweight decentralized service placement policy for performance optimization in fog computing. Journal of Ambient Intelligence and Humanized Computing, 10(6), 2435-2452. https://doi.org/10.1007/s12652-018-0914-0
[47] Guo, T., Sharma, U., Wood, T., Sahu, S., & Shenoy, P. (2014). Seagull: intelligent cloud bursting for enterprise applications. In Proceedings of the 2012 USENIX Annual Technical Conference (USENIX ATC'12) (pp. 33-45). https://www.usenix.org/system/files/conference/atc12/atc12-final37.pdf



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY