CGC International Journal of Contemporary Technology and Research
ISSN: 2582-0486 (online) Vol.-4, Issue-1 DOI: 10.46860/cgcijctr.2021.12.31.226

The Future of Scheduling in Cloud Computing: Metaheuristic Approaches

Komal*, Gaurav Goel
CEC Landran
Email: *komalsheoran3@gmail.com

Abstract: **Cloud computing has grown in importance and popularity as a platform for delivering services on-demand. It is available on basis of pay-per-use. The fundamental objective of a cloud service provider is to make effective use of resources by lowering execution time, cost, and other parameters while boosting profit. As a result, in cloud computing, adopting effective scheduling algorithms is still a significant challenge. There is no comprehensive research of cloud computing scheduling mechanisms that cover both deterministic and optimal techniques. This article explains the most often used scheduling algorithms in cloud systems, the factors that they focus on, and the simulation settings used to assess performance. It also includes brief descriptions of numerous metaheuristic approaches and the review structure, sources utilized, inclusion and exclusion criteria, and extracted data information. Finally, this study discusses the typical limitations discovered during the evaluation.**

Keywords: **Cloud Computing, Scheduling, Metaheuristic, Deterministic Schedulers, Optimized schedulers.**

I. INTRODUCTION

The effect of the internet and its services grows by the day in our technologically advanced society. Cloud computing is now a technology that leverages the internet to connect the globe by delivering a variety of services. Cloud computing provides operating systems, storage space, software, memory, and other services at a low cost [1]. Customers and service providers alike profit from the cloud. The primary goal of service providers is to maximize profit while utilizing resources efficiently [2]. The essential components of Cloud computing are Consumers, Resource Providers, and Task Scheduling. To ensure QoS, cloud providers make every effort to provide their clients with high-quality resources. Tasks are dynamically assigned to resources based on the scheduling algorithm [3]. With the aid of Virtual Machines (VMs), the Scheduler will take care of the sort of services to be delivered as well as other criteria such as resource availability, user requirements, and so on. The cloud has many resources, and each resource contains virtual machines (VMs) that enable you to perform several jobs at once [4].

In various aspects, such as server, memory, and storage virtualization [2], virtualization plays a critical role in allocating resources to their respective users. Virtual machines are created with efficiency in mind. Virtual machines (VMs) are not physical things. When the Scheduler receives a request from a user to access resources for the fulfillment of their activities in a Cloud environment, machines are assigned based on appropriateness and necessity. This dynamic resource allocation improves resource usage while simultaneously balancing the system's demand. As a result, an optimum resource usage policy is required, which will aid in the management of all resources and the equitable distribution of load.

Task scheduling is also a significant concern in cloud computing, and this problem is classified as an NP-complete problem [5]. To solve the NP-complete issue, several scholars presented various meta-heuristic techniques. These heuristic approaches also include optimization strategies, which are further split into two categories: (a) Bio-inspiring; (b) Nature-inspiring. The metaheuristic techniques, currently optimized schedulers, performance metrics utilized for study, and significant limitations and future metaheuristic approaches are described in the following parts of this article.

II. METAHEURISTIC APPROACHES

Almost all engineering issues may be expressed as optimization problems. Different approaches have been investigated in mathematical programming, operations research, and other fields to tackle optimization issues. On the other hand, conventional approaches are frequently ineffective when the issue area is big and complicated. Evolutionary computation issues are common in artificial intelligence [6]. We may never find a polynomial-time solution to these issues since they are NP-hard. Different "heuristics" have been employed to "search for the sub-optimal solution" to solve these issues effectively. Heuristics are search methods based on human intuition and creativity, and they are frequently used in local search to identify good answers in a limited region quickly. Metaheuristics are higher-level heuristics that regulate the whole search process, allowing for the systematic and efficient discovery of global optimum solutions [7]. Though metaheuristics cannot always ensure the real global optimal solution, they can produce excellent outcomes in a wide range of situations. Metaheuristics can major boon that adds significant capacity to a computer system without raising the hardware cost. The following are some characteristics of metaheuristic approaches:

- Metaheuristics are search-process-guiding techniques.
- The objective is to identify near-optimal solutions by effectively exploring the search space.
- These algorithms are made up of various techniques ranging from simple local search operations to sophisticated learning processes.
- These algorithms are non-deterministic and approximate.

- Metaheuristics aren't limited to a single problem.

Metaheuristics methods come in a wide range of types and have several classified characteristics listed in the table below.

Table 1 Classification of Metaheuristic Approaches

Category	*Description*	*Algorithms*
Local and Global search	Finding the best solution for a localized region of the search space or the global optima for problems with no local optima is known as local optimization. On issues with local optima, global optimization entails finding the best solution [8].	Hill Climbing Algorithm, Simulated Annealing, Genetic Algorithm, etc.
Single Solution and Population-based	Single solution techniques are concerned with changing and enhancing a single candidate solution, whereas population-based approaches aim to maintain and improve numerous candidate solutions [9].	Simulate Annealing, Variable Neighborhood Solution, Genetic Algorithm, Particle Swarm Optimization, etc.
Hybridization and Memetic Algorithms	A hybrid metaheuristic uses a metaheuristic in conjunction with other optimization techniques. Both halves of a hybrid metaheuristic can run simultaneously and share information to help steer the search. On the other hand, memetic algorithms are a combination of evolutionary or population-based approaches with distinct individual learning or local improvement methods for issue solving [10].	Machine Learning, Mathematical Programming, Local Search Algorithm, etc.
Parallel metaheuristics	A parallel metaheuristic employs parallel programming techniques to perform several metaheuristic searches in parallel; these might range from basic distributed systems to concurrent search runs that work together to enhance the overall solution [11].	Evolutionary Algorithms, Particle Swarm Optimization, Ant Colony Optimization, etc.
Nature-inspired and metaphor-based metaheuristics	The creation of nature-inspired metaheuristics is a particularly active field of research. Natural systems have inspired several contemporary metaheuristics, particularly evolutionary computation-based methods. Nature provides concepts, methods, and principles for developing artificial computing systems to solve complicated computational problems [12].	Simulate Annealing, Ant Colony Optimization, Evolutionary Algorithms, etc.
Ancient-inspired metaheuristics	Ancient-inspired algorithms serve as a source of inspiration for developing new methods [13] [14]. There were many constraints in the ancient world, yet different man-made structures show that constraints and a shortage of resources led to optimization. A deeper examination of these historical remains reveals that antiquity's tactics, strategies, and technology were considerably more advanced and optimized than we may expect. The elements of ancient-inspired philosophy are observed and reflected upon, and techniques of administering the project at the time are sought to be understood.	Different Combinational Approaches, etc.

Researchers have previously utilized several metaheuristic techniques to enhance the performance of schedulers in cloud computing. The table above lists all of the metaheuristic approach classes, along with examples of algorithms that fall into each category. The following section goes through the specifics of the optimal scheduling methods.

III. CLOUD SCHEDULERS

Scheduling is a procedure in cloud computing that involves mapping a set of workloads to a group of virtual machines (VMs) or allocating VMs to run on available resources to fulfil the demands of users. [15]. In a cloud context, the goal of utilizing scheduling algorithms for enhancing system load balance and throughput, maximize resource usage, conserve energy, save expenses, and decrease overall processing time. To achieve effective matching between workloads and resources, the scheduler should take into account virtualized resources and users' necessary limitations [16]. One or more methods should underpin each scheduling strategy. Time, cost, energy, quality of service, and fault tolerance are the most often utilized techniques or objectives [17].

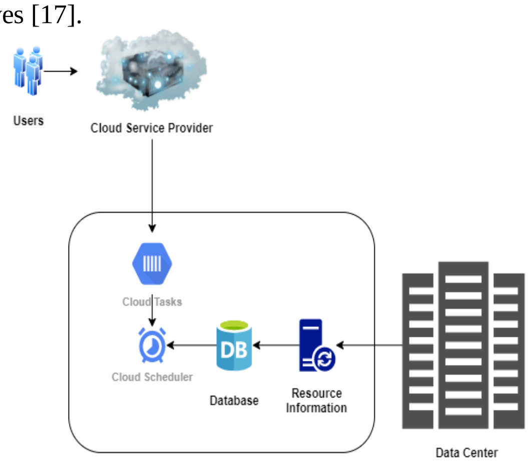

Figure 1 Task Scheduling Architecture of Cloud

ISSN: 2582-0486 (online) Vol.-4, Issue-1 DOI: 10.46860/cgcijctr.2021.12.31.226

There are two layers of resource allocation: VM-level and Host-level [18]. The goal of task scheduling is to properly map tasks to suitable virtual machine. Depending on how reliant a task is, it might be classified as independent or dependent [19]. Independent tasks have no dependencies on other activities and do not need adherence to a priority order throughout the scheduling process. Dependent tasks, on the other hand, have a priority order that must be followed during the scheduling process based on task dependencies. Workflow Scheduling is the process of scheduling tasks that are reliant on one another. Work dependence is a significant factor in determining the best task scheduling approach in a cloud environment. The fundamental goal of task scheduling is to reduce the time it takes to complete each job. If the jobs are interdependent, reducing the computation cost (the time it takes for the execution of every node) and also the cost of communication (the time it takes for data to be sent between two nodes) might help reduce the makespan. Autonomous tasks, on the other hand, can be planned without regard for order [19]. The practice of arranging virtual machines (VMs) to run on the appropriate physical machines (PMs) to assure job completion is known as VM scheduling. This will enhance resource utilization while also ensuring that all systems' load balance is maintained. The quality of service (QoS) and Service Level Agreements (SLA) agreed upon by cloud service providers and users are dependent on VM scheduling.

A. Scheduling Algorithms

When cloud computing was first introduced, the same scheduling methods used for grid and cluster scheduling were used to cloud computing. For example, First Come First Served (FCFS) scheduling [20], in which services are served in the order in which they are received in the system, and Round Robin (RR) scheduling [20], in which a set time quantum determines the duration for which tasks will be completed in one go, The Min-Min algorithm [21] assigns the task with the shortest completion time to the resource with the shortest execution time, the Max-Min algorithm[22] assigns the task with the longest completion time to the resource with the shortest execution time and the Resource Aware Scheduling Algorithm (RASA) [23] that combines the Min-Min and Max-Min algorithms. While these algorithms worked admirably, they failed to account for cloud characteristics and a variety of other QoS factors. In order to attain QoS, the scheduling algorithms are further classified into two types: (a) Deterministic Schedulers and (b) Optimized Schedulers.

Deterministic Schedulers: Deterministic schedulers use a linear method and integer programming to create their schedules. This category includes a variety of techniques, each with its own set of benefits and downsides. This category includes several schedulers, which are detailed below.

- *ABC Scheduler:* The Activity-Based Cost (ABC) method is used by this scheduler to calculate the cost of an item as well as the performance of an activity.Cao et al. [24] presented the ABC algorithm, which is used to schedule tasks based on priority and cost. Furthermore, Ingole et al. [25] used the same scheduler for Cloud Tasks and recommended that the method be improved based on specific additional parameters. Furthermore, Bhaskaran et al. [26] presented an enhanced version of this ABC scheduler. The scheduling algorithm conducts task prioritization and task grouping, and resources are distributed to the cloud tasks accordingly.
- *Priority Scheduler:* As specific tasks should be serviced earlier than others, and some jobs cannot stay in a system for an extended period, job priority is an important consideration in scheduling. PJSC was proposed by Ghanbari and Othman [27] Based on a hierarchical analytical method (AHP). AHP created a three-level priority for the PJSC algorithm using a model based on MCDM (multi-criteria decision-making) and MCDM (multi-attribute decision-making). The results of this method were satisfactory; however, the issue of complexity arose, which Amalakar and Oulika [28] address further. In addition, Kaur and Singh [29] presented a completion rate-based prioritization algorithm. The first part of this method involves calculating task priority and sorting, while the second phase involves calculating the job's anticipated completion time. The scheduler then handles the job with a minimum completion time. The results of this scheduler show the QoS parameters that were attained. Furthermore, the priority was determined using six sigma control charts based on dynamic threshold values, as described by Bala and Chana [30]. The experiments show that this proposed technique performs well in terms of execution time and makespan.
- *Credit-based Scheduler*: Thomas et al. [31] examined the problem of resource availability and user satisfaction and presented a credit-based scheduler. Task length and user priority are the two parameters used by this scheduler to produce credits. In comparison to the priority scheduler, simulation findings demonstrated that the proposed credit-based scheduler is more effective. Sharma and Bala [32] presented a KNN-based Credit Scheduler that uses deadline parameters to improve the performance of the existing scheduler and achieves more optimality.
- *Cost-based Scheduler*: The scheduler makes use of computation cost principles. Selvarani et al. [33] developed a cost-based scheduler based on an enhanced ABC algorithm to assess resource cost and computation performance. To increase performance, this algorithm also conducted task grouping. They compare the performance of different numbers of cloudlets in terms of processing cost and time. Mosleh et al. [34] examined the other issue of deadlines and presented the ACTS scheduler, an adaptive cost-based task scheduling technique. The minimum cost path is assigned to low priority activities to fulfill the time deadline, while the quick access path is assigned to high priority tasks. The suggested technique's efficacy was assessed using simulation data. CPU Time, Turnaround Time (Sharma and Tyagi [35], better ranking based (Amoon et al. [36], Processing power, and network parameters (Mansouri and Javidi [37] are some of the other cost-based schedulers developed recently to improve the performance of cloud schedulers.

Numerous more deterministic schedulers focus on various factors such as QoS [38], Task duplication [39], multiple goals [40], and many more. However, because some of these strategies fail to meet the goals due to scheduling problems'

NP-Hard nature, a current trend is to use optimization methodologies to deal with it. The specifics regarding several efficient schedulers will be provided in the following sections. *Optimized Schedulers:* These schedulers used metaheuristic techniques to optimize the scheduling process and boost cloud server performance. Researchers in cloud computing have already developed and applied a number of meta-heuristic techniques to build the optimal scheduler. Some of them are covered farther down.

- *GA Scheduler*: Genetic algorithm (GA) is a metaheuristic method for genetics and evolution that performs different operations. This method was used to solve several scheduling issues and provided the best answer. The GA algorithm is utilized in cloud computing in a variety of ways by different researchers. Ge and Wei [41] presented a novel scheduler and built an improved version based on the GA method. The completion time is the emphasis of GA's objective function. The researchers took into account complexity, and Kaur and Kinger [42] developed an upgraded method of GA Scheduler. They also enhance the fitness function, which calculates the mean and grand mean for resources with various factors. They computed specific parameters as well, but only for a small number of assignments. Furthermore, Beegom and Rajasree [43] constructed the GA scheduler with two factors: cost and makespan. Hamad and Omara [44] offered an enhanced version of the GA scheduler based on TS-GA, Wen and Wen [45] proposed improved mutation, Liu and Wang [46] proposed improved fitness, crossover, and mutation parameters (OSIG), and so on, to obtain higher performance.

- *PSO Scheduler*: Particle Swarm Optimization (PSO) is a metaheuristic technique for solving scheduling problems utilized by various platforms. Using Load Balancing Mutation, Awad et al. suggested an improved version of the PSO scheduler. This method rescheduled incorrect tasks, which implies it reallocated resources to such tasks when the standard PSO scheduler failed to manage them. This improves execution speed, round trip time, dependability, and failures, among other things. Verma and Kaushal presented another expanded form of PSO in order to reduce costs. Xu et al. added a judgment component to PSO to assess the algorithm's fairness and to enhance parameters such as execution time, allocated bandwidth, and usage. Furthermore, Khalili and Babamir offered a makespan enhancement in PSO, Kumar introduced a cost element, Huang et al. proposed time changing intra weight with PSO, Yu proposed dynamic PSO, and so on.

:

- Hybrid Scheduler: To achieve various objectives, optimized schedulers were also suggested using a mix of more than one optimization technique. The following are some of the most common hybrid schedulers: GA and Fuzzy, Tabu Search and PSO, Cuckoo Search (CS) and PSO, GA and ACO, Elephant Herd Optimization (EHO) and GA, PSO and Fuzzy, CS and Modified PSO, PSO and grey wolf optimization (GWO), GA and PSO and LJFP-PSO and MCT-PSO which were proposed to achieve various objectives to enhance the performance of cloud scheduler in terms of utilization, allocation, load balancing, fairness, cost, and many other factors.

The following table contains information on the methodology, algorithm, settings, and simulation which shows that the majority of schedulers are concerned with time, deadlines, priorities, cost, budget, and other factors. In contrast, just a few are concerned with the resources' capacity to do the work. Although metaheuristic method-based schedulers with hybrid techniques perform better, the resource capacity aspect can help to enhance performance. Another element observed is that the assessment of these techniques was done with a limited number of resources, VMs, and Tasks and that it may be improved in the future.

IV. REVIEW METHODOLOGY

This section presents the state-of-the-art review layout, a step-by-step method for the literature discussed in the previous sections. This research focuses on categorizing the current literature on cloud computing scheduling, optimization approaches and assessing the current trends.

A. *Development of Review Protocol*

This evaluation finds relevant research articles from reputable electronic databases and the top conferences in the field. After then, inclusion and exclusion criteria were used to reduce the number of papers that were considered. Following that, final research studies were chosen based on a variety of variables. The information given here is the product of a thorough investigation.

Table 2: Existing state w

Ref No.	Approach	Algorithm	Method	Simulation	Parameters			Performance Metrics
					No. of Tasks	No. of Resources	No. of VMs	
[24]	Deterministic	Enhanced ABC	Task Grouping and Priority	Conceptual	-	-	-	-
[25]	Deterministic	ABC	Priority and Cost	SimGrid	-	-	-	-
[26]	Deterministic	Improved ABC	Task Dependency and Priority	Conceptual	-	-	-	-
[27]	Deterministic	PJSC	Three level Priority	Numerical	4	3	-	Makespan
[29]	Deterministic	Priority	Priority and Completion Rate	CloudSim	20-400	-	-	Time
[30]	Deterministic	Multilevel Priority	six sigma control charts based on dynamic threshold	CloudSim	20-100	-	25	Execution Time (ET) and Makespan
[31]	Deterministic	Credit Based	Task length and user priority	CloudSim	5-50	4	8	Makespan
[32]	Deterministic	KNN based Credit Scheduling	Task Length, User Priority, Deadline, and Cost	CloudSim	100-500	10	80	Makespan
[33]	Deterministic	Enhanced ABC with cost	Task grouping and Cost	CloudSim	25-100	6	-	Processing Time (PT) and Cost
[34]	Deterministic	ACTS	Completion Time, Cost, and Deadline	CloudSim	10-50	-	10	Computation Cost (CC), ET, Bandwidth (BW), and CPU Utilization (CU)
[35]	Deterministic	Improved ABC with Cost	CPU Time, Turnaround Time	CloudSim	100-5000	1	3	ET and Cost
[36]	Deterministic	ICTS	Ranking	DAG	80-400	-	100	Makespan
[37]	Deterministic	CJS	Processing power, and network parameters	CloudSim	100-500	-	30-90	Makespan, Processor Utilization (PU), BW, and Average Waiting Time (AWT)
[38]	Deterministic	QoS Driven	Priority and Completion Time	CloudSim	200-2400	-	50-100	Makespan, and Average Latency (AL)
[39]	Deterministic	DILS	Completion Time	DAG	2-10	-	-	Makespan
[40]	Deterministic	Multi Objective	Cost and Time	MATLAB	512 and 1024	16 and 32	-	Makespan, Cost and Utilization
[41]	Optimized	Improved GA	Completion Time	Numerical	6	3	-	Completion Time (CT)
[42]	Optimized	Enhanced GA	mean and grand mean values	Numerical	5	-	-	ET
[43]	Optimized	Bi-Objective GA	Time and Cost	Numerical	99	-	-	Makespan

[44]	Optimized	GA	Completion Time	CloudSim	25-100	-	8	Time, Cost, and Utilization
[45]	Optimized	CGA	Non-Uniform Mutation	CloudSim and MATLAB	40-200	8	-	CT
[46]	Optimized	OSIG	Improve all GA Operators	CloudSim	800-1600	4	20	ET
[47]	Optimized	Enhanced PSO	Load Balancing Mutation	-	1000	-	50	Makespan, ET, and Cost
[48]	Optimized	Cost Minimized PSO	Deadline and Budget	CloudSim	-	6	-	Normalized Scheduled Cost (NSC)
[49]	Optimized	Improved PSO	Dynamic Adjustment	CloudSim	12	-	4	ET and Cost
[50]	Optimized	Improved PSO	Linear Descending Inertia Weight	CloudSim	20-200	2	-	Makespan
[51]	Optimized	PSO	Cost	MATLAB and CloudSim	40	6	20	ET
[52]	Optimized	Optimal PSO	Logarithm decreasing strategy	MATLAB	100-300	-	20	Makespan
[53]	Optimized	IPSO	Global Optimal	CloudSim	7		10	CT, BW, and Energy Consumption (EC)
[54]	Optimized	GA and Fuzzy	Job Length and Resource Capability	CloudSim	100-1000	-	50	Makespan, Degree of Imbalance (DI)
[55]	Optimized	Tabu Search and PSO	Time	CloudSim	200-1000	-	-	Schedule Length and Execution Rate
[56]	Optimized	CS and PSO	Local and Global Search	CloudSim	10-40	-	5	ET and Utilization
[57]	Optimized	GA and ACO	Weighted Sum of Makespan and Flowtime	-	10-100	-	20-100	Cost
[58]	Optimized	EHO and GA	-	CloudSim	500-10000	-	1500	Makespan and Cost
[59]	Optimized	PSO and Fuzzy	length of tasks, speed of CPU, size of RAM, and total execution time	-	100-700	-	40	Makespan and DI
[60]	Optimized	CS and MPSO	Deadline and Budget	CloudSim	100-500	-	50	Makespan and Cost
[61]	Optimized	PSO and GWO	Time and Cost	WorkflowSim	25-1000	-	-	ET
[62]	Optimized	GA and PSO	Global Optimal	CloudSim	1000-5000	-	-	Makespan, Response Time (RT), and Cost
[63]	Optimized	LJFP-PSO	Job Length and Resource Capability	MATLAB	200-1000	-	40-200	Makespan, ET, DI, and EC
		MCT-PSO	Completion Time					

CGC International Journal of Contemporary Technology and Research
ISSN: 2582-0486 (online) Vol.-4, Issue-1 DOI: 10.46860/cgcijctr.2021.12.31.226

B. Sources of Information

For this review study, various electronic database sources were investigated; some of the popular electronic databases used in this search are listed below.

Fig 2. Popular Electronic Databases

C. *Inclusion and Exclusion Criteria*

From 1990 through 2021, this literature review includes both quantitative and qualitative research investigations. As this subject is investigated for a range of scheduling techniques, the keywords "Cloud Schedulers," "Metaheuristic approaches," "Deterministic Scheduling Algorithms in Cloud Computing," and "Optimized Scheduling Algorithms in . Cloud Computing" pointed to a significant number of results. The search was done using the search term "Scheduling Approaches [in, for] [Cloud Computing]" in the abstract and title. Some research studies, for example, have included the substrings "in Cloud Computing" or "for Cloud Computing" in their titles. As a result, a search has been conducted that considers this in order to include all of the research works in this study. There are research studies from various conferences, journals, and book chapters included in this collection. Only the relevant articles have been chosen from the retrieved results. The relevance of the methodologies, publications, and conferences are the selection factors for this work.

D. Extraction Outcomes

Using the inclusion criterion, which mainly depends on the techniques, the relevant work of Scheduling algorithms is retrieved from the enormous collection of data given by search engines. Figures 3(a) and 3(b) show the interpretation of the selected kind of article and year of work. The following fig shows that journals account for most of the work in this study (51%), with conferences accounting for 40% of the work and book chapters accounting for 9%. In addition, the following graph depicts a year-by-year study of work relevant to cloud computing scheduling

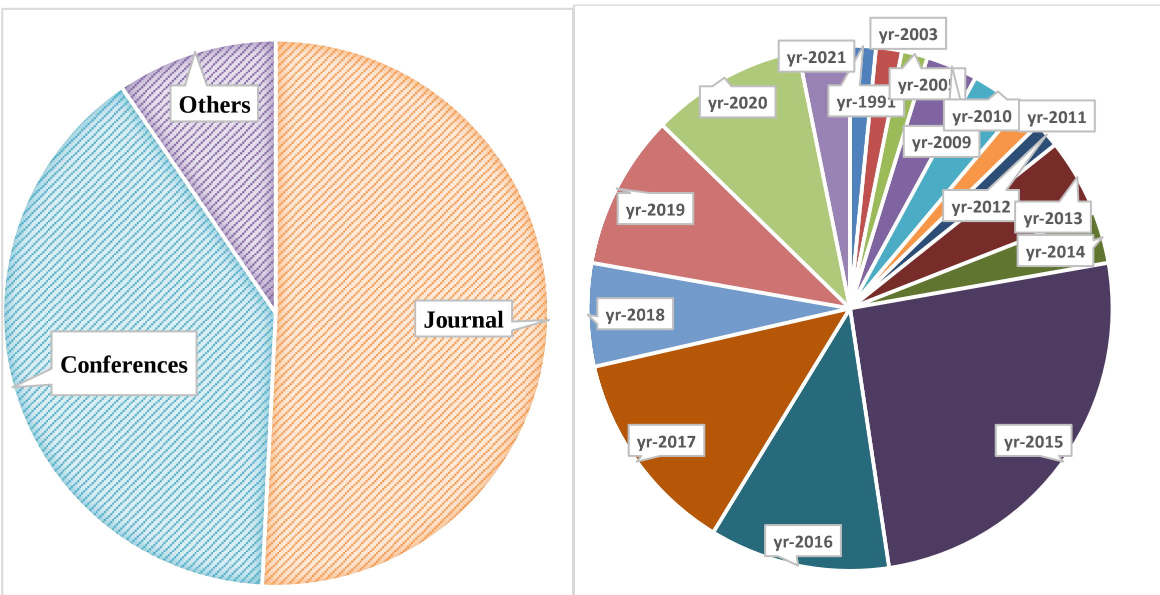

Fig 3. (a) Included research works- Type of Paper, (b) Year-wise included research

V. CONCLUSION AND FUTURE SCOPE

The cloud has become a popular technology because of its various services for both consumers and businesses. This raises the strain on the cloud servers, necessitating the management of their services, which mainly entails scheduling. Researchers have devised many techniques for efficiently allocating resources to projects to obtain the optimum scheduling. This paper discusses a few of these techniques. The approaches are divided into two groups in this paper: Deterministic schedulers and Optimized schedulers are two types of schedulers. Deterministic methods are the conventional approaches, and they don't always work. Optimized schedulers, on the other hand, deal with them more effectively. Many efficient schedulers have been proposed in the past, and they are all covered in this work. These schedulers employ single as well as hybrid techniques to improve performance. Still, there are certain drawbacks to

ISSN: 2582-0486 (online) Vol.-4, Issue-1 DOI: 10.46860/cgcijctr.2021.12.31.226

these schedulers that were discovered throughout this research and can be targeted for future directions in the field of cloud computing. These are:

- Using the resource capacity factor can help with task allocation.
- The Dynamic Objective feature can help you improve the performance.
- The real-time element can be taken into account while configuring resources.
- Hybrid approaches can be opted
- Testing can be done on both heterogeneous and homogeneous

The elements listed above are some of the suggestions that researchers may use in their future work to improve cloud performance.

REFERENCES

[1] A. Keivani, F. Ghayoor, and J.-R. Tapamo, "A review of recent methods of task scheduling in cloud computing," in *2018 19th IEEE Mediterranean Electrotechnical Conference (MELECON)*, May 2018, pp. 104–109, doi: 10.1109/MELCON.2018.8379076.

[2] M. Vishalatchi, N. Krishnamoorthy, and S. Sangeetha, "Optimised scheduling in cloud computing," in *2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET)*, Feb. 2017, pp. 1–6, doi: 10.1109/ICAMMAET.2017.8186712.

[3] M. R. Belgaum, S. Soomro, Z. Alansari, M. Alam, S. Musa, and M. M. Su'ud, "Load balancing with preemptive and non-preemptive task scheduling in cloud computing," in *2017 IEEE 3rd International Conference on Engineering Technologies and Social Sciences (ICETSS)*, Aug. 2017, pp. 1–5, doi: 10.1109/ICETSS.2017.8324145.

[4] Y. Vijay and B. V. Ghita, "Evaluating cloud computing scheduling algorithms under different environment and scenarios," in *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Jul. 2017, pp. 1–5, doi: 10.1109/ICCCNT.2017.8204070.

[5] A. Gupta and R. Garg, "Load Balancing Based Task Scheduling with ACO in Cloud Computing," in *2017 International Conference on Computer and Applications (ICCA)*, Sep. 2017, pp. 174–179, doi: 10.1109/COMAPP.2017.8079781.

[6] R. Balamurugan, A. M. Natarajan, and K. Premalatha, "Stellar-Mass Black Hole Optimization for Biclustering Microarray Gene Expression Data," *Appl. Artif. Intell.*, vol. 29, no. 4, pp. 353–381, Apr. 2015, doi: 10.1080/08839514.2015.1016391.

[7] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization," *ACM Comput. Surv.*, vol. 35, no. 3, pp. 268–308, Sep. 2003, doi: 10.1145/937503.937505.

[8] G. M. Guisewite and P. M. Pardalos, "Global search algorithms for minimum concave-cost network flow problems," *J. Glob. Optim.*, vol. 1, no. 4, pp. 309–330, 1991, doi: 10.1007/BF00130828.

[9] M. El Yafrani and B. Ahiod, "Population-based vs. Single-solution Heuristics for the Travelling Thief Problem," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, Jul. 2016, pp. 317–324, doi: 10.1145/2908812.2908847.

[10] J. E. Amaya, C. Cotta Porras, and A. J. Fernández Leiva, "Memetic and Hybrid Evolutionary Algorithms," in *Springer Handbook of Computational Intelligence*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 1047–1060.

[11] E. Alba, *Parallel Metaheuristics*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2005.

[12] K. Sörensen, "Metaheuristics-the metaphor exposed," *Int. Trans. Oper. Res.*, vol. 22, no. 1, pp. 3–18, Jan. 2015, doi: 10.1111/itor.12001.

[13] S. Harifi, "Ancient-inspired : a novel source of inspiration for development of metaheuristic algorithms," no. July, 2020, doi: 10.13140/RG.2.2.19760.38408.

[14] S. Harifi, J. Mohammadzadeh, M. Khalilian, and S. Ebrahimnejad, "Giza Pyramids Construction: an ancient-inspired metaheuristic algorithm for optimization," *Evol. Intell.*, Jul. 2020, doi: 10.1007/s12065-020-00451-3.

[15] A. Razaque, N. R. Vennapusa, N. Soni, G. S. Janapati, and khilesh R. Vangala, "Task scheduling in Cloud computing," in *2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, Apr. 2016, pp. 1–5, doi: 10.1109/LISAT.2016.7494149.

[16] L. Liu and Zhe Qiu, "A survey on virtual machine scheduling in cloud computing," in *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, Oct. 2016, pp. 2717–2721, doi: 10.1109/CompComm.2016.7925192.

[17] D. C. Poola, "Robust and fault tolerant scheduling for scientific workflows in clouds," University of Melbourne, Australia, 2015.

[18] E. Pacini, C. Mateos, and C. G. Garino, "Multi-objective Swarm Intelligence schedulers for online scientific Clouds," *Computing*, vol. 98, no. 5, pp. 495–522, May 2016, doi: 10.1007/s00607-014-0412-y.

[19] R. Annette.J, A. Banu. W, and S. Shriram, "A Taxonomy and Survey of Scheduling Algorithms in Cloud: Based on task dependency," *Int. J. Comput. Appl.*, vol. 82, no. 15, pp. 20–26, Nov. 2013, doi: 10.5120/14240-2389.

[20] A. Kaur, "Different Task Scheduling Algorithms in Cloud Computing," *Int. J. Latest Trends Eng. Technol.*, vol. 9, no. 3, pp. 217–223, 2018, doi: 10.21172/1.93.37.

[21] Huankai Chen, F. Wang, N. Helian, and G. Akanmu, "User-priority guided Min-Min scheduling algorithm for load balancing in cloud computing," in *2013 National Conference on Parallel Computing Technologies (PARCOMPTECH)*, Feb. 2013, pp. 1–8, doi: 10.1109/ParCompTech.2013.6621389.

[22] B. Kanani and B. Maniyar, "Review on Max-Min Task scheduling Algorithm for Cloud Computing," *J. Emerg. Technol. Innov. Res.*, vol. 2, no. 3, pp. 781–784, 2015.

[23] S. ; Parsa and R. Entezari-Maleki, "RASA: A New Grid Task Scheduling Algorithm," *Int. J. Digit. Content Technol. its Appl.*, vol. 3, no. 4, pp. 91–99, 2009, doi: 10.4156/jdcta.vol3.issue4.10.

[24] Q. Cao, Z. B. Wei, and W. M. Gong, "An optimized algorithm for task scheduling based on activity based costing in cloud computing," *3rd Int. Conf. Bioinforma. Biomed. Eng. iCBBE 2009*, pp. 1–3, 2009, doi: 10.1109/ICBBE.2009.5162336.

[25] A. Ingole, S. Chavan, and U. Pawde, "An optimized algorithm for task scheduling based on activity based costing in cloud computing," *2nd Natl. Conf. Inf. Commun. Technol.*, pp. 34–37, 2011.

[26] A. Bhaskaran, S. Bijlay, M. More, P. Patil, and M. Vyawhare, "An Improved Activity Based Costing Algorithm for Task Scheduling in Cloud," *ResearchGate*, no. May, pp. 1–5, 2017, doi: 10.13140/RG.2.2.20501.12005.

[27] S. Ghanbari and M. Othman, "A priority based job scheduling algorithm in cloud computing," *Procedia Eng.*, vol. 50, no. January 2018, pp. 778–785, 2012, doi: 10.1016/j.proeng.2012.10.086.

[28] K. Amalakar and M. Oulika, "A Priority Based Job Scheduling

ISSN: 2582-0486 (online) Vol.-4, Issue-1 DOI: 10.46860/cgcijctr.2021.12.31.226

Algorithm in Cloud Computing," *Sementic Sch.*, pp. 1–6, 2015.

[29] P. Kaur and P. Singh, "Priority based Scheduling Algorithm with Fast Task Completion Rate in Cloud," *Adv. Comput. Sci. Inf. Technol.*, vol. 2, no. 10, pp. 17–20, 2015.

[30] A. Bala and I. Chana, "Multilevel priority-based task scheduling algorithm for workflows in cloud computing environment," *Adv. Intell. Syst. Comput.*, vol. 408, pp. 685–693, 2016, doi: 10.1007/978-981-10-0129-1_71.

[31] A. Thomas, G. Krishnalal, and V. P. Jagathy Raj, "Credit based scheduling algorithm in cloud computing environment," *Procedia Comput. Sci.*, vol. 46, no. Icict 2014, pp. 913–920, 2015, doi: 10.1016/j.procs.2015.02.162.

[32] V. Sharma and M. Bala, "A Credits Based Scheduling Algorithm with K-means Clustering," *ICSCCC 2018 - 1st Int. Conf. Secur. Cyber Comput. Commun.*, pp. 82–86, 2018, doi: 10.1109/ICSCCC.2018.8703201.

[33] S. Selvarani and G. S. Sadhasivam, "Improved cost-based algorithm for task scheduling in cloud computing," *2010 IEEE Int. Conf. Comput. Intell. Comput. Res. ICCIC 2010*, pp. 620–624, 2010, doi: 10.1109/ICCIC.2010.5705847.

[34] M. A. S. Mosleh, G. Radhamani, M. A. G. Hazber, and S. H. Hasan, "Adaptive Cost-Based Task Scheduling in Cloud Environment," *Sci. Program.*, vol. 2016, 2016, doi: 10.1155/2016/8239239.

[35] S. Sharma and S. Tyagi, "Cost-Based Task Scheduling in Cloud Computing," *Int. Res. J. Eng. Technol.*, vol. 4, no. 6, pp. 694–699, 2017.

[36] M. Amoon, N. El-Bahnasawy, and M. ElKazaz, "An efficient cost-based algorithm for scheduling workflow tasks in cloud computing systems," *Neural Comput. Appl.*, vol. 31, no. 5, pp. 1353–1363, 2019, doi: 10.1007/s00521-018-3610-2.

[37] N. Mansouri and M. M. Javidi, "Cost-based job scheduling strategy in cloud computing environments," *Distrib. Parallel Databases*, vol. 38, no. 2, pp. 365–400, 2020, doi: 10.1007/s10619-019-07273-y.

[38] X. Wu, M. Deng, R. Zhang, B. Zeng, and S. Zhou, "A task scheduling algorithm based on QoS-driven in Cloud Computing," *Procedia Comput. Sci.*, vol. 17, pp. 1162–1169, 2013, doi: 10.1016/j.procs.2013.05.148.

[39] L. Shi *et al.*, "Multijob Associated Task Scheduling for Cloud Computing Based on Task Duplication and Insertion," *Wirel. Commun. Mob. Comput.*, vol. 2021, 2021, doi: 10.1155/2021/6631752.

[40] S. K. Panda and P. K. Jana, "A multi-objective task scheduling algorithm for heterogeneous multi-cloud environment," in *2015 International Conference on Electronic Design, Computer Networks & Automated Verification (EDCAV)*, Jan. 2015, pp. 82–87, doi: 10.1109/EDCAV.2015.7060544.

[41] Y. Ge and G. Wei, "GA-based task scheduler for the cloud computing systems," *Proc. - 2010 Int. Conf. Web Inf. Syst. Mining, WISM 2010*, vol. 2, pp. 181–186, 2010, doi: 10.1109/WISM.2010.87.

[42] R. Kaur and S. Kinger, "Enhanced Genetic Algorithm based Task Scheduling in Cloud Computing," *Int. J. Comput. Appl.*, vol. 101, no. 14, pp. 1–6, 2014, doi: 10.5120/17752-8653.

[43] A. S. Ajeena Beegom and M. S. Rajasree, "Genetic algorithm framework for Bi-Objective Task scheduling in cloud computing systems," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8956, pp. 356–359, 2015, doi: 10.1007/978-3-319-14977-6_38.

[44] S. A. Hamad and F. A. Omara, "Genetic-Based Task Scheduling Algorithm in Cloud Computing Environment," *Int. J. Adv. Comput. Sci. Appl.*, vol. 07, no. 07, pp. 550–556, 2016, [Online]. Available: http://www.hanspub.org/journal/PaperDownload.aspx?DOI=10.12677/CSA.2016.66038.

[45] J. W. Li and C. W. Qu, "Cloud computing task scheduling based on cultural genetic algorithm," *MATEC Web Conf.*, vol. 40, 2016, doi: 10.1051/matecconf/20164009008.

[46] S. Liu and N. Wang, "Collaborative Optimization Scheduling of Cloud Service Resources Based on Improved Genetic Algorithm," *IEEE Access*, vol. 8, pp. 150878–150890, 2020, doi: 10.1109/ACCESS.2020.3016762.