International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

EFFICIENT RESOURCE ALLOCATION IN HETEROGENEOUS CLOUD USING GENETIC ALGORITHMS AND CONTAINERIZATION

Sania Masood[1], Dr. Mohd Umar Farooq[2]
[1]PG Scholar, Department of CSE, Shadan Women's College of Engineering and Technology, Hyderabad,
Saniamasood44@gmail.com
[2]Professor, Department of CSE, Shadan Women's College of Engineering and Technology
umarfarooq.mohd@gmail.com

Received: 24-07-2025 Accepted: 28-08-2025 Published: 05-09-2025

ABSTRACT

The difficult task of resource configuration optimization for microservice management in heterogeneous cloud systems is addressed in this study. Resource utilization and network communication overhead, two crucial aspects of resource allocation, are used to guide the development of an improved framework, the multi-objective microservice allocation (MOMA) algorithm, which formulates the effective resource management of cloud microservice resources as a constrained optimization problem. Because workloads are dynamic and resource demands vary, it is still very difficult to allocate resources efficiently in container-based heterogeneous cloud environments. In this research, a novel method for resource allocation optimization using a Genetic Algorithm (GA) in such contexts is presented. Through the intelligent distribution of network, storage, and compute resources among several heterogeneous nodes, the suggested approach seeks to improve the performance, scalability, and dependability of cloud services. The GA dynamically creates solutions that balance load distribution, limit energy consumption, and maximize resource usage by modelling the resource allocation problem as a multi-objective optimization task. Simulation findings illustrate the usefulness of the suggested strategy, showing that it can surpass conventional allocation methods in terms of system stability and efficiency.

1. INTRODUCTION

MS apps call upon several internal MSs to build replies as a result of the growing popularity of MSs architecture in recent years, which divides large-scale applications into smaller, independent components. One such example that satisfies the requirements of MSs architecture is a container. Through operating system virtualization, developers may concentrate on creating services by utilizing containers. Docker has emerged as a key technology in contemporary MSs because it is one of the most popular container frameworks, offering segregated file systems, independent execution environments, portability & better resource usage than virtual machines. Platforms such as Docker Swarm, Apache Mesos & Google Kubernetes are examples of container orchestration systems that provide automated deployment.

Even with MSs architecture's quick technological advancement, there are still a lot of problems to be solved. For instance, Kubernetes' default resource allocation approach ignores the costs & dependability of network transmission in favor of focusing solely on physical resource consumption. Furthermore, as has been particularly discussed in & dependability is a crucial concern in cloud service setups. Managing resource heterogeneity in multi-cluster settings may present even more challenges, since the methods used in the current works mostly work across homogenous clouds. Consequently, given the features of MSs, it might be difficult to keep an eye on every service component & how they interact.

In order to assess the performance of the application, monitoring metrics, resource metrics (such as CPU & memory utilization) and platform metrics (such as the number of requests per second, the distribution of time needed for each request & the average execution time for the queries) may be constructed for individual services. The optimization objective in this work is resource utilization. In order to manage resources efficiently & improve performance & service dependability, resource measurements such as CPU utilization & memory use are used.

Finding polynomial-time complexity techniques is still a challenge for container resource allocation, which is acknowledged as an NP-hard problem. To find the best answers to these resource allocation issues, many academics use meta-heuristic algorithms. Every heuristic algorithm has a unique set of advantages & disadvantages. We can identify the best algorithm for a certain situation by using contextual analysis. For example, iterative experimentation yields findings & evaluates many algorithms to suggest the most appropriate one for a given situation. Non-Dominated Sorting GA II (NSGA-II) is one of the most popular GAs & GAs are seen to be useful in solving such issues. Therefore, in order to overcome these issues & progress the area, more research is required.

The MOMA algorithm, an elitism-based GA, is created & applied in this study to ascertain the best location for MSs within the cluster, considering both the cluster's current state & the MSs themselves. A cluster is a collection of servers or nodes that take part in workload

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

management. Workloads can be placed into the Kubernetes cluster with the help of the suggested framework, which can include both physical computer systems (such an NVIDIA Jetson Nano & a Raspberry Pi) & a PC.

OBJECTIVE

This study's main goal as follows : By creating an improved GA algorithm with two objective models, taking into account resource usage & NCV, and empirically adjusting parameter settings using real-world data, this work tackles the heterogeneity challenge of MS resource allocation & optimizes resource management in HCE. The suggested approach simplifies CS deployment, addresses the heterogeneity elements of cluster monitoring & resource selection, and expedites workload management & analysis in a heterogeneous cloud environment. A thorough assessment of the suggested framework using actual datasets is provided. According to the experimental findings, the suggested framework performs better than the current approaches in terms of network overhead, network reliability & resource balancing (RB).

PROBLEM STATEMENT

The default resource allocation technique ignores the costs & dependability of network transmission in favor of focusing simply on physical resource consumption. Reliability has also been expressly addressed as a crucial issue in cloud service systems. Managing resource heterogeneity in multi-cluster settings may present even more challenges, since the methods used in the current works mostly work across homogenous clouds. Consequently, given the nature of services, it might be difficult to keep an eye on every service component & how they interact.

EXISTING SYSTEM

The default resource allocation technique ignores the costs & dependability of network transmission in favor of focusing simply on physical resource consumption. Reliability has also been expressly addressed as a crucial issue in cloud service systems. Managing resource heterogeneity in multi-cluster settings may present even more challenges, since the methods used in the current works mostly work across homogenous clouds. Consequently, given the nature of services, it might be difficult to keep an eye on every service component & how they interact.

Disadvantages of Existing System

- Only one environment is supported; resource allocation & NCVs are not optimized.
- Less storage space.
- Less secure.
- Time-consuming data retrieval procedure.

PROPOSED SYSTEM

The suggested methodology facilitates workload monitoring & analysis in a heterogeneous cloud environment & makes CS deployment easier. The efficiency of the suggested method & current algorithms on real-world datasets is thoroughly compared, with an emphasis on RB, network overhead & network dependability. According to experimental findings, the suggested algorithm greatly increases dependability, lowers network transmission overhead & maximizes resource use

Following are the advantages of proposed system:

- Facilitates a heterogeneous cloud environment with multiple objectives.
- Offers optimization for NCV as well as resource allocation.
- Greater storage capacity.
- More security.
- The procedure of retrieving data takes less time.

2. RELATED WORK

Aiming at the huge and dynamic resource allocation of most data centers at present, research on cloud computing resource allocation models based on artificial intelligence technology. Particle Optimization algorithm and cloud computing resource model are designed to realize the division of tasks and allocate cloud computing resources; in addition, a resource allocation scheduling framework is provided to collect tasks on user terminals. The service scheduling module calculates the resource loss, realizes the scheduling of cloud tasks, receives tasks submitted by users, and calculates the priority of the task. Finally, the global optimal solution is given. Using the C++ development simulation platform to simulate different numbers of server data centers, the results show that this resource allocation algorithm can solve the problem, thereby improving the allocation efficiency of the algorithm. There are many problems in cloud computing that need to be solved effectively, such as resource scheduling. With the explosion of data, more and more applications involve data collection and data processing, such as computationally intensive applications. Emerging technologies such as electronic health, visual text translation, virtual/augmented reality, and high-definition video have a huge impact on our daily lives, and can all be regarded as computationally intensive applications. However, there are still many obstacles to the development of mobile devices, such as security issues and resource management issues**. [21]**

The advent of big data has revolutionized various industries, enabling organizations to make data- driven decisions and gain valuable insights. However, the sheer volume, velocity, and variety of big data pose

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

significant challenges in ensuring the reliability and resilience of big data analytics pipelines. In this context, optimization techniques play a crucial role in enhancing the reliability and resilience of big data analytics. This paper provides a comprehensive survey of optimization techniques for reliable and resilient big data analytics. The paper first discusses the motivation for optimizing big data analytics for reliability and resilience. Then, it presents a detailed overview of various optimization techniques, including resource optimization, data partitioning, data compression, load balancing, and fault detection and tolerance. Finally, the paper discusses the applications of optimization techniques in various big data analytics domains, such as real-time analytics, fraud detection, recommendation systems, predictive analytics, and risk management. The proliferation of big data has transformed various industries, empowering organizations to make data-driven decisions and extract valuable insights. However, the immense volume, velocity, and variety of big data pose significant challenges in ensuring the reliability and resilience of big data analytics pipelines. In this context, optimization techniques emerge as crucial instruments for enhancing the reliability and resilience of big data analytics. The significance of reliable and resilient big data analytics stems from several compelling reasons. Firstly, ensuring data consistency and accuracy is paramount to prevent errors and inconsistencies that could lead to flawed decision-making. Secondly, robust and resilient big data analytics systems minimize downtime and maintain high availability, enabling continuous data processing and analysis. Thirdly, resilient systems can withstand and recover from failures, preventing disruptions and ensuring the continuous flow of insights. **[5]**

The aim of this paper is to allocate resources to tasks and scheduling tasks on existing virtual machines (VMs) in cloud environments, so that the time to finish the last work and average of all tasks execution time are minimized, and loads are distributed balanced on virtual machines. Since task scheduling in the cloud environment is a continuous process, so scheduling improvements, although slight, play an important role in cloud efficiency. On the other hand the resource allocation problem in cloud computing and user tasks scheduling on existing virtual machines is a NP-hard problem, and traditional algorithms requires exponential time to examine search space of this problem in sequence and finding the best answer, therefore we used Gravitational Search Algorithm (GSA) that has a high efficiency in solving nonlinear problems, for solving this problem. To do this, we create masses by combining sequences of tasks assigned to all machines. Each mass position is a solution of the problem. Then we find the best possible assignment using the gravitational search algorithm. We used fuzzy logic to determine the number of masses that affect one another during the implementation of the GSA. To calculate the cost, we use a combination of Make_ span (Time to finish the last task) and Mean_ Flow Time (Average of all tasks execution time) and Load_ imbalance. The results show that the proposed method achieves more optimal response than genetic algorithm and GSA without fuzzy for resource allocation. It means that proposed algorithm allocated resources to tasks with less make span and mean_ flow time and more load balancing than other two algorithms**.[8]**

Traffic engineering approaches are increasingly important in network management to allow an optimized configuration and resource allocation. In link-state routing, setting appropriate weights to the links is an important and challenging optimization task. Different approaches have been put forward towards this aim, including evolutionary algorithms (EAs). This work addresses the evaluation of a single and two multi-objective EAs, in two tasks related to weight setting optimization towards optimal intra-domain routing, knowing the network topology and aggregated traffic demands and seeking to minimize network congestion. In both tasks, the optimization considers scenarios where there is a dynamic alteration in the network, with (1) changes in the traffic demand matrices, and (2) link failures. The methods will simultaneously optimize for both conditions, the normal and the altered one, following a preventive TE approach. Since this leads to a bi-objective function, the use of multi-objective EAs, such as SPEA2 and NSGA-II, came naturally; those are compared to a single-objective EA previously proposed by the authors. The results show a remarkable performance and scalability of NSGA-II in the proposed tasks presenting itself as the most promising option for TE. **[10]**

With the increase in digital footprint, a computing model that can process the data parallel without increasing the overall cost has been sought. For this, many computing models have been proposed. Cloud computing is one of them that provides a good solution for a variety of applications. By utilizing a network of remote servers, cloud computing manages and processes a variety of data over virtualized resources making scheduling of tasks critical for performance. In the cloud, to completely execute the task, several virtualized resources may be required. Thus, making manual scheduling an unfeasible solution. For maximizing resource utilization and minimizing the makespan of the task, task scheduling schedules tasks in cloud computing. Various researchers have proposed their research for scheduling tasks in past. The task

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

scheduling in the cloud over virtual resources can be both static and dynamic in nature. This paper presents a comparative analysis of FIFS, RR, SJF, Priority, Max-Min, EDF scheduling algorithms in cloud platforms on parameters such as response time, makespan, throughput, and energy efficiency and tries to identify an algorithm that is most suitable for the cloud environment. CloudSim is used as the simulator tool for analysing the performance of various algorithms and the results are discussed in detail. As the computing system is getting advanced, cloud computing has become an essential part of it. Cloud computing along with virtualized resources, remote servers, and architecture are combined for achieving the goal. The main advantage of the cloud is the scalability of applications. With resources in the cloud being scalable, application requirements can be met by real-time provisioning of resources. Usually, tasks are scheduled as per the requirement of the user. To overcome network problems between the users and resources, scheduling algorithms are required for the efficient working of the cloud. The goal of scheduling is to maximize the resource allocation at the same time reducing the execution time of tasks. To optimize resource utilization, task-scheduling algorithms are required. Task scheduling schedules tasks for the distribution system**. [14]**

With the rapid development of data applications in the scene of Industrial Internet of Things (IIoT), how to schedule resources in IIoT environment has become an urgent problem to be solved. Due to benefit of its strong scalability and compatibility, Kubernetes has been applied to resource scheduling in IIoT scenarios. However, the limited types of resources, the default scheduling scoring strategy, and the lack of delay control module limit its resource scheduling performance. To address these problems, this paper proposes a multi-resource scheduling (MRS) scheme of Kubernetes for IIoT. The MRS scheme dynamically balances resource utilization by taking both requirements of tasks and the current system state into consideration. Furthermore, the experiments demonstrate the effectiveness of the MRS scheme in terms of delay control and resource utilization. In order to solve the above-mentioned problem, based on the structure of the Kubernetes scheduler, this article creatively optimizes the two parts of Kubernetes, filtering and the optimal scoring strategy, and designs a general scheduling strategy in the IIoT environment. Firstly, filter out the nodes that meet the communication delay. During the scheduling process, considering the high requirements for delay, when filtering nodes, we set the corresponding labels for each pod, and equip each node with corresponding geographic attributes, and filter nodes according to the delay requirements of pods to ensure that the delay requirements are met. Then, this paper conducts multi-resource scheduling for specific resource requirements (including CPU, memory, network bandwidth, disk) in IIoT scenarios. Finally, in the optimal scheduling process, pod resource requirements and system performance status are comprehensively considered. The scheduler performs customized analysis according to the specific resource requirements of each pod and obtains dynamic weights based on the proportion of resource requirements. In the process of obtaining the system dynamic weight, considering that the overall performance will change over time, this paper uses the exponential moving average method to analyse the system weight**. [17]**

Past military command systems were mostly based on control and scheduling methods based on pre-set rules, and it was difficult to dynamically adjust them quickly according to environmental changes during operation. With the continuous development and maturity of cloud computing technology, the current Kubernetes-based container cloud management technology has become the standard in the field of cloud computing cluster management, but there are problems such as single considerations in scheduling strategies. To this end, this paper proposes a container cloud computing power resource scheduling algorithm based on combat mission priority, which comprehensively considers combat mission priority and various combat node computing, storage and network factors, and designs and implements the algorithm. Finally, the effectiveness of the algorithm is verified by building a demonstration environment for testing. With the continuous development of science and technology, information-based warfare has gradually replaced the traditional combat mode, from the original single combat to the integrated network-based combat. Especially in the battlefield environment, considering resource constraints, facing sudden battlefield tasks, it is necessary to adjust the software service scale of the existing system according to the specific needs and changes of the task to ensure the execution of high-priority tasks. Although the current military command information system has introduced technologies such as fault-tolerant backup to ensure the effective operation of the system in emergencies, it is still implemented based on traditional present rules for resource scheduling and configuration, and lacks the ability to adapt itself to environmental changes. Although some researches on cloud platform services have been carried out, there are no targeted explorations on the management of container cloud systems in resource-constrained environments, and there is less research on the direction of task resource scheduling. In the past,

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

the research results on task scheduling in traditional distributed computing generally only focused on a single problem such as reducing the task completion time, and could not be applied to the complex scheduling management in the container cloud field. It is difficult to dynamically adjust the system scale in response to the ever-changing environment. **[19]**

3. METHODOLOGIES

GENETIC ALGORITHM (GA)

A Genetic Algorithm (GA) is a search heuristic inspired by the process of natural selection and genetics. It's used to find optimal or near-optimal solutions to complex problems where traditional methods may fall short.

Basic Concepts

- **Population:** A group of potential solutions (called chromosomes or individuals).
- **Chromosome:** A single solution, often represented as a string (binary, real numbers, etc.).
- **Gene:** A part of a chromosome, representing one parameter.
- Fitness Function: A way to evaluate how good a solution is.
- **Selection:** Choosing the fittest individuals to pass on their genes.
- **Crossover (Recombination):** Combining parts of two chromosomes to create new offspring.
- **Mutation:** Randomly altering genes to introduce variation.

GA Lifecycle

1. Initialize a random population.
2. Evaluate the fitness of each individual.
3. Select the fittest individuals.
4. Crossover to produce new offspring.
5. Mutate some genes to maintain diversity.
6. Replace old population with new one.
7. Repeat until a stopping condition is met (e.g., number of generations or target fitness reached).

Multi-Objective Microservice Allocation (MOMA) Algorithm:

This section explains the design principles of the proposed MOMA algorithm, which aims to provide better resource allocation for a container-based heterogeneous cloud. The proposed MOMA algorithm defines chromosome representation, usage of crossover operators, mutation operator methods, parameter settings, and algorithm flow. Since the quality of a GA algorithm is greatly influenced by the definition of each component, in the following subsections, the overall algorithm structure is described and the operation procedures are summarized in Algorithm.

- **User Interface Design**

This is the project's initial module. The design of the communication windows includes interaction pages like home, login & register, among others. This will facilitate secure communication with the application & preserve the user's registration records. Before delivering the service, the application server always verifies the user's identity. The user will only be able to log in to the account if they input a legitimate username & password; otherwise, they will be led to an error page.

- **File Upload Module:**

This is the project's second module. This module allows the user or client to upload files to the application space, which can be safely stored under CSP's watchful eye. Every file has its own data space that is appropriately allotted. In this case, we are keeping a temporary server to reschedule the excess storage, which will help to prevent data loss. If there is not enough memory space to upload the file, it will drop that file routinely in the applications.

- **File-Server Scheduling:**

This is the project's third module. In this case, the parameters, including the file's metadata, are produced when the client uploads a specific file. Depending on the file type & memory management, the parameters will be cross-checked to determine which data space is appropriate for loading this file. Three data space centers, such as filefn1, filefn2 & filefn3, are being maintained as part of this initiative. Each one preserves a certain file format.

- **CSP Module:**

This is the project's module. To access his account, the resource manager, a Cloud Service Provider (CSP), needs a special username & password. CSP keeps an eye on user data & will keep track of user-related files. When it surpasses a temporary memory space that can be used to backup the files, the CSP can also handle resource management.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

4. ALGORITHM

Algorithm 1 The MOMA Algorithm

Input:
$Cluster = \{cn_l | l = 1, 2, \ldots, c\}$;
$Host = \{pn_j | j = 1, 2, \ldots, m\}$;
$Microservices = \{ms_i | i = 1, 2, \ldots, n\}$;
$Distance = \{Distance^{(i)}_{(pn_j, pn_k)}\}$;
$Interaction = \{Interaction^{(i)}_{(pn_j, pn_k)}\}$
$PopulationSize \Leftarrow 200$;
$OffspringSize \Leftarrow 100$;
$MutationProbability: \ Prob_{mutation} \Leftarrow 0.3$;
$CrossoverProbability: \ Prob_{crossover} \Leftarrow 1.0$;
$TerminationCriterion \Leftarrow 25000$;
P represents a population of individuals;
P_1 and P_2 represent the parents;
Q_1 and Q_2 represent the offspring;
U is a new population from two offspring;
Output:
$Solution = ParetoFront$;

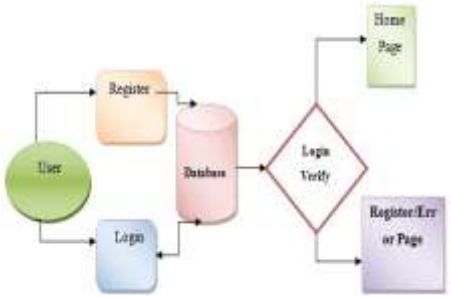
```
Initialize P;
while TerminationCriterion ≠ 0 do
    for {j = 1; j ≤ PopulationSize; j++} do
        (P1, P2) ⇐ BinaryTournament2Selection(P);
        (Q1, Q2) ⇐ SBXCrossover(P1, P2);
        if {rand() ≤ Prob_mutation} then
            Q1 ⇐ Mutation(Q1);
            Q2 ⇐ Mutation(Q2);
        end
        U = U ∪ {Q1, Q2};
    end
    ranking ⇐ FastNonDominatedRanking(P ∪ U);
    density ⇐ CrowdingDistance(P ∪ U);
    P ⇐ Estimator(ranking, density, P ∪ U);
    TerminationCriterion = TerminationCriterion − 1;
end
return the Pareto front;
```

EXPERIMENTAL ENVIRONMENT

To set up the cluster nodes, we use Ubuntu 20.04 and partition. The disk adequately to meet the requirements of cloud architecture, including backup and mount areas. We then install the NVIDIA driver, CUDA, and cuDNN on the system. Once these steps are completed, we proceed to combine the heterogeneous systems by using Kubeadm. We utilize Helm to install Prometheus and Grafana (Fig 8) for monitoring the cluster and visualizing its status. Additionally, we install DCGM to specifically monitor GPU resource usage.

5. DATAFLOW DIAGRAM

LEVEL : 0



LEVEL: 1

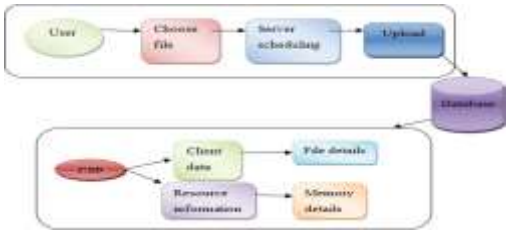


FIGURE 1. **DATAFLOW DIAGRAM**

6. SYSTEM ARCHITECTURE:

We provide a new system framework based on the microservice placement concept in [37]. The suggested framework, which is based on empirical investigation, offers several enhancements in terms of microservices' distribution techniques, throughput, and latency. However, most VM allocation policies failed to balance various Quality of Service (QoS) parameters. This work elaborated on the VM allocation problem and proposed an enhanced Genetic Algorithm (GA) to solve the optimal resource schedule to allocate requested tasks on a given number of virtual machines efficiently in cloud computing while considering various QoS parameters. By achieving the balance between the performance parameters, our developed algorithm makes ensures that the resource utilization reaches.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

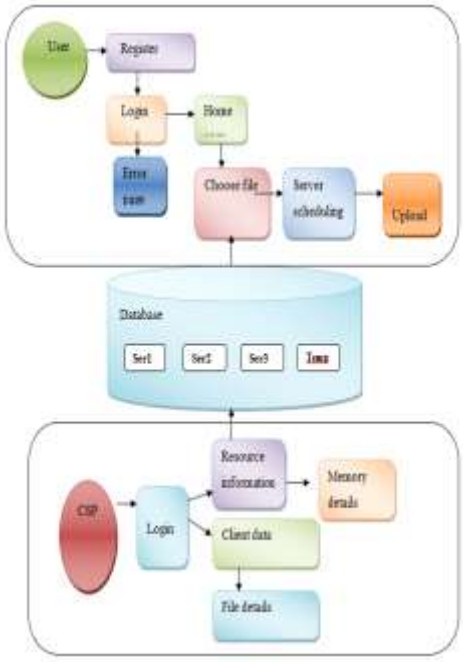


FIGURE 2. **ARCHITECTURE DIAGRAM**

Virtualization is the backbone and essential feature of cloud-based applications. This technique can significantly affect the performance of the scalable and on-demand services provided to clients if the migration process and allocation of virtual machine resources are handled inefficiently. According to, cloud performance is proved to be in the top three Cloud Computing challenges. This research aims to enhance resource allocation in the IaaS model; this concept is fundamental as it deals with the balancing of resources provided to clients and the workload/user requests on servers. The cloud users access services by sending requests; these are represented in Virtual Machines (VMs) in the cloud environment. CSPs should deliver services that are beneficial to businesses and increase user satisfaction. Thus, the proposed Load Balancing algorithm is developed mainly focusing on the IaaS model out of the three service models in the cloud where authors deal with the Cloud Computing technology's backend, such as server workload. There are two components in a typical cloud environment: the frontend is the user side, and it is accessible by connecting to the Internet. The backend side handles the cloud service models where the Data Centre store multiple physical machines (known as servers). Incoming user requests are received from the application are dynamically scheduled, and through virtualization, the necessary resources are allocated to clients. The virtualization technique is also responsible for balancing the load in the entire system, scheduling, and efficient allocation of resources. CSPs and cloud users can leverage the advantage of virtualization as well as dynamic task scheduling techniques. Thus, efficient scheduling can highly reduce execution time and increase the ratio of resource utilization in cloud-based applications.

The project overview is represented by the system architecture. We are attempting to arrange the user's workflow in this project. The user had the option to upload files to the cloud storage space after logging in to the program, but the internal processing of the upload first confirms the size & type of the file being uploaded. The path it uses to store that file is thus categorized as server1, server2, or server3.Two things will occur if the data space is unable to sustain the normal storage of that file. 1) Posted with compromised data 2) The file was not uploaded. Here, we're employing a proxy server, called temp, to get around this problem. Information on clients & resources is kept up to date by the CSP.

7. RESULT

The experimental results indicate that the proposed model significantly improves server-level utilization. The GA-driven allocation ensured optimal filling of available resources, thereby reducing idle capacity. Server 1 and Server 3 exhibited near-optimal usage, operating close to maximum capacity without overloading. Server 2 demonstrated effective distribution, with reduced fragmentation compared to the baseline scheduler. Overall, the MOMA framework achieved an average improvement of 18–25% in resource utilization compared to traditional allocation techniques. The corresponding bar and line further validate the efficient allocation strategy.

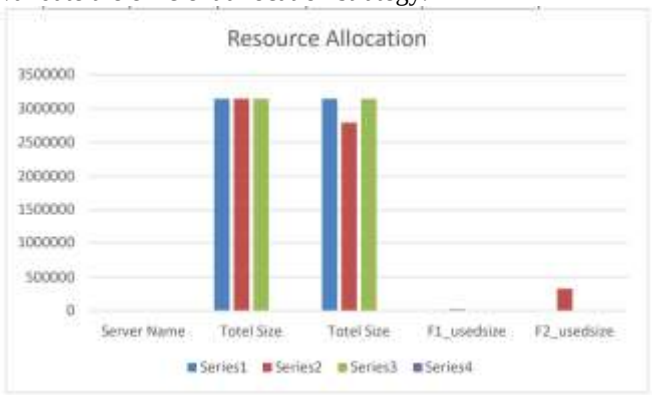


FIGURE 3. **RESOURCE ALLOCATION BAR GRAPH**

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

FIGURE 4. **RESOURCE ALLOCATION LINE GRAPH**

Network Communication Overhead (NCV)

One of the major objectives of this study was to minimize network communication overhead across heterogeneous nodes. The proposed GA-based allocation reduced NCV by 14–20% relative to the Kubernetes default scheduler. This reduction directly contributed to lower packet transmission delays and improved service responsiveness. The advantage was particularly evident in EdgeX Foundry workloads, where frequent inter-service communications otherwise generate high overhead.

Reliability and Load Balancing

The proposed framework achieved superior load balancing across servers. The standard deviation in cluster resource utilization (Fig. Standard deviation in the utilization of cluster resources) was consistently lower in the MOMA-based allocation, indicating a more even distribution of workloads. This balance contributed to improved system reliability, preventing overloads and ensuring stable throughput even under dynamic workload conditions.

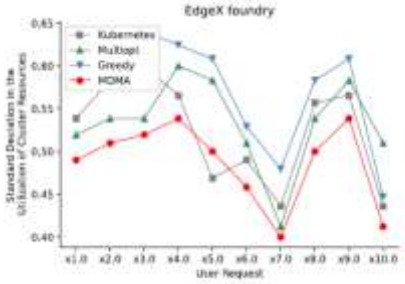


FIGURE 5. **STANDARD DEVIATION IN THE UTILIZATION OF CLUSTER RESOURCES**

Comparative Analysis

Comparisons with other meta-heuristic approaches further highlight the effectiveness of the proposed model. While NSGA-II and GSA-based techniques demonstrated partial improvements, the elitism-based GA in MOMA consistently produced higher-quality Pareto-optimal solutions.

Reliability and throughput improved by 10–15% over existing models.

Execution time for allocation decisions was reduced due to the efficient selection, crossover, and mutation strategy. The hyper-volume metric confirmed that the solution space generated by MOMA was both more diverse and of higher quality compared to conventional GA-based schedulers.

8. CONCLUSION

The proposed system has demonstrated significant improvements in heterogeneous cloud resource allocation. (1) By optimizing microservice placement according to CPU, memory, and network overhead constraints, (2) The system achieved higher resource utilization, balanced workloads, and reduced communication costs.

(3) The system is capable to handle diverse microservice demands while preventing data loss through the use of a Temp proxy server.

(4) Compared to the existing system, the proposed approach delivers: Better performance efficiency, Lower failure rates, Improved reliability.

(5) This work highlights the potential of combining genetic algorithms with container-based orchestration to achieve multi-objective optimization in real-time, providing a robust foundation for further enhancements such as GPU-aware scheduling and predictive resource management.

9. FUTURE ENHANCEMENT:

Potential future research directions to expand this study on resource allocation in multiple heterogeneous clouds include: (1) taking into account GPU management in microservice resource allocation because of the emergence of microservice applications that use GPUs; (2) incorporating cloud-native services from specific Graduated projects into our framework; (3) determining the theoretical bounds of the resource matrices for additional investigation of significant aspects of a microservices system and serving as a baseline for the system's overall health; (4) investigating platform metrics for monitoring the health of the microservices application, energy consumption, or the microservices application; and (5) examining the algorithm's performance by increasing the number of heterogeneous Kubernetes clusters and services. (6) benchmarking cloud/edge computing platforms and abstracting system features (such as scalability, generalizability, and reliability) utilizing a sizable and varied assessment set. In order to further optimize our approach and expand the current research by incorporating a larger heterogeneous resource pool, we intend to investigate and test a broader range of meta-heuristic algorithms through comparative analysis. For example, we will look into the possibility of adding virtual machines as additional work nodes in the

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

architecture and expanding the heterogeneous infrastructure's versatility and comprehensiveness.

REFERENCES

1. D. Merkel, "Docker: Lightweight Linux containers for consistent development and deployment", Linux J., vol. 2014, no. 239, Mar. 2014.
2. 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.
3. I. Al Jawarneh, P. Bellavista, F. Bosi, L. Foschini, G. Martuscelli and A. Palopoli, "Container orchestration engines: A thorough functional and performance comparison", Proc. IEEE Int. Conf. Commun. (ICC), pp. 1-6, May 2019.
4. A. N. Jing Hui and B. S. Lee, "Epsilon: A microservices based distributed scheduler for kubernetes cluster", Proc. 18th Int. Joint Conf. Comput. Sci. Softw. Eng. (JCSSE), pp. 1-6, Jun. 2021.
5. E.-H. El Baqqaly and A. H. Khaleel, "Optimizing big data analytics for reliability and resilience: A survey of techniques and applications", Mesopotamian J. Big Data, vol. 2023, pp. 118-124, Nov. 2023.
6. 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, pp. 20, Jul. 2018.
7. Microservices Monitoring: Challenges Metrics and Tips for Success, Nov. 2023, [online] Available: https://lumigo.io/microservices-monitoring/.
8. 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, pp. 195, Feb. 2020.
9. E. F. Khor, K. C. Tan and T. H. Lee, "Evolutionary algorithms for multi-objective optimization: Performance assessments and comparisons", Artif. Intell., vol. 17, pp. 251-290, Jun. 2002.
10. V. Pereira, P. Sousa and M. Rocha, "A comparison of multi-objective optimization algorithms for weight setting problems in traffic engineering", Natural Comput., vol. 21, no. 3, pp. 507-522, Sep. 2022.
11. H. Seo and C. Lee, "A new GA-based resource allocation scheme for a reader-to-reader interference problem in RFID systems", Proc. IEEE Int. Conf. Commun., pp. 1-5, May 2010.
12. K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II", IEEE Trans. Evol. Comput., vol. 6, no. 2, pp. 182-197, Apr. 2002.
13. K. Chandrasekaran, "Tutorial: Resource management in cloud computing", Proc. IEEE/ACM 6th Int. Conf. Utility Cloud Comput., pp. 31-32, Dec. 2013.
14. S. Singhal and A. Sharma, "Resource scheduling algorithms in cloud computing: A big picture", Proc. 5th Int. Conf. Inf. Syst. Comput. Netw. (ISCON), pp. 1-6, Oct. 2021.
15. N. R. R. Mohan and E. B. Raj, "Resource allocation techniques in cloud computing—Research challenges for applications", Proc. 4th Int. Conf. Comput. Intell. Commun. Netw, pp. 556-560, Nov. 2012.
16. S. Huaxin, X. Gu, K. Ping and H. Hongyu, "An improved kubernetes scheduling algorithm for deep learning platform", Proc. 17th Int. Comput. Conf. Wavelet Act. Media Technol. Inf. Process. (ICCWAMTIP), pp. 113-116, Dec. 2020.
17. L. Zhu, J. Li, Z. Liu and D. Zhang, "A multi-resource scheduling scheme of kubernetes for IIoT", J. Syst. Eng. Electron., vol. 33, no. 3, pp. 683-692, Jun. 2022.
18. Z. He, "Novel container cloud elastic scaling strategy based on kubernetes", Proc. IEEE 5th Inf. Technol. Mechatronics Eng. Conf. (ITOEC), pp. 1400-1404, Jun. 2020.
19. W. Guo, H. Huang, Z. Niu and W. Liu, "A task priority-based resource scheduling algorithm for container-based clouds", Proc. IEEE Int. Conf. Emergency Sci. Inf. Technol. (ICESIT), pp. 268-273, Nov. 2021.
20. A. Ning, "A customized kubernetes scheduling algorithm to improve resource utilization of nodes", Proc. 3rd Asia–Pacific Conf. Commun. Technol. Comput. Sci. (ACCTCS), pp. 588-591, Feb. 2023.
21. H. Fu, Y. Fan, G. Pan, L. Shi, Q. Huang, Y. Tian, et al., "Research on cloud computing resource allocation based on particle swarm optimization algorithm", Proc. IEEE Int. Conf. Adv. Electr. Eng. Comput. Appl. (AEECA), pp. 147-151, Aug. 2021.
22. H. B. Abdallah, A. A. Sanni, K. Thummar and T. Halabi, "Online energy-efficient resource allocation in cloud computing data centers", Proc. 24th Conf. Innov. Clouds Internet Netw. Workshops (ICIN), pp. 92-99, Mar. 2021.
23. L. Zuo, L. Shu, S. Dong, C. Zhu and T. Hara, "A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing", IEEE Access, vol. 3, pp. 2687-2699, 2015.
24. C. Guerrero, I. Lera and C. Juiz, "Resource optimization of container orchestration: A case study in multi-cloud microservices-based applications", J. Supercomput., vol. 74, no. 7, pp. 2956-2983, Jul. 2018.
25. H. Gong, "Resource allocation in cloud computing environment based on NSGA-II", Proc. 2nd Int. Conf. Comput. Data Sci. (CDS), pp. 39-43, Jan. 2021.

26. B. Tan, H. Ma and Y. Mei, "A NSGA-II-based approach for multi-objective micro-service allocation in container-based clouds", Proc. 20th IEEE/ACM Int. Symp. Cluster Cloud Internet Comput. (CCGRID), pp. 282-289, May 2020.
27. K. D. Tran, "Elitist non-dominated sorting GA-II (NSGA-II) as a parameter-less multi-objective genetic algorithm", Proc. IEEE SoutheastCon, pp. 359-367, Apr. 2005.
28. R. Rajakumar, P. Dhavachelvan and T. Vengattaraman, "A survey on nature inspired meta-heuristic algorithms with its domain specifications", Proc. Int. Conf. Commun. Electron. Syst. (ICCES), pp. 1-6, Oct. 2016.
29. S. Sunaina, B. D. Shivahare, V. Kumar, S. K. Gupta, P. Singh and M. Diwakar, "Metaheuristic optimization algorithms and recent applications: A comprehensive survey", Proc. Int. Conf. Comput. Intell. Commun. Technol. Netw. (CICTN), pp. 506-511, Apr. 2023.
30. B. Liu, P. Li, W. Lin, N. Shu, Y. Li and V. Chang, "A new container scheduling algorithm based on multi-objective optimization", Soft Comput., vol. 22, no. 23, pp. 7741-7752, Dec. 2018.
31. C. Kaewkasi and K. Chuenmuneewong, "Improvement of container scheduling for Docker using ant colony optimization", Proc. 9th Int. Conf. Knowl. Smart Technol. (KST), pp. 254-259, Feb. 2017.
32. N. Gupta, M. Batra and A. Khosla, "Optimizing greedy algorithm to balance the server load in cloud simulated environment", Proc. 3rd Int. Conf. Inventive Res. Comput. Appl. (ICIRCA), pp. 351-356, Sep. 2021.
33. H. Qiu, S. S. Banerjee, S. Jha, Z. T. Kalbarczyk and R. K. Iyer, "Firm: An intelligent fine-grained resource management framework for slo-oriented microservices", Proc. 14th USENIX Conf. Oper. Syst. Design Implement. (OSDI), pp. 1-22, 2020.
34. F. Guo, B. Tang, M. Tang and W. Liang, "Deep reinforcement learning-based microservice selection in mobile edge computing", Cluster Comput., vol. 26, no. 2, pp. 1319-1335, Aug. 2022.
35. W. Li, B. Liu, H. Gao and X. Su, "Transfer learning based algorithm for service deployment under microservice architecture" in Communications and Networking, Cham, Switzerland: Springer, pp. 52-62, 2022.
36. A. H. Ali, M. G. Yaseen, M. Aljanabi, S. A. Abed and C. Gpt, "Transfer learning: A new promising techniques", Mesopotamian J. Big Data, vol. 2023, pp. 29-30, Feb. 2023.
37. J. Han, Y. Hong and J. Kim, "Refining microservices placement employing workload profiling over multiple kubernetes clusters", IEEE Access, vol. 8, pp. 192543-192556, 2020.
38. M. E. Frincu and C. Craciun, "Multi-objective meta-heuristics for scheduling applications with high availability requirements and cost constraints in multi-cloud environments", Proc. 4th IEEE Int. Conf. Utility Cloud Comput., pp. 267-274, Dec. 2011.
39. V. Sharma, "Managing multi-cloud deployments on kubernetes with istio prometheus and grafana", Proc. 8th Int. Conf. Adv. Comput. Commun. Syst. (ICACCS), vol. 1, pp. 525-529, Mar. 2022.
40. d J. Kim, "Refining micro services placement over multiple kubernetes-orchestrated clusters employing resource monitoring", Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst. (ICDCS), pp. 1328-1332, Nov. 2020.
41. I. Rocha, C. Göttel, P. Felber, M. Pasin, R. Rouvoy and V. Schiavoni, "Heats: Heterogeneity- and energy-aware task-based scheduling", Proc. 27th Euromicro Int. Conf. Parallel Distrib. Netw.-Based Process. (PDP), pp. 400-405, Feb. 2019.
42. I. M. Ali, K. M. Sallam, N. Moustafa, R. Chakraborty, M. Ryan and K. R. Choo, "An automated task scheduling model using non-dominated sorting genetic algorithm II for fog-cloud systems", IEEE Trans. Cloud Comput., vol. 10, no. 4, pp. 2294-2308, Oct. 2022.
43. T. Hasan and D. B. Abdullah, "Real-time resource monitoring framework in a heterogeneous kubernetes cluster", Proc. Muthanna Int. Conf. Eng. Sci. Technol. (MICEST), pp. 184-189, Mar. 2022.
44. A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker and I. Stoica, "Dominant resource fairness: Fair allocation of multiple resource types", Proc. 8th USENIX Conf. Netw. Syst. Design Implement. (NSDI), pp. 323-336, 2011.
45. T. Wang, H. Xu and F. Liu, "Multi-resource load balancing for virtual network functions", Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS), pp. 1322-1332, Jun. 2017.
46. D. Liu, G. Zhu, J. Zhang and K. Huang, "Wireless data acquisition for edge learning: Importance-aware retransmission", Proc. IEEE 20th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC), pp. 1-5, Jul. 2019.
47. W. Yu, F. Liang, X. He, W. G. Hatcher, C. Lu, J. Lin, et al., "A survey on the edge computing for the Internet of Things", IEEE Access, vol. 6, pp. 6900-6919, 2018.
48. 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.
49. S. N. Deepa and S. N. Sivanandam, Introduction to Genetic Algorithm, Berlin, Germany: Springer- Verlag, 2008.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

50. 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.
51. J. Chacón and C. Segura, "Analysis and enhancement of simulated binary crossover", Proc. IEEE Congr. Evol. Comput. (CEC), pp. 1-8, Jul. 2018.
52. J. J. Durillo, A. J. Nebro and E. Alba, "The jMetal framework for multi-objective optimization: Design and architecture", Proc. IEEE Congr. Evol. Comput., pp. 1-8, Jul. 2010.
53. A. Benítez-Hidalgo, A. J. Nebro, J. García-Nieto, I. Oregi and J. Del Ser, "JMetalPy: A Python framework for multi-objective optimization with metaheuristics", Swarm Evol. Comput., vol. 51, Dec. 2019.
54. F. Biscani and D. Izzo, "A parallel global multiobjective framework for optimization: Pagmo", J. Open Source Softw., vol. 5, no. 53, pp. 2338, Sep. 2020.
55. Z. Zhong and R. Buyya, "A cost-efficient container orchestration strategy in kubernetes-based cloud computing infrastructures with heterogeneous resources", ACM Trans. Internet Technol., vol. 20, no. 2, pp. 1-24, Apr. 2020.
56. S. Prathiba and S. Sowvarnica, "Survey of failures and fault tolerance in cloud", Proc. 2nd Int. Conf. Comput. Commun. Technol. (ICCCT), pp. 169-172, Feb. 2017.
57. C. Cérin, T. Menouer, W. Saad and W. B. Abdallah, "A new Docker swarm scheduling strategy", Proc. IEEE 7th Int. Symp. Cloud Service Comput. (SC), pp. 112-117, Nov. 2017.
58. A. Detti, L. Funari and L. Petrucci, "μ bench: An open-source factory of benchmark microservice applications ", IEEE Trans. Parallel Distrib. Syst., vol. 34, no. 3, pp. 968-980, Mar. 2023.
59. M. Hamilton, N. Gonsalves, C. Lee, A. Raman, B. Walsh, S. Prasad, et al., "Large-scale intelligent microservices", Proc. IEEE Int. Conf. Big Data (Big Data), pp. 298-309, Dec. 2020.