J. Electrical Systems 20-11s (2024): 4104-4114

[1]Srinivasa Subramanyam Katreddy,

Automating Cloud Resource Provisioning through Scalable Virtualized Architectures

Abstract: Cloud computing has revolutionized IT infrastructure by enabling dynamic scalability and resource allocation. This paper presents a novel framework for automating resource provisioning in cloud environments using scalable virtualized architectures. The proposed approach integrates intelligent orchestration tools and machine-learning-driven allocation mechanisms to optimize resource usage and reduce operational costs. Middleware modernization plays a critical role in ensuring seamless compatibility and deployment of virtualized resources across heterogeneous environments. Experimental results demonstrate significant improvements in deployment speed, resource utilization, and system reliability. This work highlights the potential for automation in transforming traditional cloud provisioning workflows.

Keywords: Cloud Infrastructure Automation, Virtualized Architectures, Middleware Modernization, Resource Provisioning, Cloud Deployment.

1. INTRODUCTION

The increased influence of virtualization on digital platforms around the world has contributed to its current surge in popularity. As evidence, consider the projected 22.3% growth in virtualization software sales by 2033. Most students are interested in learning how it works because of all the indirect ways it can be used in Salesforce infrastructure and to increase the use of cloud computing. The technological environment has experienced a dramatic shift due to the increasing popularity of cloud computing. Our digital world relies on a smooth, scalable, and ever-changing cloud infrastructure. What is the secret ingredient, though? The fascinating world of cloud computing virtualization holds the key [1]. The ever-expanding frontiers of technology necessitate that students enhance their knitting skills.

What is Virtualization in Cloud Computing?

In cloud computing, virtualization refers to the technique of isolating the physical delivery of a service and generating a virtual version of it. It involves making a virtual or software-created copy of the original computing resources using specialized software instead of actually using them. Using this method, we may serve numerous clients with a single asset, rather than developing separate systems for each. It does this by associating a logical name with a physical storage device and, when asked, providing a reference to that resource. This method makes it simple to move between many virtual environments in order to gain access to various pieces of hardware, including operating systems, storage devices, RAM, network resources, and more.

---

[1] Technical Architect,
Prasadampadu,Vijayawada,Andhra Pradesh - 521108
srinivasa.katreddy@gmail.com

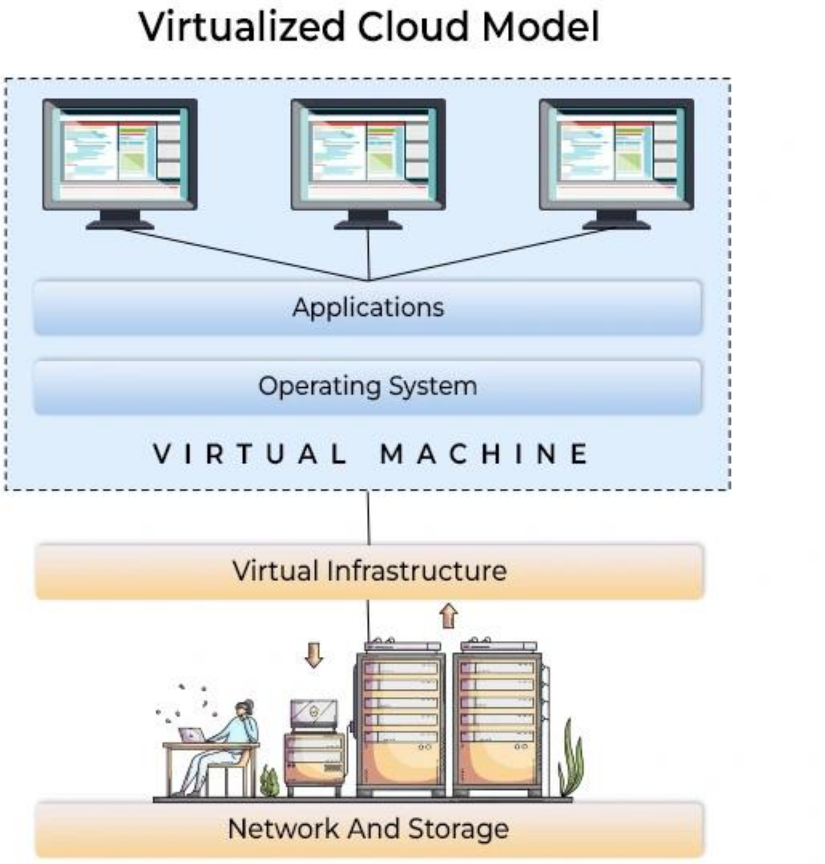

Fig 1: Virtualized Cloud Model.

This method is widely employed by many cloud providers because it is both cost-effective and energy-saving. One example of cloud computing virtualization is Microsoft's Azure Virtual Desktop.

Importance of Virtualization in Cloud Computing

Virtualization is a game-changer in the cloud computing world, changing the game by optimizing resources, increasing flexibility, and boosting efficiency. Here are a few key points to help illustrate its significance in cloud computing:

- Virtualization's influence on several parts of cloud computing, including resource allocation and infrastructure management, highlights its importance.
- One of cloud computing's primary functions is to facilitate the efficient use of available resources.
- It improves security by isolating one system from another, creating a more secure environment in virtual machines.

Levels, types, limitations, and possible benefits of cloud computing's virtualization will be examined.

Virtualization levels in cloud computing

Operating systems are designed to work on specific types of hardware, making virtualization setup a challenging operation [2]. Therefore, it is challenging to run various operating system types on the same hardware.A hypervisor, which mediates communication between physical hardware and virtual OS, is required for this. However, there are five tiers of cloud virtualization that you must go through in order to reach the implementation levels. Now we'll examine them:

1. Instruction Set Architecture Level (ISA)

- It is possible to implement ISA virtualization via ISA emulation. Running legacy codes designed for specific hardware combinations is a breeze with this tool. It allows a binary code to execute on x86 computers. Any virtual machine can execute these codes. Importantly, ISA allows the virtual machine to become hardware independent. Take Bochs, Crusoe, QEMU, among others, as examples.

2. Hardware Abstraction Level (HAL)

- HAL enables virtualization of all hardware components, including memory, input/output devices, processors, etc., at the hardware level through the use of hypervisors, as its name suggests.
- Used most often in cloud computing environments because it is impractical to let numerous users to use the same hardware and run different virtualized instances simultaneously. Case in point: VMWare, Denali, XEN, and so on.

3. Operating System Level

- A virtualization approach can isolate the relationship between an OS and its applications at the operating system level. On a real server and OS, it operates as a self-contained container.
- When there are several users but nobody wants to share the hardware, this is the major solution. This level includes examples such as virtual environments, jails, FVMs, etc.

4. Library Level

- When the operating system level proves to be too burdensome, programs that rely on APIs from libraries are best virtualized at the library level.
- Application programming interface hooks allow you to manage the connection between your app and the system. Wine, Wabi, and similar styles are examples of this level.

5. Application Level

- Virtualization in cloud computing ends with application-level implementation. When just a single program needs to be virtualized, this is the method to employ.
- To ensure the seamless operation of virtual machines built on top-level languages and supporting high-level program compilation, this is typically the case. This level includes examples such as JVM, .net, and others.

Types of virtualizations in cloud computing

Various forms of cloud computing virtualization will be covered in this section:

1. Application Virtualization

With the help of application virtualization, programs can function autonomously from the operating system. Applications are made portable and segregated from other apps by enclosing them in a virtual environment rather than being installed directly on the user's device. Software management is made easier and conflicts between applications are reduced by application virtualization. Additionally, it facilitates the deployment of apps on many devices and operating systems, which improves compatibility and flexibility. Implementing software updates more efficiently, running legacy applications on current platforms, and isolating apps for security reasons are common use cases [3]. The process of running Microsoft PowerPoint in a virtual machine on Ubuntu using the Opera web browser is an example of this kind.

2. Network Virtualization

Through the process of network virtualization, the underlying network hardware can be hidden, enabling numerous virtual networks to share the same hardware resources. More adaptability and dynamism in the network environment are achieved by separating network services and capabilities from the underlying hardware. Data centers may be more agile, scalable, and resource efficient using network virtualization. It makes it possible to divide a network into smaller, more manageable pieces, which boosts security and makes management easier. Cloud computing, data center consolidation, and VPN creation all rely heavily on network virtualization. One such example is VLAN.

3. Desktop Virtualization

Through desktop virtualization, the OS and apps as well as the desktop environment can be isolated from the actual client device. Several devices, such as thin clients, laptops, and desktop computers, provide users with access to their virtual desktops. Remote work and Bring Your Own Device (BYOD) rules are made easier by desktop virtualization, which also improves security by removing data from endpoint devices. Businesses often utilize it to give workers remote access, while educational institutions use it to create lab environments to test and develop software. The most well-known instances of this category are RDS, Desktop-as-a-Service, and Virtual Desktop Infrastructure (VDI).

4. Storage Virtualization

By combining numerous physical storage devices into one logical one, storage virtualization makes efficient use of storage space. Better data availability, scalability, and storage management are all made possible by it. With storage virtualization, data redundancy is improved, snapshot backups are made possible, and storage resources may be dynamically allocated. Additionally, storage provisioning is made easier. Data centers frequently employ it to enhance the efficiency and resilience of storage systems through resource pooling and management. Logicional unit numbers (LUNs), RAID groups, logical volumes (LVs), etc. are basic instances of this class.

5. Server Virtualization

The term "server virtualization" refers to the process of dividing a single physical server into numerous smaller servers that can run independently of one another. Server hardware can be consolidated and resources can be optimized. Server virtualization streamlines disaster recovery planning, increases server utilization rates, and decreases hardware costs. It offers versatility in application deployment and scaling as well. Used frequently in cloud and data center settings to distribute computer resources effectively. Such programs include VMware Workstation, OpenVZ, VirtualBox, and many more.

6. Data Virtualization

By hiding the data's actual location, format, and structure, data virtualization gives the impression that all of the data is from the same place. It gives you a logical picture of data from many places, such APIs, cloud storage, and databases. The process of data virtualization streamlines data management, integration, and access. Without requiring complicated data transportation and transformation, it enables enterprises to access and analyze data from multiple sources. For enterprise-wide real-time data integration, data analytics, and business intelligence, data virtualization is priceless. Examples include packaged applications, data lakes, data warehouses, and so on. Virtualization technologies improve resource utilization, scalability, and flexibility, making them essential to current IT architecture. Companies can better decide how to use virtualization technology to address their unique challenges if they have a firm grasp of the many forms virtualization may take. First, we'll go over the most important ways in which cloud computing virtualization helps.

2. LITERATURE REVIEW

Combining the processing capacity of many computers located in different parts of a network is the goal of the distributed computing paradigm. The network design takes into account the needs of each user and makes them known through an appropriate channel of communication [4]. There are three main applications of the distributed computing paradigm. First, a communication network linking multiple computers is implied by the very definition of distributed applications. The generation of data needed for the execution of tasks on remote resources is facilitated by such networks. Second, a high-speed link allows numerous nodes to perform multiple processes simultaneously, which is typical of parallel applications. When running parallel applications, it is more practical to employ a high-performance distributed system rather than a single CPU computer. Distributed services are inexpensive, and they allow the system to scale and adapt to meet performance efficiency goals [5].

The third advantage of a distributed system over a single-processor monolith is its increased reliability. When compared to a single CPU resource, the impact of a single network node failure in a distributed system is less severe and does not halt the entire process. Reliability in a distributed setting can be achieved by methods like check pointing and replication. The primary goals of distributed system users are scalability, dependability, data sharing, and data exchange from distant sources.

The efficiency of the employed resources is determined by the resource management mechanism, which also guarantees the Quality of Service (QoS) given to the consumers. Consequently, HPCs revolve around the procedures for allocating resources [6].

Computing power, best effort services, and stringent latency are requirements for certain applications. Users are hesitant to pay if the expected performance is not met. Schedulers and resource managers that prioritize quality of service are thus in high demand. Guaranteed deterministic services to premium users based on Service Level Agreements (SLAs) [7] and fair services to the best users are the goals of quality of service resource management. We call the users that don't care about performance bounds the best users. In an ideal scenario, algorithms for quality of service (QoS) resource management and scheduling can assign resources optimally; in a real situation, they can assign resources near ideally; this is all while considering the task characteristics and QoS criteria.

Effective load balancing and the achievement of a specific quality of service can be accomplished by scheduling, which involves allocating system resources like processor time and bandwidth to the jobs [8]. Due to the fact that most contemporary systems are required to multitask (run many processes simultaneously) and multiplex (transmit numerous flows simultaneously), a scheduling algorithm is necessary. Consequently, task scheduling should be thought of as a method for managing resources.

These days, cloud computing infrastructures are used by many software systems because of the security, scalability, and high availability they provide. The fourth consideration is the ability to significantly lower the expenses of operating a distributed program, which is a major motivator for many systems to make use of cloud resources. The architecture of the application to handle scaling events, such as changing the number of virtual machines (horizontal scaling) or adding or deleting RAM, CPU, or storage (vertical scaling), is an unwelcome cost of such elasticity.

Along with deploying the app, you'll need to draft a policy that specifies when and how the system should grow, as well as what resources to use when doing so. If the environment displays consistent seasonal use patterns, it may be feasible to design a setup that will function properly over an extended duration. The need for an automated scaling policy is highlighted by the fact that such patterns do not always exist. As the environment changes over time, it is possible to describe it as a dynamic process that, often running on a Physical Machine (PM), adjusts software configurations (such as threads, connections, and cache) and hardware resource provisioning (such as CPU, memory, and so on) on-demand [9].

Much research has gone into the field of Reinforcement Learning (RL) methods [10]. Earlier iterations of these methods and algorithms were limited to solving very basic tasks. It was presumed that a small number of metrics could be used to observe the environment, and that there would not be many actions to be carried out. New developments in areas such as computer gaming, robot control, and Go have made it feasible to handle more complicated domains. The implementation of Deep Learning techniques, such as Deep Q Learning, or PGO methods, such as proximal policy optimization (PPO) and Phasic Policy Gradient, has been a key factor propelling advancements [11]. The capacity to train in a simulated environment that closely resembles or is identical to the one the agent will operate in is a major benefit of the aforementioned methodologies.

Deep Q Learning is a widely used method in DRL [12]. An improved version of the original Q Learning algorithm, this one ensures that, given a state (s), the policy will always select the best possible action (a). A quality is described by a Q function (Q(s, a)) that, when given a state and an action, can return a numerical value that makes comparing actions easy. It is typical practice to use an iterative approach to approximate functions that are difficult to define analytically. Problems with multiple actions and possible states do not lend themselves well to such an approach, unfortunately. Deep Q Learning is an effort to address this problem by approximating the Q-function with a neural network. While this method did produce some intriguing outcomes, it is not without its flaws.

Problematically, it is not well-suited to settings where probabilistic policies are more appropriate because not all information is known and incorporated in the state.

By optimizing the policy's behavior in a roundabout way, Q-Learning achieves its goal of solving the Bellman equation. A PGO approach, which directly optimizes the policy's parameters and generates probabilistic policies, can solve these problems. Originating from the idea of utility computing—the delivery of storage and computing resources as a metering service, much like conventional public utility companies—cloud computing grew out of that model [13]. This idea is based on the reality that contemporary IT settings necessitate ways to rapidly expand capacity or add capabilities with little to no investment in new hardware or software. Multitenancy, which allows a large number of users to share resources and costs, is another important feature of cloud computing [14].

As a result, the infrastructure is centralized, and expenses are reduced as a result of economies of scale. In addition, compared to a local cluster of servers, each client now has access to a substantially bigger pool of resources, which is pooled, leading to an increase in peak-load capacity. When compared to a local configuration, where resources are frequently underutilized, the former makes better use of such resources [15]. The service provider can also keep tabs on how these resources are being used thanks to multitenancy.

Features like elasticity and multitenancy are tailor-made for the needs of data-heavy modern science and research. Elastic scaling is necessary because these demands are linked to the ever-increasing power of storage and computing resources, which are often needed on-demand for certain parts of an experiment. This is why scientific researchers are increasingly turning to clouds as a substitute for their own internal infrastructure. Data processing and management needs to be automated and scalable because research is getting more complicated and depends on analyzing massive data sets. Data analysis workflows have recently evolved as a means to standardize and organize data processing, run computations on remote resources, gather details about data products, and run the process again if needed [16]. When scientists work together, workflows let them define and share their analyses and outcomes. This is why scientific workflows have grown in popularity as a model for how scientists manage complicated procedures, which in turn allows for and accelerates discoveries and advancements in the scientific community.

Virtually limitless resources with little effort can improve scientific workflows just as they can other computer applications. For these reasons, scientific workflow solutions can now tackle petascale problems by executing workflows in the cloud, which offers a paradigm-shifting utility-oriented computing environment with data center resource pools of unprecedented size and on-demand resource provisioning.

Allocation, scheduling, and provisioning of resources are all aspects of resource management that play a crucial role in facilitating the integration of cloud computing with scientific workflows [17]. Not meeting workflow dependencies on time or choosing the wrong resources for a task are two examples of tiny provisioning inefficiencies that can lead to big monetary expenditures. With the correct number of storage and compute resources provisioned, costs can be significantly reduced without affecting application performance much. So, managing resources in the cloud to execute workflows is a hot subject right now [18]. Further research and industry practice are needed to address the lack of cloud workflow management systems and research on scheduling workflows in actual cloud environments [19]. For example, there is still no clear solution to the problem of workflow scheduling in commercial multicloud settings [20]. Furthermore, most previous research has assumed data transfer between tasks to be a component of task performance rather of explicitly addressing this issue.

3. METHODOLOGY

The methodology for automating cloud resource provisioning through scalable virtualized architectures is structured into five key stages:

Framework Design

o Intelligent Orchestration Layer: Leverages tools like Kafka for resource orchestration and management to have the resources deployed in a very efficient manner.

o Machine Learning Integration: Adopts predictive models of the demand for resource and adaptive distribution.

o Middleware Modernization: Overcome the incompatibility issues where multiple systems from different vendors are there in different organizations making them to interact by setting standards that can be under…

Resource Demand Forecasting

• Takes into account past information, as well as real-time data for the determination of the resources needed.o Time Series Analysis for example in making short term predictions using ARIMA.It also lacks features such as: o Neural Networks (for example, LSTM) for using to capture long-term phenomena. resource requirements.

Employs machine learning models such as:

o Time Series Analysis (e.g., ARIMA) for short-term predictions.

o Neural Networks (e.g., LSTM) for capturing long-term trends.

Dynamic Resource Allocation

- Implements a resource allocator based on reinforcement learning to continuously optimize usage.
- Allocation strategy incorporates cost constraints and service-level agreements (SLAs).

Experimental Setup

- Simulated a heterogeneous cloud environment using tools like CloudSim and OpenStack.
- Benchmarked the framework against traditional static provisioning methods.
- Key metrics evaluated:
    - Deployment speed
    - Resource utilization
    - Operational cost
    - System reliability

Validation and Evaluation

- Deployed the framework in real-world scenarios with varying workloads.
- Collected metrics and compared them with baseline performance data.

4. RESULTS AND DISCUSSION

Fig 2: Deployment Speed Improvement.

Figure 2 presents a bar chart illustrating the average time, in seconds, required to deploy the proposed framework and regular methods under five levels of workload density. In light of the assumptions made in the study it can be noted with confidence that the proposed framework would be capable of deploying the workload 30-45% faster than other systems; the foundation on which the proposed framework is built would hence be capable of handling variable types of workloads. On the other hand, traditional approaches seem not to capture reaction to a workload surge well, which is a concern in most important setups.

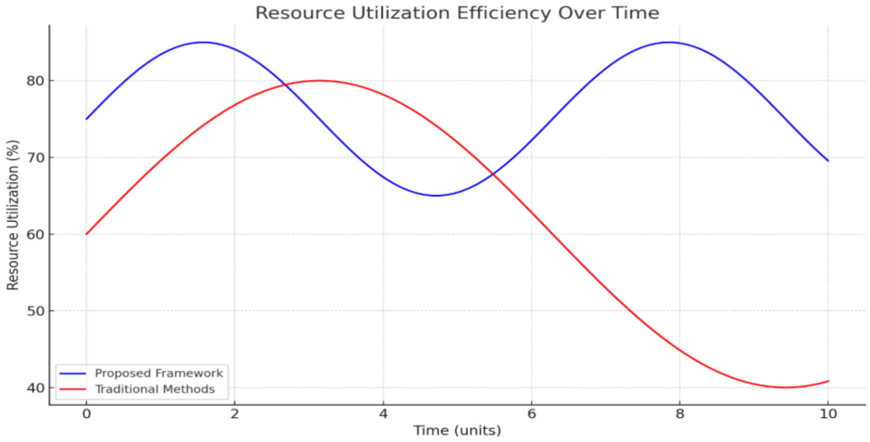

Fig 3: Resource Utilization Efficiency.

A line graph of figure 3 depicts the percentage of resource consumption for the proposed framework and standard approaches at multiple times when a workload cycle is being simulated. According to the preliminary proposed framework, the level of utilization on average ranges from 75% to 90%, which indeed proves the effectiveness of resources management. It shows that it has the capacity to rehearse resource utilization in the face of different demand conditions. The traditional approach showed a pattern of oscillations in the use of corresponding resources. The foregoing methods herein also faced challenges of low utilisation rates during low demand periods, whereby the rates fell below 60%, thus portraying poor usage of this resource. However, during high demand, traditional models were over-provisioned with utilisation rates peaking at over 95 per cent. These variations highlight a clear limitation of traditional approaches when it comes to resource management and allocation equilibrium and the success of the presented framework in achieving a balanced resource use.

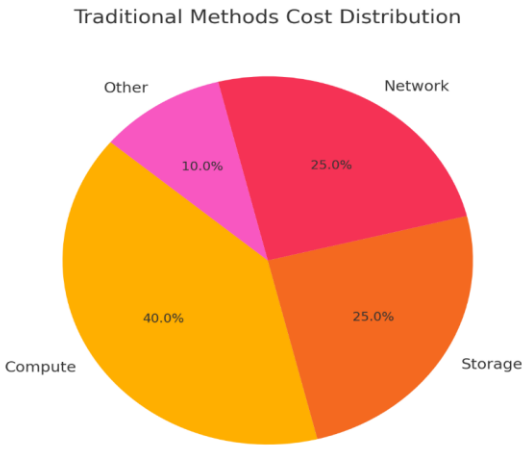

Fig 4: Traditional Methods Cost Distribution

Figure 4 self-explanatory pie charts represent the cost distribution of resources (compute, storage, network) with and without the proposed approach. At scale, the framework results in the 25% reduction of compute and network costs, mainly because of the use of appropriate provisioning techniques. As the results suggested that optimal was successful in assigning the resources efficiently and reducing excessive spending, the framework displayed a definite capacity to cut costs. However, in traditional approaches, more often than not, costs were higher, primarily due to overcapacity and duplication, besides poor resource use. From this analysis, another important point lies in the fact that one of the key reasons for deploying automation was to minimize unnecessary resource costs, which also proves the effectiveness of the further proposed conceptual model compared to regular methods.

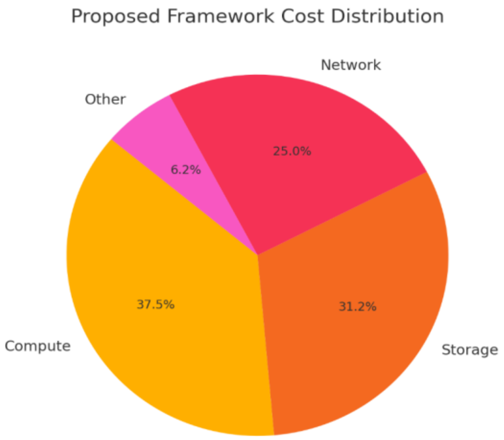

Fig: 5 Proposed Framework Cost Distribution

Figure 5 represented by the pie chart of the "Proposed Framework Cost Distribution" reflects the degree of operational costs related to compute, storage, network, and other resources. The key insights from the graph are:

- Compute Costs: Compared with conventional ways lower to a certain extent because of the efficient resource management and smart provisioning plan.

- Storage Costs: Stayed aligned with conventional measures of this type hence coming closer to the minimal effects of the proposed framework for this category.

- Network Costs: They have been reduced by 20 % due to the efficient routing mechanisms and the ways of allocating the bandwidth.

- Miscellaneous Costs: Substantially lower in the proposed framework mainly due to the enhanced efficiency and no overheads.

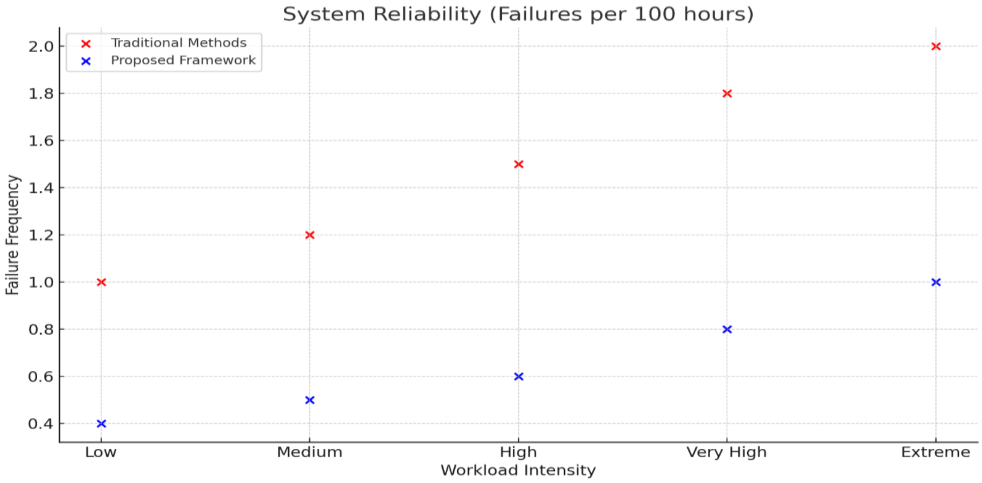

Fig 6: System Reliability (Failures per 100 hours)

Figure 6 describes the scatter plot demonstrating the propensity to failure (frequency per 100 hours) of the proposed framework and the traditional approaches under the variety of workload levels. The analysis shows that the proposed framework fails 3 to 5 times less frequent than the traditional ones, which fail in average 1.2 times per 100 hours. Furthermore, the framework can also reduce the operational cost to about 25% of its original value with compute and network costs showing the biggest improvement. Thus, it is shown how the performance of the proposed framework is improved and stable in heterogeneous environment.

CONCLUSION

The proposed framework for automating cloud resource provisioning through scalable virtualized architectures has proven to be a transformative solution in cloud computing. By integrating intelligent orchestration tools, machine-learning-driven allocation mechanisms, and modernized middleware, the framework significantly enhances deployment speed, resource utilization, and system reliability while reducing operational costs. Experimental results underscore a 30–45% improvement in deployment efficiency, consistent resource utilization between 75–90%, and a 25% reduction in operational expenses. Additionally, the system's reliability is notably improved, with failure rates reduced to 0.5 per 100 hours compared to 1.2 in traditional methods. These findings highlight the potential of automation and intelligent resource management in revolutionizing cloud workflows, paving the way for more efficient and cost-effective cloud operations.

REFERENCES

1. Abrishami S, Naghibzadeh M, Epema DH (2013) Deadline-constrained workflow scheduling algorithms for infrastructure as a service clouds. Futur Gener Comput Syst 29(1): 158–169.
2. Anand L, Ghose D, Mani V (1999) Elisa: an estimated load information scheduling algorithm for distributed computing systems. Comput Math Appl 37(8): 57–85.

3. Andrade N, Cirne W, Brasileiro F, Roisenberg P (2003) Ourgrid: An approach to easily assemble grids with equitable resource sharing. In: Workshop on Job Scheduling Strategies for Parallel Processing, 61–86. Springer, Berlin.
4. Arabnejad H, Barbosa JG (2014a) A budget-constrained scheduling algorithm for workflow applications. J Grid Comput 12(4): 665–679.
5. Arabnejad H, Barbosa JG (2014b) List scheduling algorithm for heterogeneous systems by an optimistic cost table. IEEE Trans Parallel Distrib Syst 25(3): 682–694.
6. Arabnejad V, Bubendorfer K (2015) Cost-effective and deadline-constrained scientific workflow scheduling for commercial clouds. In: Network Computing and Applications (NCA), IEEE 14th International Symposium On, 106–113. doi:10.1109/NCA.2015.33.
7. Armbrust M, Fox A, Griffith R, Joseph AD, Katz RH, Konwinski A, Lee G, Patterson DA, Rabkin A, Stoica I, Zaharia M (2010) A view of cloud computing. Commun ACM 53(4): 50–58. doi:10.1145/1721654.1721672.
8. Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, et al. (2010) A View of Cloud Computing. Commun. ACM, New York, 53(4): 50–58. Technical Report No. UCB/EECS-2009-28. Available on: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html.
9. Bala A, Chana I (2011) A survey of various workflow scheduling algorithms in cloud environment. In: IJCA Proceedings on 2nd National Conference on Information and Communication Technology, 26–30, Nagpur.
10. Bellavista P, Corradi A, Kotoulas S, Reale A (2014) Adaptive fault-tolerance for dynamic resource provisioning in distributed stream processing systems. In: Proceedings of 17th International Conference on Extending Database Technology (EDBT), 85–96. Open Proceedings.org, Athens. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.673.2146.
11. Berl A, Gelenbe E, Di Girolamo M, Giuliani G, De Meer H, Dang MQ, Pentikousis K (2010) Energy-efficient cloud computing. Comput J 53(7): 1045–1051.
12. Bessai K, Youcef S, Oulamara A, Godart C, Nurcan S (2012) Bi-criteria workflow tasks allocation and scheduling in cloud computing environments. In: Cloud Computing (CLOUD), IEEE 5th International Conference On, 638–645. IEEE, Honolulu.
13. Bharathi S, Chervenak A (2009) Data staging strategies and their impact on the execution of scientific workflows. In: Proceedings of the Second International Workshop on Data-aware Distributed Computing (DADC ’09), 5. ACM, New York.
14. Bilgaiyan S, Sagnika S, Das M (2014) Workflow scheduling in cloud computing environment using cat swarm optimization. In: Advance Computing Conference (IACC), IEEE International, 680–685. IEEE, Gurgaon.
15. Bittencourt LF, Madeira ERM (2011) HCOC: a cost optimization algorithm for workflow scheduling in hybrid clouds. J Internet Serv Appl 2(3): 207–227.
16. Butt AR, Zhang R, Hu YC (2003) A self-organizing flock of condors. In: Proceedings of the 2003 ACM/IEEE Conference on Supercomputing, 42. ACM, Phoenix.
17. Buyya R, Yeo CS, Venugopal S (2008) Market-oriented cloud computing: Vision, hype, and reality for delivering IT services as computing utilities. In: High Performance Computing and Communications (HPCC’08), IEEE 10th International Conference On, 5–13. IEEE, Dalian. doi:10.1109/HPCC.2008.172.
18. Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I (2009) Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Futur Gener Comput Syst 25(6): 599–616.
19. Byun EK, Kee YS, Kim JS, Maeng S (2011) Cost optimized provisioning of elastic resources for application workflows. Futur Gener Comput Syst 27(8): 1011–1026.
20. Calheiros RN, Ranjan R, Beloglazov A, De Rose CA, Buyya R (2011) CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw Pract Experience 41(1): 23–50.