ISSN No:-2456-2165

A Survey on Task Scheduling based on Various Meta-Heuristics and Machine Learning Algorithms in Cloud Computing

B. Suganya
Research Scholar
RVS College of Arts and Science
Coimbatore, Tamilnadu,
India.

Dr.R. Padmapriya
Associate professor & HoD-BCA,
School of Computer Studies
RVS College of Arts & Science
Coimbatore, Tamilnadu, India

Abstract:- The development of cloud computing in current decades has led to it serving as the basis for a variety of systems. It enables customers to access a list of specified resources, act immediately and adaptably to customer preferences, and only be charged for actual utilization. One of the most important problems in cloud computing is Task Scheduling (TS). The issue is how to equitably distribute and organize the user-provided tasks for Virtual Machine (VM) execution. Also, user experience is directly impacted by the effectiveness of scheduling efficiency. As a result, the TS issue in cloud computing has to be more precisely addressed. In cloud computing, the TS is essential such that the optimal scheduling of task requests may boost network efficiency. The main objective of TS is to assign tasks to appropriate processors to create the shortest deadline achievable without compromising on priority criteria. Numerous research has been conducted to design TS schemes based on various metaheuristic and machine learning algorithms that satisfy several criteria such as minimization of the makespan, execution cost and energy. They have demonstrated that conventional TS is effective only to satisfy certain criteria and have devised an optimum solution using multi-objectives in cloud computing. This paper presents a systematic and extensive analysis of TS algorithms in cloud computing depending on the different optimization and machine learning algorithms. Also, it addresses the challenges in those algorithms and recommends a few possible solutions for improving the utilization of cloud computing.

***Keywords**:- Cloud computing, Task scheduling, Virtual machine, Makespan, Metaheuristic, Machine learning, Optimization.*

I. INTRODUCTION

Internet-connected supercomputing is known as cloud computing. It is a sort of global technology that merely joins enormous computer groups utilizing a variety of techniques, including remote computing, virtualization, etc. Clients may transfer many data into the cloud systems and utilize a great processing ability with the aid of their local computer [1]. It provides clients with a range of storage, networking, and processing capabilities over the Internet. The cloud, as defined by R. Buyya, is “a concurrent and dispersed computing architecture that primarily consists of a group of interlinked and VMs that are provided flexibly and offered as 1 or as greater than 1 integrated processing facilities depending on Service-Level Agreements (SLAs) formed via negotiations between the service providers of clouds and customers” [2]. A large-scale dispersed processing architecture called cloud computing is abstract, virtualized and dynamically operated dependent on the monetary scale of the operator. The primary function of cloud computing is the management of computer resources, storage, multiple platforms and applications that are rented out to outside customers over the internet [3].

Cloud computing is a quickly developing model for processing that aims to alleviate cloud clients from the maintenance of hardware, software, networks and information resources, as well as, transfer such obligations to cloud service providers [4]. The essential features of cloud computing are distribution, virtualization and flexibility. Clouds offer a huge variety of resources, such as computing platforms, data centers, storage, networks, firewalls, and applications delivered as services. In addition, it offers strategies for controlling those services ensuring that cloud clients may utilize them without experiencing any performance-related issues. The 3 categories of cloud computing services are Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). Such categories are based on the degree of abstraction and the communication pattern of the providers [5].

A. Architecture of Cloud Computing

Different types of enterprises use cloud computing platforms to preserve information in the clouds therefore they may retrieve it anytime they need it. The 2 types of cloud infrastructure as shown in Fig. 1 are a front end and a back end, which are linked by the internet [6].

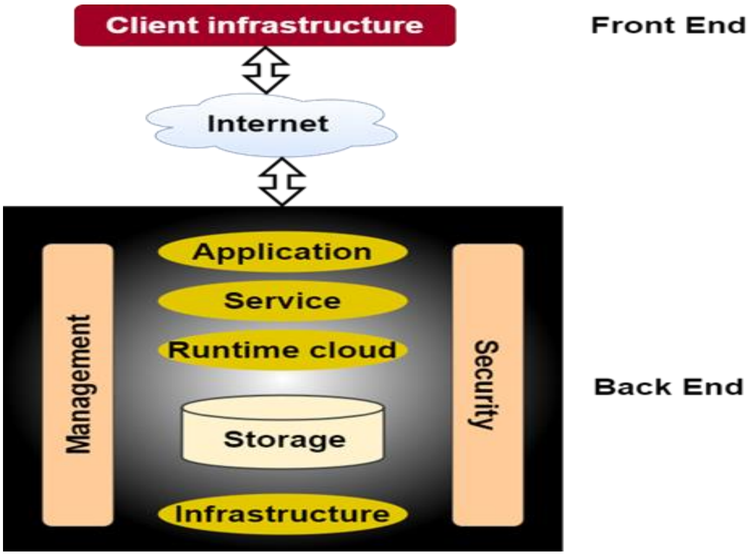

Fig. 1: Architecture of Cloud Computing

The back end is in the role of providing cloud applications with data protection. The back end is used by the network operators. It oversees the management of every resource required to provide operations. It includes a security system, a huge amount of information storage, hosts, VMs, traffic management systems, deployment models, etc.

Indeed, individuals interact with the front end. Programs and user interactions are required for front-end access to cloud computing. Computers, web browsers, and smartphones are included. The access methods for cloud storage are distinct from those for traditional storage because the cloud holds a large quantity of information from a wide range of individuals. The majority of operators implement several access methods. The following includes a few cloud computing architectural components:

- User infrastructure: It is regarded as a front-end component. It offers a Graphical User Interface (GUI) for communicating with the cloud.
- Internet: It serves as a channel for 2 ends to interact with one another.
- Application: The client might seek access to any application or network.
- Service: It offers IaaS, SaaS and PaaS.
- Runtime cloud: It provides the VMs with an operational and dynamic platform.
- Storage: It is one of the key components of cloud computing infrastructure. It provides a lot of storage capacity in the cloud for handling and storing information.
- Infrastructure: It provides functions at the application, host and network levels. It includes both hardware and software components.
- Management: It is employed to handle every component of the back end. As well, it creates cooperation among them.
- Security: It executes a privacy method in the back end.

B. Task Scheduling and Its Categories

The cloud comprises a variety of resources, which are distinct from one another in terms of various resources, and since the expense of executing jobs in the cloud with those resources is distinct, therefore TS in the cloud differs from conventional strategies of TS. As a result, TS in the cloud requires more emphasis since cloud operations rely on it. TS is crucial for increasing the adaptability and dependability of cloud-based applications. The primary aim of allocating jobs to resources in line with scheduling constraints is to determine the optimal schedule in which to perform multiple jobs such that to provide the client with the optimum outcome [7].

In cloud computing, various resources, including containers, firewalls, and networks, are often dynamically assigned by the order and specifications of the job and its subtasks. As a result, work scheduling in the cloud becomes a flexible issue because no previously established schedule can be helpful while executing a job. Since the workflow is unpredictable, processing methods are also unpredictable, and resources are also unpredictable when several workloads are using resources concurrently, the scheduling is unpredictable due to these factors.

TS in the cloud refers to selecting the optimal resources provided for workload completion or allocating system resources to workloads in a way that minimizes the workload execution period. In scheduling strategies, a collection of workloads is formed by assigning a weight to all jobs, with the significance of individual workload depending on a variety of factors. After that, workloads are selected based on their importance and given to the processing systems that can meet a predetermined target function [8].

Two major categories of TS are:

- Fixed scheduling: It schedules workloads in a well-known setting, i.e. it contains the data regarding the overall arrangement of workloads, resource allocation before processing and prediction of the workload processing period [9].

- Dynamic scheduling: It should rely not solely on the allocated workloads to the cloud system, yet also on the present conditions of systems to create scheduling choices [9].
- Direct scheduling: If new workloads exist, then they are allocated to VMs immediately [10].
- Batch scheduling: Workloads are clustered into a batch before transmission. It is also known as mapping services [11].
- Preemptive scheduling: All workloads are disturbed while processing and may be shifted to the other resource to finish processing [12].
- Non-preemptive scheduling: VMs are not rescheduled to new workloads until completing the processing of the allocated workloads [13].

The cloud computing TS process has 3 stages [14] as illustrated in Fig. 2:

- Initial stage: It comprises a collection of workloads (cloudlets), which are transmitted by the cloud clients for processing.
- Second stage: It translates workloads to appropriate resources to obtain the maximum resource usage and a less makespan.
- Third stage: It comprises a collection of VMs that are utilized to process the workloads.

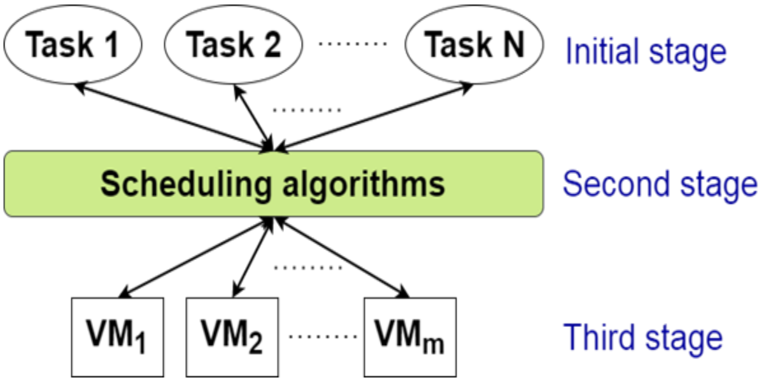

Fig. 2: Overview of Task Scheduling in Cloud Computing

C. Necessity of Task Scheduling in Cloud Computing

The primary goal of scheduling is to respond to arriving requests from end clients by identifying the optimum cloud resources that must increase both the system usage rates and essential quality metrics. Different efficiency measures for cloud computing exist, including makespan, financial cost, processing cost, reaction time, power usage, dependability, etc. To meet the needs of end clients and service providers while maintaining the SLA, an effective TS strategy should be employed to assess and enhance such factors. Owing to challenges including resource distribution, dynamism, and heterogeneity, traditional scheduling mechanisms are unable to tackle these issues [15-16]. Considering the primary goal of solving the possible issues of overloading and underloading in cloud TS, a scheduling algorithm is therefore required for fair and appropriate allocation of diverse tasks among VMs depending on the availability of resources.

The benefits of TS approaches [16] are (i) controlling the Quality-of-Service (QoS) efficiency of cloud computing, (ii) controlling the processor and storage, (iii) increasing resource usage when reducing the overall workload processing period, (iv) ensuring fairness for every workload, (v) enhancing the number of workloads that are properly finished, (vi) allocating workloads on real-time applications, (vii) obtaining a maximum network throughput and (viii) enhancing load distribution.

D. Classification of Meta-Heuristic Task Scheduling Algorithms in Cloud Computing

The purpose of TS differs from every system to others, under the QoS criteria. As a result, several studies were established that focus on TS using meta-heuristics algorithms. A new, robust classification is presented in Fig. 3 to help individuals comprehend metaheuristic TS algorithms in cloud computing more extensively and effectively [16-17]. These algorithms may be grouped into 4 distinct categories depending on the specific kind of scheduling issue, the major goal of scheduling, the task-resource mapping strategy and the scheduling restriction. Depending on the relationship between the entering workloads, these algorithms are further divided into dependent and independent workloads. Depending on the kind of scheduling algorithms (schedulers), they are classified as classical/heuristics or meta-heuristics.

- Type of scheduling issue: It is necessary to develop an optimized strategy that satisfies the goals by selecting the most optimum result because there is often a balance among optimization goals. It is feasible to evaluate the optimality of a specific strategy in contrast to another one that already exists in a single objective optimization. Whilst this cannot be done effectively in Multi-objective Optimization Problems (MOPs), it may be done indirectly [18].
- Major goal of scheduling: Whenever a task scheduling procedure is carried out, a minimum single goal value is required for obtaining higher performance. The most often used goals may be stated as follows: throughput,

makespan, economic cost, processing expense (i.e., usage of CPU, memory and so on), dependability and accessibility, flexibility or scalability, privacy and power usage [19].

- Task-resource mapping method: To effectively exploit the allocated resources depending on the state of the cloud system and the given tasks, static, dynamic, Artificial Intelligence (AI)-based and prediction-based translation of cloud resources to arriving workloads is conducted. Resources and tasks are well-known for having ambiguous features and also being possible to be altered. Thus, to address QoS demands and reduce SLA breaches, these strategies are created and implemented [20].
- Scheduling restriction: Due to the potential impact on the SLA when a huge variety of services are failing to satisfy deadline, priority, budget, and fault tolerance requirements, such variables are important in the sector of cloud scheduling [19].

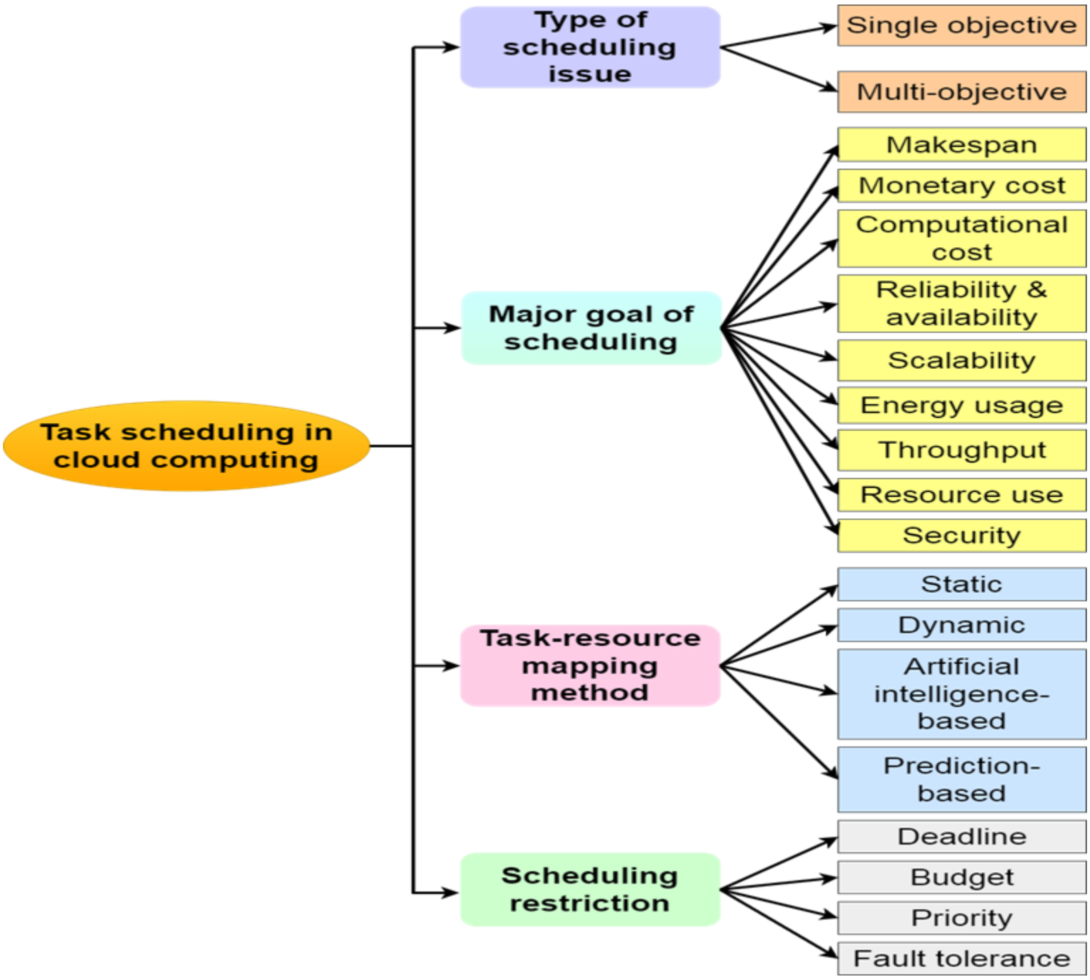

Fig. 3: Taxonomy of Task Scheduling Algorithms in Cloud Computing

Various algorithms have been developed over the past decades for TS in cloud applications. The primary purpose of this paper is to give a comprehensive overview of TS algorithms in cloud computing using various optimization and machine learning techniques. Also, a comparative analysis is presented to highlight the benefits and drawbacks of those algorithms in a tabular form, which supports us to suggest possible future directions.

The following sections have been prepared as follows: Section II studies and analyzes the TS based on various optimization algorithms, whereas Section III studies and analyzes the TS based on machine learning algorithms in cloud systems. Section IV summarizes the complete study and offers suggestions for future enhancement.

II. LITERATURE SURVEY

A. Survey on Task Scheduling Based on Optimization Algorithms in Cloud Computing

An Improved Particle Swarm Optimization (IPSO) algorithm [21] are developed to achieve the best distribution for a huge amount of tasks. This was performed by partitioning the allocated tasks into batches dynamically. Also, the resource usage condition was taken in the generation of all batches. Once obtaining a sub-optimal result for all batches, every sub-optimal result for batches was added to the absolute distribution map. Moreover, the loads over the absolute distribution map were balanced by the IPSO.

An Immune-based PSO (IMPSO) algorithm [22] were presented to allocate workflow in the cloud paradigm. The aim was to reduce the execution cost and makespan under user-defined deadline restraints. Li & Han [23] presented a flexible TS based on the hybrid discrete Artificial Bee Colony (ABC) algorithm. Initially, the TS issue was formulated as a Hybrid Flowshop Scheduling (HFS) issue. In multi-objective HFS, reduction of the maximum end period, maximum system workload and overall workloads of each system were measured concurrently. Many kinds of perturbation patterns were considered to improve hunting capabilities. Also, an enhanced adaptive perturbation pattern was included to balance the exploitation and exploration capability. Moreover, a deep-exploitation operator was applied to enhance the exploitation capabilities for effective TS.

To designed a multi-objective TS optimization depending on the fuzzy defense algorithm [24]. The main aim was to choose the shortest period, the degree of resource load balance and the cost of multi-objective task execution by creating a mathematical framework, which provides the objective factor and determines the impact of multi-objective TS. Those objective values were resolved by the fuzzy self-defense scheme to get the global best result of the objective factor.

An Improved Whale Optimization (IWC) algorithm [25] are developed to enhance the TS performance in cloud computing. Initially, a cloud computing TS and allocation system with a period, cost and VMs was built. Then, a viable strategy for all whale individuals related to the cloud computing TS was applied to obtain the optimal whale individual using the inertial weight mechanism, which enhances the local hunting capability and avoids early convergence. Also, add and delete functions were used to monitor individuals after all iterations, which were ended and modified to choose individuals with greater efficiency.

A metaheuristic model termed MDVMA [26] were developed for dynamic VM distribution with optimized TS in a cloud computing paradigm. In this model, a multi-objective scheduling scheme was adopted by the Non-dominated Sorting Genetic Algorithm (NSGA)-II algorithm to optimize TS, which reduces energy utilization, makespan and cost concurrently to achieve tradeoff to the cloud service providers according to their demands.

A multi-objective restricted optimization issue [27] were analyzed to recognize the best scheduling strategies for systematic tasks to be employed in unreliable cloud scenarios. The main aim was to reduce the estimated task execution period and monetary expense under probabilistic restraints on deadline and budget. This issue was resolved by the combined Monte Carlo method and Genetic Algorithm (MCGA), the cloud clients were permitted to select the strategy of the Pareto optimum group ensuring their demands and interests. An alternated TS method [28] were designed for IoT requests in a cloud-fog paradigm depending on the modified Artificial Ecosystem-based Optimization (AEO) by the operators of the Salp Swarm Algorithm (SSA) to improve the exploitation capability of AEO in the procedure of discovering the best decision for the issue under concern.

A 3-level scheduling framework depending on the whale-Gaussian cloud called GCWOAS2 [29], which were client task level, TS level and data center level to define the whole procedure of TS. Initially, an opposition-based training scheme was adopted to initialize the scheduling plans and find the best scheduling strategy. After that, a dynamic fine-tuning factor was applied to adaptively fine-tune the search region. To improve the arbitrariness of exploration, a whale optimization algorithm was designed depending on the Gaussian cloud scheme. Further, a multi-objective TS scheme using the Gaussian whale-cloud optimization was introduced to find the global best scheduling plan.

Introducing a framework to estimate the present condition of the active tasks based on the outcomes of the QoS forecast allocated by an Auto-regressive Integrated Moving Average (ARIMA) [30] framework optimized by the Kalman filter. After that, a scheduling strategy was determined by the joint PSO and Gravitational Search Algorithm (GSA) based on the QoS conditions to ensure the client’s QoS via allocating the workflow.

A TS technique [31] were presented to jointly reduce energy cost and mean task loss probability of clouds. In this technique, the issue was modeled and solved by an adaptive bi-objective differential growth depending on simulated annealing to compute a real-time and near-optimum group of results. Moreover, an absolute knee result was selected based on the minimum Manhattan distance to characterize appropriate servers in clouds and task distribution amid online sites.

TS method [32] were depending on the Advanced Phasmatodea Population Evolution (APPE) algorithm in a heterogeneous cloud setting. This algorithm was used to minimize the time needed to obtain solutions by enhancing the convergent progress of the closest best solutions. Also, a restart mechanism was included to avoid the algorithm from entering the local optimization and balance its search and exploitation abilities. Moreover, the valuation function was applied to discover the optimal solutions according to the makespan, resource cost and load balancing level.

A semi-adaptive real-time TS scheme named the Improved Genetic Algorithm is designed for Permutation-based Optimization Problems (IGA-POP) [33] for bag-of-tasks in the cloud-fog paradigm. In this scheme, the TS issue was modeled as a POP. First, the IGA was applied to find various permutations for arrived tasks at all scheduling cycles. After that, the tasks were allocated to the VM based on the optimal permutation to accomplish a better tradeoff between the makespan and the overall performance cost when satisfying deadline restraints.

B. Survey on Task Scheduling Based on Machine Learning Algorithms in Cloud Computing

A smart QoS-aware TS model based on Deep Reinforcement Learning (DRL) [34] was developed for applications in clouds. It can learn to create suitable online task-to-VM solutions for constant task requests immediately from its experiences with no previous data. Based on this process, the tasks were scheduled by the service providers to constrained resources under QoS demand limits.

fat-tree structure-based method named Large-scale Tasks processing using Deep Reinforcement (LTDR) [35] training to find the best TS policy. This was achieved by using a virtual network mapping scheme depending on a deep Convolutional Neural Network (CNN) and Q-learning algorithms. Also, a policy network was applied to create node mapping decisions and the link mapping method was performed using the distributed value factor. Then, tasks were scheduled to the appropriate physical nodes and processed effectively.

A novel scheduling model called Spear [36] is developed to reduce the makespan of complicated tasks when considering task dependencies and heterogeneous resource needs. In this model, a Monte Carlo Tree Search (MCTS) was applied in the TS phase and the DRL was trained to direct the processes in the MCTS. Using this DRL, exploration ability was enhanced by concentrating favorable branches of the search tree.

2-phase TS and resource distribution model [37] were designed, which utilizes many smart schedulers to resolve the cooperative scheduling issue between TS and resource distribution. A Heterogeneous Distributed Deep Learning (HDDL) framework was applied in the TS phase to allocate various tasks to several cloud data centers. Also, a Deep Q-Network (DQN) framework was applied as a resource scheduler to arrange VMs for tasks to physical servers for implementation.

TS scheme depending on the DRL model [38] such as DQN to adaptively schedule tasks with precedence connection to cloud servers to reduce the task implementation period. To achieve this, the aspects of servers and tasks were considered as state inputs and server numbers were considered as activities. To reduce the execution period, the negative change value of makespan from a particular state to the other state was described as the incentive. Also, the task precedence connection restraint was accomplished during the state shift phase. The issue of TS of cloud-based systems and intended to reduce the computational cost under resource and deadline restraints [39]. To solve this issue, a clipped double deep Q-learning method was introduced based on the target network and experience relay schemes, which allocates the tasks to their corresponding VMs.

A novel framework depending on the multi-agent system called DRL for resource distribution and TS [40] to minimize the cost and power in cloud computing. In this framework, a Quantile Regression DQN (QR-DQN) scheme was developed to create a suitable strategy and the best long-term solutions to assign resources and schedule tasks to corresponding VMs.

TS scheme in a cloud paradigm depending on the multi-criteria decision-making approach [41]. In this scheme, the TS was modeled as a non-linear restricted optimization dilemma and solved by the Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS), which incorporates an Entropy Weight Method (EWM) to reduce makespan, cost and power usage, as well as, improve the consistency.

Self-adapting TS scheme called ADATSA [42] based on the learning automata for container cloud. Initially, a learning automata scheme and the objective factor were designed for the system on the TS issue. After that, an efficient incentive-penalty strategy was performed to schedule tasks combined with the idle condition of resources and the operating condition of tasks in the present atmosphere. Additionally, the atmosphere was designed by cluster, node and task, as well as the chance of task chosen, was optimized by scheduling implementation to improve the adaptability to the cloud scenario of the allocation. Moreover, a model of task load monitoring with a buffer queue was created to perform dynamic scheduling depending on priority.

A multi-objective TS scheme depending on the Decision Tree (DT) [43] in a heterogeneous cloud scenario. A novel TS-DT algorithm was adopted to assign and implement the applications' tasks. The major aim was to resolve the multi-objective TS challenge by reducing makespan, ensuring load balancing amid VMs and increasing resource usage.

An improved training-enabled TS model depending on the task Criticality and Collapse-Aware Scheduling (CCAS) scheme [44]. In this scheme, 2 distinct strategies were designed such as the TS strategy depending on task CCAS and an ensemble forecast strategy such as Gradient Boosting DT (GBDT) to proactively estimate the system usage and task implementation status by capturing the high-level attributes via training the task variables. Also, a smart scheduling scheme was adopted for best resource distribution.

An independent TS method in cloud computing based on the utilization of the Multi-Objective Artificial Bee Colony with Q-learning (MOABCQ) algorithm [45]. This algorithm was used to compute the order of tasks for appropriate resources and schedule the most suitable tasks according to the execution time, cost and usage of resources. Also, it was integrated with the First Come First Serve (FCFS) and the Largest Job First (LJF) heuristic TS schemes to achieve load balancing among VMs.

III. COMPARATIVE ASSESSMENT

This part compares the merits and demerits of the different metaheuristic-based TS algorithms for cloud applications in Table 1.

Table 1: Assessment of Various Metaheuristic-based TS Algorithms in Cloud Systems

Ref. No.	Algorithm	Merits	Demerits	Outcomes

[21]	IPSO	It can avoid imbalanced task scenarios because of the rebalancing procedure after obtaining the final scheduling maps.	It did not consider the cost and energy use in the objective functions.	**No. of workloads=3000:**
				Makespan=540sec;
				Standard variance=15;
				Degree of imbalance=0.1

[22]	IMPSO	It could achieve the optimum result at a rapid convergence time.	It needs to group the workloads before scheduling them to cloud resources due to the dependency among workloads and tradeoffs among groups.	Genome tasks:
				Cost=7.2$/hr;
				Makespan=40000sec;
				Cybershake tasks:
				Cost=1.5$/hr;
				Makespan=700sec;
				Sipht tasks:
				Cost=1.14$/hr;
				Makespan=2500sec

[23]	Hybrid discrete ABC	It improves the convergence ability to find the best solution during TS.	The local exploration capability was not efficient. Also, the tradeoff between exploitation and exploration was not effective.	Mean makespan=23sec; Computational time=1.25sec
[24]	Fuzzy self-defense	The deadline violation rate was reduced so that the resource usage rate on the VM was comparatively satisfactory.	It did not consider the makespan and energy usage as the objective function.	**No. of workloads=180:** Highest execution period=790sec; Deadline rate=0.02%; **VM number=80:** VM resource usage=0.98%

[25]	IWC	It can reduce the cost and time for TS.	The impact on the memory load value was not clear, which needs to enhance the memory usage.	**No. of workloads=800:**
				Economic cost=0.75$;
				Time utilization=1.1msec;
				Memory load=0.41;

[26]	MDVMA using NSGA-II	It was helpful to achieve optimal TS with less energy use, makespan and cost.	It restricts the convergence speed and a few results of the Pareto front were not obtained, which may be best compared to the best results.	Total makespan=7338sec;
				Total energy utilization=31.93kWh;
				Total cost=7338$

[27]	MCGA	It can enable the clients to select the strategy of the Pareto optimum group guaranteeing their demands and interests.	The optimum results were not achieved under tight deadlines and costs, because the variability raises.	**No. of workloads:24:** Runtime=146sec; Processing period=3789sec
[28]	Modified AEO using SSA	It can achieve better mean makespan and throughput under both synthetic and real tasks.	It needs to consider additional objective values like energy usage and economic costs.	**No. of workloads:800:** Mean makespan=35.1sec; Mean throughput period=3000sec
[29]	GCWOAS2	It may decrease the workload execution period and balance the load of VMs.	It did not function efficiently in terms of operating costs.	**No. of workloads=100:** **No. of iterations=100:** Overall cost=0.263$; Time cost=0.15sec; Load cost=0.331;
[30]	Combined PSO and GSA	It can decrease the SLA violation rate efficiently while increasing the amount of workload.	It did not consider the periodicity of client workload information and the client's QoS factors. Also, it needs more objective functions to increase the efficiency of TS.	**No. of workloads=800:** SLA violation rate=0.03%; Workload completion cost=850$;
[31]	Simulated	It can minimize the energy cost and	The convergence speed and	Mean energy

	annealing	mean error probability of workloads.	diversity of acquired results were not improved.	cost=1.335×10^4\$; Convergence speed=18.09sec
[32]	APPE	It has a rapid convergence period and better makespan.	It was solely appropriate to solve the fixed TS challenges and not distribute resources based on the arriving period of workloads.	**No. of workloads=500:** Performance index evaluation function value=450

[33]	IGA-POP	It can achieve a good tradeoff between the makespan and the overall execution cost.	It considers only the static cloud computing platform so the resource usage was not enhanced. Also, it needs advanced meta-heuristics and machine learning algorithms to solve the dynamic TS issue.	**No. of workloads=400:**
				Elapsed period=615sec;
				Makespan=113.7sec;
				Execution cost=2122.4G$

Table 1 addresses that many researchers focused on optimized TS in cloud applications using various metaheuristic algorithms like PSO, GA, NSGA-II, ABC, etc. Each algorithm has disadvantages regarding exploration, exploitation and convergence abilities. To combat these issues, more advanced and machine learning algorithms must be developed to achieve TS in dynamic cloud environments. From this viewpoint, a few researchers design machine learning algorithms with and without meta-heuristics to accomplish dynamic TS in cloud systems, which are studied in Section III. Here, the merits and demerits of those machine learning-based TS algorithms are listed in Table 2.

Table 2: Assessment of Various Machine Learning-based TS Algorithms in Cloud Systems

Ref. No.	Algorithm	Merits	Demerits	Outcomes
[34]	QoS-aware TS using DRL	It can effectively decrease the mean task response period and ensure the QoS at a high level of client experience.	It needs to extend to a more sophisticated cloud platform using multiple objective values.	Success rate=98.3%; Response period=158ms;
[35]	LTDR using CNN and Q-learning	It can distribute the workloads on the appropriate physical nodes.	It needs to improve the policy model by raising the number of neural levels.	**Time=2500sec:** Throughput=3.7requests/sec; Long-term revenue/cost=0.44$

[36]	MCTS and DRL	It can decrease the makespan efficiently by enhancing the exploration ability.	It considers only a single objective, whereas more objective functions were needed to enhance the efficiency of TS.	Makespan=820.1sec;
				Runtime=500sec

[37]	HDDL and DQN	It has better scalability and computation time in real-time cloud TS.	It needs to achieve near-global optimization via enhancing the cooperative capability of many training frameworks.	Energy usage=10.48kWh; Latency rate=0.37%; Task delay=33.21sec

[38]	DQN	It has a less mean execution period as increasing the number of workloads.	It needs to consider multiple objectives such as cost, deadline of workloads, etc., to enhance TS efficiency.	**No. of workloads=100:**
				Mean makespan=45.3467sec;
				Standard variance=1.7932;
				Mean CPU period=0.0267sec

[39]	Clipped double deep Q-learning	It can achieve a better balance between exploration and exploitation.	It did not reduce the load on the cloud data center and it should select proper ϵ value for achieving effective TS.	**No. of workloads=9:** Execution period=130sec
[40]	QR-DQN	It efficiently reduces both energy and time cost.	Increasing the number of workloads discards more workloads because of exceeding intervals or resources.	**No. of workloads=1000:** Normalized energy cost=0.03; Time cost=0.1sec
[41]	TOPSIS with	It has less cost and energy	It did not consider the client’s	**Mean communication to**

	EWM	usage within significant restraints.	interest-based factors like cost and deadline restraints for TS.	**computation ratio=10:** Mean cost=1.5$; Mean energy usage=$4\times10^9$kWh

[42]	ADATSA using learning automata	It achieves good environment adaptability, resource optimization efficacy and QoS efficiency.	It did not consider the heterogeneity of cloud resources. Also, the environment system trained from the constant incentive-penalty variables was not ideal.	**Time=20min:**
				Resource imbalance degree=0.13;
				Resource residual degree=0.65;
				Response delay=440msec;
				Throughput=150req/sec

[43]	TS-DT	It can decrease the mean makespan and enhance the mean resource usage.	The energy usage was high.	**No. of VMs=40:** Mean makespan=153.65msec; Resource usage=99.297%; Deviation of load balance=0.37

[44]	CCAS and GBDT	It enhances resource usage, flexibility and dynamism.	It did not guarantee error-free workloads processing.	**No. of workloads=45:**
				Execution period=630msec;
				Latency=5msec;
				Mean system usage=90%

[45]	MOABCQ	It has less time complexity and makespan. Also, it has a good resource usage rate.	It did not ensure that this algorithm was ideal and the efficiency of the network was not optimized in each test database.	**Synthetic task database:** **No. of workloads=800:** Makespan=24sec; Mean throughput=35tasks/s; Cost=150G$; Mean resource usage rate=0.801%; Degree of imbalance=0.117

Table 2 states that some researchers have concentrated on TS based on machine learning algorithms, i.e. DQN, DRL, etc., with a few meta-heuristic algorithms. Even though those algorithms outperform single-objective optimization algorithms, TS in multi-cloud, fog-cloud, or edge-cloud platforms is problematic. It is vital to apply the other sophisticated machine learning algorithms to enhance TS in different kinds of cloud environments. Also, it must be tested in a real-time scenario to analyze the efficiency of TS algorithms.

IV. CONCLUSION

This study presents a broad review of various TS algorithms in cloud computing based on a variety of meta-heuristics and machine learning algorithms. According to the findings of this study, many academics have been experienced in designing TS algorithms that schedule the best workloads to the proper VMs in the cloud paradigm. Amongst, the MOABCQ algorithm can reduce the makespan and enhance resource usage rate with less computational time complexity. Conversely, its performance cannot be guaranteed to be perfect. Also, not all test databases can facilitate optimizing network efficiency. So, advanced machine learning and meta-heuristics algorithms can be incorporated to achieve optimized TS and analyze the effectiveness of those algorithms in both static and dynamic cloud computing applications in the future.

REFERENCES

[1.] L. Bohu, Z. Lin and C. Xudong, "Introduction to cloud manufacturing," Zte Communications, vol. *8*, no. 4, pp. 6-9, 2020.

[2.] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg and I. Brandic, "Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility," Future Generation Computer Systems, vol. 25, no. 6, pp. 599-616, 2009.

[3.] S. K. Sahana, "Emerging computing platforms for solving complex engineering problems," In Methods, Implementation, and Application of Cyber Security Intelligence and Analytics, IGI Global, pp. 165-180, 2022.

[4.] M. Taghipour, M. E. Soofi, M. Mahboobi and J. Abdi, "Application of cloud computing in system management in order to control the process," Management, vol. *3*, no. 3, pp. 34-55, 2020.

[5.] G. Bhatta and M. Pandey, "A case study on hybrid cloud approach to automate the cloud services based on decision support system," Review of International Geographical Education Online, vol. 11, no. 8, pp. 1669-1683, 2021.

[6.] Odun-Ayo, M. Ananya, F. Agono and R. Goddy-Worlu, "Cloud computing architecture: a critical analysis," In 18th IEEE International Conference on Computational Science and Applications, pp. 1-7, 2018.

[7.] M. Ibrahim, “Task scheduling algorithms in cloud computing: a review,” Turkish Journal of Computer and Mathematics Education, vol. 12, no. 4, pp. 1041-1053, 2021.

[8.] F. Ebadifard and S. M. Babamir, “Autonomic task scheduling algorithm for dynamic workloads through a load balancing technique for the cloud-computing environment,” Cluster Computing, vol. 24, no. 2, 1075-1101, 2021.

[9.] T. McSweeney, N. Walton and M. Zounon, “An efficient new static scheduling heuristic for accelerated architectures,” In International Conference on Computational Science, Springer, Cham, pp. 3-16, 2020.

[10.] J. Yao and N. Ansari, “Fog resource provisioning in reliability-aware IoT networks,” IEEE Internet of Things Journal, vol. 6, no. 5, pp. 8262-8269, 2019.

[11.] Z. Zhao, S. Liu, M. Zhou, D. You and X. Guo, “Heuristic scheduling of batch production processes based on petri nets and iterated greedy algorithms,” IEEE Transactions on Automation Science and Engineering, vol. 19, no. 1, pp. 251-261, 2020.

[12.] W. Chen, X. Zhou and J. Rao, “Preemptive and low latency datacenter scheduling via lightweight containers. IEEE Transactions on Parallel and Distributed Systems, vol. 31, no. 12, pp. 2749-2762, 2019.

[13.] N. Panwar, S. Negi, M. M. S. Rauthan and K. S. Vaisla, “TOPSIS–PSO inspired non-preemptive tasks scheduling algorithm in cloud environment,” Cluster Computing, vol. 22, no. 4, pp. 1379-1396, 2019.

[14.] T. Aladwani, “Types of task scheduling algorithms in cloud computing environment,” Scheduling Problems-New Applications and Trends, pp. 1-12, 2020.

[15.] M. Kumar, S. C. Sharma, A. Goel and S. P. Singh, “A comprehensive survey for scheduling techniques in cloud computing,” Journal of Network and Computer Applications, vol. 143, pp. 1-33, 2019.

[16.] E. H. Houssein, A. G. Gad, Y. M. Wazery and P. N. Suganthan, “Task scheduling in cloud computing based on meta-heuristics: review, taxonomy, open challenges, and future trends,” Swarm and Evolutionary Computation, vol. 62, pp. 1-41, 2021.

[17.] N. Kaur, A. Kumar and R. Kumar, “A systematic review on task scheduling in fog computing: taxonomy, tools, challenges, and future directions,” Concurrency and Computation: Practice and Experience, vol. 33, no. 21, pp. 1-25, 2021.

[18.] M. Abdullahi, M. A. Ngadi, S. I. Dishing and B. I. E. Ahmad, “An efficient symbiotic organisms search algorithm with chaotic optimization strategy for multi-objective task scheduling problems in cloud computing environment,” Journal of Network and Computer Applications, vol. 133, pp. 60-74, 2019.

[19.] R. Ghafari, F. H. Kabutarkhani and N. Mansouri, “Task scheduling algorithms for energy optimization in cloud environment: a comprehensive review,” Cluster Computing, pp. 1-59, 2022.

[20.] S. Subbaraj and R. Thiyagarajan, “Performance oriented task-resource mapping and scheduling in fog computing environment,” Cognitive Systems Research, vol. 70, pp. 40-50, 2021.

[21.] H. Saleh, H. Nashaat, W. Saber and H. M. Harb, “IPSO task scheduling algorithm for large scale data in cloud computing environment,” IEEE Access, vol. 7, pp. 5412-5420, 2018.

[22.] P. Wang, Y. Lei, P. R. Agbedanu and Z. Zhang, “Makespan-driven workflow scheduling in clouds using immune-based PSO algorithm,” IEEE Access, vol. 8, pp. 29281-29290, 2020.

[23.] J. Q. Li and Y. Q. Han, “A hybrid multi-objective artificial bee colony algorithm for flexible task scheduling problems in cloud computing system,” Cluster Computing, vol. 23, no. 4, pp. 2483-2499, 2020.

[24.] X. Guo, “Multi-objective task scheduling optimization in cloud computing based on fuzzy self-defense algorithm,” Alexandria Engineering Journal, vol. 60, no. 6, pp. 5603-5609, 2021.

[25.] L. Jia, K. Li and X. Shi, “Cloud computing task scheduling model based on improved whale optimization algorithm,” Wireless Communications and Mobile Computing, pp. 1-13, 2021.

[26.] D. Alsadie, “A metaheuristic framework for dynamic virtual machine allocation with optimized task scheduling in cloud data centers,” IEEE Access, vol. 9, pp. 74218-74233, 2021.

[27.] M. C. Calzarossa, M. L. Della Vedova, L. Massari, G. Nebbione and D. Tessera, “Multi-objective optimization of deadline and budget-aware workflow scheduling in uncertain clouds,” IEEE Access, vol. 9, pp. 89891-89905, 2021.

[28.] M. Abd Elaziz, L. Abualigah and I. Attiya, “Advanced optimization technique for scheduling IoT tasks in cloud-fog computing environments,” Future Generation Computer Systems, vol. 124, pp. 142-154, 2021.

[29.] L. Ni, X. Sun, X. Li and J. Zhang, “GCWOAS2: multiobjective task scheduling strategy based on Gaussian cloud-whale optimization in cloud computing,” Computational Intelligence and Neuroscience, pp. 1-17, 2021.

[30.] M. A. Rakrouki and N. Alharbe, “QoS-aware algorithm based on task flow scheduling in cloud computing environment,” Sensors, vol. 22, no. 7, pp. 1-20, 2022.

[31.] H. Yuan, J. Bi and M. Zhou, “Energy-efficient and QoS-optimized adaptive task scheduling and management in clouds,” IEEE Transactions on Automation Science and Engineering, vol. 19, no. 2, pp. 1233-1244, 2022.

[32.] N. Zhang, S. C. Chu, P. C. Song, H. Wang and J. S. Pan, “Task scheduling in cloud computing environment using advanced phasmatodea population evolution algorithms,” Electronics, vol. 11, no. 9, pp. 1-16, 2022.

[33.] S. Abohamama, A. El-Ghamry and E. Hamouda, “Real-time task scheduling algorithm for IoT-based applications in the cloud–fog environment,” Journal of Network and Systems Management, vol. 30, no. 4, pp. 1-35, 2022.

[34.] Y. Wei, L. Pan, S. Liu, L. Wu and X. Meng, "DRL-scheduling: An intelligent QoS-aware job scheduling framework for applications in clouds," IEEE Access, vol. 6, pp. 55112-55125, 2018.

[35.] Wu, G. Xu, Y. Ding and J. Zhao, "Explore deep neural network and reinforcement learning to large-scale tasks processing in big data," International Journal of Pattern Recognition and Artificial Intelligence, vol. 33, no. 13, pp. 1-29, 2019.

[36.] Z. Hu, J. Tu and B. Li, "Spear: Optimized dependency-aware task scheduling with deep reinforcement learning," In IEEE 39th International Conference on Distributed Computing Systems, pp. 2037-2046, 2019.

[37.] J. Lin, D. Cui, Z. Peng, Q. Li and J. He, "A two-stage framework for the multi-user multi-data center job scheduling and resource allocation," IEEE Access, vol. *8*, pp. 197863-197874, 2020.

[38.] T. Dong, F. Xue, C. Xiao and J. Li, "Task scheduling based on deep reinforcement learning in a cloud manufacturing environment," Concurrency and Computation: Practice and Experience, vol.*32,* no. 11, pp. 1-12, 2020.

[39.] S. Swarup, E. M. Shakshuki and A. Yasar, "Task scheduling in cloud using deep reinforcement learning," Procedia Computer Science, vol. 184, pp. 42-51, 2021.

[40.] T. Oudaa, H. Gharsellaoui and S. B. Ahmed, "An agent-based model for resource provisioning and task scheduling in cloud computing using DRL," Procedia Computer Science, vol. 192, pp. 3795-3804, 2021.

[41.] M. S. Kumar, A. Tomar and P. K. Jana, "Multi-objective workflow scheduling scheme: a multi-criteria decision making approach," Journal of Ambient Intelligence and Humanized Computing, vol. 12, no. 12, 10789-10808, 2021.

[42.] L. Zhu, K. Huang, Y. Hu and X. Tai, "A self-adapting task scheduling algorithm for container cloud using learning automata," IEEE Access, vol. 9, pp. 81236-81252, 2021.

[43.] H. Mahmoud, M. Thabet, M. H. Khafagy and F. A. Omara, Multiobjective task scheduling in cloud environment using decision tree algorithm," IEEE Access, vol. 10, pp. 36140-36151, 2022.

[44.] N. Khan, N. Iqbal, A. Rizwan, S. Malik, R. Ahmad and D. H. Kim, "A criticality-aware dynamic task scheduling mechanism for efficient resource load balancing in constrained smart manufacturing environment," IEEE Access, vol. 10, pp. 50933-50946, 2022.

[45.] Kruekaew and W. Kimpan, "Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning," IEEE Access, vol. 10, pp. 17803-17818, 2022.