SSURJET

Sir Syed University Research Journal of Engineering & Technology
2023, Vol. 13, No. 1,
https://doi.org/10.33317/ssurj.561

Exploring Virtual Machine Scheduling Algorithms: A Meta-Analysis

Salman Mahmood[1*], and Nor Adnan Yahaya[1]

[1]*Department of Information Technology, Malaysia University of Science and Technology, Selangor, Malaysia*

**Correspondence Author: Salman Mahmood (salman.mahmood@phd.must.edu.my)*

Received February 04, 2023; Revised March 11, 2023; Accepted March 15, 2023

Abstract

This review paper provides a comprehensive assessment of scheduling methods for cloud computing, with an emphasis on optimizing resource allocation in cloud computing systems. The PRISMA methodology was utilized to identify 2,487 articles for this comprehensive review of scheduling methods in cloud computing systems. Following a rigorous screening process, 30 papers published between 2018 and 2023 were selected for inclusion in the review. These papers were analyzed in-depth to provide an extensive overview of the current state of scheduling methods in cloud computing, along with the challenges and opportunities for improving resource allocation. The review evaluates various scheduling approaches, including heuristics, optimization, and machine learning-based methods, discussing their strengths and limitations and comparing results from multiple studies. The paper also highlights the latest trends and future directions in cloud computing scheduling research, offering insights for practitioners and researchers in this field.

Index Terms: *Heuristics-Based Scheduling, Optimization-Based Scheduling, Performance Optimization, Quality of Service (QoS), Scheduling Algorithms.*

I. INTRODUCTION

Cloud computing is defined in various ways in terms of services with many definitions not encompassing all its features. Key features include access to resources, OS, remote desktop virtual machines, web services, and databases [1].

Infrastructure as a service (IaaS) is a key service model of cloud computing where task scheduling algorithms have a direct impact on job performance and resource utilization. Achieving load balancing through task scheduling in virtual machines remains a challenge, but it is essential to optimize resource utilization and ensure efficient job execution.

Various task scheduling algorithms have been proposed in cloud systems to balance the workload among resources and virtual machines for optimal resource utilization and execution time. As the number of cloud users increases, it becomes challenging for Cloud Service Providers (CSPs) to respond to all requests, making it necessary to use task scheduling algorithms to minimize complexity [2]. Previous studies have explored different algorithms, including memory storage management and multi-objective task scheduling approaches that meet Service Level Agreements (SLAs) in virtual systems. Cloud-Sim has been used as a simulation tool to link code with cloudlets, virtual machines, and data centers [3].

This study aims to provide a systematic literature review of the role of task scheduling algorithms in optimizing resource utilization in cloud systems. Specifically, this review will analyze and evaluate existing research to understand the concept of cloud computing, including IaaS, and the various task-scheduling algorithms proposed in cloud systems. The review will also identify the gaps in the existing literature and suggest potential areas for further research.

II. LITERATURE REVIEW

A. The Architecture of Cloud Computing Review

Cloud computing architecture has 7 key components: application, client, network, platform, software, storage, and energy consumption. It is made up of numerous components that communicate through the application's programming interface, typically a web service and a three-tier architecture [4]. The architecture has two facets: a client instance for the user and the cloud serving as the backend for cloud software. The locations of servers, data storage systems, and data centers make up the cloud's design. The system is well-managed and more stable than its predecessors. Figure I depicts the cloud's entire design as shown below:

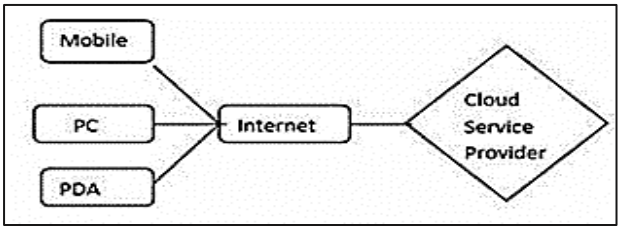

Figure I: Cloud Architecture as a Block Diagram

Cloud design has a layered architecture with four user levels as shown in figure II. Each layer is independent and

Creative Common CC BY: This article is distributed under the terms of the Creative Commons Attributes 4.0 License.
It permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

only depends on the layer above for inputs. The middleware contains frameworks of system software that allow users to build and deploy code to the cloud provider. The client has control over the provider's resources in this layer.

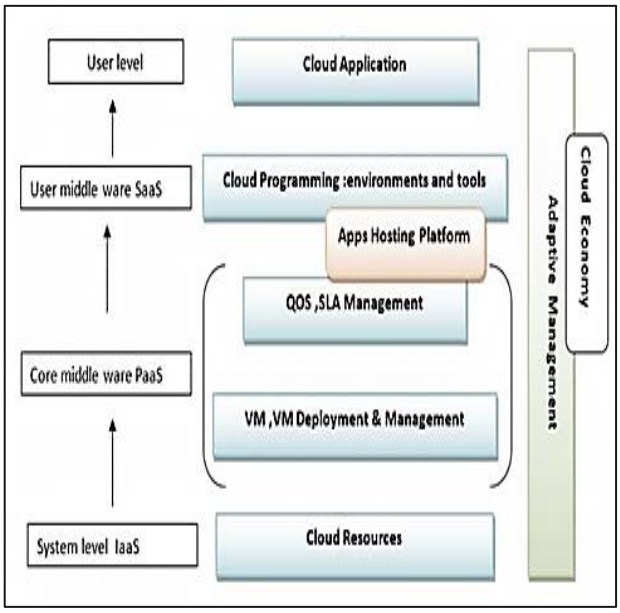

Figure II: Layered Architecture for Cloud Computing

B. Deployment Model

The National Institute of Standards and Technology (NIST) defines four traditional cloud computing models or styles: public, private, community, and hybrid [5]. These models are responsible for maintaining systems and using system funds based on the location of the hardware. The deployment model for cloud computing is depicted in figure III.

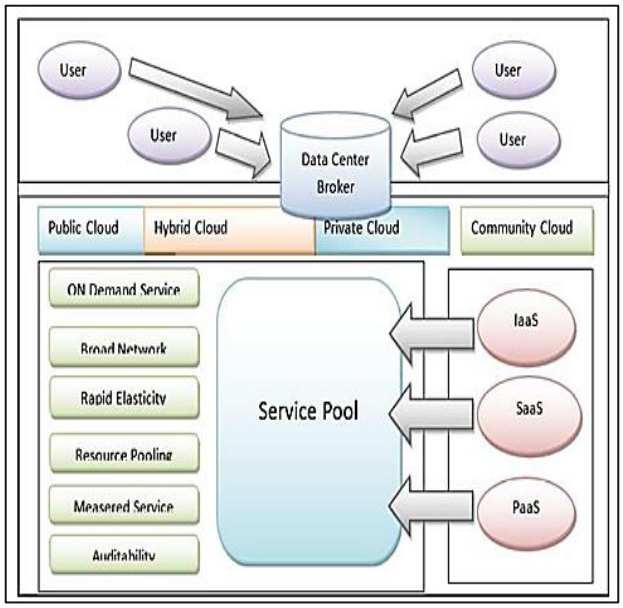

Figure III: Model for Cloud Computing Deployment

C. Scheduling in Virtual Machines

A vector-based scheduling strategy for a Virtual Machine (VM) cloud environment was proposed [6]. It involves generating permutations of tasks assigned to resources, using a three-layer approach of platforms, infrastructure, and application layers. A two-level virtual scheduling model in a cloud environment where the user interacts with the network via the application layer and can also use it to develop new applications. In the first level of scheduling, the user and the virtual machine are the parties involved, while in the second level, the virtual machine and the host assigned to it based on the first-level criteria are the parties involved. The virtual machine is selected using First Come First Serve (FCFS) scheduling and assigned to an under-loaded physical machine, which is then increased with active servers. The model is illustrated in figure IV below:

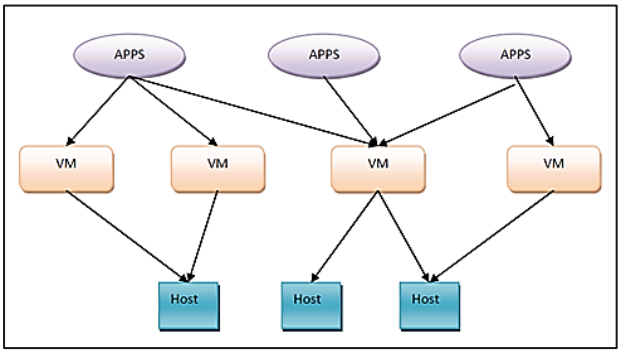

Figure IV: Model for Two-level Virtual Scheduling

A VM migration algorithm was proposed to provide quick and fair migration [7]. It uses mathematical analysis to demonstrate precision and aims to make efficient use of the network with minimal migration. The algorithm has several stages: initialization, allocation, checkpoints, and iterative log.

In an article, a genetic algorithm-based VM placement strategy was suggested to eliminate starvation [8]. The strategy prioritizes VMs using the Least Square method to determine fitness values and allocate higher priority to VMs with higher values. A VM scheduling technique was proposed for a Cloud environment to evenly distribute load among all hosts, with low cost and high performance being the primary goals [6]. The technique can be implemented in static or dynamic mode for independent or dependent tasks, respectively. The algorithm is compared to throttle and round-robin algorithms. Figure V shows the classification of VM scheduling:

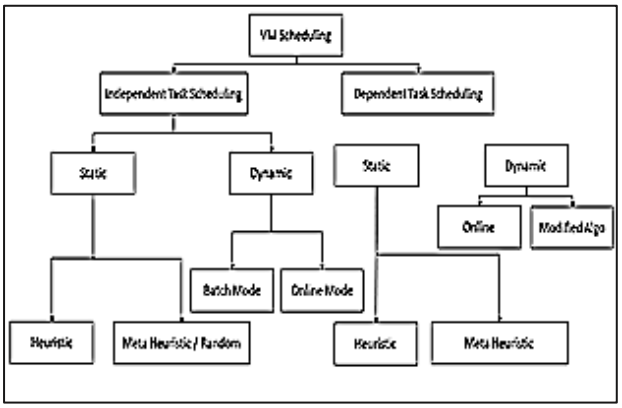

Figure V: VM Scheduling Classification

D. Dependent and Independent Tasks

Tasks can be divided into independent tasks and dependent tasks or workflow tasks based on their complexity.

Independent tasks don't require inter-task communication and are prioritized in the planning phase. The planning process of autonomous activities includes static and dynamic algorithms, as shown in figure VI. Tasks can also be categorized based on granularity as large seed and small seed assignments.

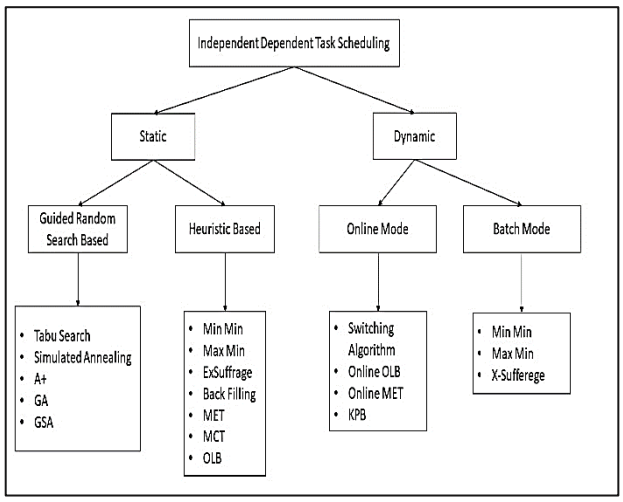

Figure VI: Scheduling Algorithms Classification for Independent Tasks

E. Heuristics and Metaheuristics Scheduling Algorithm

A heuristic is a problem-solving technique that can be successful in some cases and not in others. It's useful when optimization is difficult or impossible. Heuristics are often based on common sense or rules of thumb. A meta-heuristic is a higher-level heuristic that is used when information is limited or software capacity is insufficient. It allows for assumptions from various fields and is more relevant when an ideal solution is desired but machine feasibility is questionable [9].

Metaheuristics can consider almost perfect alternatives in comparison with other algorithms. Few algorithmic methods can be confined only to the best possible local or global optimum, but metaheuristic methods go beyond the local maxima to the global maxima. A range of storm optimization is introduced by various metaheuristic techniques so that findings are based on randomly produced variables [10].

F. Task Scheduling

Scheduling is essential for managing a large number of requests in the cloud. It's challenging to schedule tasks and allocate resources in the cloud, so an efficient scheduling algorithm is necessary.

a) Static Mode:

Scheduling uses heuristics-based algorithms to schedule independent tasks. The Balanced Minimum Completion Time (BMCT) heuristic set has been suggested to improve scheduling efficiency. It uses FCFS for initial allocation and then balances the load between computers by swapping tasks. BMCT is seen to have promising outcomes compared to Dynamic Level Scheduling. Other algorithms like Critical Path On a Processor (CPOP) and Heterogeneous Earliest Finish Time (HEFT) are used in heterogeneous settings.

b) Opportunistic Load Balancing (OLB):

It is a scheduling algorithm that aims to make all resources or computers as busy as possible. Tasks are assigned randomly to available machines without considering their execution time. Opportunistic Load Balancing (OLB) is easy to implement but may result in poor make-span as it does not consider the execution time of tasks. The efficiency of the OLB algorithm has been studied, with the aim of assigning the chosen job to VMs that are available and have the least load compared to other VMs. The algorithm scales each VM's current load and selects the VM with the minimum load to run the job [11].

c) Minimum Execution Time (MET) Algorithm:

It allocates tasks to virtual machines based on their estimated best execution time, rather than their availability. It seeks to assign the best machine for each task, leading to potentially large load imbalances.

d) Minimum Completion Time (MCT) Algorithm:

It assigns tasks to VMs based on the shortest finishing time. It can assign tasks in random order or based on predictable processing time resources. A combination of Minimum Completion Time (MCT) and the Minimum Execution Time (MET) algorithm, called MECT, has been recommended as a better scheduling technique in heterogeneous computing systems with higher efficiency in minimizing make-span compared to basic MCT and MET algorithms.

e) Min-Min Heuristic:

It is an algorithm that assumes that assigning higher jobs to the earliest and fastest executing devices, results in a lower make-span. In the case where there are both short and long tasks, the max-min algorithm may be used to decrease the wait for longer tasks and result in a better make-span and lower machine imbalance. An improved Min-Min algorithm for cloud computing task scheduling has been proposed to improve performance and satisfaction of Quality of Service (QoS). The algorithm prioritizes the earliest finishing tasks with the lowest execution time and finds the best schedule. Another variation, Mul-QoS-Min-Min, has been proposed that considers resource and task resemblance, leading to improved execution time and QoS satisfaction compared to the traditional Min-Min algorithm [12]. A method of hybrid scheduling consisting of Longest Job First and Min-Min has been suggested to lower the make-span for work scheduling in a diverse grid. The simulation results show improved output as the make-span is reduced compared to other methods. The Load Balancing Min-Min (LBMM) approach has been revised for scheduling static tasks and maximizing cloud computing resource usage [13].

f) Max-Min Heuristic:

It determines the quickest completion times for each task using the Min-Min method and assigns tasks to an overloaded machine with the maximum average completion time. It performs better than the Min-Min algorithm in scenarios where short and long tasks exist and outperforms the Min-Min method when tasks are short. Tasks with longer durations are assigned to machines in a

manner similar to the Min-Min method, and the cycle repeats as tasks are scheduled [14].

g) Backfilling Algorithms:

A study of self-adaptive backfilling algorithms, for multi-channel parallel devices, shows that the system's predictions are better than the estimated runtime for traditional backfilling [15]. The International Business Machines Corporation (IBM) study demonstrates the effectiveness of backfilling algorithms in parallel systems and compares it to the commonly used FCFS technique for job scheduling. It also considers the CONSERVATIVE and EASY backfilling algorithms and their ability to shift small jobs to fill gaps produced by FCFS. The study also mentions that FCFS, along with backfilling algorithms, is available for scheduling jobs in the backfilled queue [16].

G. Static-Metaheuristics-based Independent Task Scheduling

a) Genetic Algorithm:

A study suggested an approach based on the first fit used by cloud computing systems like Eucalyptus to solve the starvation issue but lower task make-span [17]. The technique fails to optimize resource utilization as tasks are executed on every resource. A genetic algorithm approach lays out provisions for a static instructions genetic set, which considers the overall time of completion, average task completion time, and cost considerations [18].

b) Simulated Annealing (SA):

It is a common heuristic method that uses a simulation based on the physical annealing of strong metals. It is a standard and probabilistic meta-algorithm for global optimization problems. In [19], the Simulated Annealing, Cuckoo Search Algorithm, and Firefly Algorithm were used to find the best alternative for effective resource utilization and the Firefly Algorithm outperformed the other two methods in task scheduling [20].

c) Tabu Search (TS):

The hybrid flow-shop scheduling problem has been researched as a Mixed-Integer Programming (MIP) model, followed by a TS-based algorithm. The efficiency of heuristics for flow-shop planning was evaluated, showing that the best technique varies based on problem size, required solution quality, and available time [19].

d) Gravitational Search Algorithm (GSA):

The modified GSA algorithm that combines GSA and Particle Swarm Optimization (PSO) performs better in terms of classification accuracy and choice capacity compared to Support Vector Machine (SVM) or GSA-SVM. It uses the concept of mass and gravity in task scheduling, where particles act as agents with mass that interact and move towards higher weight masses through a gravity force. This results in improved performance [21]. The author presented a meta-heuristic optimization method called GSA [22]. It was implemented for power system economic operation, where it calculates the total generated power in the internal area and the power borrowed from different areas for the most economical load specification.

H. Dynamic Mode - Independent Tasks Scheduling

It is further possible that the algorithm of OLB, MCT, and MET, are used for addressing the changes done in the previous section for dynamic planning of the autonomous assignments within the internet mode [55]. Other algorithms included are the switching algorithm as k-Percent Best (KPB).

a) Switching Algorithm (SA):

The SA algorithm uses MET and MCT heuristics cyclically for load distribution among devices. MET selects the best device for a task but can assign significant tasks to similar devices, while MCT balances the load but may not handle multiple jobs with different execution times. If a job arrives randomly, MET balances the load at a low cost and then MCT changes the load among devices.

b) K-Percent Best (KPB):

The Heuristic KPB assumes a subset of the work planning devices, consisting of the best k devices based on task completion time. A good k value would schedule the job to a computationally superior machine. The aim is to eliminate the practice of removing a device from the current task account, leading to a shorter production time than MCT.

I. Workflows or Dependent Task Scheduling

Figure VII shows categories of tasks dependent on different scheduling algorithms. Dynamic and static scheduling algorithms are used to schedule dependent tasks [23].

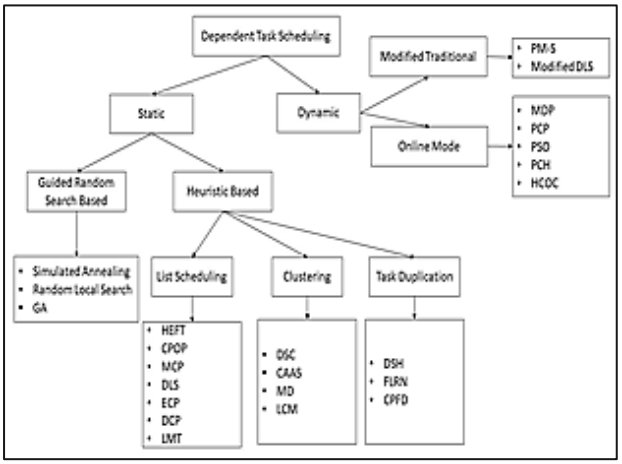

Figure VII: Scheduling Algorithms Classification for Dependent Tasks

a) Static Mode:

Heuristic-based algorithms and directed random-search-based algorithms are the two categories of static algorithms for independent and selected jobs, which may result in division, into three types: clustering heuristics, list planning heuristics, and task duplication heuristics. A study discussed a challenge with parallel constraints in cloud IaaS scheduling, focused on energy efficiency [24]. The author proposed a parallel bi-objective hybrid genetic algorithm that balances energy requirements and make-span based on DVS to save energy. The NP-hard issue with the heuristic-based approach leads to three classifications:

algorithms for lists, algorithms for clustering, and algorithms based on task duplication.

b) List Scheduling Heuristics:

Tasks based on the specified graph are described by objectives in the scheduling heuristics list and a structured list of tasks is formed. Tasks are then selected based on priorities and planned for the processor that minimizes the cost function. Besides categorization, Modified Critical Path (MCP), Mapping Heuristic (MH), Dynamic Level Scheduling (DLS), Levelized-Min Time (LMT), Critical-Path-On a Processor (CPOP), and Heteroge proposed a method for dividing a large task into smaller tasks that can be executed on multiple cloud VMs. They developed a cost-efficient task scheduling approach by combining two heuristic scheduling algorithms to list such tasks as DAG (Direct Acyclic Graphs) [25].

c) Clustering Heuristics:

The use of clustering techniques for processors with an unlimited number of processors. The method maps tasks in a specified graph to clusters, which are refined in each iteration by merging several clusters. The tasks in the same cluster are assigned to the same processor. Examples of clustering algorithms are Linear Clustering, Dominant Sequence Clustering (DSC), Directed Mobility, and Clustering and Scheduling (CASS) scheme.

d) Task Duplication Heuristic:

Task duplication algorithms aim to reduce coordination overhead by grouping redundant tasks together. The algorithms differ in task duplication selection strategy. They are not as practical as other algorithms due to higher time complexity. The different types of algorithms under this heuristic include Critical Path to Quick Duplication, Heuristic Duplication Bottom-up Top-Down, Heuristic Duplication Scheduling, Next Reduction, and First Duplication.

e) Dynamic Mode:

Researchers proposed a model for variation in VM computing rate using normal distribution re-education percentage [26]. It uses statistical models and tables to calculate the normal or exponential distribution of independent execution time and commute time. The algorithm works on a wide range of computing algorithms and reduces task delays by rescheduling tasks to faster resources, but only after delays have occurred. Research implements dynamic scheduling for a group of tasks with different arrival rates [27]. The research in [28] defines efficiency as the likelihood of completing a workflow within a set time limit and avoiding exceeding period restrictions. To fulfill SLA requirements, a proposal was made to map customer and service provider requirements in a QoS-aware workflow management and use a bi-objective function for execution cost. The implementation of a timeline-based workflow partitioning approach is discussed by researchers [29]. Several publications suggest using metaheuristic planning techniques, such as hybrid GA, to reduce execution time and cost for improved efficiency.

III. METHODOLOGY

A. Search Strategy

This article is a systematic literature review of cloud computing scheduling methods. The authors searched several databases for studies published between 2018-2023 that included keywords related to cloud computing scheduling, such as Energy Efficiency, Heuristics-Based Scheduling, Load Balancing, and Machine Learning-Based Scheduling. The authors reviewed and included only English-language articles that met their inclusion criteria and used standardized methods and tools to extract information from the articles. The inclusion and exclusion factors are described in more detail in table I. Here table II describes the search strategy of the research. The quality of articles was assessed using the PRISMA checklist as shown in figure VIII. The authors screened titles and abstracts, retrieved potential articles, and screened full-text articles for inclusion based on set criteria. The rest were excluded.

Table I: Criteria (Including and Excluding)

Inclusion	**Justification**
Published papers in 2018-2023 in journals and conference proceedings.	Use the most recent findings only
The paper presents the Quality of Service (QoS) associated with the cloud.	The review of measures for Quality of Service (QoS) would evaluate the various metrics used to assess the performance of scheduling and load balancing algorithms, such as response time, throughput, and utilization.
Exclusion	**Justification**
Papers, which are not in the English language	The standardization of English as a global language has been established.
Review papers, meta-analyses, surveys, case reports, comments, letters, presentations/posters presented at international conferences.	Focus on original research.
Articles that have redundant data.	Duplicate data has been ignored for repetition.

Table II: Search Strategy of the Research

Searching	**Search Terms**
Science Direct, Springer, IEEE, Wiley, MDPI, Hindawi, Inderscience and IGI	1. Energy Efficiency 2. Heuristics-Based Scheduling 3. Load Balancing 4. Machine Learning-Based Scheduling 5. Optimization-Based Scheduling 6. Performance Optimization 7. Quality of Service (QoS) 8. Resource Management 9. Resource Utilization 10. Scalability 11. Scheduling Algorithms 12. Virtual Machines (VMs) Strategy: #1 AND #2 AND #3 AND #4 AND #5 AND #6 AND #7 AND #8 AND #9 AND #10 AND #11 AND #12

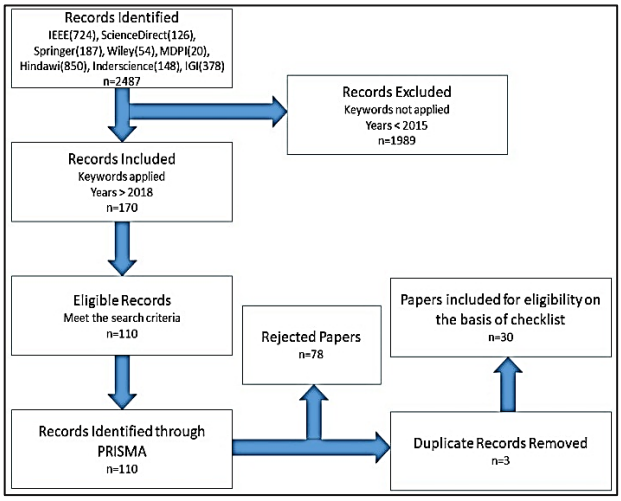

Figure VIII: Flowchart for Article Screening and Selection

IV. DISCUSSION AND FINDINGS

VM scheduling in Cloud Computing (CC) has received significant attention with several studies evaluating different scheduling strategies based on factors such as QoS, scalability, dependability, and cloud environment. The studies break down scheduling strategies by examining performance indicators such as SLA violation and power utilization. Load balancers use scheduling techniques to choose backend servers for virtual machine requests and redistribute VMs for improved workload distribution [30]. A study proposes a VM scheduling technique that considers historical VM usage for improved performance [31].

Cloud apps can quickly run out of memory without efficient load balancing. A solution using file category formatting for improved load balancing in CC with large content is proposed. The most popular scheduling methods in CC are listed in table III and use heuristics or metaheuristics to create optimal routes for jobs to accessible VMs, which cannot be achieved in a set time using traditional deterministic methods [32].

The purpose of this systematic literature review is to identify the cloud computing scheduling methods based on the QoS measures.

Table III: Cloud Computing Scheduling Methods

Mechanisms Used	Measures for QoS	Technology for Performance Evaluation	Merits	Demerits
Scheduling- Load Balancing [33]	Review	CloudSim	Emerging domains detected	Fundamental review
Metaheuristic [34]	Response Time, Cost	CloudSim	Reliable, Cost and Response time reduced	Higher computation costs, fewer services available
Scheduling-Energy Aware [35]	Execution Time, Energy	CloudSim	Resource utilization and energy efficiency are improved, and execution time is reduced	There are limitations on the volume of work and the process deadlines
Scheduling- Load Balancing [36]	Response Time, Cost	CloudSim	Lower cost, Decreased Response Time	The number of tasks and their complexity are not discussed
Framework for Resource Provisioning [37]	Response Time and Cost	CloudSim	Accuracy and reduced cost and response time	Throughput and energy efficiency characteristics are not covered
System for Controlling Elasticity [38]	Response time, elasticity, and resource utilization	CloudSim	Elasticity, lower response time, and higher resource utilization	Issues with Scalability
MFO-based Scheduling [39]	Execution Time, Makespan	iFogSim	Reduced execution time and duration	Scalability is not well proven because fewer nodes are used
BWM-VIKOR-based Scheduling [40]	Utilization of VM, Throughput Time, Makespan	CloudSim	Increased VM usage, decreased makespan, and increased throughput	Fewer virtual machines and duties
BWM-TOPSIS-based Scheduling [41]	Utilization of Resource, Makespan, Energy Consumption	CloudSim	Higher VM utilization, Decreased Makespan, Better energy consumption	Reliability challenges, small scale data centers c Considered
DVFS-PL-based Scheduling [42]	SLA Violation, Execution Time	CloudSim	Minimum SLA violation, Minimized Execution Time	More number of VMs need to be considered
Provisioning of Resources [43]	Utilization of Resources, Cost, Response Time	CloudSim	Minimized cost and Response Time	Wide-ranging user needs were not taken into account
MOB and BAT-LBRC Scheduling [44], and [45]	Accuracy, Efficiency, merging clusters, Decision Making	CloudSim	More accurate and efficient, similar clusters Mergers, improved decision making.	Response time and throughput aren't addressed, and scalability issues are present.
System for PLB-HDD Optimization [46]	Cost of Execution, Makespan	CWS	Decreased execution costs, improved timeliness	Scalability issues and the utilization of fewer virtual machines
AEFS-WOA and CSO-IRRO Scheduling [47], and [48]	Convergence, Execution Time, Throughput Time, Response Time	CloudSim	Reduced response, execution, and throughput times and faster convergence	Decision-making capability was constrained by a lack of scalability and the usage of fewer datasets

Mechanisms Used	Measures for QoS	Technology for Performance Evaluation	Merits	Demerits
TGA-EHO Scheduling [49]	Consistency, Location Search, Accuracy	CloudSim	Better accuracy and consistency, faster location search	Lesser number of nodes is considered
SA-HHO Scheduling [50]	Scheduling of jobs, makespan	CloudSim	Better job scheduling, shorter lead times	Lesser jobs and QoS metrics are considered
EELBP Scheduling [51]	Energy Use, Reaction, and Calculation Time	Eucalyptus	Computation time is decreased while response time and energy use are enhanced	Scalability problems; ML technique not employed
ICSO Scheduling [52]	F-Measures, CEC Function, Clustering Issues	MATLAB	Clustering and CEC functions have been improved	There is no mention of energy consumption, response time, or throughput
Metaheuristic Hybrid Algorithm [53]	Makespan, Throughput, execution time	CloudSim	Lower makespan, Increased throughput, better execution time	Concerns about energy use were overlooked
PSO Scheduling [54]	Execution Time, Accuracy	Google Cloud	Higher efficiency	Lower accuracy prediction
SLA-Aware Load Balancing: Scheduling [55]	Energy Consumption, Migration Time	MATLAB	Lesser migration time, improved energy consumption	Execution time and throughput are not discussed
SLA-Agile Dependent VM Scheduling [56]	SLA violations	CloudSim	Reduce SLA Violations	There are no QoS metrics available
Vanet Optimization-Metaheuristic [57]	Network overhead, Energy	NS2	Decreased overhead, improved consumption of energy	Degradation of performance, increased cost of computations
MPSO- Scheduling [58]	Utilization of Resources, Task overhead	CloudSim	Higher Resource utilization reduced task overhead	Scalability is not addressed, and a lower number of VMs and jobs are evaluated
MLP-ABC Scheduling [59]	Accuracy	CloudSim NSL-KDD	Improved Kappa Value, MAE and RMSE	Apart from accuracy, QoS measures have not been confirmed
Straggler Prediction and Mitigation Technique (START) [60]	Violation rate and execution time	CloudSim	Reduction in response time, Fewer SLA violations, and Efficient resource utilization	There are no QoS metrics available

Current load balancing techniques in cloud computing include Round Robin, Weighted Round Robin, Least Connections, Weighted Least Connections, IP Hash, and Domain Name System (DNS) load balancing. However, these load-balancing techniques have limitations that can impact cloud performance. For example, Round Robin and Weighted Round Robin may not distribute the load evenly and may not consider server capacity. Least Connections and Weighted Least Connections may not consider the location of the requesting client or the location of the servers. IP Hash may not work well for large-scale systems, and DNS load balancing may be slow to respond to changes in server availability. Load balancing can impact cloud performance by affecting response time, throughput, and resource utilization. Poor load balancing can lead to the overloading of some servers while others remain underutilized, resulting in reduced performance and potential service disruption.

To address load balancing challenges in cloud computing, several potential solutions have been proposed based on QoS, including:

- Dynamic Load Balancing: dynamically adjust server allocation based on real-time workload and resource utilization to optimize QoS.
- Machine Learning-Based Load Balancing: use machine learning algorithms to predict future workload and resource usage and make load balancing decisions accordingly.
- Cloud Orchestration: use an automated orchestration system to manage resources and allocate workloads based on QoS criteria.
- Hybrid Load Balancing: combine multiple loads balancing techniques to achieve optimal load balancing based on QoS requirements.

Overall, load balancing plays a crucial role in ensuring optimal performance and reliability in cloud computing, and ongoing research is focused on developing more efficient and effective load-balancing techniques to address the evolving needs of cloud computing.

The implications of this study are significant for both researchers and practitioners in the field of cloud computing. By analyzing the current state-of-the-art task scheduling algorithms in cloud computing systems, this study provides insights into the strengths and weaknesses of existing methods and identifies potential areas for improvement.

Researchers can use the findings of this study to develop more effective task-scheduling algorithms that can further optimize resource utilization in cloud computing systems. Additionally, the study highlights the need for further research on the application of metaheuristic algorithms to cloud computing systems, which could potentially lead to more efficient task-scheduling methods.

Practitioners in the field of cloud computing can benefit from the insights provided by this study by gaining a deeper understanding of the role of task-scheduling algorithms in optimizing resource utilization. They can use this knowledge to make informed decisions about the selection and implementation of task scheduling algorithms in their own cloud computing systems, ultimately leading to improved performance and cost savings.

V. RECOMMENDATION

The private cloud is a system that allows you to store data privately. The CC environment is made up of Eucalyptus. Eucalyptus supports a variety of hypervisors and Linux operating systems [61], and [62], and it is made up of five essential components: Examples of cloud regulators include the Cloud Regulator, Cluster Regulator, Node Regulator, Walrus, and Storage Regulator. Major scheduling choices are made by cloud regulators, who also communicate demands to cluster regulators. It is in charge of virtualized resource management. The cluster regulator manages VM instances and plans VM execution on explicit nodes. The cluster regulator and storage regulator work together. Virtual machines are started and stopped by the node regulator. When it comes to cloud VM scheduling, Round Robin (RR), Greedy, or Power Save strategies are employed by Eucalyptus. With one VM routinely assigned to each host, the RR VM scheduling strategy prioritizes equally distributing loads across all hosts.

Before connecting with the next VM to establish a connection with the next host, the scheduler assigns virtual machines to each host. Each host goes through this process until at least one virtual machine is present on each. The RR approach for VM scheduling has the main advantage of making fair use of all available resources. All hosts are given an equal number of VMs to guarantee that everything is in order. The use of the RR technique for VM scheduling has the significant disadvantage of increasing power consumption because several hosts will be turned on for an extended period of time. A Greedy VM selects the first node that can meet the underlying requirements. As a result, although Greedy's power consumption is lower, load balancing is not achieved. In comparison to these two, the PowerSave approach for VM scheduling saves more electricity.

Open Nebula, another standalone cloud, supports a variety of hypervisors and operating systems and includes components including a front-end, hosts with hypervisor support, data stores, service networks, and virtual machine networks [63]. For cloud VM scheduling, Open Nebula employs a Match Making approach. The matchmaking technique prioritizes VM allocation to hosts with high-ranking expressions, which is important when using strategies like striping, packing, and load-aware rules. Several authors sought to reduce switching times in CC by using parallel [64], and [65] approaches.

The OpenStack cloud includes computing, a dashboard, block, and object storage, identity services, a database, and image services [66]. For internal communication, the OpenStack cloud components employ the RMQ Protocol. The VMs in the Nova Computing Node hosts are scheduled using the Filter scheduling method that makes use of a weighing process and filtering. To filter compute node hosts, filter settings would be utilized. Based on the filtered hosts, a weighing operation will begin. Table IV compares the Eucalyptus, OpenStack, and Open Nebula clouds' VM scheduling techniques.

Table IV: Search Strategy of the Research

Cloud Environments	Techniques for VM Scheduling	Important Properties
OpenStack	Filter Scheduling Algorithm	Based on Memory Awareness
Open Nebula	Algorithm for finding matches	Efficiency in Terms of Costs
Eucalyptus	Round Robin Algorithm	Efficiency in Terms of Time
	PowerSave Algorithm	Saves Electricity
	Greedy Algorithm	Power Usage is Reduced

The study in reference [67] describes an IDS framework with various IDS sensors placed throughout the virtual infrastructure. Each sensor monitors a virtual component and the central management unit oversees all sensors, combining and correlating alarms. The central management unit has four parts: Event Gatherer, Event Database, Analysis Controller, and IDS Remote Controller as shown in figure IX. The Event Gatherer collects and archives alerts, the Analysis Controller uses correlation to detect multi-event attacks, and the IDS Remote Controller manages the configuration and lifespan of each IDS sensor.

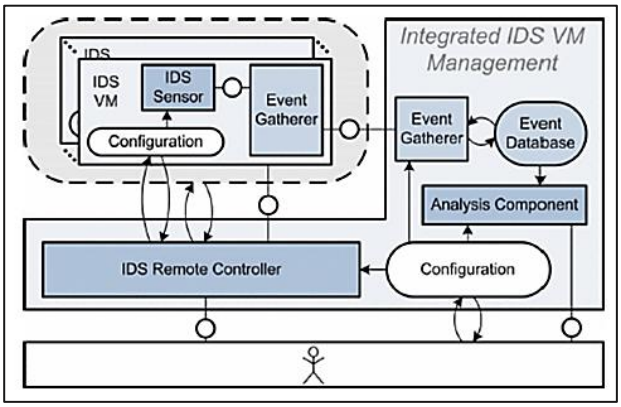

Figure IX: Cloud IDS architecture [67]

Multiple IDS sensor types but lacks consideration for the dynamic nature of virtual infrastructure. It does not address the issue of VM migration and sensor adaptation to changes, such as new services in the VMs. Another approach in reference [68] places network-based IDS sensors next to computing nodes in IaaS to reduce DoS incidents but still has limitations, such as the need for IDS reconfiguration and handling unexpected traffic increases. The approach supports only network-based IDSs, unlike the variety of IDSs in [67].

Livewire is an inactive [69], early Virtual Machine Introspection or VMI-based intrusion detection system that utilizes the hypervisor qualities of isolation, inspection, and interposition for strong isolation and host-based visibility. Figure X shows its architecture.

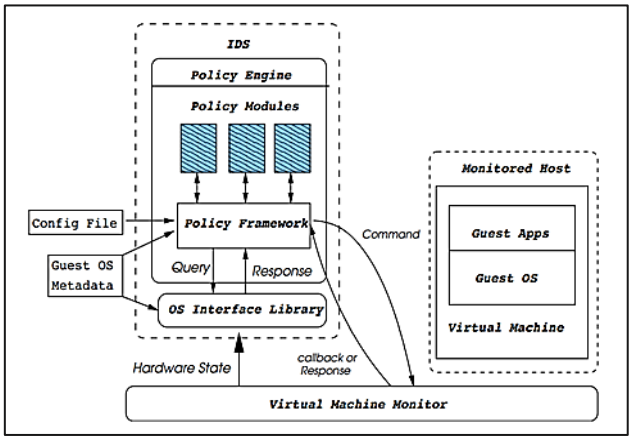

Figure X: Structure of Livewire

Livewire is an intrusion detection system using Virtual Machine Introspection (VMI) approaches. It uses three hypervisor qualities of isolation, inspection, and interposition to create strong isolation from malicious attackers while keeping the visibility of a host-based IDS. The architecture of Livewire includes an OS interface library, which bridges the semantic gap by translating hardware events into OS-level structures, and a policy engine, which determines if the system has been corrupted. The prototype was tested on a VMware workstation, but it has limitations including the inability to manage dynamic events in a cloud environment, lack of adaptation to new services added to the virtual machine, and support for only one virtual machine per IDS.

CloudSec aims to provide active monitoring of several collocated VMs without putting any sensitive code inside the untrusted VMs. [70] used VMI to identify kernel data rootkits by creating dynamic guest kernel data structures. Instead of direct access to the memory pages of the VMs, it interacts with the hypervisor and stores the pertinent pages in a unique memory page buffer. A specialty module is required for CloudSec (KDS) to load information about the OS kernel data structures of the VMs. The Semantic Gap Builder (SGB) narrows the semantic gap and generates a profile of the monitored VM.

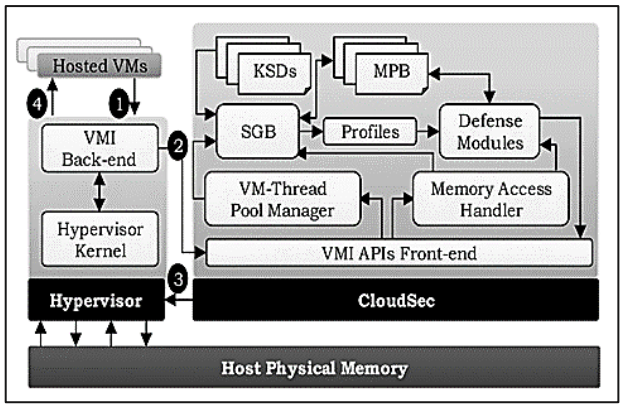

Figure XI: Architecture for CloudSec [70]

The Defense Modules, which perform the actual detection, use the profile. The architecture of CloudSec is shown in figure XI.

Although CloudSec permits concurrent active monitoring of several VMs, its performance costs in a multi-tenant context have not been studied. Furthermore, turning off an infected VM is the extent of the active monitoring capabilities. CloudSec is only compatible with the VMware ESXi hypervisor and does not address dynamic events.

We modify NIDSs to account for the dynamic shifts in a cloud context, ensuring that network traffic traveling to and from the information system housed in the cloud is properly monitored. Our contribution deals with the adaptation-related concerns that the earlier-presented solutions neglect to address.

This list of firewall options for cloud settings includes cloud-specific firewalls. We concentrate on industrial solutions because developing fresh cloud firewall solutions or enhancing existing ones requires a lot of work. Our emphasis is divided between firewalls for apps and firewalls for the future. Future-Generation Firewalls One of the most significant cyber hazards to a cloud environment is large-scale distributed assaults, which cause several security incidents at different levels of a cloud architecture. One way to stop these assaults is by integrating a next-generation firewall into the cloud architecture. Deep packet inspection for network traffic analysis, access control for online apps that are user-focused, and access control for Internet applications that are application-focused are just a few of the features that the future generation of firewalls will be able to execute in one device.

VMware and Palo Alto Networks have partnered to create VMSeries, a next-generation firewall with application-driven access management and the ability to dynamically adjust security policies [71]. The firewall includes a new feature, "tags", which allow security rules based on VM attributes. The system integrates with the NSX security suite and provides access to network traffic and topology data, but does not allow component sharing among tenants and does not account for tenant-specific security requirements. The VMSeries is linked to Amazon EC2.

Application-level firewalls have emerged as a solution for monitoring network traffic caused by specific applications. These firewalls filter network packets based on rules that consider the protocols and states of the relevant apps. Implementing this solution for web apps in a cloud environment provides defense against recognized application-level dangers like SQL injection or cross-site scripting [72]. Amazon Web Application Firewall (WAF) is available to tenants who can create custom security policies based on their hosted apps and use load balancers to implement the rules. Tenants have a lot of latitudes to customize the ruleset, but dynamic events are not considered by the WAF and it's unclear if component sharing is possible for rules from multiple tenants.

VI. CONCLUSION

In summary, cloud computing is a multifaceted concept that incorporates various services, with IaaS being a key feature that includes remote desktop VMs, web services, and databases. Task scheduling algorithms play a significant role in optimizing resource utilization and balancing the workload among resources and VMs. However, achieving optimal resource utilization and execution time is challenging, necessitating a multi-

objective task scheduling approach that adheres to SLAs. Researchers have proposed algorithms for task scheduling in IaaS and used simulation tools like Cloud-Sim. Additionally, CSPs offer low-cost storage services on a pay-per-use basis, making cloud computing an attractive option for managing and providing computing applications. The private cloud is a system for storing data privately, with Eucalyptus being a CC environment with five components. Eucalyptus uses Round Robin, Greedy, or Power Save strategies for cloud VM scheduling. Similarly, Open Nebula and OpenStack use Match Making and Filter scheduling methods, respectively. Table IV provides a comparison of VM scheduling techniques used by these standalone clouds.

Various firewall options for cloud environments, with a focus on industrial solutions. The three options reviewed include Next-Generation Firewalls, Application-level Firewalls, and CloudSec. Next-Generation Firewalls provide deep packet inspection and access control, but the VMSeries firewall has limitations such as a lack of component sharing among tenants and not accounting for tenant-specific security requirements. Application-level firewalls, such as Amazon Web Application Firewall, offer protection against specific application-level dangers, but dynamic events are not considered, and it remains unclear if component sharing is possible for rules from multiple tenants. CloudSec allows for concurrent active monitoring of VMs, but its performance cost in a multi-tenant context has not been studied, and it only turns off infected VMs as a response. Future research could focus on modifying Network Intrusion Detection Systems to account for dynamic changes in cloud environments and improve the security of cloud systems.

Acknowledgment

The authors would like to thank the management of Malaysia University of Science and Technology, Selangor, Malaysia, for their support and their assistance throughout this study.

Authors Contributions

Both authors equally contribute to achieve the objectives of this research study.

Conflict of Interest

The authors declare no conflict of interest and confirm that this work is original and not plagiarized from any other source, i.e., electronic or print media. The information obtained from all of the sources is properly recognized and cited below.

Data Availability Statement

The testing data is available in this paper.

Funding

This research received no external funding.

References

[1] Al Hasani, I. M. M., Kazmi, S. I. A., Shah, R. A., Hasan, R., & Hussain, S. (2022). IoT based Fire Alerting Smart System. *Sir Syed University Research Journal of Engineering & Technology, 12*(2), 46-50.

[2] Chawla, Y., & Bhonsle, M. (2012). A study on scheduling methods in cloud computing. *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), 1*(3), 12-17.

[3] Madni, S. H. H., Abd Latiff, M. S., Abdullahi, M., Abdulhamid, S. I. M., & Usman, M. J. (2017). Performance comparison of heuristic algorithms for task scheduling in IaaS cloud computing environment. *PloS one, 12*(5), e0176321.

[4] Smanchat, S., & Viriyapant, K. (2015). Taxonomies of workflow scheduling problem and techniques in the cloud. *Future Generation Computer Systems, 52*, 1-12.

[5] Mell, P., & Grance, T. (2011). The NIST definition of cloud computing. Retrieved from: https://csrc.nist.gov/publications/detail/sp/800-145/final

[6] Mishra, N. K., & Mishra, N. (2016). CELBT: An Algorithm for Efficient Cost based Load Balancing in Cloud Environment. *International Journal of Computer Applications, 134*(1).

[7] Liu, J., Pacitti, E., Valduriez, P., & Mattoso, M. (2015). A survey of data-intensive scientific workflow management. *Journal of Grid Computing, 13*, 457-493.

[8] Pilavare, M. S., & Desai, A. (2015, March). A novel approach towards improving performance of load balancing using genetic algorithm in cloud computing. In *2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)* (pp. 1-4). IEEE.

[9] Mandal, T., & Acharyya, S. (2015, December). Optimal task scheduling in cloud computing environment: meta heuristic approaches. In *2015 2nd International Conference on Electrical Information and Communication Technologies (EICT)* (pp. 24-28). IEEE.

[10] Rubrico, J. I. U., Ota, J., Higashi, T., & Tamura, H. (2008). Metaheuristic scheduling of multiple picking agents for warehouse management. *Industrial Robot: An International Journal, 35*(1), 58-68.

[11] Topcuoglu, H., Hariri, S., & Wu, M. Y. (2002). Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE transactions on parallel and distributed systems, 13*(3), 260-274.

[12] Wang, G., & Yu, H. C. (2013). Task scheduling algorithm based on improved Min-Min algorithm in cloud computing environment. In *Applied Mechanics and Materials* (Vol. 303, pp. 2429-2432). Trans Tech Publications Ltd.

[13] Tsai, C. W., Huang, W. C., Chiang, M. H., Chiang, M. C., & Yang, C. S. (2014). A hyper-heuristic scheduling algorithm for cloud. *IEEE Transactions on Cloud Computing, 2*(2), 236-250.

[14] Devipriya, S., & Ramesh, C. (2013, December). Improved max-min heuristic model for task scheduling in cloud. In *2013 international conference on green computing, communication and conservation of energy (ICGCE)* (pp. 883-888). IEEE.

[15] Barry, D. K., & Dick, D. (2013). *Web Services, Service-Oriented Architectures, and Cloud Computing: The Savvy Manager's Guide.*

[16] Tsafrir, D., Etsion, Y., & Feitelson, D. G. (2007). Backfilling using system-generated predictions rather than user runtime estimates. *IEEE Transactions on Parallel and Distributed Systems, 18*(6), 789-803.

[17] Brent, R. P. (1989). Efficient implementation of the first-fit strategy for dynamic storage allocation. *ACM Transactions on Programming Languages and Systems (TOPLAS), 11*(3), 388-403.

[18] Fang, Y., Wang, F., & Ge, J. (2010). A task scheduling algorithm based on load balancing in cloud computing. In *Web Information Systems and Mining: International Conference, WISM 2010, Sanya, China, October 23-24, 2010. Proceedings* (pp. 271-277). Springer Berlin Heidelberg.

[19] Voß, S., & Fink, A. (2012). Hybridizing reactive tabu search with simulated annealing. In *Learning and Intelligent Optimization: 6th International Conference, LION 6, Paris, France, January 16-20, 2012, Revised Selected Papers* (pp. 509-512). Springer Berlin Heidelberg.

[20] Miao, Y. (2014). Resource scheduling simulation design of firefly algorithm based on chaos optimization in cloud computing. *International Journal of Grid and Distributed Computing, 7*(6), 221-228.

[21] Gu, B., & Pan, F. (2013). Modified gravitational search algorithm with particle memory ability and its application. *International Journal of Innovative Computing, Information and Control, 9*(11), 4531-4544.

[22] Roy, P. K. (2013). Solution of unit commitment problem using gravitational search algorithm. *International Journal of Electrical Power & Energy Systems, 53*, 85-94.

[23] Durillo, J. J., Prodan, R., Camarasu-Pop, S., Glattard, T., & Suter, F. (2014). Bi-objective workflow scheduling in production clouds: Early simulation results and outlook. Retrieved from: https://e-archivo.uc3m.es/handle/10016/21872

[24] Mezmaz, M., Melab, N., Kessaci, Y., Lee, Y. C., Talbi, E. G., Zomaya, A. Y., & Tuyttens, D. (2011). A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. *Journal of Parallel and Distributed Computing, 71*(11), 1497-1508.

[25] Su, S., Li, J., Huang, Q., Huang, X., Shuang, K., & Wang, J. (2013). Cost-efficient task scheduling for executing large programs in the cloud. *Parallel Computing, 39*(4-5), 177-188.

[26] Yi, S., Wang, Z., Ma, S., Che, Z., Liang, F., & Huang, Y. (2010, June). Combinational backfilling for parallel job scheduling. In *2010 2nd International Conference on Education Technology and Computer* (Vol. 2, pp. V2-112). IEEE.

[27] Bansal, N., Awasthi, A., & Bansal, S. (2016). Task Scheduling Algorithms with Multiple Factor in Cloud Computing Environment. *Information Systems Design and Intelligent Applications, 619*.

[28] Poola, D., Garg, S. K., Buyya, R., Yang, Y., & Ramamohanarao, K. (2014, May). Robust scheduling of scientific workflows with deadline and budget constraints in clouds. In *2014 IEEE 28th international conference on advanced information networking and applications* (pp. 858-865). IEEE.

[29] Arabnejad, H., & Barbosa, J. G. (2015, October). Multi-workflow QoS-constrained scheduling for utility computing. In *2015 IEEE 18th International Conference on Computational Science and Engineering* (pp. 137-144). IEEE.

[30] Rekha, S., & Kalaiselvi, C. (2019). Review of Scheduling Methodologies of Virtual Machines (VMs) In Heterogeneous Cloud Computing. *International Journal of Scientific & Technology Research, 8*(09).

[31] Sotiriadis, S., Bessis, N., & Buyya, R. (2018). Self managed virtual machine scheduling in cloud systems. *Information Sciences, 433*, 381-400.

[32] Junaid, M., Sohail, A., Ahmed, A., Baz, A., Khan, I. A., & Alhakami, H. (2020). A hybrid model for load balancing in cloud using file type formatting. *IEEE Access, 8*, 118135-118155.

[33] Tiwari, P. K., Rani, G., Jain, T., Mundra, A., & Gupta, R. K. (2019). Load Balancing in Cloud Computing. *Critical Approaches to Information Retrieval Research, 294*.

[34] Ghobaei-Arani, M., Rahmanian, A. A., Aslanpour, M. S., & Dashti, S. E. (2018). CSA-WSC: cuckoo search algorithm for web service composition in cloud environments. *Soft Computing, 22*(24), 8353-8378.

[35] Safari, M., & Khorsand, R. (2018). Energy-aware scheduling algorithm for time-constrained workflow tasks in DVFS-enabled cloud environment. *Simulation Modelling Practice and Theory, 87*, 311-326.

[36] Hamdani, M., Aklouf, Y., & Chaalal, H. (2020, June). A Comparative Study on Load Balancing Algorithms in Cloud Environment. In *Proceedings of the 10th International Conference on Information Systems and Technologies* (pp. 1-4).

[37] Ghobaei-Arani, M., Khorsand, R., & Ramezanpour, M. (2019). An autonomous resource provisioning framework for massively multiplayer online games in cloud environment. *Journal of Network and Computer Applications, 142*, 76-97.

[38] Ghobaei-Arani, M., Souri, A., Baker, T., & Hussien, A. (2019). ControCity: an autonomous approach for controlling elasticity using buffer Management in Cloud Computing Environment. *IEEE Access, 7*, 106912-106924.

[39] Ghobaei-Arani, M., Souri, A., Safara, F., & Norouzi, M. (2020). An efficient task scheduling approach using moth-flame optimization algorithm for cyber-physical system applications in fog computing. *Transactions on Emerging Telecommunications Technologies, 31*(2), e3770.

[40] Rafieyan, E., Khorsand, R., & Ramezanpour, M. (2020). An adaptive scheduling approach based on integrated best-worst and VIKOR for cloud computing. *Computers & Industrial Engineering, 140*, 106272.

[41] Khorsand, R., & Ramezanpour, M. (2020). An energy-efficient task-scheduling algorithm based on a multi-criteria decision-making method in cloud computing. *International Journal of Communication Systems, 33*(9), e4379.

[42] Safari, M., & Khorsand, R. (2018). PL-DVFS: combining Power-aware List-based scheduling algorithm with DVFS technique for real-time tasks in Cloud Computing. *The Journal of Supercomputing, 74*, 5578-5600.

[43] Khorsand, R., Ghobaei-Arani, M., & Ramezanpour, M. (2019). A self-learning fuzzy approach for proactive resource provisioning in cloud environment. *Software: Practice and Experience, 49*(11), 1618-1642.

[44] Strumberger, I., Tuba, E., Bacanin, N., Beko, M., & Tuba, M. (2020). Modified and hybridized monarch butterfly algorithms for multi-objective optimization. *Advances in intelligent systems and computing (923)*, pp. 449–458. Springer International Publishing.

[45] Adhikari, M., Nandy, S., & Amgoth, T. (2019). Meta heuristic-based task deployment mechanism for load balancing in IaaS cloud. *Journal of Network and Computer Applications, 128*, 64-77.

[46] Kaur, A., & Kaur, B. (2022). Load balancing optimization based on hybrid Heuristic-Metaheuristic techniques in cloud environment. *Journal of King Saud University-Computer and Information Sciences, 34*(3), 813-824.

[47] Strumberger, I., Bacanin, N., Tuba, M., & Tuba, E. (2019). Resource scheduling in cloud computing based on a hybridized whale optimization algorithm. *Applied Sciences, 9(*22), 4893.

[48] Torabi, S., & Safi-Esfahani, F. (2018). A dynamic task scheduling framework based on chicken swarm and improved raven roosting optimization methods in cloud computing. *The Journal of Supercomputing, 74*(6), 2581-2626.

[49] Attiya, I., Abd Elaziz, M., & Xiong, S. (2020). Job scheduling in cloud computing using a modified harris hawks optimization and simulated annealing algorithm. *Computational intelligence and neuroscience, 2020*.

[50] Li, C., Li, J., Chen, H., & Heidari, A. A. (2021). Memetic Harris Hawks Optimization: Developments and perspectives on project scheduling and QoS-aware web service composition. *Expert Systems with Applications, 171*, 114529.

[51] Patel, D., Gupta, R. K., & Pateriya, R. K. (2019). Energy-aware prediction-based load balancing approach with VM migration for the cloud environment. *Data, Engineering and Applications: Volume 2*, 59-74.

[52] Kumar, Y., & Singh, P. K. (2018). Improved cat swarm optimization algorithm for solving global optimization problems and its application to clustering. *Applied Intelligence, 48*, 2681-2697.

[53] Anwar, N., & Deng, H. (2018). A hybrid metaheuristic for multi-objective scientific workflow scheduling in a cloud environment. *Applied sciences, 8*(4), 538.

[54] Zhong, W., Zhuang, Y., Sun, J., & Gu, J. (2018). A load prediction model for cloud computing using PSO-based weighted wavelet support vector machine. *Applied Intelligence, 48*, 4072-4083.

[55] Ashouraei, M., Khezr, S. N., Benlamri, R., & Navimipour, N. J. (2018, August). A new SLA-aware load balancing method in the cloud using an improved parallel task scheduling algorithm. In *2018 IEEE 6th international conference on future internet of things and cloud (FiCloud)* (pp. 71-76). IEEE.

[56] Sharma, N., & Maurya, S. (2019, February). SLA-based agile VM management in cloud & datacenter. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)* (pp. 252-257). IEEE.

[57] Toutouh, J., & Alba, E. (2015). Metaheuristics for energy-efficient data routing in vehicular networks. *International Journal of Metaheuristics, 4*(1), 27-56.

[58] Mohanty, S., Patra, P. K., Ray, M., & Mohapatra, S. (2018). A Novel Meta-Heuristic Approach for Load Balancing in Cloud Computing. *International Journal of Knowledge-Based Organizations (IJKBO), 8*(1), 29-49.

[59] Hajimirzaei, B., & Navimipour, N. J. (2019). Intrusion detection for cloud computing using neural networks and artificial bee colony optimization algorithm. *ICT Express*, *5*(1), 56-59.

[60] Tuli, S., Gill, S. S., Garraghan, P., Buyya, R., Casale, G., & Jennings, N. (2021). START: Straggler prediction and mitigation for cloud computing environments using encoder lstm networks. *IEEE Transactions on Services Computing*.

[61] Mathew, M. (2018). Virtualization and Scheduling In Cloud Computing Environment – A Study. *IOSR Journals 20*(4), pp. 23–32.

[62] Varma, N. M. K., & Choi, E. (2016). Study and comparison of virtual machine scheduling algorithms in open source clouds. In *Advanced Multimedia and Ubiquitous Engineering: FutureTech & MUE* (pp. 349-355). Springer Singapore.

[63] Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., & Zagorodnov, D. (2009, July). Eucalyptus: an open-source cloud computing infrastructure. In *Journal of Physics: Conference Series* (Vol. 180, No. 1, p. 012051). IOP Publishing.

[64] Basthikodi, M., Faizabadi, A. R., & Ahmed, W. (2019). HPC Based Algorithmic Species Extraction Tool for Automatic Parallelization of Program Code. *International Journal of Recent Technology and Engineering*, *8*, 1004-1009.

[65] Basthikodi, M., & Ahmed, W. (2016, December). Classifying a program code for parallel computing against hpcc. In *2016 Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC)* (pp. 512-516). IEEE.

[66] Varma, N. M. K., Min, D., & Choi, E. (2011, November). Diagnosing CPU utilization in the Xen virtual machine environment. In *2011 6th International Conference on Computer Sciences and Convergence Information Technology (ICCIT)* (pp. 58-63). IEEE.

[67] Roschke, S., Cheng, F., & Meinel, C. (2009, December). Intrusion detection in the cloud. In *2009 eighth IEEE international conference on dependable, autonomic and secure computing (pp. 729-734)*. IEEE.

[68] Mazzariello, C., Bifulco, R., & Canonico, R. (2010, August). Integrating a network ids into an open source cloud computing environment. In *2010 sixth international conference on information assurance and security (pp. 265-270)*. IEEE.

[69] Garfinkel, T., & Rosenblum, M. (2003, February). A virtual machine introspection based architecture for intrusion detection. In Ndss (Vol. 3, No. 2003, pp. 191-206).Retrieved from : http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.11.8367&rep=rep1&type=pdf%5Cnhttp://www.isoc.org/isoc/conferences/ndss/03/proceedings/papers/13.pdf

[70] Ibrahim, A. S., Hamlyn-Harris, J., Grundy, J., & Almorsy, M. (2011, September). Cloudsec: a security monitoring appliance for virtual machines in the iaas cloud model. In *2011 5th International Conference on Network and System Security (pp. 113-120)*. IEEE.

[71] E. Summary. (2014). WHITE PAPER 2 Cybersecurity Problems Today 2 What Is an NGFW? 3 Best Practices for Selecting an NGFW. Next-Generation Firewalls: The New Norm in Defense. Retrieved from: https://webobjects.cdw.com/webobjects/media/pdf/Solutions/Security/148649-Next-Generation-Firewalls-The-New-Norm-In-Defense.pdf

[72] Naidu, V. R., Bhat, A. Z., & Singh, B. (2019). Cloud Concept for Implementing Multimedia Based Learning in Higher Education. In *Smart Technologies and Innovation for a Sustainable Future: Proceedings of the 1st American University in the Emirates International Research Conference—Dubai, UAE 2017 (pp. 81-84)*. Springer International Publishing.