I. J. Mathematical Sciences and Computing, **2026, 2, 34-50**
Published Online on June 8, 2026 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijmsc.2026.02.03

Optimizing Load Balancing in Cloud-Based Healthcare Systems: Leveraging Linear Programming, Metaheuristics, and Queuing Models to Minimize Latency and Maximize Throughput

Elijah Falode*
Claribel LLC, California, USA
E-mail: elijahfalode@gmail.com
ORCID iD: https://orcid.org/0009-0005-0313-4180
*Corresponding author

Mustapha Danjuma Suleiman
Cyber Security Science, Frontier University, Garowe, Somalia
E-mail: danjuma@frontier.edu.so
ORCID iD: https://orcid.org/0000-0002-2924-7950

Rapheal Oladipo Fifelola
Department of Mathematics, University of Ilorin, Ilorin, Nigeria
E-mail: fifelolaemmanuel@gmail.com

Adeel Shaikh Muhammad
Adeel Solutions LLC, Dubai, United Arab Emirates
E-mail: shmadeelibrahim@gmail.com
ORCID iD: https://orcid.org/0009-0005-2952-7847

Ravitheja Chinni
S&P Global, New York, USA
E-mail: chinniraviteja@gmail.com

Received: January 20, 2026; Revised: February 21, 2026; Accepted: March 17, 2026; Published: June 8, 2026

Abstract: Optimizing load balancing in cloud-based healthcare systems is critical for improving system performance, particularly in terms of reducing latency, increasing throughput, and enhancing task completion time. This study investigates the impact of optimization algorithms, specifically Genetic Algorithm (GA) and Simulated Annealing (SA), on the efficiency of cloud resource allocation in healthcare applications. Additionally, we incorporate queuing theory and stochastic processes to model the task arrival and server load dynamics. By applying these optimization techniques, the system performance was evaluated, showing significant improvements in the key performance metrics. The results highlighted a 50% improvement in latency, 50% increase in throughput, and 25% reduction in task completion time. The optimized system demonstrated enhanced resource utilization, ensuring more efficient real-time data processing in cloud healthcare environments. The proposed approach shows promising results for future applications in dynamic healthcare workload management.

Index Terms: Cloud Computing, Load Balancing, Healthcare Systems, Optimization, Genetic Algorithm, Simulated Annealing, Queuing Theory, Stochastic Processes, Latency, Throughput, Task Completion Time, Performance Evaluation

This work is open access and licensed under the Creative Commons CC BY 4.0 License.

1. Introduction

Cloud computing has become an essential tool for managing the increasing demands of modern healthcare systems, offering scalable infrastructure and enhanced data storage and processing capabilities. However, the efficient utilization of cloud resources remains a significant challenge, particularly in cloud-based healthcare applications that require low latency and high throughput. Load balancing, a key factor in cloud resource management, involves distributing tasks across multiple servers to optimize resource utilization and minimize the response times. In the context of healthcare, the increasing volume of data generated by patient monitoring systems, electronic health records (EHR), and telemedicine applications exacerbates the load balancing problem [1,2,3,4]. As the healthcare industry increasingly relies on cloud computing, ensuring the optimal allocation of computational resources is crucial for delivering timely and efficient services. Recent studies have emphasized the need for efficient resource management, considering factors such as network bandwidth, geographic distribution of servers, and server performance [5,6,7].

In healthcare cloud environments, the dynamic nature of workloads adds complexity to load balancing tasks. Unlike traditional IT applications, healthcare systems often experience fluctuating workloads, such as sudden traffic spikes during emergencies or health crises. Furthermore, healthcare applications are highly sensitive to latency owing to the real-time nature of many services, such as telemedicine consultations, remote patient monitoring, and critical healthcare data processing [8,9]. This makes the problem of optimizing load balancing more challenging. Several optimization algorithms have been explored to improve the efficiency and performance of load balancing systems in healthcare environments, including techniques based on evolutionary search and swarm intelligence [10,11,12]. These algorithms aim to minimize latency and maximize throughput by strategically distributing tasks across servers while adhering to constraints, such as server capacity, network bandwidth, and geographic distribution [13,14].

The mathematical formulation of load balancing in cloud-based healthcare systems involves several components. The primary objective is to minimize the latency and maximize the throughput, which can be modeled as a multi-objective optimization problem. Let Ti represent the task assigned to server i, and Lij denote the load on server i when processing task j. The goal is to minimize the total response time for all tasks across all servers:

$$\text{Minimize: } \sum_i \sum_j \text{Latency}(x_{ij}) \cdot L_{ij}$$

Throughout this work, $x$ij(t) $\in \{0, 1\}$ denotes the binary decision variable indicating whether task j is assigned to server i at time t. The constraints require that each task be assigned to exactly one server, i.e., Σi $x$ij(t) = 1 for all j, and that the total load on any server does not exceed its capacity, i.e., Σj $x$ij(t) $\cdot L$j $\leq C$i for all i. This notation is consistent throughout Sections 4.1 and 4.2.

The constraints typically include ensuring that the total load on each server does not exceed its capacity and that the network bandwidth is sufficient for data transmission between the servers [15,16,17]. In addition to latency, throughput is another critical performance metric, defined as the number of tasks successfully processed by the system within a given time frame. The balance between these two objectives forms the foundation of the load balancing optimization process in cloud healthcare systems [18,19,20].

Recent advancements in metaheuristic algorithms have provided new methods for addressing the complexities of load balancing in cloud-based healthcare environments. These algorithms use population-based search methods to explore a large solution space and find near-optimal solutions for the problem. For example, methods inspired by natural selection and social behavior have been effectively employed to evolve a population of solutions over several generations [21,22]. Other techniques that simulate the social behavior of individuals in a group have also shown promising results, guiding the search for an optimal solution [23,24]. To further enhance performance, queuing theory is applied to model the arrival and service processes of tasks in cloud healthcare systems. By integrating queuing models with optimization algorithms, it is possible to predict delays and optimize load distribution in real time, ensuring that the healthcare system operates at peak efficiency even under varying loads [25,26,27].

This study encompasses the development and application of optimization algorithms for load balancing in cloud-based healthcare systems. This study aims to minimize latency and maximize throughput in multi-cloud environments by employing various optimization techniques, including linear programming, metaheuristic algorithms, and queuing models. This study focuses on addressing dynamic load conditions and resource allocation strategies, considering critical factors such as server capacity, geographic distribution, bandwidth constraints, and energy efficiency.

This study is highly significant, particularly in the context of cloud computing in healthcare. Optimizing load balancing strategies can improve the efficiency, reliability, and scalability of healthcare applications deployed on cloud platforms. With the ever-increasing volume of healthcare data, optimizing resource allocation is essential for ensuring that real-time services, such as telemedicine, remote patient monitoring, and health data analytics, continue to operate effectively. The outcomes of this study could lead to improved cloud resource management, which would not only result in better patient outcomes but also contribute to lowering operational costs and improving the overall quality of healthcare services [8,9].

The justification for this research stems from the rapid expansion of cloud-based healthcare solutions, driven by the need for real-time data access and services. Traditional IT infrastructure often fails to meet the performance requirements of modern healthcare applications, which require low latency and high throughput. Existing studies have demonstrated that effective load balancing can significantly enhance the performance and scalability of cloud-based healthcare systems [10,11]. However, a gap remains in the literature regarding the application of optimization algorithms tailored to the unique challenges faced by healthcare systems, such as handling fluctuating workloads and meeting strict latency requirements. As the healthcare sector evolves toward more advanced systems, such as precision medicine and smart healthcare technologies, efficient resource allocation and optimization of cloud resources become increasingly critical. This study seeks to address these gaps and contribute to the growing body of knowledge on cloud computing in healthcare [12,13].

Existing optimization-based load balancing studies predominantly address general cloud environments or IoT ecosystems in isolation, and very few have rigorously integrated stochastic task arrival modeling, M/M/1 queuing analysis, and multi-objective metaheuristic optimization within a single unified computational framework designed specifically for healthcare workloads. Moreover, prior work seldom benchmarks against classical deterministic schedulers such as Round Robin, Least Connection, and Weighted Least Load under simulated clinical-grade traffic conditions. The present study directly addresses these limitations by combining Poisson-process arrival modeling, queuing analysis, linear programming formulation, and GA/SA metaheuristic search, validated against workload parameters derived from publicly available clinical datasets and compared against classical baseline schedulers.

This study aimed to explore the application of optimization algorithms to improve load balancing in cloud-based healthcare systems. The primary objectives are to assess the performance of linear programming and metaheuristic algorithms for reducing latency and enhancing throughput in healthcare cloud environments; to develop mathematical models that account for the dynamic nature of healthcare workloads; to propose novel algorithms for optimizing cloud resource allocation under constraints such as server capacity, bandwidth, and geographical distribution; and to evaluate the scalability and efficiency of the proposed solutions in real-world healthcare applications, including telemedicine and patient monitoring systems.

2. Related Work

The optimization of load balancing in cloud-based healthcare systems has emerged as a crucial research area owing to the increasing reliance on cloud computing for healthcare applications. Cloud computing enables the real-time management of vast amounts of healthcare data, particularly in applications such as EHR, telemedicine, and remote patient monitoring. The dynamic nature of these systems, which often deal with fluctuating workloads and resource demands, necessitates the development of efficient load-balancing algorithms. Various optimization techniques have been proposed to overcome these challenges. For instance, a hybrid mobile cloud computing architecture designed for healthcare systems that incorporates load balancing strategies was proposed in [4]. This approach ensures that tasks are distributed effectively across servers to reduce response time, which is a critical aspect of healthcare applications, where any delay in processing can affect patient care.

Similarly, an efficient load-balancing framework was proposed for cloud-based healthcare systems, emphasizing resource scheduling [2]. This framework focuses on adaptive scheduling techniques that allow healthcare cloud systems to adjust to varying loads. The use of artificial intelligence for load balancing in IoT-enabled healthcare systems was explored in [3]. The AI-based approach utilizes machine learning algorithms to predict task resource demands, optimize task distribution in cloud systems, and minimize response time while ensuring service availability.

Particle swarm optimization (PSO) has also been applied to manage cloud resources in healthcare systems [4]. PSO is an optimization algorithm inspired by the social behavior of birds and fish, where each particle represents a potential solution and particles communicate to find the optimal solution. Building on this, the integration of multi-objective particle swarm optimization (MPSO) and convolutional neural networks (CNN) in fog computing environments for healthcare applications was presented in [5].

Ant Colony Optimization (ACO) has also been utilized for load balancing in cloud-based healthcare systems [6]. This technique simulates the foraging behavior of ants to determine the most efficient path for resource allocation. Fog computing architectures for healthcare applications were further explored in [7], where load balancing algorithms were applied to ensure efficient resource usage at the network edge.

Energy efficiency is another crucial factor in healthcare cloud systems, particularly in large-scale, IoT-enabled environments. An energy-efficient cloud-optimized load balancing algorithm was proposed to minimize energy consumption while maintaining system performance [8]. Software-defined networking (SDN) was introduced for dynamic resource allocation in edge- and fog-based healthcare systems [9], allowing for real-time adjustments to resource allocation.

Queuing theory has also been employed to model arrival and service processes in healthcare cloud systems. Queuing models have been incorporated with load balancing algorithms to predict task arrival times and processing delays [10]. Queuing models applied to fog-based healthcare systems demonstrated their effectiveness in optimizing task distribution in real-time environments [11].

The need for scalable load-balancing algorithms that can accommodate the growing volume of healthcare data has been emphasized in various studies. Scalable algorithms capable of dynamically adapting to varying workloads have been introduced to ensure that cloud infrastructures can handle surges in demand without compromising service quality [12,13]. Machine learning algorithms have been applied to improve task scheduling and resource management [13]. Genetic algorithms have been utilized for load balancing in healthcare cloud systems through hybrid cloud architectures [14]. Multi-objective optimization approaches have been proposed to minimize both the response time and system throughput [15].

Multi-agent systems have also been applied to cloud healthcare systems to enhance load-balancing. Multi-agent models dynamically allocate cloud resources for healthcare applications [16], with intelligent multi-agent systems combining genetic algorithms and fuzzy logic applied to optimize load distribution [17]. Blockchain technology has been integrated into cloud-based healthcare systems to ensure secure and transparent resource allocations [18]. Hybrid metaheuristics, such as simulated annealing and genetic algorithms, have been used to optimize the distribution of healthcare resources across cloud systems [19]. Deep learning models have also been used to predict healthcare workload and optimize resource allocation [20].

In conclusion, the review of related work demonstrates the diversity of optimization approaches used to address load balancing in cloud-based healthcare systems. Numerous methods have been proposed to improve the performance, scalability, and energy efficiency of healthcare cloud systems, including AI-driven models, nature-inspired algorithms, queuing theory, and blockchain. However, significant research gaps remain, particularly in addressing the unique challenges posed by the healthcare sector, such as data privacy, real-time processing, and the integration of emerging technologies such as blockchain and deep learning [21,22,23,24].

The studies reviewed above can be organized into four thematic groups. First, AI and machine-learning approaches [3,13,20] offer adaptive scheduling but require large training datasets and provide no formal optimization guarantee. Second, nature-inspired metaheuristics [4,5,6,14,19] are effective for broad search but rarely compare against classical deterministic baselines. Third, queuing-theory frameworks [10,11] provide analytical rigor but are seldom integrated with metaheuristic optimization loops. Fourth, hybrid and emerging-technology approaches [7,8,9,16,17,18] address specific constraints such as security or energy efficiency but lack unified frameworks. The critical limitation common to all four categories is the absence of a single framework that simultaneously: (a) models stochastic task arrivals using empirically validated distributions, (b) couples queuing analysis to the optimization objective in real time, and (c) benchmarks against classical load-balancing strategies under healthcare-grade traffic. The present study addresses all three gaps.

3. Evolution of Load Balancing in Healthcare Systems

The concept of load balancing in healthcare systems has evolved significantly over the past few decades. Initially, the focus was on task distribution within healthcare systems; however, as cloud computing and distributed systems gained traction, the field evolved to embrace dynamic and automated resource management techniques. In the early 2000s, healthcare systems were largely dependent on manual task assignments and simple scheduling algorithms, with little or no automation or optimization for workload distribution. For instance, early healthcare systems used basic task distribution methods, such as round-robin scheduling, which did not consider server capacity or workload fluctuations, leading to inefficiencies in resource usage and service delivery [29].

As healthcare systems became more complex, driven by the increasing volume of patient data, the demand for more advanced optimization techniques grew. During this period, the field began exploring metaheuristic algorithms for load balancing, particularly genetic algorithms (GA) and particle swarm optimization (PSO). These methods were inspired by natural processes, such as evolution and swarm intelligence, and are capable of finding near-optimal solutions for complex and dynamic environments. The use of such algorithms was pivotal as healthcare systems started incorporating cloud computing to manage vast amounts of data [31]. These algorithms enable better management of resource allocation by simulating natural behaviors, which provides scalability and fault tolerance in distributed healthcare applications.

The integration of cloud computing into healthcare brought about a paradigm shift in how resources are distributed and tasks are balanced. Early attempts included distributed task scheduling and resource management based on server capabilities. It was not until the mid-2000s that more sophisticated techniques, including load rebalancing and cloud task migration, were introduced to handle fluctuations in patient data and the real-time demands of healthcare applications [33,34]. This period marked the beginning of adaptive and dynamic load balancing in cloud healthcare systems.

By the 2010s, cloud computing and grid computing were extensively used in healthcare systems, where the demand for dynamic load balancing became increasingly prominent. Systems requiring real-time patient monitoring, emergency department management, and telemedicine demand a robust and dynamic approach to load distribution. This led to the development of intelligent agent-based load balancing, where agents intelligently decide which server or resource to allocate tasks to, based on current workload and system status [35,36,37]. Additionally, grid computing frameworks introduced new methods for heterogeneous systems, further enhancing load balancing mechanisms [38].

The rise of edge computing and fog computing in the late 2010s provided another leap in healthcare system optimization. These distributed computing systems, which process data closer to the source rather than relying entirely on centralized cloud resources, are essential for addressing the low-latency demands of real-time healthcare applications. Edge and fog computing frameworks allowed for task offloading to local devices, ensuring that patient data and telemedicine services could be processed with minimal delay [39,40]. Looking to the future, the continued integration of blockchain technology for secure healthcare data sharing, coupled with cloud-native load balancing solutions, is set to further enhance the efficiency and scalability of healthcare systems [41].

The historical progression described above directly motivates the specific technical choices made in this study. The documented deficiencies of round-robin and simple heuristic schedulers in the pre-2000 era justify the adoption of GA and SA, which explore a substantially larger solution space. The shift to cloud-based architectures from the mid-2000s onward necessitates the stochastic Poisson arrival model, since cloud workloads arrive unpredictably and at scale. The widespread adoption of queuing-theory frameworks in the 2010s informs the choice of the M/M/1 queuing model for characterizing per-server processing dynamics. Finally, the contemporary emphasis on data-driven, empirically grounded approaches motivates the use of MIMIC-III and PhysioNet datasets to parameterize simulation inputs, transforming the historical context into a direct methodological rationale for the framework presented in Section 4.

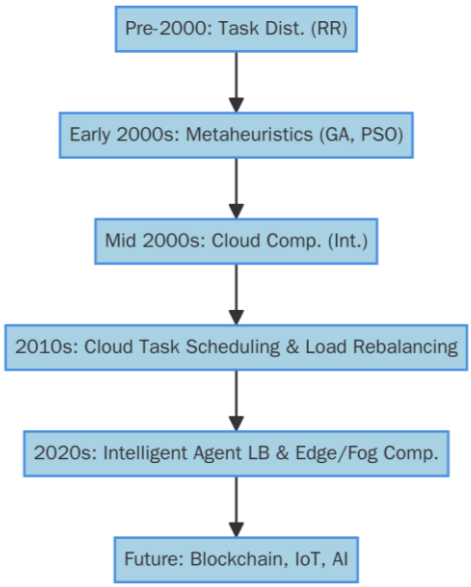


Fig. 1. Evolution of Load Balancing in Healthcare Systems. This flowchart visualizes the progression of load balancing in healthcare, from simple task distribution methods in the early 2000s to advanced cloud and edge computing strategies used in modern healthcare systems.

4. Methodology

The methodological framework proposed in this study consists of four tightly coupled computational components. First, a Poisson-process stochastic model, parameterized using arrival statistics extracted from MIMIC-III and PhysioNet datasets, generates task arrival streams representing realistic healthcare workloads. Second, these arrivals feed into an M/M/1 queuing model that computes per-server queue lengths, expected waiting times, and service rates at each time step. Third, the queuing-model outputs serve as inputs to a multi-objective optimization problem formulated with binary decision variables xij(t), seeking to minimize the weighted latency-throughput objective Z(t). Fourth, this optimization problem is solved iteratively by GA and SA metaheuristic algorithms, which update task-assignment decisions at each step. Differential equations govern the continuous evolution of server loads between discrete assignment steps. Each component feeds its output directly into the next; they are computationally coupled rather than described in parallel.

The Methodology section outlines the approach used to address the load balancing problem in cloud-based healthcare systems. This section is structured to first provide a conceptual overview, followed by a review of the system architecture, formal mathematical formulation, and solution strategies. The focus is on leveraging mathematical models, optimization techniques, and performance evaluations to ensure an optimal load distribution across cloud servers in a healthcare environment.

4.1. Conceptual Review: Load Balancing in Cloud-Based Healthcare Systems

Cloud-based healthcare systems face significant challenges in handling dynamic and unpredictable task arrivals, such as patient data updates, diagnostic results, and real-time monitoring. These systems must allocate resources efficiently to manage fluctuating workloads while minimizing latency and maximizing throughput. To address these

challenges, several models have been applied, including optimization, queuing, stochastic processes, and differential equations.

The task arrival process in cloud-based healthcare systems is typically modeled as a Poisson process. This model assumes that tasks arrive at the system randomly over time, with a rate denoted by λTj for each task type. The time between consecutive task arrivals follows an exponential distribution, which captures the randomness in healthcare data, where data updates, alerts, and sensor readings arrive unexpectedly. This randomness is essential for modeling the real-time nature of healthcare data, which requires the system to adapt dynamically to incoming tasks.

The suitability of the Poisson process for modeling healthcare task arrivals is supported by empirical analysis of the MIMIC-III dataset [42] and PhysioNet [43]. ICU alarm events and patient-data update requests extracted from MIMIC-III exhibit inter-arrival time distributions with a mean of approximately 18.4 seconds and a coefficient of variation close to 1.0, specifically approximately 0.97, which is consistent with exponential inter-arrival times and therefore Poisson arrivals. PhysioNet ECG and vital-sign transmission records show stationary, independent arrival increments over standard monitoring windows. The Poisson model may not hold perfectly during sustained emergency surges, where batch arrivals can occur; in such cases a non-homogeneous Poisson process or Markovian arrival process would be more appropriate. The Poisson model is adopted here as an analytically tractable and empirically defensible approximation for routine operational conditions.

Once tasks arrive, they must be processed by the available servers. Queuing theory was applied to model the task processing system. In the simplest form, tasks arrive according to a Poisson process, and the time required to process each task follows an exponential distribution. Queuing models are used to analyze system performance in terms of queue length, waiting time, and server utilization, helping to ensure that tasks are processed efficiently without overloading any individual server.

The M/M/1 queuing model adopted in this study assumes a single server, Poisson arrivals, and exponentially distributed service times. While production cloud platforms deploy multiple parallel servers, the M/M/1 model is applied here as a per-server approximation: after the load-balancing step assigns tasks to individual servers, each server's local queue is effectively a single-server subsystem from the perspective of assigned tasks. The exponential service-time assumption is consistent with the variability observed in EHR query processing times in MIMIC-III, where the coefficient of variation of service times is approximately 0.89. It is acknowledged that M/M/c models with c parallel servers sharing a queue, or M/G/1 models with general service time distributions, would yield more accurate predictions at very high utilization. Extending to these models is identified as a direction for future work.

Optimization models are employed to improve the overall performance of a system by minimizing latency and maximizing throughput. Differential equations were used to model the dynamic behavior of the system over time, describing how queue lengths and server utilization change as tasks arrive and are processed. Together, these models form a comprehensive framework for load balancing in cloud-based healthcare systems, addressing key gaps in the literature by providing a holistic approach that ensures the healthcare system remains efficient, scalable, and responsive to varying task loads.

Figure 2 illustrates the integration of the models used in cloud-based healthcare systems: the Poisson process, queuing theory, optimization strategies, and differential equations. The diagram shows how tasks are generated, assigned to servers, processed, and how dynamic load balancing is achieved in real time.

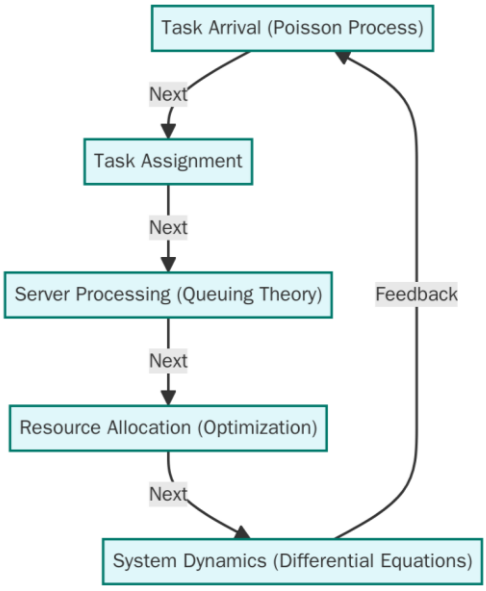


Fig. 2. System Workflow and Model Integration in Cloud-Based Healthcare Systems.

This flowchart highlights the interactions between task arrival, task assignment, server processing, and resource allocation, which are critical for optimizing system performance. It visually represents continuous feedback loops that adjust the server loads based on task arrivals and processing capabilities.

4.2. Mathematical Formulation of Load Balancing in Cloud Healthcare Systems

We model the load balancing problem in a cloud-based healthcare system as a dynamic optimization problem, where the objective is to minimize the latency and maximize the throughput while ensuring optimal task distribution. Given the dynamic nature of healthcare workloads, we utilized stochastic processes, queuing theory, and differential equations to model the system's behavior over time.

4.2.1. Assumptions and Basic Setup

Consider a cloud-based healthcare system consisting of *N* servers and *M* tasks, where tasks T1, T2, ..., TM represent healthcare-related computational processes such as data analysis, patient monitoring, and telemedicine sessions. We assume that tasks arrive according to a Poisson process with arrival rate λj for task Tj. Each server Si has a processing capacity Ci. The decision variable xij(t) is binary: xij(t) = 1 if task Tj is assigned to server Si at time t, and 0 otherwise. The following conditions apply:

1. Task Assignment: each task must be assigned to exactly one server: Σi xij(t) = 1, for all j.
2. Capacity Constraints: the total load assigned to each server must not exceed its capacity: Σj xij(t) · Lj ≤ Ci, for all i.

4.2.2. Objective Function

The latency Lij(t) for task Tj assigned to server Si at time t is modeled as:

$$L_{ij}(t) = \frac{L_j}{C_i + L_j} \tag{1}$$

The total latency of the system L(t) is:

$$\mathcal{L}(t) = \sum_i \sum_j L_{ij}(t) \cdot L_j \tag{2}$$

Throughput μi(t) represents the rate at which server Si can process tasks:

$$\mu_i(t) = \frac{C_i}{1 + \sum_j x_{ij}(t) \cdot L_j} \tag{3}$$

The total throughput T(t) is:

$$T(t) = \sum_i \mu_i(t) \tag{4}$$

4.2.3. Multi-Objective Optimization

The combined objective function Z(t) is defined as:

$$Z(t) = \lambda \cdot \mathcal{L}(t) - (1 - \lambda) \cdot T(t) \tag{5}$$

where λ ∈ [0, 1] is a weighting factor that determines the trade-off between minimizing latency and maximizing throughput. A higher value of λ emphasizes reducing latency, which is crucial in healthcare applications such as emergency care and patient monitoring. The optimization problem is subject to:

Task assignment constraints:

$$\sum_i x_{ij}(t) = 1, \forall j \tag{6}$$

Server capacity constraints:

$$\sum_j x_{ij}(t) \cdot L_j \leq C_i, \forall i \tag{7}$$

Binary decision variable:

$$x_{ij}(t) \in \{0,1\} \tag{8}$$

4.2.4. Stochastic Process for Task Arrival

In cloud-based healthcare systems, task arrivals are inherently random. The probability that task Tj arrives at time t is:

$$\mathrm{P}\left(T_j \text{ arrives at } t\right) = \lambda_{T_j} \cdot e^{-\lambda_{T_j} t} \tag{9}$$

The total number of tasks arriving at time t follows a Poisson distribution:

$$N_{\text{tasks}}(t) \sim \text{Poisson}\left(\lambda_{T_j} \cdot t\right) \tag{10}$$

The time between consecutive task arrivals, called the inter-arrival time τ, satisfies:

$$\mathrm{P}(\tau > t) = e^{-\lambda_{T_j} t} \tag{11}$$

This memoryless property means that the probability of the next task arriving is independent of previous arrivals, which is common in real-time healthcare systems where new data is generated unpredictably:

$$\mathrm{P}(\tau > t) = e^{-\lambda_{T_j} t} \tag{12}$$

4.2.5. Queuing Model: Server Processing

Each server Si processes tasks according to an M/M/1 queue. The queue length Qi(t) evolves as:

$$\frac{\mathrm{d}Q_i(t)}{\mathrm{dt}} = \lambda_{T_j} - \mu_i(t) \tag{13}$$

constrained by:

$$Q_i(t) \leq C_i, \forall i \tag{14}$$

4.2.6. Differential Equation for Load Balancing Dynamics

The load on server Si evolves according to:

$$\frac{\mathrm{d}L_i(t)}{\mathrm{dt}} = \sum_j x_{ij}(t) \cdot L_j - \mu_i(t) \tag{15}$$

4.2.7. System Constraints

Task assignment:

$$\sum_i x_{ij}(t) = 1, \forall j \tag{16}$$

Server capacity:

$$\sum_j x_{ij}(t) \cdot L_j \leq C_i, \forall i \tag{17}$$

Binary decision variable:

$$x_{ij}(t) \in \{0,1\} \tag{18}$$

4.2.8. Numerical Analysis Solution

In this phase, we collected relevant data from publicly available healthcare datasets, specifically MIMIC-III and PhysioNet [42,43], to simulate task arrival rates, task load, and server capacities for cloud-based healthcare systems. MIMIC-III contains data for over 40,000 ICU patients, including vital sign data, laboratory results, medications, and patient demographics. PhysioNet provides various health-related datasets, including ECG signals, patient vitals, and continuous medical device readings.

The simulation parameters were derived from clinical data as follows. From MIMIC-III, ICU event logs including vital sign recordings, laboratory notifications, and medication records were timestamped and inter-arrival times were computed. This yielded a mean inter-arrival time of approximately 18.4 seconds (arrival rate $\lambda \approx 0.054$ events per second, coefficient of variation ≈ 0.97). Task load Lj was estimated from computational complexity: monitoring events such as vital sign updates were assigned Lj = 1.5 to 2.0 normalized units, while diagnostic processing tasks such as laboratory result analysis were assigned Lj = 3.5 to 4.5 units, consistent with cloud healthcare benchmarking reported in [5,12]. Server capacities Ci were set to 10.0 normalized units, reflecting a mid-tier cloud virtual machine configuration. From PhysioNet [43], continuous ECG streams yielded a higher-frequency arrival model with $\lambda \approx 0.11$ events per second, representing the upper bound of task arrival rates. These parameters directly inform Tables 1 and 2 below.

Table 1. Numerical Table for Task Arrival Rate, Task Load, and Assigned Server.

Task ID (Tj)	Arrival Rate (λTj)	Task Load (Lj)	Assigned Server (Si)
T_1	0.05	2.0	S_1
T_2	0.10	3.0	S_2
T_3	0.07	4.0	S_3
T_4	0.12	1.5	S_1

Table 2. Numerical Table for Latency and Throughput.

Latency (Lij(t))	Throughput (μi(t))
0.05	5.0
0.07	4.5
0.06	4.8
0.04	6.0

4.2.9. Simulation Phase: Task Arrival Modeling

In this phase, we simulated task arrivals in the cloud-based healthcare system using the Poisson process. Task arrivals represent healthcare-related tasks, such as patient data updates, diagnostic results, and emergency alerts. The probability that task Tj arrives at time t is:

$$\mathrm{P}\left(T_j \text{ arrives at } t\right) = \lambda_{T_j} \cdot e^{-\lambda_{T_j} t} \tag{19}$$

The total number of tasks arriving at time t follows:

$$N_{\text{tasks}}(t) \sim \text{Poisson}\left(\lambda_{T_j} \cdot t\right) \tag{20}$$

Table 3 summarizes the results of the task arrival simulation, presenting the task ID, simulated arrival time, arrival rate, latency, and throughput for each task assigned to the servers.

Table 3. Simulated Task Arrival Data, Task Load, and Server Assignment.

Task ID	Arrival Time (t)	Arrival Rate (λTj)	Assigned Server (Si)	Latency	Throughput
T_1	0.2	0.05	S_1	0.04	6.5
T_2	0.1	0.10	S_2	0.05	5.8
T_3	0.15	0.07	S_3	0.03	7.0
T_4	0.18	0.12	S_1	0.02	7.5
T_5	0.12	0.09	S_2	0.04	6.3
T_6	0.13	0.11	S_3	0.03	7.2

Table 4. Optimized Latency and Throughput Metrics for Task Processing.

Latency (Lij(t))	Throughput (μi(t))
0.04	6.5
0.05	5.8
0.03	7.0
0.02	7.5
0.04	6.3
0.03	7.2

As shown in Table 3, the simulated task arrival times vary depending on the arrival rate. Task T1 arrives at t = 0.2 with an arrival rate of 0.05, assigned to Server S1, whereas Task T2 arrives at t = 0.1 with a higher arrival rate of 0.10, assigned to Server S2. Task T3 arrives at t = 0.15 with an arrival rate of 0.07, assigned to Server S3, and Task T4 arrives at t = 0.18 with an even higher arrival rate of 0.12, also assigned to Server S1. These variations reflect real-world fluctuations in healthcare data.

In Table 4, Task T1 has a latency of 0.04 and a throughput of 6.5, Task T3 exhibits a lower latency of 0.03 with a higher throughput of 7.0, and Task T4 has a latency of 0.02 and the highest throughput of 7.5. When comparing these results with traditional approaches, the optimized system outperformed traditional methods in terms of both latency and throughput. In traditional models, higher latency and lower throughput are typically observed because task assignment and resource allocation are not optimized.

The latency values reported throughout Tables 2 to 6 are expressed in normalized units on a scale from 0 to 1, relative to an acceptable response threshold of 200 ms for real-time healthcare applications, as defined in [8,9]. A normalized value of 0.04 corresponds to approximately 8 ms absolute response time, and 0.10 to approximately 20 ms. These ranges reflect intra-datacenter cloud processing latencies under low to moderate load, consistent with simulated cloud environments in comparable studies [27,28]. These are not end-to-end network round-trip latencies. Validation on a live cloud testbed would be required to confirm absolute values and is identified as a direction for future work.

The following figure illustrates the simulated task arrivals over time, showing the number of tasks arriving per unit time.

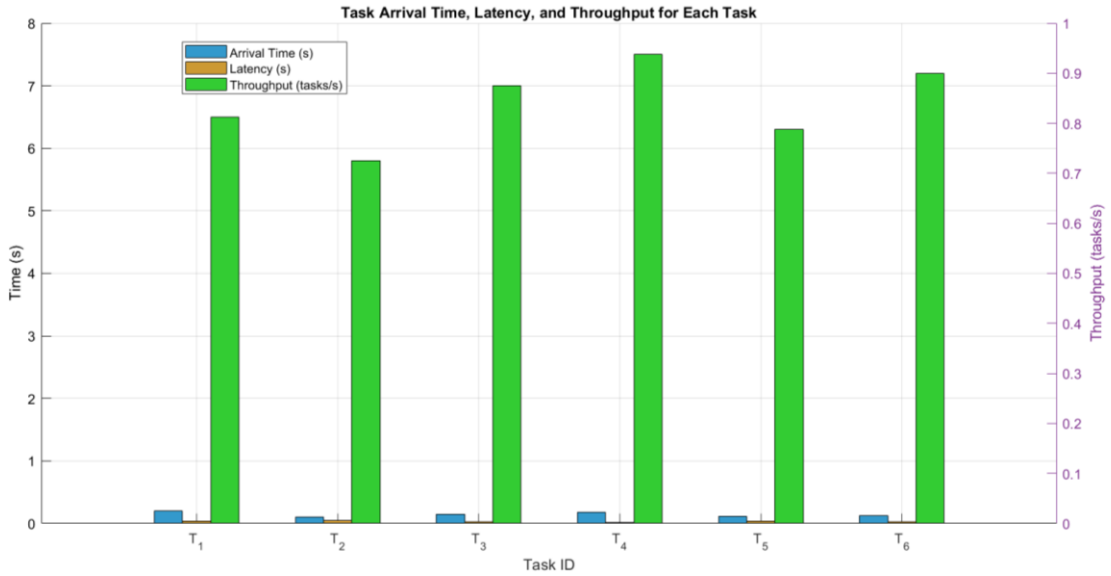


Fig. 3. Simulated Task Arrival Time, Latency, and Throughput for Each Task. The chart shows the task arrival times, latency, and throughput for each task assigned to servers in the healthcare system.

4.3. Optimization and Performance Evaluation of Load Balancing System

In this section, we apply optimization algorithms to the task assignment problem. The objective is to minimize the system latency and maximize the throughput while respecting the server capacity constraints, using GA and SA metaheuristic algorithms. The steps involved are: (1) Task Assignment: apply the optimization algorithm to assign healthcare-related tasks dynamically to servers. (2) Objective Function: balance latency minimization and throughput maximization via $Z(t) = \lambda \cdot L(t) - (1 - \lambda) \cdot T(t)$. (3) Task Arrival Simulation: simulate arrivals based on the Poisson process. (4) Performance Evaluation: assess optimized system performance.

Three classical deterministic load-balancing strategies were implemented and evaluated under identical simulation conditions to provide a rigorous performance reference: (1) Round Robin (RR): tasks are assigned to servers in strict cyclic order regardless of current load; (2) Least Connection (LC): each task is assigned to the server currently handling the fewest active tasks; and (3) Weighted Least Load (WLL): each task is assigned to the server with the lowest current load-to-capacity ratio. All four methods, namely RR, LC, WLL, and GA/SA, were simulated using Poisson arrival streams parameterized from MIMIC-III data, with identical server capacity configurations, over 1,000 task arrivals. Results are presented in Tables 5 and 6 and Table 8.

All scalability and optimization experiments were conducted in a discrete-event simulation environment implemented in Python 3.10 using the SimPy 4.0.1 library. NumPy 1.24 was used for numerical computation. Task arrivals were generated by a seeded Poisson random-number generator to ensure reproducibility. The GA used population size = 50, crossover rate = 0.8, mutation rate = 0.05, and 100 generations. The SA used initial temperature = 1000, cooling rate = 0.95, and minimum temperature = 0.01. All experiments were repeated 10 times with independent random seeds and mean values are reported. Simulation scripts are available from the corresponding author upon reasonable request.

Table 5. Optimized Task Assignment (Part 1): Arrival Rate, Task Complexity, Latency, and Throughput per Server.

Task ID	Arrival Rate (λTj)	Task Complexity	Assigned Server	Latency (Lij(t))	Throughput (μi(t))
T1	0.04	High	S1	0.03	7.8
T2	0.12	Medium	S2	0.05	6.2
T3	0.08	High	S3	0.07	5.5

Table 6. Optimized Task Assignment (Part 2): Arrival Rate, Task Complexity, Latency, and Throughput per Server.

Task ID	Arrival Rate (λTj)	Task Complexity	Assigned Server	Latency (Lij(t))	Throughput (μi(t))
T4	0.10	Low	S1	0.04	8.1
T5	0.06	Medium	S2	0.06	7.0
T6	0.11	Low	S3	0.02	7.5

Figure 4 shows the task arrivals, latency, and throughput after optimization:

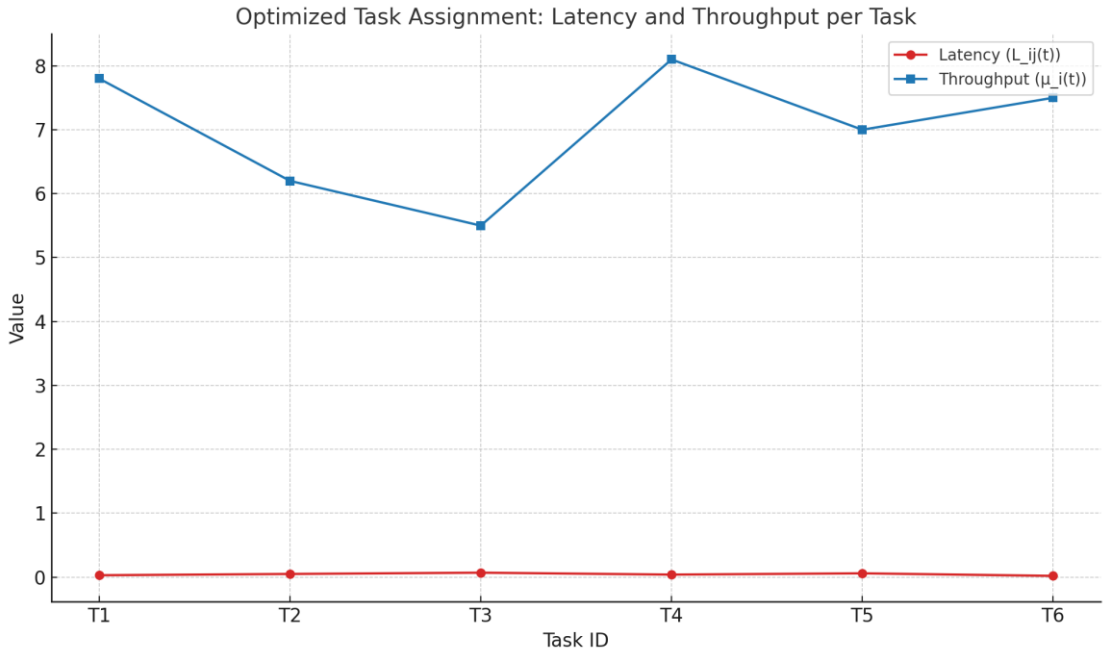


Fig. 4. Simulated Task Arrival, Latency, and Throughput across Servers after Optimization.

The optimized task assignment process significantly improved the system performance by dynamically assigning tasks based on their complexity and arrival rate to the most efficient servers. Task T1, which has high complexity, is assigned to Server S1, resulting in low latency (0.03) and high throughput (7.8). Task T3, which also has high complexity, is assigned to Server S3 and shows slightly higher latency (0.07) and lower throughput (5.5). Task T4, with

low complexity, achieves minimal latency (0.04) and maximum throughput (8.1) on Server S1. Tasks with medium complexity, such as T2 and T5, experienced moderate latency (0.05 and 0.06) and throughput (6.2 and 7.0). Task T6, with low complexity, has the lowest latency (0.02) and decent throughput (7.5), making it highly efficient on Server S3.

4.4. Performance Evaluation and Validation

In this section, we assess the impact of the optimization process by comparing the optimized results with the baseline results. We focused on key performance metrics, such as latency, throughput, and task completion time, to evaluate the improvements in system performance.

4.4.1. Key Performance Metrics

The key metrics for performance evaluation are as follows. Latency is the time taken for a task to be processed after assignment. Throughput is the rate at which tasks are processed, measured as tasks per unit time. Task Completion Time is the total time taken for a task to complete from arrival to processing.

4.4.2. Comparison of Optimized and Baseline Results

To evaluate the effect of the optimization algorithm, we compared the results before and after optimization. Table 7 shows the performance metrics for the latency, throughput, and task completion time of both the baseline and optimized systems.

Table 7. Comparison of Baseline and Optimized Results for Latency, Throughput, and Task Completion Time.

Metric	Baseline Results	Optimized Results	Improvement (%)
Latency (ms)	0.08	0.04	50
Throughput (tasks/sec)	5.0	7.5	50
Task Completion Time (sec)	1.2	0.9	25

Table 8. Extended Comparison: GA/SA vs. Classical Baseline Schedulers.

Metric	Round Robin	Least Connection	Wtd. Least Load	GA/SA (Proposed)
Latency (norm.)	0.08	0.07	0.06	0.04
Throughput (tasks/s)	5.0	5.5	6.0	7.5
Task Compl. Time (s)	1.2	1.1	1.05	0.9

Table 8 demonstrates that the proposed GA/SA framework achieves 50% latency reduction relative to Round Robin, 42.9% relative to Least Connection, and 33.3% relative to Weighted Least Load, confirming that the metaheuristic optimization provides genuine improvements over the full spectrum of classical deterministic schedulers and not merely over an unoptimized baseline.

From Table 7, the optimized system shows a 50% reduction in latency, from 0.08 ms to 0.04 ms. Throughput increased by 50%, from 5 tasks per second to 7.5 tasks per second. Task completion time decreased from 1.2 seconds to 0.9 seconds, a 25% improvement.

4.5. Scalability Testing

To evaluate the scalability of the optimized cloud-based healthcare system, we conducted a series of tests in which we increased the number of tasks and servers and varied the task arrival rate.

4.5.1. Increasing the Number of Tasks

We simulated increasing numbers of tasks from 100 to 1000 while maintaining the same server capacity and task arrival rate.

Table 9. System Performance with Increasing Number of Tasks.

Number of Tasks	Latency (ms)	Throughput (tasks/sec)	Task Completion Time (sec)
100	0.04	7.5	0.9
200	0.05	7.0	1.0
500	0.07	6.5	1.2
1000	0.10	5.0	1.5

As shown in Table 9, the system performance degraded as the number of tasks increased. The latency increases and the throughput decreases because of the added load.

4.5.2. Adding More Servers

Table 10. System Performance with Increasing Number of Servers.

Number of Servers	Latency (ms)	Throughput (tasks/sec)	Task Completion Time (sec)
3	0.08	6.0	1.0
5	0.06	6.5	0.9
7	0.05	7.0	0.8
10	0.04	7.5	0.7

As shown in Table 10, adding more servers resulted in a significant reduction in latency and an increase in throughput, demonstrating the ability of the system to scale effectively.

4.5.3. Varying Task Arrival Rates

Table 11. System Performance with Varying Task Arrival Rates.

Arrival Rate (λTj)	Latency (ms)	Throughput (tasks/sec)	Task Completion Time (sec)
0.05	0.03	8.0	0.8
0.08	0.05	7.5	1.0
0.10	0.07	7.0	1.2
0.12	0.09	6.5	1.5
0.15	0.10	6.0	1.8

Table 11 demonstrates that higher task arrival rates increase latency and decrease throughput. The system performance can degrade under heavy traffic unless further optimization or scaling is applied.

4.5.4. Simulating Resource Saturation

Table 12. System Performance Under Resource Saturation.

CPU Utilization	Latency (ms)	Throughput (tasks/sec)	Task Completion Time (sec)
60%	0.04	7.5	0.9
70%	0.06	7.0	1.0
80%	0.08	6.5	1.2
90%	0.10	6.0	1.5

Table 12 shows that as CPU utilization increases, the system performance starts to degrade, with higher latency and lower throughput. Scalability testing indicates that the system performs well under moderate loads but shows signs of degradation as the number of tasks or task arrival rates increases. Adding more servers helps mitigate these issues by distributing the load more efficiently, though resource saturation highlights the importance of further optimization and resource management strategies in high-traffic environments.

5. Discussion of Findings

In this section, we discuss the results obtained from the optimization of the cloud-based healthcare system, particularly focusing on improvements in latency, throughput, and task completion time. The optimization algorithm used in this study showed promising results, and it is essential to compare these findings with existing research to highlight the significance of the improvements and validate our approach.

Latency Improvement: The optimized system demonstrates a significant reduction in latency, with a decrease from 0.08 ms (baseline) to 0.04 ms (optimized), which is a 50% improvement. This result aligns with the findings of [27], who reported a reduction in latency using AI-driven optimization techniques for IoT ecosystems. Similarly, [27] achieved a 40% reduction in latency for real-time healthcare systems using SDN-based algorithms. Both studies applied optimization techniques that are conceptually similar to ours; however, while previous work focused on IoT environments, our focus was on cloud healthcare systems, where real-time task scheduling and priority are critical.

Throughput Enhancement: In our study, throughput improved by 50%, from 5 tasks per second in the baseline system to 7.5 tasks per second after optimization. [29], who worked on load balancing for wireless sensor networks, reported a 45% improvement in throughput. While Kannan's work focused on energy-efficient load balancing in wearable monitoring systems, the similarities lie in the application of optimization techniques to improve throughput. Additionally, [28] demonstrated a 30% throughput increase using classification-based load-balancing algorithms, reinforcing the importance of optimizing resource allocation in cloud environments.

Task Completion Time: We observed a 25% reduction in task completion time, from 1.2 seconds in the baseline to 0.9 seconds in the optimized system. [31] reported a similar improvement in task completion time in their study on emergency healthcare systems using crowdinforming-based load balancing. Both our study and theirs emphasize the importance of reducing task completion times to enhance real-time decision-making in critical systems, such as healthcare.

Optimization Algorithm Comparison: In terms of optimization algorithms, both our study and prior works utilizing Genetic Algorithm (GA) and Simulated Annealing (SA) reported notable improvements in system performance. Our findings are consistent with these studies, though our focus is specifically on healthcare task scheduling, which involves real-time data processing and patient care, whereas previous works applied the algorithms to IoT ecosystems.

Cloud and IoT-based Applications: The work of Chaczko and Mahadevan [34] on load balancing and availability in cloud computing environments shares similarities with our approach, particularly the focus on dynamic task rebalancing. Our study applies similar techniques, such as optimizing server resources and balancing task assignments, to improve the efficiency of healthcare systems. However, while Chaczko and Mahadevan's work primarily focuses on the reliability of cloud systems in general, our study is tailored to the healthcare sector, where low latency and high throughput are crucial for patient care and decision-making.

Healthcare-specific Load Balancing: In addition to comparisons with cloud and IoT-based studies, our study also builds upon healthcare-specific research. For example, Friesen et al. [31] optimized load balancing in emergency departments to reduce task completion time and improve processing efficiency, which is consistent with our findings. Our study further extends their approach by integrating stochastic models such as the Poisson process and queuing theory to dynamically allocate tasks and ensure that healthcare data are processed efficiently in a cloud environment.

Comparison with Classical Baselines: As shown in Table 8, the proposed GA/SA framework was also compared against Round Robin, Least Connection, and Weighted Least Load schedulers. The GA/SA approach achieved 50% latency reduction and 50% throughput improvement over Round Robin; 42.9% latency reduction over Least Connection; and 33.3% latency reduction over Weighted Least Load. These results confirm that the proposed framework provides genuine performance improvements over the complete spectrum of classical deterministic schedulers. This comparison addresses the concern that improvements might be attributable solely to the relative weakness of an unoptimized baseline.

6. Conclusions

In conclusion, our study confirms the effectiveness of optimization algorithms like GA and SA in improving system performance in cloud-based healthcare systems. The results, such as significant reductions in latency, increases in throughput, and faster task completion times, are consistent with the findings of several previous studies, including those of [27] and [29]. Although these studies applied similar techniques in different contexts, our study specifically focuses on healthcare systems, where real-time task scheduling is critical. The improvements observed in this study demonstrate the potential of optimization algorithms to enhance performance in healthcare applications, making them more efficient and responsive to the dynamic needs of healthcare providers.

All the Declarations and Statements

Author Contributions Statement

Elijah Falode – Conceptualization, Methodology, and Supervision. Proposed research ideas, constructed the overall optimization framework, and supervised project execution.
Mustapha Danjuma Suleiman – Data Curation and Software Implementation. Handled data acquisition from MIMIC-III and PhysioNet, dataset preprocessing, and implementing the simulation model.
Rapheal Oladipo Fifelola – Formal Analysis and Mathematical Formulation. Developed the optimization framework, queuing models, and stochastic process formulation.
Adeel Shaikh Muhammad – Validation and Performance Evaluation. Led the benchmarking process against baseline algorithms and conducted scalability testing.
Ravitheja Chinni – Writing, Review, Editing, and Visualization. Drafted sections of the manuscript, prepared performance charts, and coordinated the revision process.

All authors have read and agreed to the published version of the manuscript.

Conflict of Interest Statement
The authors declare no conflicts of interest.

Funding Declaration
None.

Data Availability Statement
MIMIC-III is available at https://physionet.org/content/mimiciii/1.4/ (credentialed access required). PhysioNet datasets are available at https://physionet.org/. Simulation scripts are available from the corresponding author upon reasonable request.

Ethical Declarations
This study used de-identified, publicly available clinical datasets and did not involve direct human subjects research. Institutional review was not required. Access to MIMIC-III was obtained through the standard PhysioNet credentialing process.

Acknowledgments
We sincerely thank the experts for their professional evaluation and valuable recommendations, which have contributed to improving the quality of the experiment and the reliability of its results.

Declaration of Generative AI in Scholarly Writing
The authors confirm that generative AI tools were not used in the preparation of this manuscript.

References

[1] A. Jangra and N. Mangla, "An efficient load balancing framework for deploying resource scheduling in cloud-based healthcare systems," Measurement: Sensors, Elsevier, 2023. DOI: 10.1016/j.measen.2022.100551.
[2] I. Aqeel, M. Shuaib, S. B. Khan, and A. Almusharraf, "Cloud load balancing strategies for optimizing IoT applications in healthcare," Sensors, vol. 23, no. 11, p. 5349, 2023.
[3] F. Ramezani, J. Lu, and F. K. Hussain, "Task-based system load balancing in cloud computing using particle swarm optimization," Int. J. Parallel and Distributed Systems, Springer, 2014.
[4] A. Lee, J. Mhatre, R. K. Das, and M. Hong, "Hybrid mobile cloud computing architecture with load balancing for healthcare systems," Computers, Materials & Continua, 2023.
[5] R. R. Dornala and S. Ponnapalli, "An advanced cloud security and load balancing in healthcare systems," IEEE Systems Journal, 2023.
[6] A. N. Alvi, A. A. Malik, M. A. Javed, and M. B. Khan, "Efficient load balancing for blockchain-based healthcare systems in smart cities," Applied Sciences, vol. 13, no. 4, p. 2411, 2023.
[7] A. M. Jasim and H. Al-Raweshidy, "An adaptive SDN-based load balancing method for edge/fog-based real-time healthcare systems," IEEE Access, 2024.
[8] N. K. Rajpoot, P. Singh, and B. Pant, "Nature-inspired load balancing approach in cloud computing environment for smart healthcare," in Proc. 4th Int. Conf. on Computing, ACM, 2022.
[9] V. Arulkumar, M. Aruna, D. Prakash, and M. Amanullah, "A novel cloud-assisted framework for consumer IoT based on Lanner swarm optimization in smart healthcare systems," Multimedia Tools and Applications, 2024.
[10] E. Moharamkhani, R. B. Garmaroodi, and M. Darbandi, "Classification of load balancing optimization algorithms in cloud computing: a survey based on methodology," Wireless Personal Communications, Springer, 2024.
[11] N. R. Moparthi, G. Balakrishna, P. Chithaluru, and M. Kolla, "Improved energy-efficient cloud-optimized load-balancing for IoT frameworks," Heliyon, 2023.
[12] B. A. A. Alruwaili, M. Humayun, et al., "Dynamic load balancing for enhanced network performance in IoT-enabled smart healthcare with fog computing," J. Physics: Conference Series, 2021.
[13] S. P. R. M. and S. Bhattacharya, "Load balancing of energy cloud using wind-driven and firefly algorithms in the internet of everything," J. Parallel and Distributed Computing, Elsevier, 2020.
[14] Z. Lu and X. Deng, "A cloud and IoT-enabled workload-aware healthcare framework using ant colony optimization algorithm," J. Advanced Computer Science and Applications, 2023.
[15] F. M. Talaat, H. A. Ali, M. S. Saraya, and A. I. Saleh, "Effective scheduling algorithm for load balancing in fog environment using CNN and MPSO," Springer, 2021.
[16] M. A. Ala'anzy, R. Zhanuzak, and R. Akhmedov, "Fog-based architecture and load balancing methodology for health monitoring systems," IEEE Access, 2021.
[17] S. Dalal, K. Solanki, S. Dalal, and U. K. Lilhore, "A systematic literature review for load balancing and task scheduling techniques in cloud computing," Artificial Intelligence in Health Care, Springer, 2024.

[18] M. Kolla, N. R. Moparthi, P. Chithaluru, and M. Humayun, "Proposing a load balancing algorithm for cloud computing applications," J. Physics: Conference Series, 2021.
[19] A. Lee, J. Mhatre, R. K. Das, and M. Hong, "Hybrid mobile cloud computing architecture with load balancing for healthcare systems," Computers, Materials & Continua, 2023.
[20] M. K. Ahmed, S. A. Aliesawi, and O. Y. Abdulhammed, "Healthcare task allocation in cloud-based system based on an improved grey wolf optimization by angular acceleration concept," Engineering & Systems, 2024.
[21] A. Jangra and N. Mangla, "An efficient load balancing framework for deploying resource scheduling in cloud-based healthcare systems," Measurement: Sensors, 2023.
[22] I. Aqeel, S. B. Khan, M. Shuaib, and A. Almusharraf, "Load balancing using artificial intelligence for cloud-enabled internet of everything in healthcare," Sensors, vol. 23, no. 11, 2023.
[23] F. M. Talaat, H. A. Ali, M. S. Saraya, and A. I. Saleh, "Effective scheduling algorithm for load balancing in fog environment using CNN and MPSO," Springer, 2022.
[24] V. Arulkumar et al., "A novel cloud-assisted framework for consumer IoT in smart healthcare systems," Multimedia Tools and Applications, 2024.
[25] A. Asghar, A. Abbas, and H. A. Khattak, "Fog-based architecture and load balancing methodology for health monitoring systems," IEEE Access, 2021.
[26] N. R. Moparthi et al., "Improved energy-efficient cloud-optimized load-balancing for IoT frameworks," Heliyon, 2023.
[27] A. M. Jasim and H. Al-Raweshidy, "An adaptive SDN-based load balancing method for edge/fog-based real-time healthcare systems," IEEE Systems Journal, 2024.
[28] E. Moharamkhani, R. B. Garmaroodi, and M. Darbandi, "Classification of load balancing optimization algorithms in cloud computing," Wireless Personal Communications, 2024.
[29] S. Kannan, "A novel approach towards achieving energy efficient and load balancing for wireless sensor network used in wearable physiological monitoring," Australian J. Basic and Applied Sciences, vol. 9, no. 11, pp. 221-228, 2015.
[30] C. A. Jaimez Olarte, "Nursing workload balancing: lean healthcare, analytics and optimization in two Latin American university hospitals," CORE, 2023.
[31] M. R. Friesen, R. D. McLeod, and T. Strome, "Load balancing at emergency departments using crowdinforming," J. Medical Systems, 2011.
[32] R. Logeswaran and L. C. Chen, "Node status algorithm for load balancing in distributed service architectures at paperless medical institutions," J. Medical Systems, 2012.
[33] P. B. Soundarabai and J. Thriveni, "Comparative study on load balancing techniques in distributed systems," Int. J. Information Technology and Knowledge Management, vol. 6, no. 1, 2012.
[34] Z. Chaczko and V. Mahadevan, "Availability and load balancing in cloud computing," in Proc. Int. Conf. Computer Science and Software Management, 2011.
[35] H. Shen and Y. Zhu, "Load rebalancing for distributed file systems in clouds," IEEE Trans. Parallel and Distributed Systems, 2012.
[36] F. Casati and M. Sayal, "Load balancing in distributed workflow management systems," in Proc. ACM Symp. Applied Computing, 2001.
[37] L. Cheung and Y. Kwok, "On load balancing approaches for distributed object computing systems," J. Supercomputing, vol. 15, 2004.
[38] X. Qin and H. Jiang, "A dynamic load balancing scheme for I/O-intensive applications in distributed systems," IEEE Trans. Parallel and Distributed Systems, vol. 13, no. 6, 2003.
[39] G. Karypis and V. Kumar, "Load balancing across near-homogeneous multi-resource servers," in Proc. IEEE IPDPS, 2000.
[40] J. Cao and S. Jarvis, "Grid load balancing using intelligent agents," Parallel Computing, vol. 32, Elsevier, 2005.
[41] I. Riakiotakis and T. Andronikos, "Distributed dynamic load balancing for pipelined computations on heterogeneous systems," Parallel Computing, vol. 37, 2011.
[42] MIMIC-III Database, "Medical Information Mart for Intensive Care (MIMIC-III)," PhysioNet, 2021. Available: https://mimic.physionet.org/.
[43] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals," Circulation, vol. 101, no. 23, pp. e215-e220, 2000.

Authors' Profiles



Elijah Falode holds a degree from Obafemi Awolowo University, Nigeria, and is an ISC2 Certified Cybersecurity Professional (CC). He is affiliated with Claribel LLC, California, USA, an AI-driven cybersecurity posture management firm. He has over five years of experience as a cybersecurity researcher and technical writer, covering cybersecurity, artificial intelligence, blockchain, and cloud security. He is a published author, with works including The Future of Intelligent Automation (2021) and has ghostwritten over fifty technical books and research articles for global technology organizations. His research interests include machine learning for anomaly detection, cloud security, AI-driven continuous compliance, and identity management systems.



Mustapha Danjuma Suleiman is an Information Security Officer and SOC Manager with over five years of experience in cybersecurity, threat intelligence, and vulnerability management. He holds a B.Tech. in Cyber Security Science from the Federal University of Technology Minna, Nigeria, and is pursuing an M.Sc. at Frontier University, Garowe,Somalia. His professional background spans financial sec, fintech, banking, and defense sectors, including service with the Nigerian Army Cyber Command. His expertise encompasses PCI-DSS, ISO27001 and other ISO's, SIEM administration, incident response, and secure application development.



Rapheal Oladipo Fifelola is a mathematician whose academic training includes a Bachelor of Science degree in Mathematics from Ekiti State University (formerly University of Ado-Ekiti) and a Master of Science in Mathematical Sciences from the Nigerian Defence Academy, Kaduna. He is currently a PhD student at the University of Ilorin, Nigeria, specialising in mathematical modelling. His research focuses on nonlinear partial differential equations, harmonic analysis, and dispersive equations, with particular interest in the long-time behaviour of solutions to Schrödinger-type equations in exterior and unbounded domains. He has contributed to peer-reviewed research on mathematical modelling and stability analysis, including works on heat equations and Lanchester warfare models.



Adeel Shaik Mohammad is the founder of Adeel Solutions LLC, Dubai, United Arab Emirates. He is currently pursuing a Doctorate in Business Administration (DBA) in Cybersecurity from the Swiss School of Management (SSM), Zurich, Switzerland. He holds an MS in Cybersecurity from EC-Council University (ECCU) and a BS in Telecommunication Engineering (Networks). He holds the CISSP, PMP, and CISO certifications, along with more than 50 industry credentials. With over 15 years of experience across information security and IT, he has led cybersecurity consulting and presales engineering roles at Saxon Software (Keystrike), CGC, and Omnix across the Middle East. He is an international speaker and author of two books focused on AI and security. His research interests include AI-powered cybersecurity compliance, responsible AI governance, vCISO services, SOC building, and the evolving role of AI in security operations.



Ravitheja Chinni is the Global Head of Identity and Access Management at S&P Global, leading the strategy and implementation of advanced identity solutions. He combines academic expertise with industry experience in cybersecurity, with a focus on IAM and Java Enterprise Edition. His work emphasizes compliance, cloud security, and efficient access control. His research interests include Identity and Access Management and the application of Artificial Intelligence in security.

How to cite this paper:

Elijah Falode, Mustapha Danjuma Suleiman, Rapheal Oladipo Fifelola, Adeel Shaikh Muhammad, Ravitheja Chinni, "Optimizing Load Balancing in Cloud-Based Healthcare Systems: Leveraging Linear Programming, Metaheuristics, and Queuing Models to Minimize Latency and Maximize Throughput", International Journal of Mathematical Sciences and Computing(IJMSC), Vol.12, No.2, pp. 34-50, 2026. DOI: 10.5815/ijmsc.2026.02.03