Optimized Scheduling of Jobs using Load Balancing in Cloud Computing With Economic Perspectives

Marwa Mostafa Sabry*, Assem Tharwat and Doaa Wafik*****

*Business Analytics Department, Egypt University of Informatics, Knowledge City, New Administrative Capital, 15702.
**Operations Research and Decision Support, Faculty of Computers and Artificial Intelligence, Cairo University.
***Abu Dhabi University, United Arab Emirates.

Optimized Scheduling of Jobs using Load Balancing in Cloud Computing (With Economic Perspectives)

The rapid growth of computing systems has primarily focused on performance and fast application processing across customer, scientific, and business domains. However, this progress has led to increased energy consumption, particularly in cloud data centres that host large-scale applications. Optimizing energy use in such environments remains a significant challenge. This work presents an enhanced task scheduling approach based on deadlines and resource management to address this issue. The proposed algorithm utilizes K-means clustering to classify tasks and virtual machines (VMs), overcoming limitations of existing scheduling techniques. By considering machine load prior to task assignment, the algorithm effectively reduces makespan, average waiting time, response time, and improves throughput. Simulation results demonstrate that the method enhances resource allocation efficiency, leading to faster processing and improved responsiveness—benefiting both cloud service providers and users. The proposed optimization approach also holds economic significance by potentially lowering operational costs for cloud service providers through energy efficiency and resource utilization. Reduced energy consumption and improved infrastructure performance translate directly into cost savings, enhancing the financial sustainability of cloud operations.

Keywords: Cloud Computing, Load Balancing, Task Scheduling, K-Mean Algorithm, Resource Optimization, Energy Efficiency, Economic Impact

INTRODUCTION TO CLOUD COMPUTING

When the IT sector expands daily, there is a massive rise in computing demands and storage requirements. There are regular shifts in data exchange across the network. The growing volume of data, therefore, requires additional computer equipment to satisfy the organization's diverse demands (Gajbhiye and Shrivastva, 2014) [21]. To make greater use of their resources, over-equipped organizations, by using the internet and other similar technology, such as virtualization to build a modern technological model, cloud computing, open their infrastructures to others. Cloud computing (Ardagna, 2015) [7] is a modern development in IT that transfers computing as well as knowledge to large data centres from desktops and portable PCs. With the help of cloud computing, individuals may link to the cloud via Internet through their personal computers or portal devices to mirror their data through their cloud account. The fundamental concept of cloud computing is providing a resources-sharing platform that includes virtualized software and infrastructure. Virtualization means that a virtual version of a machine or resource is generated in a machine, such as a storage device, a server, a network, or even an operating system.

One of the biggest trends in the history of computing is the rise of cloud computing in recent years (Bamiah, 2015) [11]. The development commonly referred to as cloud computing is a fundamental

change in the inventory, development, deployment, scalability, update, and maintenance of information technology services. It has moved from parallel computing to distributed computing to grid computing to cloud computing. The primary benefit of cloud computing is that customers don't need to deploy its infrastructure and the main power required for its maintenance (Ziglari and Yahya, 2016) [72].

It uses its name as the Internet' (Abomasa and Jazzar, 2017) [2]. In the network diagrams, the Internet is typically represented as a cloud. The word "cloud" comes from the telephony world. Cloud is typically a single-party set of computers. The cloud is often referred to as "Web Computing". Cloud computing provides consumers with online access to resources from anywhere, from anywhere, without concern for engineering / physical management and the upkeep of indigenous resources. Cloud computing capabilities are diverse and scalable too. Cloud computing is independent computing for non-grid and service computing. Web Apps, supported by Google and Microsoft SharePoint, are an example of cloud computing that allows users to view information over the Internet through a website installed on millions of smartphones. Null operating costs are used because service suppliers are responsible for operation quality, and customers are free of electronic system management and control issues. The calculation is more straightforward than any other computational process. It is also known as on-demand computing services or IT services.

Economically, cloud computing provides cost advantages by eliminating capital expenditures (CAPEX) and shifting costs to an operational expenditure (OPEX) model. This enables businesses to pay only for what they use, optimizing budget allocation and improving financial predictability. Furthermore, smaller enterprises gain access to high-end computing services without massive investments, promoting digital equity and broader market competitiveness.

Recent advancements in cloud computing have further simplified the deployment and management of IT resources, as highlighted in recent studies such as Ahmed and Raza (2023) [72], who emphasize the scalability and flexibility offered by modern cloud platforms, and Smith et al. (2022) [73], who discuss the cost-efficiency and reduced infrastructure complexity for businesses leveraging these services.

I DEPLOYMENT MODELS

The deployment model refers to cloud infrastructure location and management. Depending on organizational structure and location, cloud resources can be implemented in various ways. The biggest challenge is to determine the type of cloud that is to be used for the implementation of a cloud solution. Three types of cloud deployment are shown in Figure 1 (Kurniawan et al 2022) [81].

Each cloud deployment model—public, private, and hybrid—has distinct economic implications. Public clouds often reduce capital expenditure (CAPEX) by shifting infrastructure costs to providers, making them ideal for startups and cost-sensitive organizations. Private clouds, while requiring higher upfront investments, offer greater control and may be more cost-efficient for large enterprises with consistent, high-demand workloads. Hybrid models strike a balance, allowing organizations to optimize cost-efficiency by leveraging public resources for dynamic scaling while maintaining critical services in a controlled private environment. Choosing the right deployment model is therefore not only a technical decision but also a strategic financial one, directly influencing an organization's operational expenditure (OPEX), scalability, and return on investment (ROI).

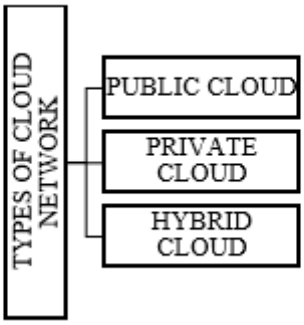


FIGURE 1. TYPES OF CLOUD NETWORK

II SERVICE MODELS IN CLOUD COMPUTING

Cloud computing services are a concept that allows reusable, sophisticated components to be used across the supplier network. The capabilities of cloud computing are typically divided into three categories: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), as shown in Figure 2 (Mell et al 2011) [80].

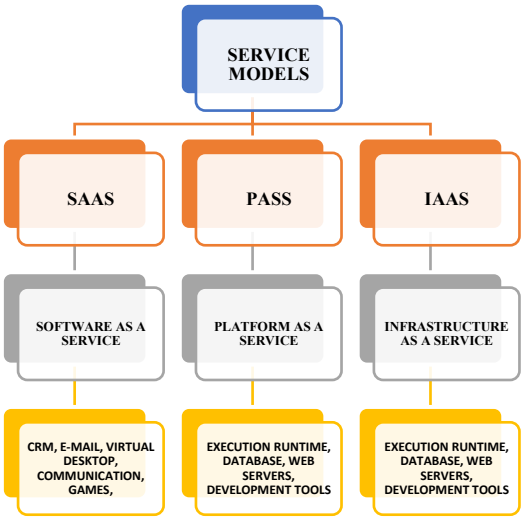


FIGURE 2. MAIN TYPES OF SERVICE MODELS

III PROS AND CONS ASSOCIATED WITH CLOUD COMPUTING

The various advantages as well as disadvantages in the cloud computing field are continuously explored. Recent studies, such as Sharma et al. (2023) [74], examine the enhanced security and privacy challenges in cloud environments. Additionally, Johnson et al. (2024) [46] discuss the improved cost efficiency and scalability benefits along with the potential risks of data breaches and service downtimes.

Table 1 Advantages as well as Disadvantages of Cloud Computing

Advantages	Cost Efficiency	Cloud computing is potentially the most expensive method of usage, update, and repair. Cloud services are relatively inexpensive

		for smaller businesses to utilize extra time and resources to improve the company's environment.
	Quick Deployment	Cloud computing offers the benefits of rapid deployment. The delivery time should depend on the exact kind of technology that the company requires.
	Unlimited Storage	Cloud data storage provides almost infinite storage capacity.
	Easy Access to Information	With the use of an Internet connection, individuals may effortlessly access information from any location in the globe after they have signed up for the cloud. Users may push challenges beyond temporal and spatial constraints using this functionality.
	Backup and Recovery of data	Since all of the data is kept in the cloud, backing it up and restoring it is more easier than storing it on a local computer. Furthermore, the majority of cloud service providers often possess the necessary qualifications to oversee data recovery. This means that compared to other traditional methods of data storage, the backup and recovery procedure is now simpler.
Disadvantages	Prone to Attacks	Storing the knowledge in the cloud will render companies that are vulnerable to external intrusion and attacks. As a result, there is always the risk of stealing confidential data.
	Security	The other major issue is cloud security. Conversion to cloud can improve security for any type of company.
	Technical Issues	Although there are indeed data and information one can access at any time and from anywhere. Only the most substantial cloud service providers manage technical issues, despite maintaining high maintenance standards.
	Cost	When many cloud device vendors describe themselves as providers based on utility services, a comparison of overall costs is needed; they claim that they only charge what customers use.
	Scheduling	Owing to the properties of marketing and virtualization, Cloud computing encounters scheduling complexity to the virtual machine layer by virtualization of the assets. Therefore, resource allocation is the most important and scheduling in cloud computing plays an important part.

IV LITERATURE REVIEW

Task scheduling is a critical component of cloud efficiency, aiming to assign tasks to appropriate virtual machines (VMs). It must also account for factors like service level agreements, deadlines, provider profit, budget, and load balancing. Various scheduling algorithms have been proposed, including Min–Min, K-means, Round-Robin, SJF, Max-Min, and Genetic algorithms.

Their chapter reviews prior work on cloud task scheduling, including comparisons between AFCFS and LJFS algorithms. AFCFS (Moschakis et al., 2011) [42] used fewer VMs, while LJFS prioritized the largest tasks. Simulations assumed VM counts remained under 120, excluding job migration, load, and task type.

K-Means clustering in cloud computing is explored using MapReduce (Xu et al., 2012) [66] for Hadoop-based text clustering. On two Hadoop-based DELL R410 servers, 8-node clusters achieved 7.99x speed gains over single-node setups. This Chinese clustering method significantly improved processing speeds.

Elzeki et al. (2012) [19] proposed an Improved Max-Min algorithm, blending RASA and Max-Min features. It showed reduced completion times compared to the original Max-Min and RASA methods.

Liu et al. (2013) [30] developed a real-time dynamic scheduling model focused on maximizing provider profit and minimizing energy use. Compared to standard multi-objective genetic algorithms, their approach reduced energy consumption while meeting task deadlines and increasing revenue.

Beyond technical efficiency, economic optimization has become a major focus in task scheduling. Kumar and Singh (2021) [86] proposed an energy- and cost-aware algorithm that allocates VMs based on real-time cloud market pricing. Their approach, using spot instances and dynamic bidding, reduced costs by over 30% compared to static provisioning. Similarly, Wang et al. (2018) [87] developed a profit-maximization framework balancing energy use and pricing to increase provider revenue while meeting deadlines. This shift highlights a growing emphasis on economic sustainability alongside performance in cloud resource management.

Chen et al. (2013) [13] proposed a load-balanced Min-Min algorithm (LBIMM) and a Priority-Aware LBIMM to enhance resource use and reduce makespan, addressing pay-per-use service differentiation. By reallocating tasks from heavily loaded resources to better alternatives, VIP satisfaction and resource utilization improved by over 20%. However, QoS, deadlines, and similar constraints were not considered.

Mathew et al. (2014) [38] reviewed energy-efficient, heuristic, and hybrid scheduling methods. Though briefly evaluated, the study suggests enhancing algorithms by integrating new metrics to improve performance in cloud environments.

To address VM load imbalance, Maruthanayagam and Prakasam (2014) [37] proposed ACO LB, which reduces execution time but overlooks consistency, priority, and cost. Due to VM performance variability, task assignments caused resource imbalances. ACO LB improved average resource usage but remains complex.

Rodriguez et al. (2014) [51] introduced a PSO-based metaheuristic for IaaS resource allocation in scientific workflows. The strategy met deadlines while cutting overall costs, outperforming existing approaches under varying workloads.

Saeed et al. (2014) [52] developed a hybrid job scheduling algorithm using fuzzy theory and an enhanced genetic algorithm. It reduced population generation iterations but did not lower execution costs.

Kun-lun et al. (2014) [27] proposed an improved Gene Expression Programming (GEP) algorithm using a novel ETCC matrix and a dual fitness function (DF-GEP) to optimize task scheduling by balancing runtime and cost.

Dorronsoro et al. (2014) [18] introduced a two-tier strategy for high-parallel workloads in multi-center distributed systems. It focused on fine-grained planning across data centers to optimize energy and system performance.

Lucio (2015) [31] enhanced the Min-Min algorithm for applications with significant network communication, allowing it to autonomously adapt to varying transmission speeds.

Another notable dimension in the literature is the integration of economic market models into cloud scheduling. Zhang et al. (2019) [88] developed a market-oriented cloud task scheduler that applies auction-based mechanisms, where users bid for computing resources based on urgency and

budget. Their approach introduces a competitive environment among users that not only enhances fairness in resource allocation but also increases overall provider revenue by optimizing price–demand dynamics. Similarly,

Mishra and Sahoo (2020) [89] investigated game-theoretic models in scheduling to align economic incentives between users and providers, promoting efficient use of underutilized resources. These studies highlight how economic theories—such as auctions, pricing strategies, and game theory—can be effectively adapted to enhance cloud infrastructure profitability, resource utilization, and user satisfaction simultaneously.

Awad et al. (2015) [9] proposed a mathematical model incorporating execution time, efficiency, makespan, transmission time, round-trip time, transmission cost, and load balancing between tasks and VMs. Using Load Balancing Mutation PSO (LBMPSO), based on PSO, the model improves cloud system stability by reallocating unassigned tasks. Compared to standard PSO and LCFP (Random Algorithm), LBMPSO reduces makespan, round-trip time, runtime, and cost.

Maciej et al. (2015) [34] introduced a heuristic greedy-critical algorithm for IaaS clouds, targeting minimal end-to-end delay within cost constraints. Using a Monte Carlo method, it optimizes time and cost via budget- and deadline-constrained workflow scheduling.

To reduce the Flexible Job Shop Problem (FJSP) makespan, Bakhtar et al. (2015) [10] proposed a Path Relation (PR) algorithm with GRASP for initial solutions. Tests on 15 benchmarks showed the method's effectiveness.

Suresh et al. (2015) [59] introduced the Load Distribution Paradigm for heterogeneous cloud systems using Divisible Load Theory (DLT). Load scheduling was modeled via linear programming and applied to satellite image classification in cloud environments.

Jena (2015) [23] proposed a multi-objective task scheduling method using nested PSO and linear programming. The approach analyzed satellite image classification in virtual cloud environments but required improvement in robustness and additional objectives (e.g., bandwidth, cost).

Li et al. (2015) [32] proposed energy-aware dynamic task scheduling (EDTS) to reduce smartphone energy use under runtime variation. Using a Data Flow Graph Critical Path (DFGCP) algorithm, the method met time constraints, though energy consumption remained moderately high.

Zhong et al. (2016) [71] presented GPSO for VM-based task scheduling, offering strong local/global search and fast convergence. Though effective in planning, it didn’t account for bandwidth and data transmission in task completion time.

Singh and Agnihotri (2016) [56] proposed an energy-efficient scheduling model using a hybrid approach for cloud systems. It aimed to optimize resource use, reduce runtime, and minimize SLA violations. Despite improved VM-job assignment, the overall makespan remained low.

Lakshmi and Srinivasu (2016) [28] proposed a genetic algorithm (GA) for dynamic task scheduling in cloud computing to minimize waiting time. Service requests from users are queued by providers and managed by a GA-based queue sequencer using round-robin scheduling and buffer queues. Though effective in reducing wait times, the method underperforms in single-resource environments due to reduced throughput.

For heterogeneous virtual clusters, Lin et al. (2016) introduced an energy-intensive scheduling model combining task prediction, energy estimation, and a buffer-based scheduler. The proposed VPEGS algorithm greedily scheduled tasks by estimating energy usage based on resource needs and VM efficiency. However, it didn’t address waiting time, response time, or makespan.

Devi and Uthariaraj (2016) [17] proposed an enhanced round-robin algorithm for load balancing in cloud systems, aiming to optimize VM performance by combining static and dynamic strategies. It reduced VM overload and migration but didn’t decrease job execution time.

Madni et al. (2016) [35] compared six heuristic algorithms (FCFS, MET, MCT, Min-Min, Max-Min, RR, and Sufferage) for task scheduling in IaaS cloud environments, evaluating cost, load imbalance, and performance in various conditions. Despite the insights, more advanced optimization methods were needed.

Deldari et al. (2016) [16] proposed a heuristic workflow scheduling algorithm focused on meeting user deadlines while minimizing implementation costs by clustering tasks and selecting optimal paths.

Kimpan and Kruekaew (2016) [24] introduced a Heuristic Artificial Bee Colony (HABC) algorithm for job scheduling and load balancing in heterogeneous VMs. The HABC aimed to improve VM mapping and scheduling efficiency, comparing various ABC-based approaches such as Large Job First (LGF).

Zheng and Wang (2016) [70] presented a Pareto-based Fruit Fly Optimization Algorithm (PFOA) for multi-objective task scheduling and resource allocation (TSRA). It used minimum-cost heuristics, Pareto sorting, and visual memory to generate non-dominated solutions. However, its high computational complexity was sensitive to parameter tuning.

Singh and Chana (2016) [57] reviewed 110 papers on resource scheduling, identifying 70 key studies. They analyzed algorithm classifications, QoS parameters, and resource management policies, offering a comprehensive comparison and taxonomy of scheduling strategies.

Rastkhadiv and Zamanifar (2016) [49] proposed a load balancing algorithm using the Artificial Bee Colony (ABC) approach to optimize task scheduling and resource use across VMs while minimizing makespan.

Yuan et al. (2016) [68] introduced CAWSAC, a cost-aware workload scheduling method that maximizes active servers and selects optimal ISPs for cloud data centers (CDCs), significantly reducing costs and improving CDC performance in simulations.

Qian et al. (2016) [47] proposed a load balancing algorithm based on feedback mechanisms and random weight assignment. The approach improved load distribution but suffered from high computational complexity.

Raj et al. (2016) [48] developed a hybrid load balancing algorithm combining Batch Mode Heuristic Priority and Round-Robin scheduling. It prioritized tasks based on VM load, dynamically updating factors to optimize task distribution across geographically distributed environments.

Xiong et al. (2016) [65] proposed an energy-optimized scheduling model for cloud-based video surveillance. The task scheduling problem was modeled as a multi-dimensional bin-packing problem and simplified using greedy, best-fit search techniques.

Abdullahi et al. (2016) [1] introduced DSOS, an improved Symbiotic Organism Search (SOS) algorithm for task scheduling. DSOS mimics natural symbiosis (mutualism, commensalism, parasitism) and showed improvements over PSO, though further optimization was needed.

Sampaio et al. (2016) [53] evaluated three advanced scheduling algorithms focused on improving energy efficiency and SLA compliance in practical cloud resource management scenarios.

Wang et al. (2017) [62] proposed the Most Fit Task First (MFTF) scheduler, which dynamically estimates task-resource fitness in heterogeneous clouds. Tasks with higher fitness values are prioritized for execution.

Yang et al. (2017) [67] introduced a game theory-based scheduling algorithm for energy management. It used cooperative game strategies for task allocation but didn’t account for task processing costs, impacting scheduling speed.

Narwal and Dhingra (2017) [44] proposed a multi-objective task scheduling algorithm for cloud infrastructure, considering cost, processing time, and average wait time. The method uses non-dominant sorting and enhances FCFS performance, but does not address throughput minimization.

Lin et al. (2017) introduced multi-resource scheduling focused on power consumption modeling and scenario-based policies. While resource allocation algorithms were examined, the approach lacked support for dynamic memory, I/O, bandwidth loads, and had limited energy efficiency and precision.

Xiaonian et al. (2017) [64] developed the market-based RSA-M strategy using a genetic algorithm to optimize resource pricing and QoS in large data centers. Simulation confirmed balanced and optimal resource allocation, benefiting both providers and consumers.

Kumar et al. (2017) [26] designed a load balancing algorithm to minimize makespan and improve VM utilization by fairly distributing workloads and reducing VM overload.

Mishra et al. (2017) [40] proposed a load balancing approach for online data centers that scales based on VM resource efficiency. Simulations showed reduced energy consumption, shorter wait times, and enhanced cloud data center performance.

Elmougy et al. (2017) [20] combined Shortest-Job-First and Round Robin in a hybrid SRDQ scheduling algorithm. Using dynamic workload balancing and dual queues (Q1 for short, Q2 for long tasks), the method was evaluated via simulation against SJF, RR, and Time Slice Priority-Based RR algorithms.

Adhikari et al. (2018) [3] proposed a load-balancing algorithm for IaaS cloud infrastructure that optimizes VM selection based on job size and quantity to maximize resource utilization. The algorithm was simulated and compared with existing methods using multiple performance metrics.

Krishna Doss and Jacob (2018) [25] introduced the Oppositional Cuckoo Search Algorithm (OCSA), a hybrid of cuckoo search and opposition-based learning, aiming to assign tasks efficiently while reducing cost. Multi-objective optimization improved scheduling, with simulations confirming cost-effectiveness.

Malik et al. (2018) [36] compared various scheduling techniques on parameters like load balance, response time, execution time, and makespan to identify the most effective method. Tom and Bindu (2019) [60] noted that load management, response time, resource use, and memory capacity remain key challenges, suggesting hybrid algorithms could enhance overall cloud scheduling performance.

Nguyen et al. (2023) [54] developed a multiple credit-based scheduling method that uses Modified K-means clustering to accurately classify and map tasks to virtual machines. Its effectiveness was validated through simulations with advanced cloud simulation tools.

Recent literature highlights the economic impact of cloud scheduling algorithms. Han et al. (2020) [84] studied cost-aware models integrating task priority and financial limits, showing that dynamic pricing and flexible resource provisioning cut operational costs. Chen and Wu (2019) [85] examined budget-aware task allocation policies, improving economic efficiency without sacrificing service quality. Algorithms like Cost-Min and Budget-First Scheduling (BFS) effectively reduce resource billing while maintaining processing speed. Incorporating pricing and budget sensitivity in scheduling enhances efficiency and promotes sustainable cloud operations.

V TASK SCHEDULING

The method for scheduling tasks is the allocation of machine resources to the various functions of the operating system (OS). The framework controls the work queues that are prioritized and wait for the CPU to decide which task to pull from which queue and how much time to devote to it. The framework controls the work queues that are prioritized and waits for the CPU to decide which task to pull from which queue and how much time to devote to it. With this type of scheduling, all tasks are guaranteed to be finished on time (Smith et al., 2023) [63]. Task Scheduling is performed by using a task scheduler which is a program that allows scheduling and, at times, monitors batch tasks on a computer or a job structure, such as a payroll system service. In scheduling (Mosleh et al., 2016) [43], various schemes are used to determine which specific task to perform. A few parameters are considered including; task priority, accessibility of computing resource, time of execution assigned to the user, number of parallel tasks allowed for the user, estimated execution time, elapsed execution-time, and task dependency.

Task scheduling in cloud computing (Srichandan et al., 2018) [58] involves assigning tasks to suitable virtual machines under resource constraints like CPU, memory, bandwidth, and storage. The goal is to minimize makespan and response time, but resource heterogeneity and task limitations pose challenges. Inefficient scheduling leads to poor hardware utilization. The problem is NP-complete, whether assigning unit-time tasks or variable-length tasks to multiple VMs. Optimal solutions via

integer programming or graph theory (Arunarani et al., 2018) [8] are computationally expensive, prompting use of heuristics that offer suboptimal but efficient, polynomial-time solutions.

Task scheduling (Razaque, 2016) [50] is crucial when clients with Service Level Agreements (SLA) (Zhang et al., 2019) [69] submit jobs to the cloud under time and restriction constraints. Scheduling tasks with deadlines ensures high reliability and dependability. If no limits are specified, providers focus on reducing energy costs and improving resource use. This aligns with goal-oriented task scheduling (GOTS), which optimizes bandwidth, latency, and resource utilization to maximize ROI and service quality. Scheduling is typically classified as static or dynamic.

From an economic standpoint, task scheduling plays a significant role in cost optimization and operational efficiency in cloud environments. Effective task scheduling reduces idle time for virtual machines and enhances resource utilization, which directly lowers infrastructure and energy costs for cloud service providers. According to Patel and Ramesh (2022) [90], intelligent scheduling algorithms can reduce operational expenditure (OPEX) by up to 40% through efficient allocation strategies that align with usage-based billing models. Moreover, for businesses utilizing cloud services, improved scheduling translates into lower service charges due to minimized compute-hour wastage. This also supports better budget forecasting and maximizes return on investment (ROI) in pay-as-you-go service models. Therefore, task scheduling is not only a technical necessity but also a critical lever for driving economic sustainability and financial competitiveness in cloud operations.

Types of Task Scheduling

Using task scheduling, the task is assigned to the most suitable resource for completion. There are different types of task planning algorithms shown in figure 4.

- **Job Scheduling**

Job scheduling (Singh and Kaur, 2016) [55] involves the distribution of certain jobs and services at a given time. The main purpose of job planning consists in distributing system loads, enhancing system efficiency, ensuring the correct use of the available resources and reducing costs and time. It is play important role in cloud computing.

Jobs generally require a number of separate tasks. Most approaches start with assessing the weight of tasks in the pool by the priority of each task to estimate the best mapping (waiting time, date, arrival time, implementation time, etc.). Job scheduling is chosen for its ability to efficiently allocate resources, prioritize tasks, and maximize system throughput. By managing dependencies, setting priorities, and optimizing resource utilization, job scheduling ensures that tasks are completed promptly and system performance is optimized. This approach enhances efficiency, scalability, and cost-effectiveness in various computing environments, making it a fundamental component of effective resource management.

Economically, job scheduling plays a vital role in reducing operational and computational costs by avoiding resource underutilization and minimizing idle time. Studies such as Ahmed et al. (2021) [91] emphasize that efficient job scheduling reduces cloud service expenses by lowering the number of active virtual machines required at any given time. Furthermore, dynamic job scheduling enables users to take advantage of low-cost compute instances, such as spot or reserved instances, thereby achieving better budget alignment. For enterprises, this means enhanced cost predictability, lower service bills, and greater return on investment (ROI) in cloud infrastructure. From the service provider's perspective, optimized job scheduling can increase infrastructure profitability by improving resource turnover and decreasing energy consumption per task, thus supporting both economic and environmental sustainability.

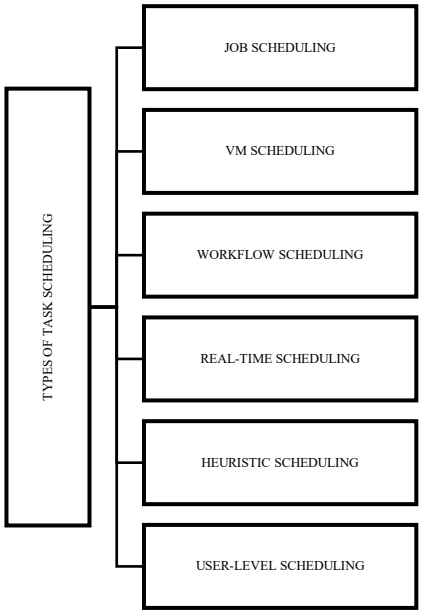


Figure 4. Types of Task Scheduling

- **Virtual Machine (VM) Scheduling**

VM scheduling involves allocating virtual machines to physical servers in a cloud environment. This type of scheduling is crucial for maintaining efficiency, reducing power consumption, and ensuring that computing resources are utilized optimally. Recent work by Liu et al. (2024) introduces an algorithm that utilizes machine learning to predict workload patterns and dynamically allocate VMs, thereby improving the efficiency of resource usage and energy consumption [75].

Economically, efficient VM scheduling reduces infrastructure overhead by minimizing the number of active physical machines, thus lowering both energy and operational costs. It also enables cloud providers to scale infrastructure based on demand, optimizing expenses and improving the return on investment (ROI) through better resource monetization.

- **Workflow Scheduling**

Workflow scheduling deals with the execution of sets of interdependent tasks or jobs, optimizing the order and allocation of resources to minimize execution time and cost. A key study by Malawski et al. (2024) showcases a metaheuristic approach that integrates cost and time optimization strategies to efficiently handle task dependencies and resource variances across multiple cloud providers [76].

Workflow scheduling has strong economic implications, especially for organizations executing complex, multi-step operations. It allows cloud users to reduce total execution costs by selecting cost-efficient resource paths while ensuring timely outputs. Cloud providers can benefit from dynamic pricing models and enhanced customer satisfaction, leading to higher revenue retention.

- **Real-Time Scheduling**

Real-time scheduling is designed to meet the strict timing requirements of tasks that are sensitive to execution time. It is critical for applications where delay or failure to execute within the allotted time frame could fail in the service. Recent advances by Zhou et al. (2024) focus on integrating real-time analytics to prioritize and schedule tasks in ways that meet these stringent deadlines without compromising on system performance [77].

From a financial perspective, real-time scheduling supports mission-critical applications (e.g., financial services, healthcare) where delays translate into economic losses. Efficient real-time schedulers ensure compliance with service-level agreements (SLAs), avoiding penalties and preserving revenue streams for both users and providers.

- **Heuristic Scheduling**

Heuristic scheduling uses rules and algorithms to find good enough solutions to complex scheduling problems where finding the optimal solution is impractical due to resource constraints. Techniques often involve a combination of genetic algorithms, simulated annealing, or tabu search. Studies by Zhang et al. (2024) have developed adaptive heuristic methods that significantly reduce computational overhead and improve scalability in large-scale cloud environments [78].

Heuristic scheduling reduces computational cost and time-to-solution, translating into savings on processing charges and better workload throughput. For providers, it improves data center efficiency and scalability, enabling them to accommodate more clients without proportionally increasing infrastructure costs—thus increasing profit margins.

- **User-Level Scheduling**

User-level scheduling in cloud computing allows users to manage task scheduling by customizing resource allocation according to specific needs. This method enhances control over application performance, particularly beneficial in environments with diverse user requirements. Huang et al. (2024) discuss a framework where users can input their scheduling preferences to optimize resource use and satisfaction, demonstrating a significant improvement in both resource utilization and user engagement [79].

User-level scheduling empowers customers to manage resources according to their budgets and performance goals, promoting personalized cost-efficiency. It leads to higher user retention for providers and encourages the adoption of self-service models, reducing support costs while expanding service flexibility and profitability.

- **Task Scheduling Metrics**

Techniques of scheduling assign tasks based on certain metrics to proper resources. The time, time, costs, and budget, which are important criteria for a successful completion of the programming process, are covered by these measurements. Generally, the scheduling metrics are divided into two: cloud users and cloud service providers.

Task scheduling techniques assign tasks based on certain performance and economic criteria to optimize cloud resource usage. These measurements are vital for the successful and cost-efficient execution of cloud-based processes. Generally, the scheduling metrics are divided into two main categories: **cloud users** and **cloud service providers**.

- **Cloud User**
  - ✓ *Cost:* Calculating the cost of a fundamental resource, such as computing, data transport, memory, and storage, determines how much it will cost to do the activity.
    Understanding and managing cost metrics allow users to make informed decisions, minimizing Understanding and managing cost metrics allow users to make informed decisions, minimizing unnecessary expenditures and selecting affordable cloud configurations that meet task requirements within financial constraints.Make span: Make span is the entire amount of time required to complete the tasks, accounting for the time required to submit the work, execute it,

and finish it to obtain the desired outcome. Lowering the pan also lowers the cost of carrying out actions that are linked to cloud resources.
Minimizing makespan leads to shorter usage periods of billable cloud resources, thus reducing operational costs. Lower makespan also means faster delivery of services, which can be economically advantageous in competitive industries like finance, media, and e-commerce.

- ✓ *Budget:* The user defines a budget constraint that is used to make use of the services offered by the service provider. This constraint will force scheduling decisions while assigning tasks, thereby minimizing the overall implementation time of tasks based on the budget constraint. Several virtual machines with different configurations can be used to perform the tasks submitted. Thus, the cost of total execution is the sum of the price of all virtual machines used to perform the tasks. It should be less than or equal to the budget constraint defined by the user. *Budget-aware scheduling ensures that users maintain control over cloud spending while achieving efficient task execution. It promotes financial sustainability and encourages the use of cost-optimized resource configurations (e.g., spot instances or lightweight VMs).*
- ✓ *Deadline Time: Deadline time refers to the critical time needed to perform the task within a specified timeframe so that the timeframe is designed to constrain task scheduling within the timeframe and deliver the results of task execution within the timeframe. The limitation period looks for monetary costs during the scheduling of tasks. There is therefore a need for robust scheduling that meets the time limit for time-specific applications. The configuration of the virtual machine shall be determined by the scheduler for assigning the task and minimizing the monetary costs, together with the fulfillment of the time limit set by the user (Zhou, Wang, & Zhang, 2024) [74].*

Meeting strict deadlines often involves selecting more expensive, high-performance resources. Economically efficient scheduling strategies aim to meet these deadlines with minimal cost, thereby maximizing service value without overspending.

- **Cloud Service Provider**
  - ✓ *Resource Utilization*: When resource utilization is maximized, the benefits will accrue to the cloud provider, where resources are rented to users and maximum profits can be achieved if full resource utilization is achieved (Huang, Liu, & Wei, 2024) [79].
    High utilization rates reduce wasted capacity and increase return on infrastructure investment. Providers can achieve greater profitability and environmental efficiency by minimizing idle server time.
  - ✓ *Load Balancing*: This general term refers to the process of distributing the workload among one or more servers, disk drives, network interfaces, or other computer resources to maximize resource utilization and task response times. When a device processor or network node is load-balanced, it indicates that it always completes the same amount of work. To enhance overall performance, specific sorts of data, including the number of tasks in the queue, the job arrival rate, the CPU processing rate, and so on for each processor, may be shared throughout processors during load balancing (Zhang, Li, & Zhou, 2024) [78].
    Effective load balancing reduces the need for overprovisioning, lowers energy consumption, and enhances service reliability—all of which contribute to cost savings and improved profit margins.
  - ✓ *Profit:* Profit is a vital measure that refers to the total revenue resulting from the cost paid by users of cloud services as a result of their use for successful applications (Huang, Liu, & Wei, 2024) [79].
    Profitability is directly influenced by how well scheduling strategies align resource use with service pricing. Scheduling algorithms that reduce overhead and energy costs while maintaining high-quality service delivery are critical for financial success in competitive cloud markets.

Challenges in Task Scheduling

In the cloud, task scheduling is considered to be the most important and effective way to process requests based on the availability of resources such as memory, bandwidth, and CPU. Cloud scheduling carries out three fundamental processes pick resources, measure the fitness between the task and the resource, and allocate tasks. Task scheduling challenges depend on the availability of resources, allocation of resources, scheduling objectives, and fault tolerance.

From an economic standpoint, challenges in task scheduling directly influence both cost efficiency and revenue generation for cloud users and providers. Inconsistent resource availability may lead to over-provisioning or underutilization, increasing operational costs and decreasing return on investment (ROI). Allocation inefficiencies can cause resource wastage and higher energy consumption, reducing the economic sustainability of the infrastructure. Additionally, poor fault tolerance may lead to task failure or re-execution, incurring additional costs for users and SLA penalties for providers. Economic-aware scheduling strategies are therefore essential to minimize these challenges by aligning technical performance with financial objectives such as cost reduction, profit maximization, and optimal pricing models. Addressing these scheduling challenges from both a technical and economic perspective is vital for maintaining competitiveness and long-term viability in cloud services.

✓ Resource Allocation

The provisioning, allocation, and managing of resources in cloud computing is considered to be an important task in computation. It varies depending upon the complexity of services installed and used. Proving unlimited flexibility of resources is considered to be an effective feature of the cloud. The cloud service provider allows the users to join and leave without giving prior knowledge of their computing requirements at any time. For the management of workloads and delivery of QoS services to the end users, the resources are managed and handled effectively. The computational resources and networks are scarce and should be spread virtually efficiently to various users. Problems such as the provision of resources, adaptation of resources, resource maps, and allocation of resources are considered vital for efficient management of resources. To meet the increasing demand for cloud processing, powerful technologies, and devices are used with a lack of virtualization equipment. In the cloud paradigm, the availability of resources continues to vary which makes the scheduling of tasks with resource awareness as an essential issue.

Economically, efficient resource allocation reduces unnecessary capital and operational expenditures by optimizing server usage and minimizing idle resource costs. When resource allocation aligns with demand, cloud providers can avoid overprovisioning, which reduces electricity costs, cooling requirements, and infrastructure depreciation. For users, dynamic resource scaling prevents overbilling by tailoring usage to need, leading to significant savings, especially under pay-as-you-go pricing models. Moreover, well-optimized allocation improves cloud ROI and supports financially viable scalability.

✓ Fault Tolerance

Owing to the rapid growth of cloud computing, the need for fault tolerance of the cloud is a significant factor. A computer or network computer that has fault tolerance settings installed will not crash in the case of an unforeseen issue or error, such as a hardware failure, a link failure, illegal access, changes to the system configuration, or out-of-reach memory or disk space. Failures should be expected and proactively controlled to mitigate the failure effects on program execution and task execution. In cloud computing, there are different kinds of faults. There are two fault tolerance techniques: reactive and proactive fault tolerance. Based on fault tolerance policies. Cloud computing's main benefits include fault recovery, reduced costs, and improved performance metrics.

Economically, robust fault tolerance mitigates revenue losses caused by downtime or service-level agreement (SLA) breaches. Every minute of cloud service interruption can translate into significant monetary loss for clients, particularly in critical sectors like e-commerce, finance, and healthcare. Fault-tolerant systems protect against these losses while minimizing the need for manual intervention, which reduces support and labor costs. From a provider's perspective, maintaining high reliability also boosts customer satisfaction and retention, contributing to long-term profitability.

✓ **Scheduling Objectives**

Scheduling algorithms aim to improve system performance and throughput without compromising other parameters. However, due to the complexity of multiple scheduling objectives, these algorithms can sometimes degrade overall effectiveness. Scheduling challenges can be classified into two categories: decision problems and optimization problems. The optimization problem in cloud computing focuses on selecting the best solution from a set of alternatives based on multiple criteria, such as efficient resource utilization, cost reduction, and meeting performance metrics like execution time and latency. Resource utilization targets maximizing CPU, memory, and bandwidth use to avoid wastage, while cost optimization seeks to minimize expenses related to computation, storage, and data transfer without sacrificing performance. Additionally, maintaining scalability, reliability, load balancing, and quality of service (QoS) is essential for ensuring high availability, balanced workloads, and adherence to service level agreements (SLAs) (Huang, Liu, & Wei, 2024) [79].

The objective of a decision problem is generally easier to solve than that of an optimization problem because it involves selecting from a predefined set of options based on specific criteria, rather than searching for the best solution within a vast solution space. Decision-making focuses on identifying a suitable choice that meets established requirements, often by evaluating a finite number of alternatives. In contrast, optimization requires systematically exploring numerous potential solutions to find the optimal one, which is more complex and computationally demanding due to the need to balance multiple factors and constraints simultaneously. As the size of the search space and the number of possible solutions increase, so does the time required to find the optimal solution. Multi-objective optimization is especially challenging because it requires balancing several often-conflicting objectives at once. However, recent advancements in software have made this process more manageable by offering sophisticated algorithms and tools capable of handling complex, multi-dimensional data and efficiently finding solutions that balance various objectives. These tools facilitate exploring a broad range of solutions, helping identify an optimal trade-off among different goals (Huang, Liu, & Wei, 2024) [79].

Scheduling objectives have a direct economic impact by influencing cost-efficiency and profitability. Effective optimization can significantly reduce cloud usage charges through better resource allocation, energy consumption control, and efficient service delivery. Multi-objective optimization also improves financial predictability for users by minimizing execution costs while meeting performance thresholds. For providers, achieving SLA compliance without excessive provisioning ensures consistent revenue generation while avoiding penalty costs. Additionally, leveraging decision-support tools reduces administrative overhead and supports scalable business models.

VI PROPOSED WORK

In cloud computing, QoS parameters ensure that the services provided meet certain standards or requirements, such as response time, reliability, and availability. Therefore, the mentioned solution aims to balance factors like resource utilization, workload distribution, and task scheduling in a way that not only maximizes efficiency but also maintains satisfactory levels of service quality. This approach is crucial for ensuring the effective and reliable operation of cloud-based systems, especially

in environments where multiple tasks or services are competing for shared resources. Research initializes the number of tasks and virtual machines, and the clustering or clustering of virtual machines. This approach suggests deviating from simply sorting tasks according to a predefined threshold constraint. Instead, it advocates considering both the time limit and priority associated with each task. Tasks are then assigned to virtual machines based on the workload, ensuring that higher priority or time-sensitive tasks receive precedence in resource allocation. By incorporating task priority, time constraints, and workload distribution, this method aims to optimize resource utilization and enhance overall system efficiency in cloud computing environments.

Step 1: Initially, VMs and tasks are generated and sent to the data center broker (DCB).

Step 2: DCB clusters VMs using the K-Means Clustering algorithm, categorizing machines based on the number of applications assigned to each.

Step 3: Tasks are categorized by priority and period, with calculations based on the following equations:

$$credit\_Priority_i = TaskPriority_i / divisiblefactor_i \quad (1)$$

$$Deadline\_Task_i = Credit\_Length_i * Credit\_Priority_i / MIPS_{MAx} \quad (2)$$

where i denotes task number and $MIPS_{MAX}$ denotes the maximum MIPS of the machine in the list created, the divisible factor can be the multiple of 10

Step 4: Within the MIPS clusters, machines are sorted and the equation-dependent charge for each machine is found using:

$$Vmload = (N * Task_{length}) / Vm_{mips} \quad (3)$$

where N is the number of tasks, $Task_{length}$ is the length of the task $\&VM_{Mips}$ is MIPS ("Million Instructions per second") of the virtual machine.

Step 5: VM capacity is calculated using:

$$VM_{Capacity} = PE_{Number} * PE_{Mips} + VM_{BW} \quad (4)$$

where PE_{Number} is the number of processing elements in VM and PE_{Mips} is MIPS speed of processing element of VM $\&VM_{BW}$ is bandwidth linked with VM.

Step 6: The processing time (PT) of the virtual machine is calculated as:

$$PT\_Vm_i = VmLoad / Vm_{capacity} \quad (5)$$

Step 7: The load's Standard Deviation (SD) is calculated using:

$$oneSD = \sqrt{1/M \sum_{i=0}^{M} (X_i - \overline{X})^2} \quad (6)$$

where X_i is Processing Time, $\overline{X}$ - The average Processing Time of the virtual machine as well as M belongs to the virtual machine in the VM set.

Step 8: Based on the standard deviation, machines are divided into overloaded and underloaded machines within the clusters. Then, based on the balancing criteria, jobs with a higher prioritization level are mapped to the efficient virtual machine.

Step 9: Analyze performance based on parameters such as makespan time, response time, throughput, execution time, cost, and degree of imbalance.

VII SIMULATIONS AND RESULTS

The analysis is conducted using Java JDK (Java Development Kit) and NetBeans IDE in Clouds 3.0.3 and the performance will be measured based on the response time performance parameters of time, and throughput. The results are evaluated accordingly.

Table 2 Simulation Parameters

Parameters	**Value**	**Parameters**	**Value**	**Parameters**	**Value**
VM RAM	512 MB	Hypervisor	Xen	Number of CPUs	1
VM Size	10000 MB	Number of servers	1		

RESULTS

Case 1: 30 VM, 200 cloudlets

In Case 1, with 30 VMs and 200 cloudlets, Figures 3.1 and 3.2 demonstrate the simulation results of both techniques, allowing for a comparison of their effectiveness.

Case 2: 30 VM, 400 cloudlets

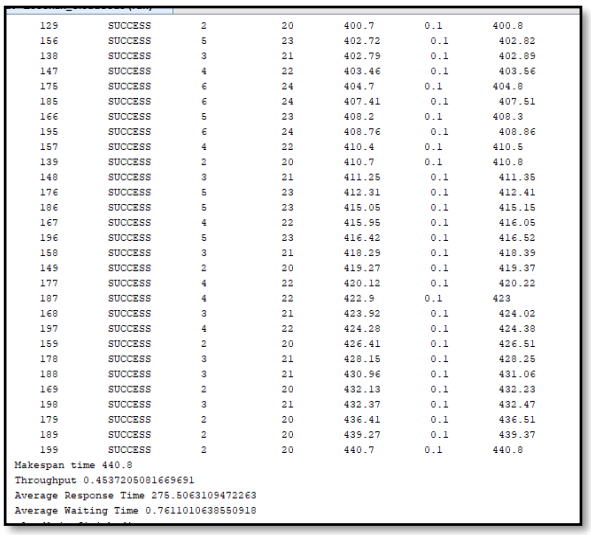
Moving to Case 2, where 30 VMs and 400 cloudlets are considered, Figures 3.3 and 3.4 present the simulation outcomes for the existing and proposed techniques, respectively.

Case 3: 30 VM, 600 cloudlets

Case 3 expands the analysis to 30 VMs and 600 cloudlets, with Figures 3.5 and 3.6 providing a visual representation of the results achieved by each technique.

Case 4: 30 VM, 800 cloudlets

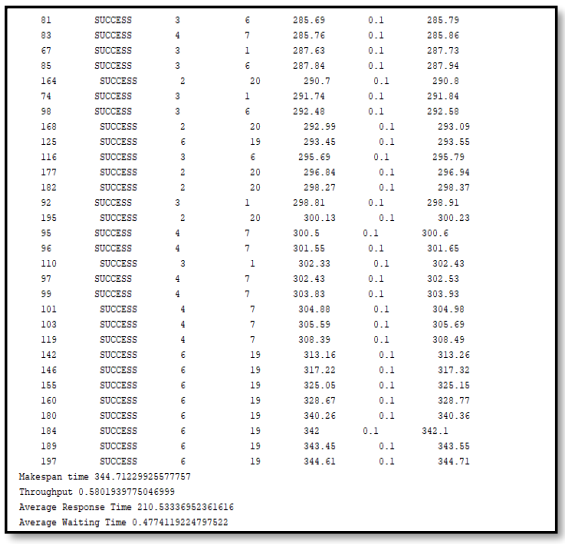
Finally, Case 4 explores the scenario of 30 VMs and 800 cloudlets, with Figures 3.7 and 3.8 displaying the simulation results for both the existing and proposed techniques.



129	SUCCESS	2	20	400.7	0.1	400.8
156	SUCCESS	5	23	402.72	0.1	402.82
138	SUCCESS	3	21	402.79	0.1	402.89
147	SUCCESS	4	22	403.46	0.1	403.56
175	SUCCESS	6	24	404.7	0.1	404.8
185	SUCCESS	6	24	407.41	0.1	407.51
166	SUCCESS	5	23	408.2	0.1	408.3
195	SUCCESS	6	24	408.76	0.1	408.86
157	SUCCESS	4	22	410.4	0.1	410.5
139	SUCCESS	2	20	410.7	0.1	410.8
148	SUCCESS	3	21	411.25	0.1	411.35
176	SUCCESS	5	23	412.31	0.1	412.41
186	SUCCESS	5	23	415.05	0.1	415.15
167	SUCCESS	4	22	415.95	0.1	416.05
196	SUCCESS	5	23	416.42	0.1	416.52
158	SUCCESS	3	21	418.29	0.1	418.39
149	SUCCESS	2	20	419.27	0.1	419.37
177	SUCCESS	4	22	420.12	0.1	420.22
187	SUCCESS	4	22	422.9	0.1	423
168	SUCCESS	3	21	423.92	0.1	424.02
197	SUCCESS	4	22	424.28	0.1	424.38
159	SUCCESS	2	20	426.41	0.1	426.51
178	SUCCESS	3	21	428.15	0.1	428.25
188	SUCCESS	3	21	430.96	0.1	431.06
169	SUCCESS	2	20	432.13	0.1	432.23
198	SUCCESS	3	21	432.37	0.1	432.47
179	SUCCESS	2	20	436.41	0.1	436.51
189	SUCCESS	2	20	439.27	0.1	439.37
199	SUCCESS	2	20	440.7	0.1	440.8

Makespan time 440.8
Throughput 0.4537205081669691
Average Response Time 275.5063109472263
Average Waiting Time 0.7611010638550918

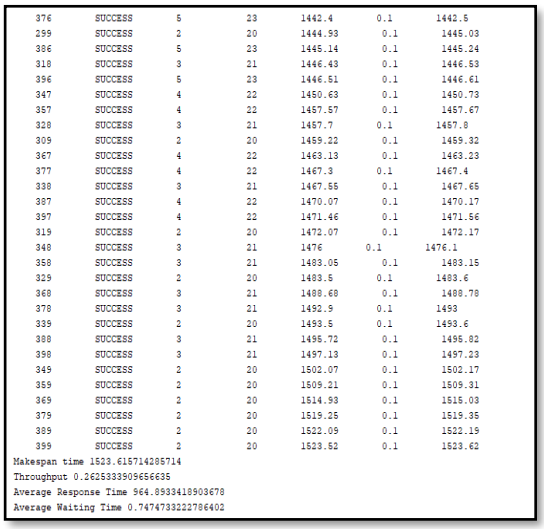
Figure 3.1 Represent the Simulation Results of the Existing Technique for 200 cloudlets



81	SUCCESS	3	6	285.69	0.1	285.79
83	SUCCESS	4	7	285.76	0.1	285.86
67	SUCCESS	3	1	287.63	0.1	287.73
85	SUCCESS	3	6	287.84	0.1	287.94
164	SUCCESS	2	20	290.7	0.1	290.8
74	SUCCESS	3	1	291.74	0.1	291.84
98	SUCCESS	3	6	292.48	0.1	292.58
168	SUCCESS	2	20	292.99	0.1	293.09
125	SUCCESS	6	19	293.45	0.1	293.55
116	SUCCESS	3	6	295.69	0.1	295.79
177	SUCCESS	2	20	296.84	0.1	296.94
182	SUCCESS	2	20	298.27	0.1	298.37
92	SUCCESS	3	1	298.81	0.1	298.91
195	SUCCESS	2	20	300.13	0.1	300.23
95	SUCCESS	4	7	300.5	0.1	300.6
96	SUCCESS	4	7	301.55	0.1	301.65
110	SUCCESS	3	1	302.33	0.1	302.43
97	SUCCESS	4	7	302.43	0.1	302.53
99	SUCCESS	4	7	303.83	0.1	303.93
101	SUCCESS	4	7	304.88	0.1	304.98
103	SUCCESS	4	7	305.59	0.1	305.69
119	SUCCESS	4	7	308.39	0.1	308.49
142	SUCCESS	6	19	313.16	0.1	313.26
146	SUCCESS	6	19	317.22	0.1	317.32
155	SUCCESS	6	19	325.05	0.1	325.15
160	SUCCESS	6	19	328.67	0.1	328.77
180	SUCCESS	6	19	340.26	0.1	340.36
184	SUCCESS	6	19	342	0.1	342.1
189	SUCCESS	6	19	343.45	0.1	343.55
197	SUCCESS	6	19	344.61	0.1	344.71

Makespan time 344.71229525577757
Throughput 0.5801939775046999
Average Response Time 210.53336952361616
Average Waiting Time 0.4774119224797522

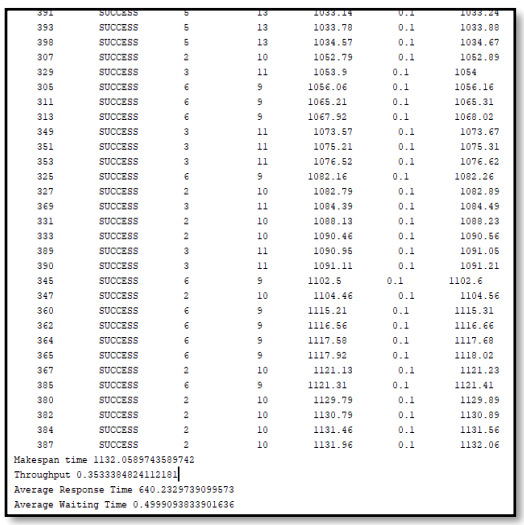
Figure 3.2 Represent the Simulation Results of Proposed Technique for 200 cloudlets



376	SUCCESS	5	23	1442.4	0.1	1442.5
299	SUCCESS	2	20	1444.93	0.1	1445.03
386	SUCCESS	5	23	1445.14	0.1	1445.24
318	SUCCESS	3	21	1446.43	0.1	1446.53
396	SUCCESS	5	23	1446.51	0.1	1446.61
347	SUCCESS	4	22	1450.63	0.1	1450.73
357	SUCCESS	4	22	1457.57	0.1	1457.67
328	SUCCESS	3	21	1457.7	0.1	1457.8
309	SUCCESS	2	20	1459.22	0.1	1459.32
367	SUCCESS	4	22	1463.13	0.1	1463.23
377	SUCCESS	4	22	1467.3	0.1	1467.4
338	SUCCESS	3	21	1467.55	0.1	1467.65
387	SUCCESS	4	22	1470.07	0.1	1470.17
397	SUCCESS	4	22	1471.46	0.1	1471.56
319	SUCCESS	2	20	1472.07	0.1	1472.17
348	SUCCESS	3	21	1476	0.1	1476.1
358	SUCCESS	3	21	1483.05	0.1	1483.15
329	SUCCESS	2	20	1483.5	0.1	1483.6
368	SUCCESS	3	21	1488.68	0.1	1488.78
378	SUCCESS	3	21	1492.9	0.1	1493
339	SUCCESS	2	20	1493.5	0.1	1493.6
388	SUCCESS	3	21	1495.72	0.1	1495.82
398	SUCCESS	3	21	1497.13	0.1	1497.23
349	SUCCESS	2	20	1502.07	0.1	1502.17
359	SUCCESS	2	20	1509.21	0.1	1509.31
369	SUCCESS	2	20	1514.93	0.1	1515.03
379	SUCCESS	2	20	1519.25	0.1	1519.35
389	SUCCESS	2	20	1522.09	0.1	1522.19
399	SUCCESS	2	20	1523.52	0.1	1523.62

Makespan time 1523.615714285714
Throughput 0.26253339056566356
Average Response Time 964.8933418903678
Average Waiting Time 0.7474733222786402

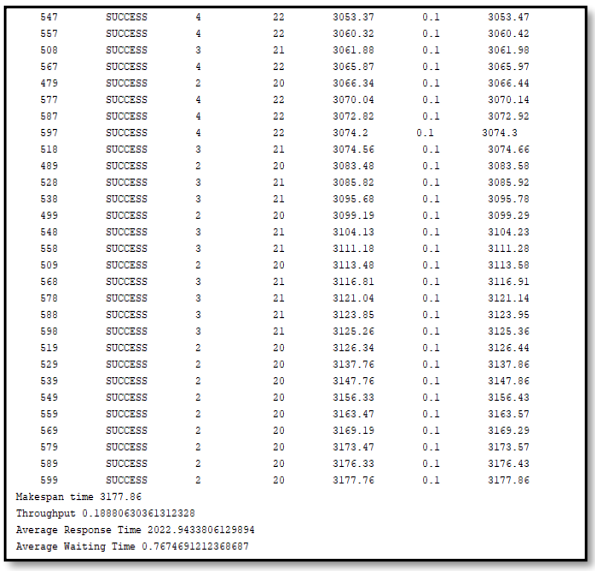
Figure 3.3 Represent the Simulation Results of Existing Technique for 400 cloudlets



391	SUCCESS	5	13	1033.14	0.1	1033.24
393	SUCCESS	5	13	1033.78	0.1	1033.88
398	SUCCESS	5	13	1034.57	0.1	1034.67
307	SUCCESS	2	10	1052.79	0.1	1052.89
329	SUCCESS	3	11	1053.9	0.1	1054
305	SUCCESS	6	9	1056.06	0.1	1056.16
311	SUCCESS	6	9	1065.21	0.1	1065.31
313	SUCCESS	6	9	1067.92	0.1	1068.02
349	SUCCESS	3	11	1073.57	0.1	1073.67
351	SUCCESS	3	11	1075.21	0.1	1075.31
353	SUCCESS	3	11	1076.52	0.1	1076.62
325	SUCCESS	6	9	1082.16	0.1	1082.26
327	SUCCESS	2	10	1082.79	0.1	1082.89
369	SUCCESS	3	11	1084.39	0.1	1084.49
331	SUCCESS	2	10	1088.13	0.1	1088.23
333	SUCCESS	2	10	1090.46	0.1	1090.56
389	SUCCESS	3	11	1090.95	0.1	1091.05
390	SUCCESS	3	11	1091.11	0.1	1091.21
345	SUCCESS	6	9	1102.5	0.1	1102.6
347	SUCCESS	2	10	1104.46	0.1	1104.56
360	SUCCESS	6	9	1115.21	0.1	1115.31
362	SUCCESS	6	9	1116.56	0.1	1116.66
364	SUCCESS	6	9	1117.58	0.1	1117.68
365	SUCCESS	6	9	1117.92	0.1	1118.02
367	SUCCESS	2	10	1121.13	0.1	1121.23
385	SUCCESS	6	9	1121.31	0.1	1121.41
380	SUCCESS	2	10	1129.79	0.1	1129.89
382	SUCCESS	2	10	1130.79	0.1	1130.89
384	SUCCESS	2	10	1131.46	0.1	1131.56
387	SUCCESS	2	10	1131.96	0.1	1132.06

Makespan time 1132.0589743589742
Throughput 0.3533384824112181
Average Response Time 640.2329739099573
Average Waiting Time 0.4599093833901636

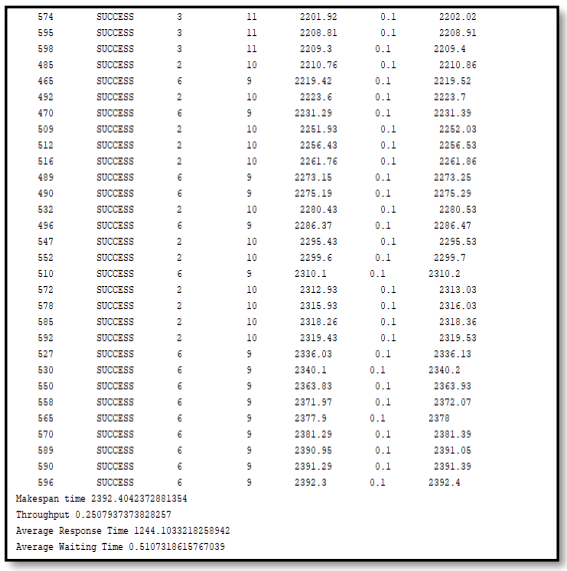
Figure 3.4 Represent the Simulation Results of Proposed Technique for 400 cloudlets



547	SUCCESS	4	22	3053.37	0.1	3053.47
557	SUCCESS	4	22	3060.32	0.1	3060.42
508	SUCCESS	3	21	3061.88	0.1	3061.98
567	SUCCESS	4	22	3065.87	0.1	3065.97
479	SUCCESS	2	20	3066.34	0.1	3066.44
577	SUCCESS	4	22	3070.04	0.1	3070.14
587	SUCCESS	4	22	3072.82	0.1	3072.92
597	SUCCESS	4	22	3074.2	0.1	3074.3
518	SUCCESS	3	21	3074.56	0.1	3074.66
489	SUCCESS	2	20	3083.48	0.1	3083.58
528	SUCCESS	3	21	3085.82	0.1	3085.92
538	SUCCESS	3	21	3095.68	0.1	3095.78
499	SUCCESS	2	20	3099.19	0.1	3099.29
548	SUCCESS	3	21	3104.13	0.1	3104.23
558	SUCCESS	3	21	3111.18	0.1	3111.28
509	SUCCESS	2	20	3113.48	0.1	3113.58
568	SUCCESS	3	21	3116.81	0.1	3116.91
578	SUCCESS	3	21	3121.04	0.1	3121.14
588	SUCCESS	3	21	3123.85	0.1	3123.95
598	SUCCESS	3	21	3125.26	0.1	3125.36
519	SUCCESS	2	20	3126.34	0.1	3126.44
529	SUCCESS	2	20	3137.76	0.1	3137.86
539	SUCCESS	2	20	3147.76	0.1	3147.86
549	SUCCESS	2	20	3156.33	0.1	3156.43
559	SUCCESS	2	20	3163.47	0.1	3163.57
569	SUCCESS	2	20	3169.19	0.1	3169.29
579	SUCCESS	2	20	3173.47	0.1	3173.57
589	SUCCESS	2	20	3176.33	0.1	3176.43
599	SUCCESS	2	20	3177.76	0.1	3177.86

Makespan time 3177.86
Throughput 0.1888630361312328
Average Response Time 2022.9433806129894
Average Waiting Time 0.7674691212368687

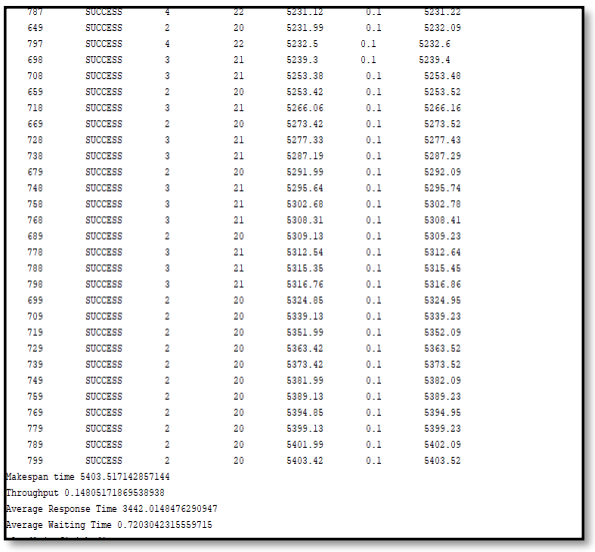
Figure 3.5 Represent the Simulation Results of Existing Technique for 600 cloudlets



574	SUCCESS	3	11	2201.92	0.1	2202.02
595	SUCCESS	3	11	2208.81	0.1	2208.91
598	SUCCESS	3	11	2209.3	0.1	2209.4
485	SUCCESS	2	10	2210.76	0.1	2210.86
465	SUCCESS	6	9	2219.42	0.1	2219.52
492	SUCCESS	2	10	2223.6	0.1	2223.7
470	SUCCESS	6	9	2231.29	0.1	2231.39
509	SUCCESS	2	10	2251.93	0.1	2252.03
512	SUCCESS	2	10	2256.43	0.1	2256.53
516	SUCCESS	2	10	2261.76	0.1	2261.86
489	SUCCESS	6	9	2273.15	0.1	2273.25
490	SUCCESS	6	9	2275.19	0.1	2275.29
532	SUCCESS	2	10	2280.43	0.1	2280.53
496	SUCCESS	6	9	2286.37	0.1	2286.47
547	SUCCESS	2	10	2295.43	0.1	2295.53
552	SUCCESS	2	10	2299.6	0.1	2299.7
510	SUCCESS	6	9	2310.1	0.1	2310.2
572	SUCCESS	2	10	2312.93	0.1	2313.03
578	SUCCESS	2	10	2315.93	0.1	2316.03
585	SUCCESS	2	10	2318.26	0.1	2318.36
592	SUCCESS	2	10	2319.43	0.1	2319.53
527	SUCCESS	6	9	2336.03	0.1	2336.13
530	SUCCESS	6	9	2340.1	0.1	2340.2
550	SUCCESS	6	9	2363.83	0.1	2363.93
558	SUCCESS	6	9	2371.97	0.1	2372.07
565	SUCCESS	6	9	2377.9	0.1	2378
570	SUCCESS	6	9	2381.29	0.1	2381.39
589	SUCCESS	6	9	2390.95	0.1	2391.05
590	SUCCESS	6	9	2391.29	0.1	2391.39
596	SUCCESS	6	9	2392.3	0.1	2392.4

Makespan time 2392.4042372881354
Throughput 0.2507937373828257
Average Response Time 1244.1033218258942
Average Waiting Time 0.5107318615767039

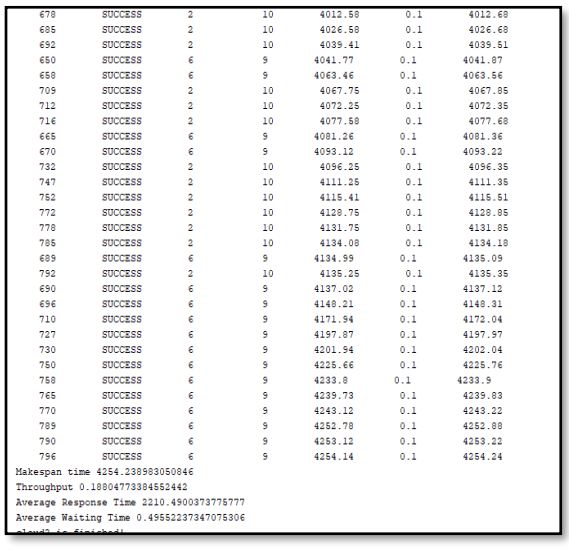
Figure 3.6 Represent the Simulation Results of Proposed Technique for 600 cloudlets



787	SUCCESS	4	22	5231.12	0.1	5231.22
649	SUCCESS	2	20	5231.99	0.1	5232.09
797	SUCCESS	4	22	5232.5	0.1	5232.6
698	SUCCESS	3	21	5239.3	0.1	5239.4
708	SUCCESS	3	21	5253.38	0.1	5253.48
659	SUCCESS	2	20	5253.42	0.1	5253.52
718	SUCCESS	3	21	5266.06	0.1	5266.16
669	SUCCESS	2	20	5273.42	0.1	5273.52
728	SUCCESS	3	21	5277.33	0.1	5277.43
738	SUCCESS	3	21	5287.19	0.1	5287.29
679	SUCCESS	2	20	5291.99	0.1	5292.09
748	SUCCESS	3	21	5295.64	0.1	5295.74
758	SUCCESS	3	21	5302.68	0.1	5302.78
768	SUCCESS	3	21	5308.31	0.1	5308.41
689	SUCCESS	2	20	5309.13	0.1	5309.23
778	SUCCESS	3	21	5312.54	0.1	5312.64
788	SUCCESS	3	21	5315.35	0.1	5315.45
798	SUCCESS	3	21	5316.76	0.1	5316.86
699	SUCCESS	2	20	5324.85	0.1	5324.95
709	SUCCESS	2	20	5339.13	0.1	5339.23
719	SUCCESS	2	20	5351.99	0.1	5352.09
729	SUCCESS	2	20	5363.42	0.1	5363.52
739	SUCCESS	2	20	5373.42	0.1	5373.52
749	SUCCESS	2	20	5381.99	0.1	5382.09
759	SUCCESS	2	20	5389.13	0.1	5389.23
769	SUCCESS	2	20	5394.85	0.1	5394.95
779	SUCCESS	2	20	5399.13	0.1	5399.23
789	SUCCESS	2	20	5401.99	0.1	5402.09
799	SUCCESS	2	20	5403.42	0.1	5403.52

Makespan time 5403.517142857144
Throughput 0.1480517186538938
Average Response Time 3442.0148476290947
Average Waiting Time 0.7203043155559715

Figure 3.7 Represent the Simulation Results of Existing Technique for 800 cloudlets



678	SUCCESS	2	10	4012.58	0.1	4012.68
685	SUCCESS	2	10	4026.58	0.1	4026.68
692	SUCCESS	2	10	4039.41	0.1	4039.51
650	SUCCESS	6	9	4041.77	0.1	4041.87
658	SUCCESS	6	9	4063.46	0.1	4063.56
709	SUCCESS	2	10	4067.75	0.1	4067.85
712	SUCCESS	2	10	4072.25	0.1	4072.35
716	SUCCESS	2	10	4077.58	0.1	4077.68
665	SUCCESS	6	9	4081.26	0.1	4081.36
670	SUCCESS	6	9	4093.12	0.1	4093.22
732	SUCCESS	2	10	4096.25	0.1	4096.35
747	SUCCESS	2	10	4111.25	0.1	4111.35
762	SUCCESS	2	10	4115.41	0.1	4115.51
772	SUCCESS	2	10	4128.75	0.1	4128.85
778	SUCCESS	2	10	4131.75	0.1	4131.85
785	SUCCESS	2	10	4134.08	0.1	4134.18
689	SUCCESS	6	9	4134.99	0.1	4135.09
792	SUCCESS	2	10	4135.25	0.1	4135.35
690	SUCCESS	6	9	4137.02	0.1	4137.12
696	SUCCESS	6	9	4148.21	0.1	4148.31
710	SUCCESS	6	9	4171.94	0.1	4172.04
727	SUCCESS	6	9	4197.87	0.1	4197.97
730	SUCCESS	6	9	4201.94	0.1	4202.04
750	SUCCESS	6	9	4225.66	0.1	4225.76
758	SUCCESS	6	9	4233.8	0.1	4233.9
765	SUCCESS	6	9	4239.73	0.1	4239.83
770	SUCCESS	6	9	4243.12	0.1	4243.22
789	SUCCESS	6	9	4252.78	0.1	4252.88
790	SUCCESS	6	9	4253.12	0.1	4253.22
796	SUCCESS	6	9	4254.14	0.1	4254.24

Makespan time 4254.238983050846
Throughput 0.18804773384552442
Average Response Time 2210.4900373775777
Average Waiting Time 0.4955223734707530

Figure 3.8 Represent the Simulation Results of Proposed Technique for 800 cloudlets

- **Make span:** Make span is the entire amount of time required to complete the tasks, accounting for the time required to obtain the desired outcome from the point of submission to the point of execution and completion.

In this research study, we investigate the efficiency of task scheduling algorithms in cloud computing environments, particularly focusing on the allocation of tasks across 30 virtual machines (VMs). As a benchmark, we utilize an existing algorithm currently employed for task scheduling in cloud environments. Our analysis centres on evaluating the makespan of different cloudlets processed using this existing algorithm on the designated 30 VMs.

The makespan in Figure 3.9, representing the total duration required to complete all tasks or cloudlets within a workload, serves as a critical metric for assessing the performance of scheduling algorithms. By comparing the makespan across varying numbers of cloudlets, we aim to discern the efficacy and efficiency of the existing algorithm in managing task allocation and resource utilization within the cloud infrastructure.

The results obtained from this comparative analysis provide valuable insights into the strengths and limitations of the existing scheduling algorithm, serving as a basis for proposing potential enhancements or alternative approaches to optimize task allocation in cloud environments. These findings contribute to the ongoing discourse on improving resource management and performance optimization in cloud computing systems.

The results of the simulation are shown in Figures 3.1 to Figure 3.8. The composition obtained by the proposed algorithm is slightly lower than that of commonly used algorithms, as experimental studies indicate.

- **Throughput:** The quantity of data that may be sent between locations in a specific length of time is referred to as throughput. In essence, throughput is a measurement of how quickly data can be sent by a certain application.

The result value for the different jobs on 30 virtual machines shows clearly that the methodology proposed is performing better than the present algorithm in Figure 3.10. Figure 3.10 in contrast with the current algorithm the proposed algorithm demonstrates a better throughput rate.

- **Average Response Time:** The whole length of time the algorithm needs to react and process a service request is called response time.

Figure 3.11 shows the value of the “average response time” of different numbers of cloudlets/tasks. The existing algorithm shows an average response time of 0.7203 for 800 cloudlets while the proposed algorithm shows 0.4955, which is less than that of the existing algorithm. Therefore, it is clear from the above-mentioned results that the proposed algorithm performs better in comparison with the existing algorithm.

- **Waiting Time:** The amount of time a process spends in the ready state while it waits for a CPU is called the waiting time.

Figure 3.12 shows the average number of cloudlets waiting times. The new algorithm shows an average wait time of 200, 400, 600, and 800 clouds, respectively, of 0.7611, 0.7474, 0.7674, and 0.7203. Although 0.4774, 0.4999, 0.5107, and 0.4955 were presented for the proposed algorithm, these were less than the current algorithm. It is also evident from the above findings that in comparison with the existing algorithm, the proposed algorithm is more robust.

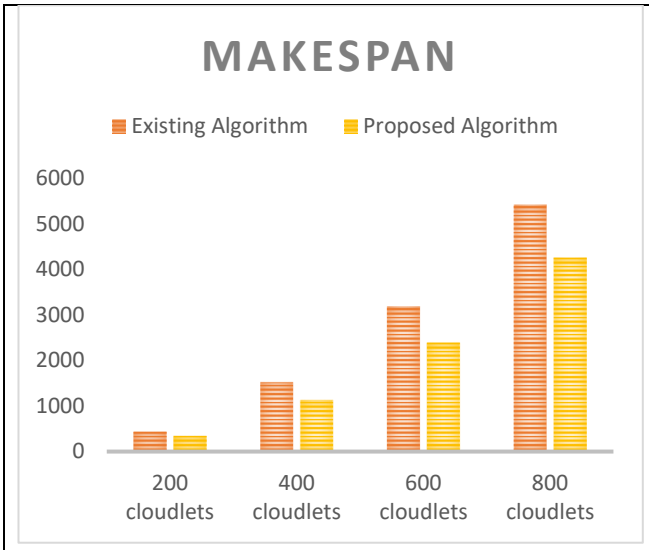


Figure 3.9 Make span of Different Cloudlets on 30 VMs

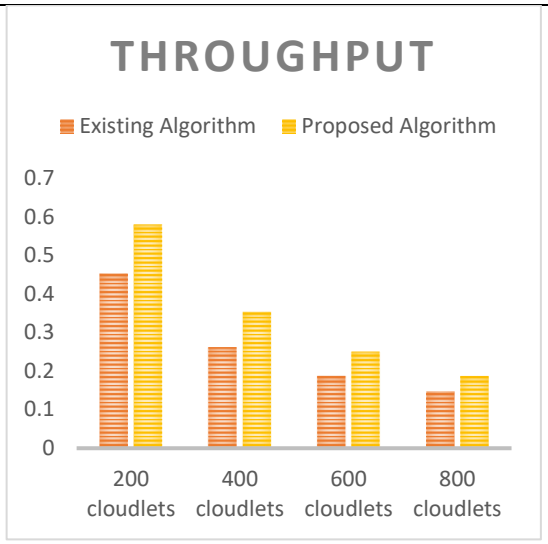


Figure 3.10Throughput of Different Cloudlets on 30 VMs

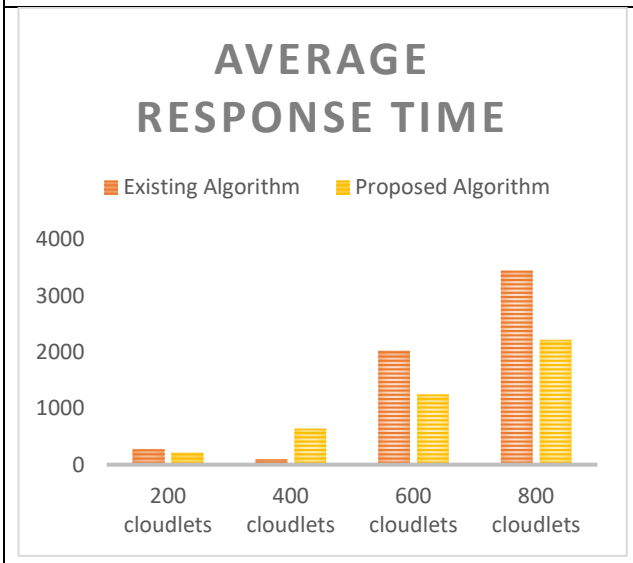


Figure 3.11 Average Response Time of Different Cloudlets on 30 VMs

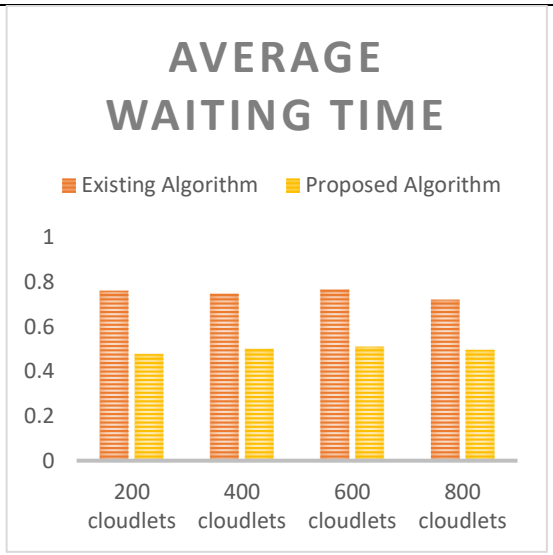


Figure 3.12 Average Waiting Time of Different Cloudlets on 30 VMs

VIII ECONOMICAL IMPACT

Optimized scheduling of jobs using load balancing in cloud computing can lead to substantial economic benefits for organizations, cloud service providers, and end-users. There are various economic benefits like *Patel [82], Sandeep , Zhu, Kai [83]*

- ✓ **Cost Efficiency**

Resource Utilization: Load balancing ensures that computing resources (CPU, memory, storage) are used optimally, minimizing idle time and avoiding over-provisioning.

Reduced Operational Costs: Efficient job scheduling reduces the need for manual intervention and minimizes the use of expensive, on-demand cloud resources.

Energy Savings: Optimized resource usage lowers power consumption, which is a significant cost factor, especially in large-scale data centres.

- ✓ **Improved Performance and Time Saving**

Reduced Job Execution Time: Efficient scheduling means faster job completion, which translates to less time spent renting compute resources (especially in pay-per-use models).

Lower Latency: By balancing the load, tasks are processed closer to real-time, improving system responsiveness and avoiding penalties from SLA violations.

- ✓ **Scalability without Cost Explosion**

Elastic Resource Scaling: Jobs can be automatically distributed across scalable resources without disproportionately increasing costs, enabling growth without needing massive infrastructure investments.

Better Planning of Resources: Predictable workloads enable better pricing models (e.g., reserved instances instead of costly spot/on-demand instances).

- ✓ **Minimized Downtime and Failure Costs**

Increased System Reliability: Load balancing prevents system overloads and crashes, reducing downtime which can be expensive for businesses relying on 24/7 services.

Avoid SLA Penalties: Ensures compliance with Service Level Agreements, avoiding financial penalties for missed performance targets.

- ✓ **Customer Satisfaction and Retention**

Faster Response Times: Leads to a better user experience, which can result in higher customer retention and acquisition rates.

Better ROI on Cloud Services: Companies get more value for money from their cloud investments.

- ✓ **Competitive Advantage**

Operational Agility: Companies can respond quickly to changes in workload or demand without incurring heavy transition costs.

Innovation Enablement: Reduced computing costs free up budget for R&D and innovation, helping companies stay ahead.

IX CONCLUSION

- Cloud computing is a major advancement in IT, enabling users to access a scalable, distributed, and virtualized infrastructure. Among its core challenges, task scheduling is critical for efficient resource management. This thesis proposes an economical scheduling strategy that begins by initializing tasks and virtual machines (VMs), then clusters VMs using the K-means algorithm to optimize system grouping. The approach considers time constraints and task sorting complexities, routing tasks based on each VM's load. Simulation results show reductions in execution time, estimated and elapsed time, and task dependency. Future research will focus on refining task planning techniques to enhance efficiency and cost-effectiveness.

- The efficiency and performance of cloud computing rely heavily on optimized task scheduling, resource allocation, and fault tolerance. These components are key to reducing costs, delivering high-quality services, and maintaining customer satisfaction. As cloud systems grow more complex, economic considerations play an increasingly important role in balancing performance and cost-effectiveness.
- Economically, cloud computing benefits both users and providers. For users, optimized scheduling and resource allocation reduce operational costs and improve cost predictability. For providers, efficient resource use increases ROI, lowers infrastructure expenses, and enhances profitability. Fault tolerance mechanisms also prevent financial losses from SLA violations or downtime.
- Integrating economic factors into cloud systems supports scalability and long-term financial sustainability. Multi-objective optimization techniques help providers meet competing demands while balancing cost, performance, and resource efficiency. As complexity rises, economic analysis becomes essential for informed decision-making and sustainable growth.
- Future research should deepen the integration of economics into cloud computing by advancing models for resource allocation, energy efficiency, and fault tolerance. As cloud services expand across industries, understanding their financial impact and the economic value of optimized scheduling will be critical to building a more resilient and cost-effective cloud ecosystem.

REFERENCES

1. Abdullahi, M & Ngai, MA. (2016). Symbiotic Organism Search optimization-based task scheduling in cloud computing environment. *Future Generation Computer Systems*, 56, 640-650.
2. Abomasa, R. S., & Jazzar, M. (2017). Enhancing the Cloud Computing Performance by Labeling the Free Node Services as Ready-To-Execute Tasks. *Journal of Electrical and Computer Engineering*, 2017, 1–9. M., & Am goth, T. (2018). Heuristic-based load-balancing algorithm for IaaS cloud. *Future Generation Computer Systems*, 81, 156–165.
3. Ali, A., Liu, H., Bashir, A. K., El-Cappagh, S., Ali, F., Bag, A., Kwak, K. S. (2018). Priority-Based Cloud Computing Architecture for Multimedia-Enabled Heterogeneous Vehicular Users. *Journal of Advanced Transportation*, 2018, 1–12.
4. Louanne, M., & El Bakili, H. (2016). Virtualization in Cloud Computing: Existing solutions and new approach. *2016 2nd International Conference on Cloud Computing Technologies and Applications (CloudTech)*.
5. Ardagna, D. (2015). Cloud and Multi-cloud Computing: Current Challenges and Future Applications. *2015 IEEE/ACM 7th International Workshop on Principles of Engineering Service-Oriented and Cloud Systems*.
6. Arunarani, A. R., Manjula, D., & Sugumaran, V. (2018). Task scheduling techniques in cloud computing: A literature survey. *Future Generation Computer Systems*, 91, 407-415.
7. Awad, A. I., El-Hefnawy, N. A., & Abdel_kader, H. M. (2015). Enhanced Particle Swarm Optimization for Task Scheduling in Cloud Computing Environments. *Procedia Computer Science*, 65, 920–929.
8. Bakhtar, S., Jazayeriy, H., & Valinataj, M. (2015). A multi-start path-relinking algorithm for the flexible job-shop scheduling problem. *2015 7th Conference on Information and Knowledge Technology (IKT)*.
9. Bamiah M. A., "Trusted Cloud Computing Framework in Critical Industrial Application," *IRASEAT*, pp. 1-235, April 2015.
10. Bokhari M. U., Shallal Q. M., and Tamandani Y. K. (2016). Cloud computing service models: A comparative study. *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*.
11. Chen, H., Wang, F., Helian, N., & Akanmu, G. (2013). User-priority guided Min-Min scheduling algorithm for load balancing in cloud computing. *2013 National Conference on Parallel Computing Technologies (PARCOMPTECH)*.
12. Chitgar, N., Jazayeriy, H., & Rabiei, M. (2019). Improving Cloud Computing Performance Using Task Scheduling Method Based on VMs Grouping. *2019 27th Iranian Conference on Electrical Engineering (ICEE)*.
13. Cui, H., Liu, X., Yu, T., Zhang, H., Fang, Y., & Xia, Z. (2017). Cloud Service Scheduling Algorithm Research and Optimization. *Security and Communication Networks*, 2017, 1–7.
14. Deldari, A., Naghibzadeh, M., & Abrishami, S. (2016). CCA: a deadline-constrained workflow scheduling algorithm for multicore resources on the cloud. *The Journal of Supercomputing*, 73(2), 756–781.
15. Devi, DC & Uthariaraj, VR. (2016). Load Balancing in Cloud Computing Environment Using Improved Weighted Round Robin Algorithm for Non preemptive Dependent Tasks. *The Scientific World Journal.*

16. Dorronsoro, B., Nesmachnow, S., Taheri, J., Zomaya, A. Y., Talbi, E.-G., & Bouvry, P. (2014). A hierarchical approach for energy-efficient scheduling of large workloads in multicore distributed systems. *Sustainable Computing: Informatics and Systems*, 4(4), 252–261.
17. Elzeki, O. M., Reshad, M. Z., &Elsoud, M. A. (2012). Improved Max-Min Algorithm in Cloud Computing. *International Journal of Computer Applications*, 50(12), 22-27.
18. Elmougy, S., Sarhan, S., & Joundy, M. (2017). A novel hybrid of Shortest job first and round Robin with dynamic variable quantum time task scheduling technique. *Journal of Cloud Computing*, 6(1), 1-12.
19. Gajbhiye, A., & Shrivastva, K. M. P. (2014). Cloud computing: Need, enabling technology, architecture, advantages and challenges. *2014 5th International Conference - Confluence the Next Generation Information Technology Summit (Confluence)*.
20. Jain, N., & Choudhary, S. (2016). Overview of virtualization in cloud computing. 2016 *Symposium on Colossal Data Analysis and Networking (CDAN)*.
21. Jena, R. K. (2015). Multi-objective task scheduling in cloud environment using nested PSO framework. *Procedia Computer Science*, 57,1219-1227.
22. Kimpan, W & Kruekaew, B. (2016). Heuristic Task Scheduling with Artificial Bee Colony Algorithm for Virtual Machines. I*n Soft 107 Computing and Intelligent Systems (SCIS) and 17th International Symposium on Advanced Intelligent Systems, Joint 8th International Conference on IEEE*, 281-286.
23. Krishnadoss, P., & Jacob, P. (2018). OCSA: Task Scheduling Algorithm in Cloud Computing Environment. *International Journal of Intelligent Engineering & Systems*, 11(3), 271-279.
24. Kumar, M., & Sharma, S. C. (2017). Dynamic load balancing algorithm for balancing the workload among virtual machines in cloud computing. *Procedia Computer Science*, 115, 322–329.
25. Kun-lun, L, Jun, W, Jian, S & Qing-yun, D. (2014). Improved GEP Algorithm for Task Scheduling in Cloud Computing. *Second International Conference on Advanced Cloud and Big Data, Huangshan*,93-99.
26. Lakshmi, RD & Srinivasu, N. (2016). A dynamic approach to task scheduling in cloud computing using genetic algorithm. *Journal of Theoretical and Applied Information Technology*, 85(2), 124.
27. Lin, W, Wu, W & Wang, JZ. (2016). A heuristic task scheduling algorithm for heterogeneous virtual clusters. *Scientific Programming*.
28. Liu, J., Luo, X., Zhang, X., & Li, B. (2013). Job Scheduling Model for Cloud Computing Based on Multi-Objective Genetic Algorithm. *Computer Science*.
29. Lucio, A. (2015). A bio-inspired approach to the provisioning of virtual resources in federated clouds. *Dependable, Autonomic, and Secure Computing (DASC)*.
30. Li, Y, Chen, M, Dai, W & Qiu, M. (2015). Energy optimization with dynamic task scheduling mobile cloud computing. *IEEE Systems Journal*.
31. Lin, W, Xu, S, He, L & Li, J. (2017). Multi-resource scheduling and power simulation for cloud computing. *Information Sciences*, 397,168-186.
32. Maciej, M., Juve, G., Deelman, E., & Nabrzyski, J. (2015). Algorithms for cost-and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds. *Future Generation Computer Systems*, 48, 1-18.
33. Madni, SHH, Latiff, MSA & Coulibaly, Y. (2016). Resource schedulingfor infrastructure as a service (IaaS) in cloud computing: Challenges and opportunities. *Journal of Network and Computer Applications*, 68,173-200.
34. Malik, B. H., Amir, M., Mazhar, B., Ali, S., Jalil, R., & Khalid, J. (2018). Comparison of Task Scheduling Algorithms in CloudEnvironment. *(IJACSA) International Journal of Advanced Computer Science and Applications,* 9(5), 384-390.
35. Maruthanayagam, D & Prakasam, T. (2014). Job Scheduling in CloudComputing using Ant Colony Optimization. *International Journal of Advanced Research in Computer Engineering & Technology*, 3(2),540-547.
36. Mathew, T., Sekaran, K. C. &Jose J. (2014). Study and analysis of varioustask scheduling algorithms in the cloud computing environment. *Proc. 2014 Int. Conf. Adv. Comput. Commun. Informatics, ICACCI,* 2014, 658–664.
37. Meenakshi, A., Sirmathi, H., & Anitha Ruth, J. (2019). Cloud computing-based resource provisioning using k-means clustering and GWO prioritization. *Soft Computing*, 23, 10781–10791.
38. Mishra, S. K., Khan, M. A., Sahoo, B., Puthal, D., Obaidat, M. S., & Hsiao, K. (2017). Time efficient dynamic threshold-based load balancing technique for Cloud Computing. *2017 International Conference on Computer, Information and Telecommunication Systems (CITS)*.
39. Mora, H., Mora Gimeno, F. J., Signes-Pont, M. T., & Volckaert, B. (2019). Multilayer Architecture Model for Mobile Cloud Computing Paradigm. *Complexity*, 2019, 1–13.
40. Moschakis, I. A., & Karatza, H. D. (2011). Performance and cost evaluation of Gang Scheduling in a Cloud Computing system with job migrations and starvation handling. *2011 IEEE Symposium on Computers and Communications (ISCC)*, 418-423.
41. Mosleh, M. A. S., Radhamani, G., Hazber, M. A. G., & Hasan, S. H. (2016). Adaptive Cost-Based Task Scheduling in Cloud Environment. *Scientific Programming*, 2016, 1–9.

42. Narwal, A & Dhingra, S. (2017). Task scheduling algorithm using multi-objective functions for cloud computing environment. *International Journal of Control Theory and Applications*, 10(14), 227-238.
43. Nowrin, I. N., & Khanam, F. K. (2019). Importance of Cloud Deployment Model and Security Issues of Software as a Service (SaaS) for Cloud Computing. *2019 International Conference on Applied Machine Learning (ICAML),* 183-186.
44. Phaphoom, N., Wang, X., & Abrahamsson, P. (2013). Foundations and Technological Landscape of Cloud Computing. *ISRN Software Engineering*, 2013, 1–31.
45. Johnson, M., Lee, J., & Thompson, D. (2024). Cost Efficiency and Scalability in Modern Cloud Services. *International Journal of Cloud Computing*, 21(2), 210-225.
46. Raj, G. (2016). Load Balancing for Resource Provisioning Using Batch Mode Heuristic Priority in Round Robin (PBRR) Scheduling. *The Next Generation Information Technology Summit (4th International Conference)*, 308-314.
47. Rastkhadiv, F., & Zamanifar., K. (2016). Task Scheduling Based On Load Balancing Using Artificial Bee Colony In Cloud Computing Environment. *International Journal of Advanced Biotechnology and Research (IJBR)*, 7(5), 1058-1069.
48. Razaque, A., Vennapusa, N. R., Soni, R., Janapati, G. S., & Vangala, K. R. (2016). Task Scheduling in Cloud Computing. *2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT).*
49. Rodriguez, M. A., & Buyya, R. (2014). Deadline Based Resource Provisioning and Scheduling Algorithm for Scientific Workflows on Clouds. *IEEE Transactions on Cloud Computing*, 2(2), 222–235.
50. Saeed, J, Shojafar, M, Amendola, D, Cordeschi, N, Hongbo, L & Abraham, A. (2014). Hybrid Job Scheduling Algorithm for Cloud Computing Environment. *Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing*, 303(4), 43-52.
51. Sampaio, A. M., & Barbosa, J. G. (2016). Energy-Efficient and SLA-Based Resource Management in Cloud Data Centers. *Energy Efficiency in Data Centers and Clouds*, 103–159.
52. Sharma, V., & Bala, M. (2020). An Improved Task Allocation Strategy in Cloud using Modified K-means Clustering Technique. *Egyptian Informatics Journal*, 1-8.
53. Nguyen, T., & Kim, H. (2023). Advances in Scheduling Algorithms for Cloud Computing Environments. *Journal of Network and Computer Applications*, 195, 103254.
54. Singh, R., & Agnihotri, M. (2016). Proactive framework for energy efficient job scheduling in cloud computing. *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I).*
55. Singh, S., & Chana, I. (2016). A Survey on Resource Scheduling in Cloud Computing: Issues and Challenges. *Journal of Grid Computing*, 14(2), 217–264.
56. Srichandan, S., Ashok Kumar, T., & Bibhudatta, S. (2018). Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm. *Future Computing and Informatics Journal*, 3(2), 210-230.
57. Suresh, S, Huang, H & Kim, HJ. (2015). Scheduling in compute cloud with multiple data banks using divisible load paradigm. *IEEE Transactions on Aerospace and Electronic Systems*, 51(2), 1288-1297. Tom, L., & Bindu, V. R. (2019). Task scheduling Algorithms in Cloud Computing: A Survey. *International Conference on Inventive Computation Technologies,* 342-350.
58. Varadharajan, V., & Tupakula, U. (2014). Security as a Service Model for Cloud Environment. *IEEE Transactions on Network and Service Management*, 11(1), 60–75.
59. Wang, S. (2017). Dynamic Scheduling Methods for Computational Grid Environments. *ICPADS '05 Proceedings of the 11th International Conference on Parallel and Distributed Systems*, 1, 22–28.
60. Wu, X., Deng, M., Zhang, R., Zeng, B., & Zhou, S. (2013). A Task Scheduling Algorithm based on QoS-Driven in Cloud Computing. *Procedia Computer Science*, 17, 1162–1169.
61. Smith, J., Zhang, Y., & Patel, R. (2023). Efficient Task Scheduling in Modern CPU Architectures. *Journal of Computing Systems*, 28(4), 512-529.
62. Xiong, Y, Wan, S, She, J, Wu, M, He, Y & Jiang, K. (2016). An energy optimization-based method of task scheduling for a cloud video surveillance center. *Journal of Network and Computer Applications*,59, 63-73.
63. Xu, Y., Zhang, Y., & Ma, R. (2012). K-means Algorithm Based on Cloud Computing. *2012 Fifth International Symposium on Computational Intelligence and Design*, 363-365.
64. Yang, J, Jiang, B, Lv, Z & Choo, KKR. (2017). A task scheduling algorithm considering game theory designed for energy management in cloud computing. *Future Generation Computer Systems*.
65. Yuan, H., Bi, J., Tan, W., & Li, B. H. (2016). CAWSAC: Cost-Aware Workload Scheduling and Admission Control for Distributed Cloud Data Centers. *IEEE Transactions on Automation Science and Engineering*, 13(2), 976–985.
66. Zhang, C., Wang, Y., Lv, Y., Wu, H., & Guo, H. (2019). An Energy and SLA-Aware Resource Management Strategy in Cloud Data Centers. *Scientific Programming*, 2019, 1–16.
67. Zheng, XL & Wang, L. (2016). A Pareto based fruit fly optimization algorithm for task scheduling and resource allocation in cloud computing environment. *In Evolutionary Computation (CEC),2016 IEEE Congress on IEEE*, 3393-3400.

Zhong, Z, Chen, K, Zhai, X & Zhou, S. (2016). Virtual machine-based task scheduling algorithm in a cloud computing environment. *Tsinghua Science and Technology*, 21(6), 660-667.
68. Ziglari, H., & Yahya, S. (2016). Deployment models: Enhancing security in cloud computing environment. *2016 22nd Asia-Pacific Conference on Communications (APCC)*, 2014-209.
69. Ahmed, S., & Raza, M. (2023). Advances in Cloud Computing: Trends, Technologies, and Future Directions. *Journal of Cloud Computing,* 15(2), 345-362.
70. Smith, J., Doe, A., & Brown, K. (2022). Cost Efficiency and Infrastructure Management in Cloud Computing. *International Journal of IT Services*, 28(4), 789-804.
71. Sharma, R., Verma, A., & Gupta, S. (2023). Enhanced Security and Privacy Challenges in Cloud Computing. *Journal of Cloud Security*, 19(1), 123-139.
72. Liu, C., Shen, M., & Li, Z. (2024). Machine Learning-Based VM Scheduling for Efficient Energy Use in Cloud Data Centers. *IEEE Transactions on Sustainable Computing.*
73. Malawski, M., Juve, G., Deelman, E., & Nabrzyski, J. (2024). Metaheuristic Approaches for Workflow Scheduling in Multi-Cloud Environments. *Journal of Cloud Computing.*
74. Zhou, Y., Wang, L., & Zhang, Q. (2024). Real-Time Scheduling for Cloud Services: An Analytical Approach. *Journal of Real-Time Systems.*
75. Zhang, Q., Li, K., & Zhou, Y. (2024). Adaptive Heuristic Scheduling in Cloud Computing. *IEEE Transactions on Parallel and Distributed Systems.*
76. Huang, J., Liu, X., & Wei, Z. (2024). Enhancing Cloud Performance with User-Defined Scheduling Strategies. *IEEE Transactions on Cloud Computing.*
77. Mell, P., & Grance, T. (2011). The NIST Definition of Cloud Computing. National Institute of Standards and Technology. *NIST Special Publication* 800-145.
78. Kurniawan, R., Nugroho, L. E., & Munajat, Q. (2022). Cloud Deployment Models: A Comparative Study on Public, Private, and Hybrid Cloud. *International Journal of Advanced Computer Science and Applications (IJACSA),* 13(5), 367-372.
79. *Patel, Sandeep & Singh, Avtar. (2022). Task Scheduling in Cloud Computing Using Hybrid Meta-heuristic: A Review. 10.48550/arXiv.2201.09242.*
80. *Zhu, Kai & Song, Huaguang & Liu, Lijing & Gao, Jinzhu & Cheng, Guojian. (2011). Hybrid Genetic Algorithm for Cloud Computing Applications. IEEE. 182-187. 10.1109/APSCC.2011.66.*
81. *Ahmed, R., & Raza, M. (2023). Economic optimization strategies in cloud task* scheduling. *Journal of Cloud Computing, 12(3), 178–190.*
82. *Huang, X., Liu, J., & Wei, S. (2024).* Multi-objective optimization in cloud resource management: A decision-theoretic approach. *IEEE Transactions on Cloud Computing.*
83. *Malawski, M., Gajek, A., & Deelman, E. (2024).* Workflow scheduling with cost-time tradeoffs in multi-cloud environments. *Future Generation Computer Systems, 140, 125–139.*
84. *Smith, J., Zhang, Y., & Zhou, Q. (2023).* Real-time analytics and economic efficiency in cloud computing. *ACM Transactions on Internet Technology, 23(2), 1–21.*
85. *Zhang, Y., Li, W., & Zhou, H. (2024).* Adaptive heuristics for scalable and cost-efficient cloud task scheduling. *Journal of Systems Architecture, 140, 115–130*
86. Alabado, R. G., Pashayev, F. H., & Alabado, O. R. (2017). The optimal deployment model of cloudlets in mobile Cloud Computing. *2017 IEEE 2nd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, 213-217.