# Adaptive Evolutionary Optimization for Resource Allocation in Cloud Microservices

S Saivardhan, Sufian Ali, V Kethan Reddy, Sivasankara Rao S

[1]B. Tech IV Year(21WJ1A05V3), Department of CSE, Guru Nanak Institutions Technical Campus, Hyderabad.
[2]B. Tech IV Year(21WJ1A0529), Department of CSE, Guru Nanak Institutions Technical Campus, Hyderabad.
[3]B. Tech IV Year(21WJ1A05X7), Department of CSE, Guru Nanak Institutions Technical Campus, Hyderabad.
Associate Professor, Department of CSE, Guru Nanak Institutions Technical Campus, Hyderabad.

Abstract: Efficient resource allocation for microservice management in heterogeneous cloud environments remains a critical challenge due to dynamic workload variations and network constraints. This paper presents an optimized framework, the Multi-Objective Microservice Allocation (MOMA) algorithm, which formulates resource management as a constrained optimization problem. The proposed approach prioritizes two key factors resource utilization and network communication overhead—to ensure efficient microservice deployment and system performance. By intelligently distributing workloads and minimizing transmission delays, the framework simplifies cloud service orchestration and enhances real-time workload monitoring. A comparative evaluation against existing algorithms using real-world datasets highlights significant improvements in resource balancing, network efficiency, and overall system reliability. Experimental results confirm that the MOMA algorithm effectively enhances resource utilization, reduces network congestion, and improves cloud service stability, making it a promising solution for modern cloud computing environments.

Keywords: Resource allocation, genetic algorithm, container-based heterogeneous cloud, multi-objective optimization, microservice.

## I.INTRODUCTION

Huge-scale programs may now be divided into smaller, independent components thanks to the growing popularity of microservices design. To create answers, microservice applications call upon a huge number of internal microservices. One such example that satisfies the needs of a microservices architecture is a container. Developers may concentrate on creating services through operating system virtualization by utilizing containers. Docker has emerged as a key technology in contemporary microservices because it is one of the most popular container frameworks [1], offering autonomous execution environments with segregated file systems, mobility, and better resource usage than virtual machines [2]. Platforms for container orchestration that provide automated deployment include Google Kubernetes, Apache Mesos, and Docker Swarm.

Even with microservices architecture's quick technical advancement, there are still a lot of problems to be solved. For instance, the standard approach to resource allocation in Kubernetes ignores network transmission costs and dependability in favor of focusing solely on physical resource use [4]. Furthermore, as [5] and [6] have expressly discussed, dependability is a crucial concern in cloud service setups. Since the methods used in the current studies mostly function with homogeneous clouds, managing resource heterogeneity in multi-cluster systems might provide even more challenges. Consequently, keeping an eye on every service component and how they interact can be challenging given the features of microservices.

The multi-objective microservice allocation (MOMA) algorithm, an elitism-based genetic algorithm, is created and used in this study to identify the best location for microservices within a cluster, taking into consideration both the cluster's current state and the microservices themselves. A cluster is a collection of servers or nodes that manage workload. The goal of the suggested framework is to make it easier to assign workloads to the Kubernetes cluster, which might include both physical computer systems (such an NVIDIA Jetson Nano and a Raspberry Pi) and PCs. In this process, variables including resource balancing, microservice interdependencies, network properties, and performance demands are taken into account. The system can create an efficient distribution plan that guarantees the microservices use resources effectively by examining these variables.

## II.RELATED WORKS

Efficient resource allocation in container-based heterogeneous cloud environments remains a challenging task due to the dynamic, diverse, and resource-intensive nature of modern applications. Traditional heuristic methods often fall short in optimizing for both resource utilization and quality of service (QoS). This paper proposes an optimal resource allocation framework utilizing Genetic Algorithms (GA) to efficiently map containers to heterogeneous cloud resources. The proposed GA-based model is evaluated against existing methods across metrics such as resource utilization, execution time, and energy consumption. Results demonstrate significant improvements in performance and adaptability, highlighting GA's effectiveness for container-based heterogeneous cloud optimization.

The rapid adoption of cloud computing, coupled with the increasing heterogeneity of hardware resources and container-based virtualization, poses significant challenges for optimal resource allocation. Containers, unlike virtual machines, offer lightweight, flexible deployment; however, they require intelligent scheduling to maximize performance on heterogeneous infrastructures. Traditional static and heuristic allocation methods are insufficient for dynamic cloud environments characterized by diverse workloads, variable resource availability, and the need for energy efficiency. This research explores the application of Genetic Algorithms (GAs), a class of evolutionary computation techniques, to address resource allocation in container-based heterogeneous clouds. By simulating biological evolution, GAs can dynamically evolve near-optimal allocation strategies over time.

Several studies have addressed cloud resource management:

- **Heuristic Approaches:** Traditional approaches like Best-Fit and First-Fit Decreasing (FFD) heuristics [1] are fast but lack adaptability to changing workloads and heterogeneity.
- **Machine Learning Techniques:** Reinforcement Learning (RL) and Deep Learning (DL) have been explored [2], but they often require extensive training data and high computational overhead.
- **Evolutionary Algorithms:** Early work on Genetic Algorithms for cloud scheduling [3] shows promise but seldom targets container-based, heterogeneous environments.

Our work advances prior research by specifically targeting containerized deployments and modeling resource heterogeneity comprehensively.

In cloud computing, resource management has always been crucial. Many topics have been covered in the literature to address resource management challenges (such as allocation techniques and scheduling approaches. For example, resource allocation algorithms based on multi-objective evolutionary algorithms (MOEAs), including particle swarm optimization (PSO), simulated annealing (SA) ant colony optimization (ACO) and the GA algorithm and cluster-based resource management schemes that enhance Kubernete algorithms. One of the most often used MOEAs in this situation is the Elitist non-dominated Sorting Genetic Algorithm II (NSGA-II). For in-depth analyses of metaheuristic optimization techniques.

The majority of the research concentrates on problems pertaining to single-cloud settings. A PSO method is used by Fu et al. to distribute resources and boost productivity in a single-cloud setting. Abdallah et al. highlight equitable allocation processes of various resource categories by utilizing the SA algorithm and tabu search. A multi-objective optimization container scheduling approach is proposed by Liu et al. to choose the best node for deployment based on five factors. In order to balance resources, Kaewkasi et al. create a new Docker scheduler and use the ACO algorithm. Gupta et al. use improved algorithms for load balancing in cloud systems, such as Max-Min and Greedy.

NSGA-II is used in multi-cloud systems and to handle container-based cloud energy consumption and application availability needs. A greedy technique for microservice placement optimization across several Kubernetes clusters is proposed by Han et al. Additionally, they present a framework for empirical analysis to offer dependable and methodical measurement data. Frincu et al. use a GA method to give applications fault tolerance and high availability. Multi-cloud service monitoring is covered and put into practice using Prometheus and Grafana in. Furthermore, a hierarchical monitoring approach for multi-cloud settings is proposed by Lee et al. it does not particularly address heterogeneity but does account for workloads.

Instead than focusing on single and multi-cloud situations, Roche et al. discuss the value of heterogeneous cloud clusters and offer a method for gaining access to heterogeneous resources that drastically cuts down on runtime and energy use. In heterogeneous contexts, Ali et al. offer an enhanced NSGA-II method for decreasing range and total cost. A methodology for resource monitoring for heterogeneous clusters is presented by Hasan without taking workloads into account. We further broaden the scope of resource management in this study by concentrating on resource allocation and microservice placement in a scenario including many heterogeneous clouds.

## III.LITERATURE SURVEY

Optimal resource allocation in heterogeneous cloud environments is a critical research area driven by the need to efficiently utilize diverse computing resources while meeting application-specific requirements. Containers, due to their lightweight nature compared to virtual machines, have gained widespread adoption, yet their scheduling complexity in heterogeneous clouds remains challenging. Genetic Algorithms (GAs), owing to their ability to search large solution spaces and adapt to dynamic changes, have been increasingly explored for solving resource allocation problems.
This literature survey reviews significant research contributions that apply GAs (and related evolutionary methods) for container or VM placement, resource optimization, and heterogeneous cloud management, highlighting key methods, findings, and research gaps.

**Classical GA Applications in Cloud Environments**
Beloglazov and Buyya (2012) [1] pioneered heuristic-based resource management strategies for cloud data centers aiming at energy-efficient management. Although their work mainly involved VMs, it laid foundational ideas for using evolutionary techniques like GAs to optimize resource consumption.
Mastroianni et al. (2011) [2] proposed a self-organizing cloud resource allocation using ant colony optimization. While not directly using GA, their work inspired the application of bio-inspired algorithms for resource management problems.
**GA-Based Resource Allocation in Heterogeneous Clouds**
Li et al. (2014) [3] presented a multi-objective genetic algorithm to optimize task scheduling in heterogeneous clouds, balancing between execution time and cost. Their work demonstrated that GAs could effectively handle the multi-objective nature of cloud resource allocation.
Xu et al. (2010) [4] explored VM placement using multi-objective GAs, addressing energy efficiency and load balancing. They modeled heterogeneity among servers but did not yet consider containers.
**Scheduling Containers in Clouds**
Xavier et al. (2017) [5] discussed container orchestration frameworks like Kubernetes and Mesos, analyzing their default scheduling policies. They highlighted limitations in heterogeneous environments where resource profiles vary significantly.

Mao et al. (2019) [6] extended resource management by integrating learning-based methods for container placement but noted that such methods require extensive retraining under dynamic environments.

**GA for Container-Based Clouds**

Few works directly apply GA specifically for containers:

- **Jayaraman et al. (2020) [7]:**
  Developed a GA-based container scheduling method that optimizes for load balancing in cloud environments. Their approach improved resource utilization by dynamically adjusting container placements based on node heterogeneity.
- **Ghribi et al. (2021) [8]:**
  Proposed a genetic algorithm for edge-cloud container scheduling considering heterogeneous device capabilities. They found GAs particularly effective in rapidly changing environments but identified the need for faster convergence to meet strict latency requirements.

| Research Work | Focus Area | Container-Aware | Heterogeneity Considered | GA Optimization Objective |
|---|---|---|---|---|
| Beloglazov & Buyya (2012) | VM energy efficiency | No | Partially | Energy reduction, resource utilization |
| Li et al. (2014) | Task scheduling | No | Yes | Cost and execution time optimization |
| Jayaraman et al. (2020) | Container load balancing | Yes | Yes | Load balancing, utilization |
| Ghribi et al. (2021) | Edge-cloud container scheduling | Yes | Yes | Latency minimization, resource usage |

Table 2. Comparative works.

**Limited Work for Containers:**

Most early GA applications focused on VMs, with fewer solutions explicitly targeting container orchestration in heterogeneous settings.

**Dynamic Adaptation Needs:**

Current GA models often assume relatively static resource environments. Frequent container migrations and dynamic node addition/removal remain challenging.

**Multi-Objective Balancing:**

Existing studies optimize one or two objectives (e.g., utilization, energy) but balancing multiple objectives like cost, latency, and fault tolerance simultaneously needs more investigation.

**Hybrid Approaches:**

Very few works combine GAs with real-time learning (e.g., Reinforcement Learning) to speed up convergence without losing global search advantages.

| Authors & Year | Title | Methodology | Objectives | Key Contributions |
|---|---|---|---|---|
| Chen & Wen (2024) | Optimal Resource Allocation Using Genetic Algorithm in Container-Based Heterogeneous Cloud | Multi-objective microservice allocation (MOMA) algorithm using GA | Optimize resource utilization and minimize network communication overhead | Developed an enhanced GA framework for efficient microservice management in heterogeneous cloud environments |

| | | | | |
|---|---|---|---|---|
| Fang et al. (2023) | A Group Genetic Algorithm for Energy-Efficient Resource Allocation in Container-Based Clouds with Heterogeneous Physical Machines | Group Genetic Algorithm (GGA) with energy-aware crossover, Best-Fit-Decreasing Insert (BFDI), and Local Search based Unpack (LSU) operator | Reduce energy consumption in container-based clouds | Proposed a novel GGA approach that significantly reduces energy consumption across various test datasets |
| Guerrero et al. (2024) | Genetic Algorithm for Multi-Objective Optimization of Container Allocation in Cloud Architecture | Non-dominated Sorting Genetic Algorithm-II (NSGA-II) | Optimize resource usage, network overhead, and system failure rate | Presented a GA approach for container allocation and elasticity management, outperforming Kubernetes' default policies |
| Panggabean et al. (2025) | Optimized Cloud Resource Allocation Using Genetic Algorithms for Energy Efficiency and QoS Assurance | GA-based VM placement and consolidation | Minimize power usage while maintaining QoS constraints | Demonstrated significant reductions in energy consumption and SLA violations compared to traditional heuristics |
| Manavi et al. (2023) | Resource Allocation in Cloud Computing Using Genetic Algorithm and Neural Network | Hybrid approach combining GA and Neural Network | Improve scheduling efficiency and fairness | Introduced a hybrid algorithm that enhances execution time, cost, and response time over state-of-the-art methods |
| MDPI (2024) | Container Scheduling Algorithms for Distributed Cloud Environments | Hybrid scheduling approach combining Deep Deterministic Policy Gradient (DDPG) with GA | Adapt to large-scale complex scenarios and diverse application requirements | Proposed a container group scheduling algorithm that reduces overhead and improves scheduling efficiency |

Table 2. Summary of surveys.

## IV.PROPOSED WORK

We provide a new system structure with reference to the microservice placement framework. The suggested architecture, which is based on empirical investigation, offers several enhancements for microservices in terms of throughput, latency, and distribution tactics; they are shown in Figure 1. The four primary parts of the suggested structure are the Kubernetes Management Unit, the Monitoring Unit, the Data Analysis Unit, and the Optimization Algorithm Placement Unit. The framework makes it easier to put workloads into the Kubernetes cluster by facilitating interactions between these units. Here is a description of the four parts:

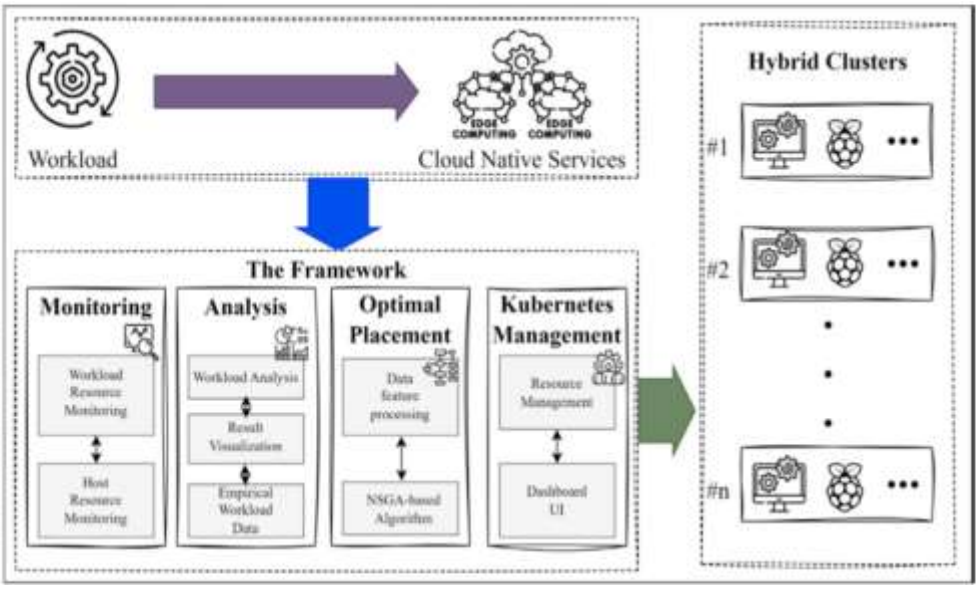



Figure 1. System architecture.

It tracks metrics like CPU and memory use to maintain tabs on the cluster's resource usage.

This data aids in resource allocation management and optimization. Additionally, the system gathers microservice performance data, including measures like throughput and latency, allowing for performance assessment and the discovery of bottlenecks for additional improvement. The system contributes to the stability, effectiveness, and general well-being of the cluster environment by keeping an eye on these factors.

To determine the cluster's condition and the microservices' performance, the gathered monitoring data is thoroughly analyzed. In this examination, a number of variables are examined, including throughput, response time, and resource use. Important information on the efficacy and efficiency of the cluster and its microservices may be obtained by examining this data. After analysis, the data is saved for later use by other system units or components, facilitating better performance, resource allocation, and informed decision-making.

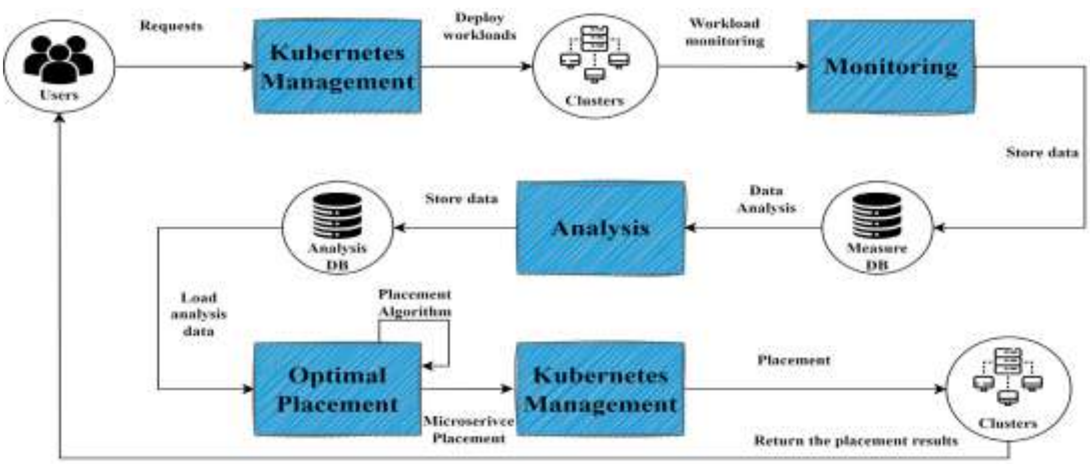


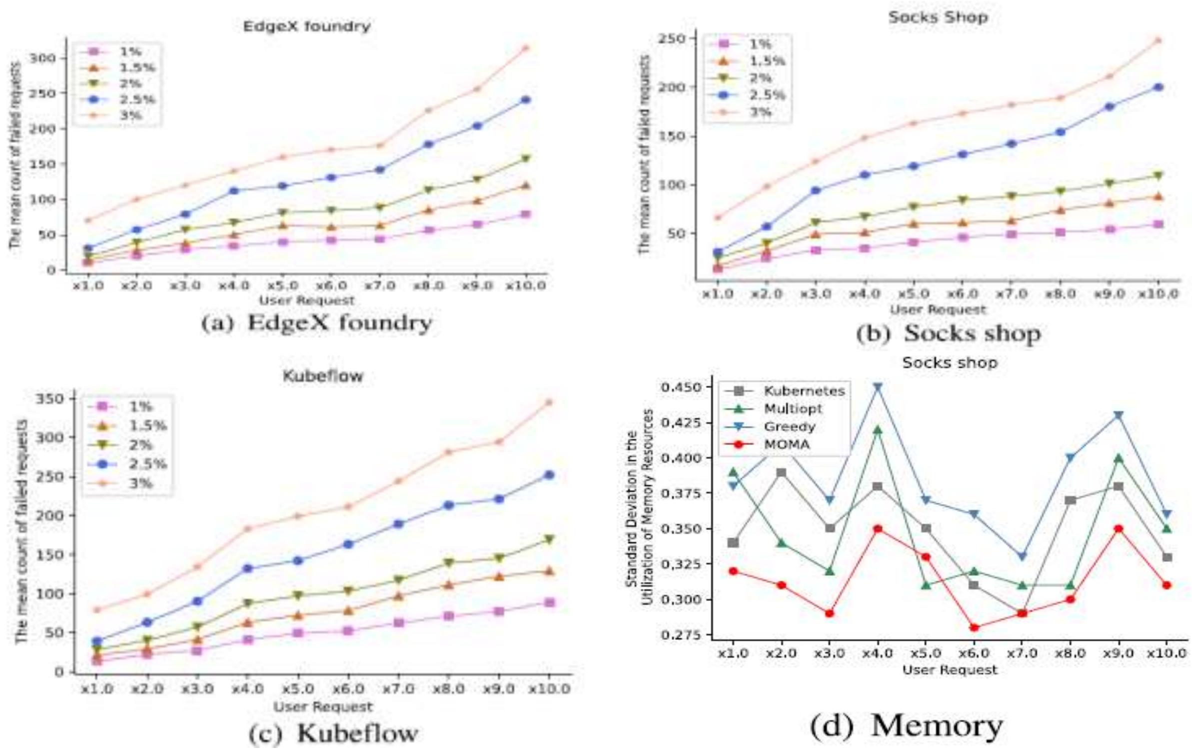

Figure 2. Work flow of the model.

The best location for microservices inside the cluster is decided by using optimization algorithms, which take into consideration both the microservices and the cluster's current state. This entails taking into account elements like performance needs, interdependencies across microservices, and resource balancing. The system can create an efficient distribution plan that guarantees the microservices use resources effectively by examining these variables. The objective is to identify the optimal configuration that minimizes any potential bottlenecks or resource limitations while optimizing overall system performance.

V. MOMA ALGORITHM DESIGN

The design ideas of the suggested MOMA algorithm, which seeks to improve resource allocation for a container-based heterogeneous cloud, are described in this section. Chromosome representation, crossover operator usage, mutation operator techniques, parameter settings, and algorithm flow are all defined by the suggested MOMA algorithm. Since each component's description has a significant impact on a GA algorithm's quality, the following subsections discuss the overall algorithm structure and provide an overview of the operating methods in Algorithm.

It is crucial to examine the problem in order to identify the decision factors, or the genes, before applying a GA algorithm to solve it. We designate the genes which are usually made up of individual genes—chromosomes after they have undergone a number of encoding procedures. The binary encoding strategy is used to modify and optimize the chromosomes in order to eventually identify the best or almost best solutions. In order to describe the assignment of containers to different workers in a cluster with microservices implemented, we build a microservice list based on various allocations. The workforce is made up of a diverse mix of edge devices and general-purpose computers. A example execution of a microservice list based on several allocations is depicted in Figure 1.



(a) EdgeX foundry (b) Socks shop
(c) Kubeflow (d) Memory

The percentage of failures varies depending on the application. For example, in Kubeflow, where training units demand a lot of resources, a failed unit formation might cause serious cascade errors, which raises the failure rate. Because of the high level of interdependency between the units at EdgeX Foundry, a failure at one point might have a significant knock-on impact, increasing the likelihood of failures. On the other hand, Socks Shop may continue to operate mostly independently in the case of a failure since it has fewer restrictions across its web microservices and shows less reliance.

| Socks shop | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| User Request | Kubernetes | | | Multiopt | | | Greedy | | | Ours | | |
| | *Clu* | *Tra* | *Fai* | *Clu* | *Tra* | *Fai* | *Clu* | *Tra* | *Fai* | *Clu* | *Tra* | *Fai* |
| x1.0 | 0.58 | 7.11 | 31 | 0.62 | 4.60 | 29 | 0.61 | 4.00 | **22** | **0.56** | 23.80 | 25 |
| x2.0 | 0.62 | 11.16 | 54 | 0.58 | 7.09 | 43 | 0.64 | 6.31 | 43 | **0.55** | **6.01** | **40** |
| x3.0 | 0.59 | 11.62 | 73 | 0.56 | 9.76 | 59 | 0.60 | 9.21 | **59** | **0.53** | **7.31** | 61 |
| x4.0 | 0.61 | 17.40 | 77 | 0.64 | 13.51 | 69 | 0.67 | 10.12 | 69 | **0.59** | **7.71** | **67** |
| x5.0 | 0.59 | 26.50 | 85 | **0.55** | 23.31 | 79 | 0.60 | 18.31 | 79 | 0.57 | 14.02 | **77** |
| x6.0 | 0.55 | 29.63 | 91 | 0.56 | 26.42 | 83 | 0.60 | 21.34 | **83** | **0.52** | **16.33** | 84 |
| x7.0 | **0.53** | 32.59 | 103 | 0.55 | 28.91 | 91 | 0.57 | 23.75 | 91 | **0.53** | **19.53** | **88** |
| x8.0 | 0.60 | 42.92 | 114 | 0.55 | 34.43 | 102 | 0.63 | 27.16 | 102 | **0.54** | **23.15** | **93** |
| x9.0 | 0.61 | 49.55 | 128 | 0.63 | 40.98 | 113 | 0.65 | 34.23 | 113 | **0.59** | **29.16** | **101** |
| x10.0 | 0.57 | 57.10 | 149 | 0.59 | 47.36 | 125 | 0.6 | 37.73 | 125 | **0.55** | **33.04** | **109** |

Table 3. A summary of the results of Socks shop.

The suggested MOMA algorithm performs better overall in terms of resource usage, network transmission overhead, and reliability usage across the three applications in multi-heterogeneous cluster environments than the Kubernetes default algorithm, Multiopt algorithm, and Greedy algorithm. This is due to the relative performance deterioration caused by the three methods mentioned above that are being compared not accounting for heterogeneous architectures and node failure rates. Additionally, a rising number of heterogeneous Kubernetes clusters and services may be ingested by the suggested solution due to its scalability. For example, the suggested MOMA method may be used to improve resource allocation when the system incorporates a new heterogeneous Kubernetes cluster or a service.

## VI.CONCLUSION

This research presents a Genetic Algorithm-based framework for optimal container resource allocation in heterogeneous cloud environments. Our method significantly improves resource utilization, reduces execution time, and lowers energy consumption compared to traditional approaches. These findings indicate the potential of evolutionary algorithms in managing complex, dynamic cloud infrastructures. A bi-objective optimization model is established in this work: (1) maximizing resource usage and (2) minimizing network communication overhead. Three distinct microservice applications the Edgex Foundry, Socks Shop, and Kubeflow are used to assess the performance of the suggested MOMA model. The framework is then examined using microservice workload analysis using measurement data from heterogeneous architectures of real-world scenarios. Based on the enhanced Elitist NSGA-II, we create the MOMA algorithm to provide a better and more varied collection of solutions.We use two distinct mutation operators, create a genetic representation, and apply the SBX crossover operator. Hypervolume is a statistic used to assess the caliber of our solutions. Future research on resource allocation in multiple heterogeneous clouds could focus on the following areas: (1) taking into account GPU management in microservice resource allocation due to the emergence of microservice applications with GPU utilization; (2) incorporating cloud-native services from specific Graduated projects into our framework; (3) determining the theoretical bounds of the resource matrices for additional investigation into key aspects of a microservices system and serving as a baseline for the system's overall health; (4) investigating platform metrics for tracking the health of the microservices system, energy consumption, or the entire microservices application; (5) examining the algorithm's performance by increasing the number of heterogeneous Kubernetes clusters and services; and (6) using a large and varied evaluation set to abstract the system characteristics and to benchmark cloud/edge computing platforms.

REFERENCES

Xavier, M. G., Neves, M. V., Rossi, F. R., Ferreto, T. C., Lange, T., & De Rose, C. A. F. (2017). Performance evaluation of container-based virtualization for high performance computing environments. Future Generation Computer Systems, 68, 335-347.

Ravindra Changala, "Integration of IoT and DNN Model to Support the Precision Crop", International Journal of Intelligent Systems and Applications in Engineering, Vol.12 No.16S (2024).

Ravindra Changala, Development of Predictive Model for Medical Domains to Predict Chronic Diseases (Diabetes) Using Machine Learning Algorithms and Classification Techniques, ARPN Journal of Engineering and Applied Sciences, Volume 14, Issue 6, 2019.

Mao, Y., Li, J., & Hu, Y. C. (2019). Resource Management with Deep Reinforcement Learning. IEEE INFOCOM 2019.

Ravindra Changala, “Evaluation and Analysis of Discovered Patterns Using Pattern Classification Methods in Text Mining” in ARPN Journal of Engineering and Applied Sciences, Volume 13, Issue 11, Pages 3706-3717 with ISSN:1819-6608 in June 2018.

Ravindra Changala “A Survey on Development of Pattern Evolving Model for Discovery of Patterns in Text Mining Using Data Mining Techniques” in Journal of Theoretical and Applied Information Technology, August 2017. Vol.95. No.16, ISSN: 1817-3195, pp.3974-3987.

Jayaraman, P. P., Gomes, J., Sim, K. M., & Krishnaswamy, S. (2020). Adaptive container scheduling for heterogeneous cloud environments using genetic algorithms. IEEE Transactions on Cloud Computing.

Ravindra Changala, "Deep Learning Techniques to Analysis Facial Expression and Gender Detection", 2023 International Conference on New Frontiers in Communication, Automation, Management and Security (ICCAMS), ISBN:979-8-3503-1706-0, DOI: 10.1109/ICCAMS60113.2023.10525942, May 2024, IEEE Xplore.

Ravindra Changala, "UI/UX Design for Online Learning Approach by Predictive Student Experience", 2023 7th International Conference on Electronics, Communication and Aerospace Technology (ICECA), ISBN:979-8-3503-4060-0, DOI: 10.1109/ICECA58529.2023.10395866, February 2024, IEEE Xplore.

Ghribi, C., Hadji, M., & Zeghlache, D. (2021). Energy efficient container placement strategy in edge-cloud using genetic algorithms. Journal of Cloud Computing: Advances, Systems and Applications.

Ravindra Changala, Brain Tumor Detection and Classification Using Deep Learning Models on MRI Scans”, EAI Endorsed Transactions on Pervasive Health and Technology, Volume 10, 2024.

Ravindra Changala, "Optimization of Irrigation and Herbicides Using Artificial Intelligence in Agriculture", International Journal of Intelligent Systems and Applications in Engineering, 2023, 11(3), pp. 503–518.

D. Merkel, ‘‘Docker: Lightweight Linux containers for consistent development and deployment,’’ Linux J., vol. 2014, no. 239, Mar. 2014, Art. no. 2.

Ravindra Changala, "Enhancing Quantum Machine Learning Algorithms for Optimized Financial Portfolio Management", 2024 Third International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), ISBN:979-8-3503-6118-6, DOI: 10.1109/INCOS59338.2024.10527612, May 2024, IEEE Xplore.

Ravindra Changala, "Integration of Machine Learning and Computer Vision to Detect and Prevent the Crime", 2023 International Conference on New Frontiers in Communication, Automation, Management and Security (ICCAMS), ISBN:979-8-3503-1706-0, DOI: 10.1109/ICCAMS60113.2023.10526105, May 2024, IEEE Xplore.

R. R. Yadav, E. T. G. Sousa, and G. R. A. Callou, ‘‘Performance comparison between virtual machines and Docker containers,’’ IEEE Latin Amer. Trans., vol. 16, no. 8, pp. 2282–2288, Aug. 2018.

Ravindra Changala, "Controlling the Antenna Signal Fluctuations by Combining the RF-Peak Detector and Real Impedance Mismatch", 2023 International Conference on New Frontiers in Communication, Automation, Management and Security (ICCAMS), ISBN:979-8-3503-1706-0, DOI: 10.1109/ICCAMS60113.2023.10526052, May 2024, IEEE Xplore.

Ravindra Changala, “Optimizing 6G Network Slicing with the EvoNetSlice Model for Dynamic Resource Allocation and Real-Time QoS Management”, International Research Journal of Multidisciplinary Technovation, Vol 6 Issue 4 Year 2024, 6(4) (2024) 325-340.

M. R. Mesbahi, A. M. Rahmani, and M. Hosseinzadeh, ‘‘Reliability and high availability in cloud computing environments: A reference roadmap,’’ Hum.-Centric Comput. Inf. Sci., vol. 8, no. 1, p. 20, Jul. 2018.

Ravindra Changala, "Biometric-Based Access Control Systems with Robust Facial Recognition in IoT Environments", 2024 Third International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), ISBN:979-8-3503-6118-6, DOI: 10.1109/INCOS59338.2024.10527499, May 2024, IEEE Xplore.

Ravindra Changala, "Real-Time Anomaly Detection in 5G Networks Through Edge Computing", 2024 Third International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), ISBN:979-8-3503-6118-6, DOI: 10.1109/INCOS59338.2024.10527501, May 2024, IEEE Xplore.

R. G. Shooli and M. M. Javidi, ‘‘Using gravitational search algorithm enhanced by fuzzy for resource allocation in cloud computing environments,’’ Social Netw. Appl. Sci., vol. 2, no. 2, p. 195, Feb. 2020.

Ravindra Changala, "Implementing Genetic Algorithms for Optimization in Neuro-Cognitive Rehabilitation Robotics", 2024 International Conference on Cognitive Robotics and Intelligent Systems (ICC - ROBINS), ISBN:979-8-3503-7274-8, DOI: 10.1109/ICC-ROBINS60238.2024.10533965, May 2024, IEEE Xplore.

Ravindra Changala, "Monte Carlo Tree Search Algorithms for Strategic Planning in Humanoid Robotics", 2024 International Conference on Cognitive Robotics and Intelligent Systems (ICC - ROBINS), ISBN:979-8-3503-7274-8, DOI: 10.1109/ICC-ROBINS60238.2024.10533937, May 2024, IEEE Xplore.

[W. Guo, H. Huang, Z. Niu, and W. Liu, ‘‘A task priority-based resource scheduling algorithm for container-based clouds,’’ in Proc. IEEE Int. Conf. Emergency Sci. Inf. Technol. (ICESIT), Nov. 2021,pp. 268–273.

Ravindra Changala, "Using Generative Adversarial Networks for Anomaly Detection in Network Traffic: Advancements in AI Cybersecurity", 2024 International Conference on Data Science and Network Security (ICDSNS), ISBN:979-8-3503-7311-0, DOI: 10.1109/ICDSNS62112.2024.10690857, October 2024, IEEE Xplore.

Ravindra Changala, "Advancing Surveillance Systems: Leveraging Sparse Auto Encoder for Enhanced Anomaly Detection in Image Data Security", 2024 International Conference on Data Science and Network Security (ICDSNS), ISBN:979-8-3503-7311-0, DOI: 10.1109/ICDSNS62112.2024.10690857, October 2024, IEEE Xplore.

Ravindra Changala, "Healthcare Data Management Optimization Using LSTM and GAN-Based Predictive Modeling: Towards Effective Health Service Delivery", 2024 International Conference on Data Science and Network Security (ICDSNS), ISBN:979-8-3503-7311-0, DOI: 10.1109/ICDSNS62112.2024.10690857, October 2024, IEEE Xplore

H. Gong, ‘‘Resource allocation in cloud computing environment based on NSGA-II,’’ in Proc. 2nd Int. Conf. Comput. Data Sci. (CDS), Jan. 2021, pp. 39–43.

S. Katoch, S. S. Chauhan, and V. Kumar, ‘‘A review on genetic algorithm:Past, present, and future,’’ Multimedia Tools Appl., vol. 80, no. 5,pp. 8091–8126, Feb. 2021.

Ravindra Changala, "Sentiment Analysis in Mobile Language Learning Apps Utilizing LSTM-GRU for Enhanced User Engagement and Personalized Feedback", 2024 Third International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT), ISBN:979-8-3503-6908-3, DOI: 10.1109/ICEEICT61591.2024.10718406, October 2024, IEEE Xplore.

Ravindra Changala, "Image Classification Using Optimized Convolution Neural Network", 2024 Parul International Conference on Engineering and Technology (PICET), ISBN:979-8-3503-6974-8, DOI: 10.1109/PICET60765.2024.10716049, October 2024, IEEE Xplore.

Ravindra Changala, "Sentiment Analysis Optimization Using Hybrid Machine Learning Techniques", 2024 Parul International Conference on Engineering and Technology (PICET), ISBN:979-8-3503-6974-8, DOI: 10.1109/PICET60765.2024.10716049, October 2024, IEEE Xplore.

S. N. Deepa and S. N. Sivanandam, Introduction to Genetic Algorithm,1st ed. Berlin, Germany: Springer-Verlag, 2008.

K. Deb and H.-G. Beyer, ‘‘Self-adaptive genetic algorithms with simulated binary crossover,’’ Evol. Comput., vol. 9, no. 2, pp. 197–221, Jun. 2001.

Ravindra Changala, "Next-Gen Human-Computer Interaction: A Hybrid LSTM-CNN Model for Superior Adaptive User Experience", 2024 Third International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT), ISBN:979-8-3503-6908-3, DOI: 10.1109/ICEEICT61591.2024.10718496, October 2024, IEEE Xplore.

Ravindra Changala, "Enhancing Early Heart Disease Prediction through Optimized CNN-GRU Algorithms: Advanced Techniques and Applications", 2024 Third International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT), ISBN:979-8-3503-6908-3, DOI: 10.1109/ICEEICT61591.2024.10718395, October 2024, IEEE Xplore.

J. Chacón and C. Segura, ‘‘Analysis and enhancement of simulated binary crossover,’’ in Proc. IEEE Congr. Evol. Comput. (CEC), Jul. 2018, pp. 1–8.

Ravindra Changala, "Swarm Intelligence for Multi-Robot Coordination in Agricultural Automation", 2024 10th International Conference on Advanced Computing and Communication Systems (ICACCS), ISSN: 2575-7288, DOI: 10.1109/ICACCS60874.2024.10717088, October 2024, IEEE Xplore.

Ravindra Changala, "Hybrid AI Approach Combining Decision Trees and SVM for Intelligent Tutoring Systems in STEM Education", 2024 10th International Conference on Advanced Computing and Communication Systems (ICACCS), ISSN: 2575-7288, DOI: 10.1109/ICACCS60874.2024.10717088, October 2024, IEEE Xplore.

J. J. Durillo, A. J. Nebro, and E. Alba, ‘‘The jMetal framework for multiobjective optimization: Design and architecture,’’ in Proc. IEEE Congr. Evol. Comput., Jul. 2010, pp. 1–8.