**International Journal of Emerging Research in Engineering and Technology**
Pearl Blue Research Group| Volume 6 Issue 3 PP 95-102, 2025
ISSN: 3050-922X | https://doi.org/10.63282/3050-922X.IJERET-V6I3P111

*Original Article*

# Cloud Task Scheduling Techniques: A Survey of Greedy, Machine Learning, and Metaheuristic Approaches

Sandeep Gupta
Department of Artificial Intelligence, Samrat Ashok Technological Institute (SATI), Vidisha, Madhya Pradesh (MP), India.

***Received On: 30/06/2025*** ***Revised On: 18/07/2025*** ***Accepted On: 23/08/2025*** ***Published On: 03/09/2025***

***Abstract*** *- Cloud computing has emerged as an enabling technology, that offers scalability, elastic, on-demand computing and storage resources through internet servers. Appropriate scheduling of tasks in cloud systems is a requisite in achieving an optimal performance scale, minimizing costs of operation and achieving Quality of Service (QoS) demands. This paper is a review of the existing task scheduling methods with a detailed description of the most popular of them related to greedy algorithms, machine learning (ML), and metaheuristic approaches. Greedy algorithms including First Come First Serve (FCFS), Shortest Job First (SJF) are simple and computation efficient, yet they are not usually adaptable and, might not be globally optimized. Schedulers based on machine learning have been developed to use historical information, adaptive models to better assist decision-making under dynamic conditions, and use supervised, reinforcement and online learning techniques. Natural-yet-inspired metaheuristic algorithms, (e.g. Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO)) offer strong global search properties well-suited to complex multi-objective optimization. Comparative analyses suggest that metaheuristic and machine learning approaches inheritably deliver better productivity in terms of makespan, resource consumption, and energy efficiency than conventional greedy based methods. The combination of these methods holds great potential in terms of cloud systems concerning heterogeneity, scalability, and dynamic workload handling issues. The survey exclaims its significance of adaptive, hybrid scheduling structures to address changing needs within the large-scale cloud infrastructures.*

***Keywords*** *- Cloud Computing, Task Scheduling Algorithms, Greedy Algorithms, Machine Learning, QOS, Computational Efficiency, Adaptive Scheduling, Reinforcement Learning.*

## 1. Introduction

One of the most important parts of today's technology is cloud computing, which uses the internet to provide computer and storage resources on demand in an elastic, scalable, and incredibly modern way [1]. It rids its customers of the burden of managing the physical infrastructure needed to run a number of services, including storage, processing power, and other applications. As cloud services are incorporated into many more fields today, including healthcare, finance, education, and entertainment, the effective utilization of resources has acquired extreme significance [2]. All the big problems with managing resources in the cloud revolve around task scheduling.. Task scheduling: Task scheduling can be thought of as assigning workloads to cloud resources such that the performance is optimal and at the same time cost, latency and energy is kept to a minimum.

Task scheduling in cloud computing needs to be dealt with due to both internal and external requirements including capacity of storage, response time, cost of resources and bandwidth, which may change between different tasks. The issues that are major obstacles in this area are scalability, reliability, tool balancing, performance optimization, dynamic resource reshaping [3]. The above difficulties can be overcome with utilization of powerful task scheduling schemes that improve the performance of cloud computing environments [4]. As a result, the task schedulers form a major part of any cloud infrastructure. Task scheduling contributes directly to the realization of user QoS goals such as decreased response time, minimized total execution time (Makespan), enhanced throughput and other performance indicators as a consequence of its action of ensuring efficient workload distribution.

Greedy algorithms were early methods of task scheduling where each scheduling decision is made locally, on a step-wise basis, to achieve optimality locally. Such other examples are FCFS, Min-Min and Max-Min algorithms. Such approaches are computationally cheap and do not require complex computation, but in most cases they cannot guarantee global optimization and this is especially the case where the variability in workload is great [5]. Although greedy techniques have limitations, they are still employed in low-competence environments and in programs designed to be carried out in a non-dynamic condition because of speed and predictability. In order to address the disadvantage of greedy methods, researchers have started looking toward scheduling techniques based on machine learning (ML). ML can help systems to be able to learn using the data in the past and base their predictions on resource demands and task completion time and scheduling priority. Deep learning, reinforcement learning, and supervised learning are methods that allow schedulers to adjust to every new workload pattern and dynamic infrastructure state [6]. Real time, dynamically and over time-improving capabilities of ML-based schedulers as well as predicting possible system failures means that they are adapted to large-scale and real-time cloud activities.

Metaheuristics, in particular, have been the focus of much research into finding solutions to the work scheduling issue [7]. Genetic Algorithms (GA), Ant Colony optimisation (ACO), and Particle Swarm Optimisation (PSO) are a few examples of algorithms that mimic natural processes and offer powerful capabilities for global optimisation [8][9]. The algorithms are very suitable when solutions are not possible to derive in complex multi objectives environments with many objectives. Metaheuristics considerably search large solution spaces and can be tailored or blended to particular cloud applications. Their added exploitation and exploration capabilities position them well to be candidates in task scheduling in dynamic, high-load Cloud Security's.[10].

### *1.1. Structured of the paper*

The following is the outline of the paper: Task scheduling in cloud environments is covered in Section II, along with any gaps and objectives. Section III deals with Greedy Algorithm, its advantages, and disadvantages. Section IV talks about Machine Learning, and Section V talks about Metaheuristic algorithms. Section VI provides the Literature Review of recent researches. Lastly, it ends by providing important results and the future course of research in Section VII.

## 2. Task Scheduling In Cloud Environment

The cloud computing spaces are made up of very many data centers which accommodate so many physical computers popularly known as the hosts. Each of the hosts hosts several virtual machines (VMs) which are in-turn charged with the running of the tasks that are submitted by users and have varying QoS requirements. Figure 1 shows the steps involved in scheduling jobs in a cloud computing environment [11]. Assume there are n cloudlets (tasks), denoted as $T = \{T_1, T_2, T_3, \ldots, T_n\}$, which are scheduled for execution on m virtual machines, represented as $VM = \{VM_1, VM_2, VM_3, \ldots, VM_m\}$. These activities vary in both length and resources requirements and the VMs are also heterogenous based on bandwidth, RAM and CPU capacity in each VM.

The cloud broker provides the service as a mediator between the users of the cloud service vendors' products [12]. It directs queries to the cloud information service to determine information about available resources and their capabilities to schedule the tasks granted in an efficient mode. Scheduling of the broker is based on several variables and QoS parameters which determine the choices and assigning tasks to appropriate resources. The cloud broker is in charge of connecting with the user and provider, as well as setting the schedule and location to run tasks on available resources. It is also at the centre of the task scheduling process.. There are however some difficulties in the process of scheduling. Tasks remain usually queued and have to be awaited to have resource availability, which may result in the delay as resources are assigned and are becoming occupied.

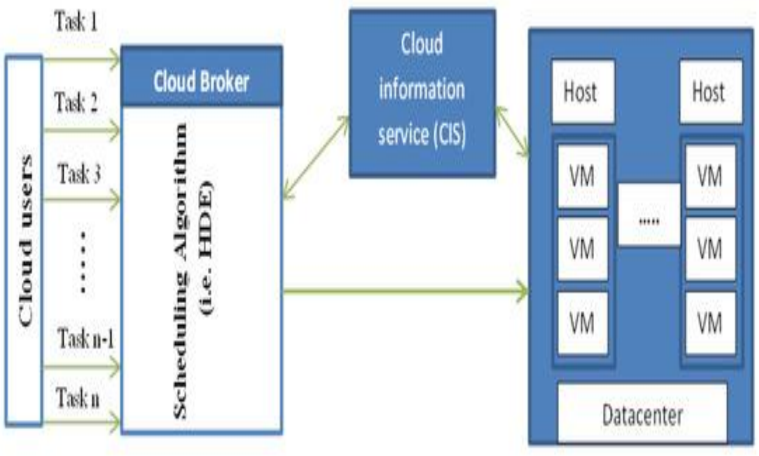

**Fig 1: Task Scheduling Architecture in Cloud Environment**

The objectives of scheduling tasks in a cloud environment are:

- Minimising the makespan also known as the total job piece-time of all tasks.
- Decreasing the expenditures on the computing and storage resources.
- Improving energy efficiency to help green computing efforts.
- Distribution of the work so that particular nodes are not overloaded.
- Timeliness and accuracy of work are two aspects of QoS that they must be able to provide.
- Maximizing throughput, which is the corresponding tasks per unit time.

### *2.1. Key Challenges in Task Scheduling for Cloud Environment*

The nature of cloud environment naturally makes task scheduling challenging because of several dynamic interacting factors. The major issues are:

- **Resource Heterogeneity:** Cloud infrastructures entail a variety of resources that include CPUs, GPUs, memory, and storage devices, each with different capabilities, and performance attributes. This heterogeneity makes it difficult to manage the scheduling processes since work is to be matched accordingly to the most appropriate resources.
- **Dynamic Workload Arrival:** The quality and quantity of the work that comes along as well as its order might change tremendously depending on the period. Scheduling algorithms should be dynamic in facilitating such flexible and unpredictable workloads in an efficient manner.
- **Multi-objective Optimization:** Scheduling tasks frequently requires juggling competing priorities, such as reducing expenses without sacrificing performance or throughput [13]. A trade-off between these various criteria is substantial to attain an ideal trade-off among these criteria.
- **Scalability:** The scheduling algorithms should be effective and efficient when the size of the cloud infrastructure is increased, supporting more resources and a greater number of tasks without any loss of performance.
- **Resource Contention:** There are several users and applications that are fighting over a shared reservoir of resources. Intelligent priorities and allocation to handle contention and achieve fairness is essential to good scheduling.
- **Fault Tolerance and Reliability:** Cloud infrastructures experience hardware malfunctions, network problems and software outreach. The mistake in the scheduling systems must be tolerant in order to sustain services that are reliable and constant.
- **Energy Efficiency:** The rising energy concerns and operating costs have made task scheduling to look at the aspect of energy consumption besides encouraging green computing skills by optimal utilization of resources in order to save on power consumption.

### *2.2. Key Parameters and Constraints in Task Scheduling*
There are a number of important parameters and constraints that should be taken into account to schedule tasks efficiently in cloud environment, and they are as follows:

- **Task Characteristics:** Task size, task priority, task deadline, estimated task execution time and task dependency are attributes which have a big impact in scheduling decisions.
- **Resource Capabilities:** The convenience in terms of processing power, memory capacity, storage availability, energy efficiency and presence of the resources would have to be considered to ensure optimum distribution of tasks [14].
- **Quality of Service (QoS) Constraints:** The demands in latency, throughput, availability, and fault tolerance are instrumental in segmenting the appropriate scheduling strategies.
- **Budget Constraints:** Financial constraints- These variables are typically determined by the user(s) or written into the service level agreement (SLAs) and prevent them spending more on task performance than what is allowed.
- **Dependency and Precedence:** Dependencies are common in a large number of tasks when the starting or finish of a task waits on the presentations of another task and thus requires stringent scheduling and synchronization.
- **Load Balancing and Resource Utilization:** It is critical to ensure that work is evenly apportioned among all the available resources to eliminate bottlenecks and overall efficiency of the system [15].

## 3. Greedy Algorithms for Task Scheduling
Greedy algorithms represent one of the most fundamental and widely employed strategies for task scheduling in cloud environments. The goal of these algorithms is to find the best solution on a global scale, and they do this by iteratively making locally optimum decisions. In the context of cloud task scheduling, greedy methods typically select the best task-resource pairing based on predefined heuristics such as shortest job duration, earliest arrival time, or minimal resource consumption. Due to their simplicity, low computational complexity, and ease of implementation, greedy algorithms are particularly suited for real-time scheduling and scenarios with low system complexity [16]. There are several well-known greedy scheduling algorithms that have been adapted for cloud computing:

- **First Come First Serve (FCFS):** The tasks are tightly scheduled in the order that they done. This method is clear and simple to use, but it doesn't take task size or resource efficiency into account, so it often doesn't give the best results, especially for bigger or more complicated tasks.
- **Shortest Job First (SJF):** This algorithm prioritises tasks based on their predicted execution time, aiming to decrease average waiting time. But SJF can cause famine if shorter jobs keep arriving, eventually delaying longer ones.
- **Min-Min:** The Min-Min algorithm first puts every task on the time the task could be finished within the shortest time possible in a resource. In these assignments, the assignment that requires the least amount of time to be completed is picked to be undertaken. Such a strategy is more preferential towards small tasks and tends to cause an overall decrease in the make span, yet has higher chances of causing more important tasks to be overlooked.
- **Max-Min:** Max-Min, like Min-Min, finds tasks by minimising their time to complete on all resources, and then it executes the task with the longest time to completion. Workload is spread equally across this strategy and starvation is mitigated as the greater work not continuously missed.

### *3.1. Advantages and Limitations of Greedy Algorithms in Cloud Task Scheduling*

Greedy algorithms have many advantages in relation to the task scheduling in cloud platforms such as simplicity, speedy decision making, and minimal computational burden. These features are why they can be applied in cases where there is need to issue a replica instantly, particularly in homogeneous or sparsely loaded systems. Nevertheless, in spite of all these benefits, greedy algorithms have some significant drawbacks which impair their performance in highly diversified cloud environments.

#### *3.1.1. Advantages*

- **Low Computational Complexity:** Greedy algorithms have negligible processing demands, making them easy and fast in scheduling decisions with high applicability in real time applications.
- **Ease of Implementation:** These are rule-based algorithms that do not need any training data or optimization process to deploy without difficulties.
- **Effective in Homogeneous and Lightly Loaded Environments:** Greedy techniques scale well in cloud systems with relatively standard resources and relative resource smoothness and workloads.
- **Deterministic Behavior:** Scheduling results are more predictable and reliable when implemented using greedy algorithms since they consistently produce the same output using the same input.
- **Suitability for Small-scale Systems:** They are useful algorithms in environments where the number of both tasks and resources is limited and a simple scheduling is required.

#### *3.1.2. Limitations*

- **Lack of Global Optimization:** Greedy algorithms place emphasis on the local optimization of decisions that normally translates to poor global performance and inability to meet the global optimum in terms of scheduling.
- **Inflexibility:** These are non-dynamic approaches which not change with change of workload or diverse and varying state of the system and tend to be limited to constant environment use.
- **Unfair Scheduling:** Scheduling algorithms such as Shortest Job First (SJF) can cause starvation, with longer tasks being prioritized progressively less and never run to completion.
- **Poor Performance in Heterogeneous and Highly Dynamic Environments:** This failure to properly deal with diversity of resources and workload volatility make them less effective in the real-world cloud infrastructures.
- **Neglect of Task Dependencies:** Greedy algorithms do not often take inter-task relationships into account, or precedence constraints, which are important in work flows with dependent tasks.

### *3.2. Performance in Static and Dynamic Environments*

Greedy algorithms usually work well in a static cloud in which workload characteristics and system states do not vary much. They can be easily implemented, have low overheads and predictability in behavior and hence, find useful utility in simple schedule tasks. Nevertheless, greedy algorithms may not be efficient and fair in a dynamic workload where workloads often vary, resources are often not always available, and the requirements of the users may be constantly changing. There is the problem of resource underutilization and delays in execution of tasks due to a lack of adaptability in them. Greedy algorithms are therefore usually used in simple or time-constrained scheduling that do not need to take into consideration many factors or are applied with more complex algorithms that pursue their complexity in large dynamic cloud systems.

## 4. Machine Learning Approaches To Task Scheduling In Cloud Environments

Machine learning (ML) has recently become a compelling paradigm of task scheduling in cloud settings that enables scheduled tasks to take advantage of the automated, data-driven approaches that can eventually learn over time and respond efficiently to the evolution of the state easily. Such ML methods process workload patterns, estimate execution times and optimize resource allocation at runtime and in contrast to static or rule-based algorithms, the observations are used as input [17]. This is what makes ML especially efficient in circumstances that include complex, but also dynamic and multi-objective applications of cloud where traditional approaches tend to fail.

### *4.1. Supervised Learning Techniques*

Supervised learning models are very common in predicting the task attributes like the usage time, resource used and the optimum resource allocation by learning models on past labelled data. The main skills are:[18].

- **Support Vector Machines (SVM):** Used for task categorization and priority setting, building a resource map.
- **Decision Trees and Random Forests:** Provide interpretable models for scheduling decisions based on task features.
- **K-Nearest Neighbors (KNN), Naive Bayes, and Gradient Boosting:** Effective when large labeled datasets are available.

These methods excel in relatively stable environments with consistent workload patterns.

### *4.2. Reinforcement Learning and Deep Reinforcement Learning*

RL techniques enable adaptive scheduling by learning through trial and error, optimizing policies based on feedback from the environment [19]. Notable methods include:

- **Q-Learning and Deep Q-Networks (DQN):** Model scheduling as a Markov Decision Process, facilitating dynamic, context-aware decisions.
- **Policy Gradient and Actor-Critic Methods:** Suitable for complex environments with large state-action spaces.

### *4.3. Online Learning and Adaptive Scheduling*
Online learning algorithms continuously update scheduling policies in response to incoming data, unlike batch learning, which relies on fixed datasets [20]. Key approaches include:

- **Incremental Models:** Adapt to fluctuating workloads and system performance.
- **Contextual Bandits:** Manage task allocation under uncertainty with limited feedback.
- **Multi-Armed Bandit (MAB) Frameworks:** Strike a balance between discovering new and exploiting existing, efficient resources.

These approaches are especially valuable for real-world cloud systems characterized by frequent changes in workloads and resource availability.

## 5. Metaheuristics Approach in Scheduling
In cloud task scheduling, where numerous competing goals and limitations are present, metaheuristic algorithms have gained renown as powerful tools for resolving optimisation problems. Due to the computational complexity of exact optimization methods in large-scale cloud environments, metaheuristics provide a practical alternative by efficiently exploring vast solution spaces to identify near-optimal scheduling solutions. These algorithms are flexible and adaptable to dynamic, nonlinear scheduling problems without requiring detailed problem-specific information, making them suitable for both single and multi-objective optimization challenges [21].

Common metaheuristic methods applied in cloud task scheduling include:

- **Genetic Algorithms (GA):** Genetic algorithms (GAs) take their cue from natural selection and employ genetic operators such as mutation, selection, and crossover to generate a pool of possible solutions. When a fitness function is used to evaluate solutions, many metrics are considered, including makespan, cost, and energy usage.
- **Particle Swarm Optimization (PSO):** Particle swarm optimisation (PSO) mimics the cooperation seen in natural groups of individuals, such schools of fish or flocks of birds, by iteratively moving a population of possible solutions, or "particles," through the search space and adjusting their positions along the way.
- **Ant Colony Optimization (ACO):** The ACO algorithm mimics the ant's foraging behaviour by building solutions using heuristic information and pheromone trails, and by adding more solutions as iterations rise.

## 6. Literature Review
This section of the literature review outlines the most recent developments in cloud task scheduling, namely all the improved algorithms, meta heuristic optimization, priority-based models, adaptive approaches and issues in heterogeneous a multi-cloud environment. Aloudah and Banimelhem (2025) Task scheduling problem is still relevant in the cloud computing environment because it is an efficient problem in ensuring the necessary performance, in relation to the makespan and power consumption aspects. This paper gives a better task scheduling algorithm, which has been named Enhanced Energy Management Algorithm (EEMA). When compared with an earlier Energy Management Algorithm (EMA) to task scheduling in a cloud environment, the simulation results have indicated that EEMA has resulted in the reduction of makespan by 5 to 28 percent and also by 44 percent against first-come-first-serviced (FCFS) algorithm using the different numbers of tasks and the different virtual machines used in the simulation results [22].

Habibpour Roudsari (2024) Intelligent Harris Hawk Optimization is the target of this study's recommendations. The method not only fixed typical issues with related algorithms, such as low exploration capabilities during the exploration stage and premature convergence, but it also addressed the work scheduling problem in heterogeneous systems. In order to address them, the exploration phase was put aside and it was distinct with the exploitation phase. This was in an attempt to make sure that the algorithm sees the full space of problems without necessarily giving optimum solution. Additionally, the algorithm may become stuck in a local optimum due to the exploitation process, making early convergence impossible [23]. Lipsa et al. (2023) This research models the scheduling of cloud computing jobs using the M/M/n queuing paradigm. A waiting time matrix, a recently discovered data structure, is used to construct an algorithm that prioritizes incoming jobs. In addition, the waiting queue employs an independent idea derived from the Fibonacci heap to prioritise tasks according to their urgency. A parallel approach to task scheduling is suggested in the work that is currently under consideration. This approach involves building heaps and assigning priorities to tasks in a manner that is parallel to both preemptive and non-preemptive tasks [24].

M. Almufti et al. (2023) Natural processes like evolution, swarm behaviour, and genetics serve as models for the methods employed to determine the global optimum of a problem. Then, a more specific area of interest is searched using them. Metaheuristic algorithms include some popular ones like genetic algorithms, ant colony optimizations, particle swarm optimizations, simulated annealing, and tabu search. Computer scientists, engineers, and financiers have all made extensive use of these techniques to tackle challenging problems. Overall, metaheuristic algorithms have a history of development over several decades, in which many different optimization algorithms have been developed which are inspired by natural systems [25]. Mukherjee et al. (2022) This study proposes a method for managing digital device tasks in cloud data centres called ASA-TL. Thanks to ASA, all of the virtual servers are now working together to handle the workload, and the cloud data is safe from being overloaded. Consideration of the digital device's significance and status is exerted during data assignment in order to facilitate equitable and efficient work allocation. TL handles these requests effectively, and the input to the tasks needs to be distributed reliably and equitably among the processors [26].

Gupta, Batra and Khosla (2021) Cloud scheduling strategies such as the Max-Min, Min-Min, Enhanced Max-Min, and Greedy strategies were employed to distribute server workloads evenly in this research. The optimal scheduling algorithm was then determined by analysing the outcomes of various algorithms. The results talked about in this study show that the greedy algorithm works better when there are more tasks than other scheduling algorithms. On the other hand, the Enhanced Max-Min algorithm works much better when there are fewer tasks than another task scheduling algorithm [27]. Karaja, Ennigrou and Said (2020) Cloud computing has exploded in popularity due to its convenient on-demand service model, which allows users to pay only for the resources they really use. The demand for cloud services has led to the development of a multi-cloud architecture, which allows for the interconnection of multiple clouds to better serve individual users. Scheduling tasks in these kinds of settings is hard because the tools aren't all the same. For a heterogeneous multi-cloud setting, this study proposes a budget-limited dynamic Bag-of-Tasks scheduling protocol. By running tests on synthetic data sets that come up with, show that the method works well in terms of makespan [28].

The Table I summarizes key studies on cloud task scheduling, highlighting diverse approaches, their effectiveness in optimizing performance, and limitations related to scalability, dynamic workloads, and adaptation to heterogeneous cloud environments.

**Table 1: Summary of Key Literature on Task Scheduling Approaches in Cloud Computing**

| Reference | Study Focus | Approach | Key Findings | Limitations / Future Work |
|---|---|---|---|---|
| Aloudah and Banimelhem (2025) | Task scheduling focusing on makespan and power consumption | Enhanced Energy Management Algorithm (EEMA) | EEMA reduces makespan by 5–28% and power consumption by 44% compared to FCFS and EMA | Needs testing on larger heterogeneous environments; integration with dynamic workloads could be explored |
| Habibpour Roudsari (2024) | Tackling task scheduling with metaheuristic optimization | Intelligent Harris Hawk Optimization (metaheuristic) | Separated exploration and exploitation phases to avoid early convergence | Algorithm may get trapped in local optima during exploitation; needs enhancement to improve global search |
| Lipsa et al. (2023) | Task scheduling in the cloud based on priorities | Priority assignment + Parallel algorithm using Fibonacci heap | Efficient task priority assignment with parallel heap building | Applicability to large-scale heterogeneous cloud environments and dynamic workloads remains to be validated |
| M. Almufti et al. (2023) | Metaheuristic algorithms overview across domains | Genetic Algorithm, PSO, ACO, SA, Tabu Search (Metaheuristics) | Metaheuristics effective for large and complex search spaces | Lack of specific cloud computing focus; need to adapt algorithms for cloud-specific constraints and dynamics |
| Mukherjee et al. (2022) | Adaptive scheduling for cloud data centers | Adaptive Scheduling Algorithm-Guided Task Loading | Fair task distribution among virtual servers; protects against overload | Requires evaluation in multi-cloud environments; scalability and energy efficiency improvements needed |
| Gupta, Batra and Khosla (2021) | Egocentric cloud scheduling algorithms: a comparative study | Malicious Algorithms: Max-Min, Min-Min, and Improved Max-Min | Greedy algorithms outperform others for large tasks; Enhanced Max-Min better for fewer tasks | Limited exploration of hybrid or machine learning-based approaches; focus mostly on static workload scenarios |
| Karaja, Ennigrou and Said (2020) | Scheduling in heterogeneous multi-cloud environments | Budget-constrained dynamic Bag-of-Tasks algorithm | Effective makespan reduction in multi-cloud settings with resource heterogeneity | Requires real-world deployment and consideration of dynamic task arrivals; energy consumption not addressed |

## 7. Conclusion and Future Work

Task scheduling is a very important aspect of cloud computing infrastructure, where it is important to optimize the performance and the usage of the resources. Different methods have been implemented to find solutions to intricacies associated with cloud task management, which include greedy algorithms, machine learning and meta heuristic methods. Greedy algorithms are simple and reasonably fast, and thus good in low complexity and real-time situations, but can generally fail to perform global optimization and may be sensitive to dynamic workloads. Machine learning techniques also offer adaptive and information driven scheduling functions, which effectively manage variability and thus increases the quality of decisions to be used over time. Metaheuristic techniques are characterized by strong global search capabilities that search effectively through large solution spaces to trade-off conflicting goals, e.g., cost, makespan, and energy consumption. All these different techniques contribute towards the overall progress of the efficiency, and flexibility of cloud task scheduling under different operational constraints and workload situations.

Further investigations and work should be done to formulate the hybrid frameworks which combine the advantages of different methods to adapt the dynamic and heterogeneous nature of the cloud environments. Also, it will be essential to integrate energy-saving and fault-resistant systems into scheduling schemes that uphold productive and dependable cloud services. Real-time adaptation through continual learning and scalability to multi-cloud infrastructures also represent promising directions for future exploration.

## References

[1] S. P. Bheri and G. Modalavalasa, "Advancements in Cloud Computing for Scalable Web Development: Security Challenges and Performance Optimization," *J. Comput. Technol. Int. J.*, vol. 13, no. 12, 2024.

[2] S. Shilpashree, R. R. Patil, and C. Parvathi, "'Cloud computing an overview,'" *Int. J. Eng. Technol.*, 2018, doi: 10.14419/ijet.v7i4.10904.

[3] C. Shyalika, T. Silva, and A. Karunananda, "Reinforcement Learning in Dynamic Task Scheduling: A Review," *SN Comput. Sci.*, vol. 1, no. 6, Nov. 2020, doi: 10.1007/s42979-020-00326-5.

[4] G. Maddali and S. J. Wawge, *Site Reliability Engineering*. 2025.

[5] A. Jain, M. Saini, and M. Kumar, "Greedy Algorithm," *J. Adv. Res. Comput. Sci. Eng. (ISSN 2456-3552)*, 2015, doi: 10.53555/nncse.v2i4.451.

[6] A. Chadha, S. Sharma, and V. Arora, *Computational Science and Its Applications*. 2023. doi: 10.1201/9781003347484.

[7] B. K. R. Janumpally, "Intelligent Energy Aware Efficient Task Scheduling in Cloud Computing: Leveraging Swarm Optimization Algorithms for Improve Resource Utilization," in *2025 1st International Conference on Radio Frequency Communication and Networks (RFCoN)*, IEEE, Jun. 2025, pp. 1–6. doi: 10.1109/RFCoN62306.2025.11085278.

[8] S. S. S. Neeli, "Heart Disease Prediction For A Cloud-Based Smart Healthcare Monitoring System Using Gans And Ant Colony Optimization," *Int. J. Med. Public Heal.*, vol. 14, no. 4, p. 11, 2024.

[9] P. Agrawal, H. F. Abutarboush, T. Ganesh, and A. W. Mohamed, "Metaheuristic algorithms on feature selection: A survey of one decade of research (2009-2019)," *IEEE Access*, 2021, doi: 10.1109/ACCESS.2021.3056407.

[10] D. D. Rao, S. Madasu, S. R. Gunturu, C. D'britto, and J. Lopes, "Cybersecurity Threat Detection Using Machine Learning in Cloud-Based Environments: A Comprehensive Study," *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 12, no. 1, 2024.

[11] G. Maddali, "An Efficient Bio-Inspired Optimization Framework for Scalable Task Scheduling in Cloud Computing Environments," *Int. J. Curr. Eng. Technol.*, vol. 15, no. 3, 2025.

[12] M. Abdel-Basset, R. Mohamed, W. Abd Elkhalik, M. Sharawi, and K. M. Sallam, "Task Scheduling Approach in Cloud Computing Environment Using Hybrid Differential Evolution," *Mathematics*, vol. 10, no. 21, p. 4049, Oct. 2022, doi: 10.3390/math10214049.

[13] C. Carrión, "Kubernetes Scheduling: Taxonomy, Ongoing Issues and Challenges," *ACM Comput. Surv.*, vol. 55, no. 7, pp. 1–37, Jul. 2023, doi: 10.1145/3539606.

[14] T. Rausch, A. Rashed, and S. Dustdar, "Optimized container scheduling for data-intensive serverless edge computing," *Futur. Gener. Comput. Syst.*, 2021, doi: 10.1016/j.future.2020.07.017.

[15] A. R. Duggasani, "Scalable and Optimized Load Balancing in Cloud Systems: Intelligent Nature-Inspired Evolutionary Approach," *Int. J. Innov. Sci. Res. Technol.*, vol. 10, no. 5, May 2025, doi: 10.38124/ijisrt/25may1290.

[16] C. F. Kurz, W. Maier, and C. Rink, "A greedy stacking algorithm for model ensembling and domain weighting," *BMC Res. Notes*, 2020, doi: 10.1186/s13104-020-4931-7.

[17] H. Lee, S. H. Moon, J. Y. Hong, J. Lee, and S. H. Hyun, "A Machine Learning Approach Using FDG PET-Based Radiomics for Prediction of Tumor Mutational Burden and Prognosis in Stage IV Colorectal Cancer," *Cancers (Basel).*, 2023, doi: 10.3390/cancers15153841.

[18] A. I. Kadhim, "Survey on supervised machine learning techniques for automatic text classification," *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 273–292, Jun. 2019, doi: 10.1007/s10462-018-09677-1.

[19] B. R. Kiran *et al.*, "Deep Reinforcement Learning for Autonomous Driving: A Survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 4909–4926, Jun. 2022, doi: 10.1109/TITS.2021.3054625.

[20] J. Bassen *et al.*, "Reinforcement Learning for the Adaptive Scheduling of Educational Activities," in *Conference on Human Factors in Computing Systems - Proceedings*, 2020. doi: 10.1145/3313831.3376518.

[21] A. Peiris, F. K. P. Hui, C. Duffield, and T. Ngo, "Production scheduling in modular construction: Metaheuristics and future directions," *Autom. Constr.*, vol. 150, p. 104851, Jun. 2023, doi: 10.1016/j.autcon.2023.104851.

[22] A. A. Aloudah and O. Banimelhem, "An Enhanced Task Scheduling Algorithm for Cloud Computing Environments," in *2025 16th International Conference on Information and Communication Systems (ICICS)*, IEEE, Jul. 2025, pp. 1–5. doi:

10.1109/ICICS65354.2025.11073117.

[23] M. N. H. Roudsari, "Improved task scheduling in heterogeneous distributed systems using intelligent greedy harris hawk optimization algorithm," *Evol. Intell.*, vol. 17, no. 5–6, pp. 4199–4226, Oct. 2024, doi: 10.1007/s12065-024-00979-8.

[24] S. Lipsa, R. K. Dash, N. Ivković, and K. Cengiz, "Task Scheduling in Cloud Computing: A Priority-Based Heuristic Approach," *IEEE Access*, vol. 11, pp. 27111–27126, 2023, doi: 10.1109/ACCESS.2023.3255781.

[25] S. M. Almufti, A. A. Shaban, R. I. Ali, and J. A. Dela Fuente, "Overview of Metaheuristic Algorithms," *Polaris Glob. J. Sch. Res. Trends*, vol. 2, no. 2, pp. 10–32, Apr. 2023, doi: 10.58429/pgjsrt.v2n2a144.

[26] D. Mukherjee, S. Ghosh, S. Pal, A. A. Aly, and D.-N. Le, "Adaptive Scheduling Algorithm Based Task Loading in Cloud Data Centers," *IEEE Access*, vol. 10, pp. 49412–49421, 2022, doi: 10.1109/ACCESS.2022.3168288.

[27] N. Gupta, M. Batra, and A. Khosla, "Optimizing Greedy Algorithm to Balance the Server Load in Cloud Simulated Environment," in *Proceedings of the 3rd International Conference on Inventive Research in Computing Applications, ICIRCA 2021*, 2021. doi: 10.1109/ICIRCA51532.2021.9544107.

[28] M. Karaja, M. Ennigrou, and L. Ben Said, "Budget-constrained dynamic Bag-of-Tasks scheduling algorithm for heterogeneous multi-cloud environment," in *2020 International Multi-Conference on: "Organization of Knowledge and Advanced Technologies" (OCTA)*, IEEE, Feb. 2020, pp. 1–6. doi: 10.1109/OCTA49274.2020.9151737.