Journal of Population Therapeutics & Clinical Pharmacology

RESEARCH ARTICLE
DOI: 10.53555/jptcp.v31i1.6977

ADVANCED TECHNIQUES FOR DISTRIBUTING AND TIMING ARTIFICIAL INTELLIGENCE BASED HEAVY TASKS IN CLOUD ECOSYSTEMS

Suri Babu Nuthalapati[1*], Aravind Nuthalapati[2]

[1*]Cloudera, Santa Clara, California, United States 95054/ Trida Labs, Bengaluru, Karnataka, India 560048
[2]Microsoft, Charlotte, NC, USA 28273

***Corresponding author:** Suri Babu Nuthalapati
*Email: thammuio@ieee.org

ABSTRACT

Today, the proliferation of Artificial Intelligence (AI) workloads has brought to light a requirement for nuanced cloud infrastructures that can efficiently process AI-heavy workloads. The results of this study, focusing on workload allocation and scheduling in cloud systems that are experiencing the additional challenge emanating from AI-intensive jobs. In this work, we aimed to investigate the prevailing techniques and limitations in current methodologies, with an eye toward designing new methods that can improve how AI resources are allocated and scheduled in cloud environments. The reason for this problem is that AI-related workloads are of diverse resources (why), change due to their nature, and require scalability. Artificial intelligence is also extremely computationally intensive and has a diversification of compute needs making distributing resources efficiently another beast. In addition, the nature of these activities is dynamic, which means responsive approaches are needed to cope with changing computing demands over time. In addition, given that both AI models and datasets are growing in complexity as well size this makes scalability an imperative for cloud environments to be able operate effectively. In this literature review, we will describe traditional and state-of-the-art task allocation systems to give a complete picture of their pros and cons. Moreover, we discuss the scheduling strategy used for managing tasks requiring more significant AI and provide an extensive review of state-of-the-art. To address these challenges, we introduce a sophisticated framework that emphasizes the deployability of resources and scheduling by leveraging machine learning tools, as well as more efficient job transfer methods. The system is designed to allocate resources elastically and match the dynamic requirements of AI based workloads. This is accomplished by employing machine learning algorithms to predict workload information and introducing task migration mechanisms for adjusting changes in workloads. The study concludes through an empirical assessment of the proposed solutions in a virtual environment using different datasets. We objectively measure the performance of our methods with respect to traditional approaches using important metrics such as throughput, latency and resource utilization. Such analysis can provide useful insights on cost effective deployment of AI rich workloads onto cloud infrastructures. It helps current initiatives to improve the scalability and performance of cloud environments as AI applications multiply.

Keywords: - Cloud Infrastructure, Task Distribution, Scheduling Techniques, Machine Intelligence (MI), Resource Efficiency, Adaptive Resource Allocation, Expandability, Job Migration.

I. INTRODUCTION

With the rapid development of artificial intelligence (AI) and cloud computing technologies, a myriad amount of sophisticated AI-centric tasks are becoming increasingly available in complex-cloud ecosystems that demands optimal allocation and scheduling. Those tasks, which are kind of compute-bound and big-data hungry by definition require efficient valuable scheduling techniques to guarantee a scalability. In this paper, the expanding research is employed to demonstrate that backhauling and scheduling resource-intensive AI tasks within cloud ecosystems can take advantage of multiple novel distribution/timing strategies.

AbstractOne of the biggest challenges in cloud-fog computing environments is IoT task scheduling which must be handled efficiently. In this paper a modified artificial ecosystem-based optimization (AEO) technique referred to as AEOSSA is proposed which aims at improving the exploitation ability of AEO by integrating Salp Swarm Algorithm (SSA). This leads the way to outperform other metaheuristic among many good results this approach has shown in task scheduling, such as makespan time and throughput [1]. A modified symbiotic organisms search (SOS) algorithm for task scheduling in cloud computing is also reported that improved makespan minimization and convergence speed [2].

A game-based offloading scheme for data-as-a-service (DAAS) platforms in the Internet of Things with Edge and Cloud Environment This method employs a hierarchical game model with fictitious play for optimal task distribution and improves the global quality of computation offloading [3]. Hybrid ML methods have also been used to solve resource allocation problems [26] and task scheduling on Cloud computing environment [27]. The RATS-HM approach has been proven effective for a near optimal shuffling solution by reconstructing both cat swarm optimization and deep neural network to optimize make-span time and throughput [4].

Moreover, bio-inspired algorithm is also employed for resource provisioning and task scheduling in clouds. Recently, a neural-bio inspired GA-ANN methodology has been proposed to schedule tasks on virtual machines more efficiently which in turn reduces the fault rates and overall execution time [5]. Hybrid swarm intelligence techniques have also been put forward to improve IoT application tasks scheduling in the cloud, such as MRFOSSA algorithm [6], showing promising results and performances with other metaheuristic strategies.

Furthermore, machine learning techniques have also been used to predict the execution times of workflow tasks in the cloud and managed to achieve high confidence results that surpass previous prediction models [7]. The two-stage strategy, which employs a Bayes classifier for task classification and dynamic matching of tasks with VMs is one kind of those strategies that has shown benefits over cloud scheduling performance and load-balancing [8]. Finally, in heterogeneous multi-cloud environments task scheduling approaches based on normalization have been proposed to effectively manage workloads across multiple clouds [9].

In conclusion, the incorporation of advanced optimization algorithms and bio-inspired techniques from swarm intelligence theory with sufficiently robust bottom-up AI-heavy tasks distribution along space (clouds) and time has enabled drastically improved performance in cloud ecosystems. In this paper, we will focus on the efficient methods that are applied to optimize cloud computing performance and elasticity.

One of the biggest changes to artificial intelligence (AI) has come via cloud computing, thanks in large part to advances and availability of infrastructure that makes deploying AI-heavy tasks much easier. But efficient distribution and timing of these tasks are still critical problems, General cloud ecosystems make design proposals challenging. The Events features federate Items across openHAB instances (cloud/edge), distribute tasks based on machine learning; and create hybrid algorithms to solve these issues. For example, federated cloud/edge (FCE) framework can achieve significant gains in AI processing time, model accuracy and fault tolerance [10], thus being a competitive solution to modern large-scale multitasking AI problems. An example is the Machine Learning based Task Distribution (MLTD), which uses Artificial Neural Networks (ANNs) to automatically balance and distribute tasks among fog/cloud resources; this way it minimizes response times of requests, reducing internet bandwidth consumption [11]. This way of thinking can

also be useful in combination with a hybrid strategy, since neural networks and genetic algorithms not only evolve for task scheduling; they are employed as well to assess high-performance distributed computation reducing completion times [12]. It can also involve decentralized distributed AI (DDAI) systems, which allow flexible and efficient task execution with dynamic component registration and management [13]. Moreover, modified G-SOS (symbiotic organisms search) algorithm is used for task scheduling algorithm by reducing execution time & cost and improving convergence speed [14]. These important considerations are security which has provided great efficiency for intelligent systems, such as using ANNs and evolutionary algorithms to automate various aspects of system software management including the scheduling processes in terms of both computing tasks and their desired levels on a secure platform while considering makespan [15]. In order to improve the way of handling incoming job requests. there is another approach which involves integrating AI techniques like genetic algorithms and ANNs in various resource allocation methodologies as well, this can help provide scalable solution along with being reliable towards multi-tenant cloud environment [16]. After all, worthwhile task scheduling systems are essential to cope with a variety of user requirements and get high performance on the cloud environment since they offer task priority-based work management which reduces not only idle time but also communication delays between operations [17]. Heuristic Initialization, Parallel execution using Particle Swarm Optimization (HIP-PSO), etc., are optimization algorithms [18] that enhance the task scheduling to reduce the time of computation and improve in overall system throughput. Together, these advanced techniques emphasize the need to develop creative ways of trading off and scheduling AI-intensitive jobs within cloud ecosystems so as deal with key challenges such as throughput scalability, resources allocation & access control mechanisms, security issues or energy optimization to provide reliable and efficient services into clouds.

II. LITERATURE REVIEW

Cloud computing has been rapidly evolving and many of the apps involve AI-heavy tasks that are best served by advanced techniques for task distribution and timing. This paper provides literature review works on different methodologies and algorithms that have been developed to improve task scheduling, resource allocation, data execution among cloud environments by particularly concentrated mainly around the AI-intensive tasks.

Background Task scheduling is essentially a key component in cloud computing since it heavily decides the task throughput, makespan and response time has attracted many research communities. The above problem in scheduling tasks within a cloud has led to many complex techniques proposed for the same. 4.2 Modified Artificial Ecosystem-Based Optimization (AEOSSA): A new task scheduling algorithm, named as AEOSSA, is developed by incorporating the Salp Swarm Algorithm (in short SSA) to improve search around global best position of original AV-PSO [21]. This method has shown better performance in makespan time & throughput as compare to other metaheuristic methods [19]. G_Symbiotic Organisms Search Algorithm (adaptive symbiosis) G_SOS: In view of the mutualism among species, it develops a task scheduling method with the objectives to minimize execution time and cost as well response time. Compared with traditional SOS and PSO-SA algorithms, it has demonstrated significant enhancement on large-scale problems [2]. Game-Based Offloading Scheme: This scheme utilizes a hierarchical game model to offload Data-as-a-Service (DAAS) computing tasks among IoT, Edge and Cloud ecosystems. The algorithm organizes tasks into data-intensive and CPU- intensive, then it proceeding to the computation offloading which results practically overall system performance improvement [20].

To manage such large amount of data and a variety of resources in cloud environments efficient resource allocation is required. AbstractHybrid machine learning approaches and bio-inspired systems are put forth to overcome these issues. Resource Allocation and Task scheduling for Software DefinEd Satellite IoT systems using Hybrid MaChine Learning Techniques (RATS-HM): It is a hybrid technique which combines enhanced cat swarm optimization of improved task sCHeduling Method, deep neural networks as resource allocation Mechanism, Lightweight Authentication Scheme to ensure data Security. This holistic view allows for leaner resource use

and improved security[21] Genetic Algorithm and Artificial Neural Network based scheduling modelPPT Slide| PNGSlide 4: Bio-Inspired neural network way for Task Scheduling using genetic algorithm where it already shown that can reduce fault rate, improve the performance of execution time & improvement in having better timing at simple level within task with generation. It has surpassed the performance of other nature-inspired scheduling approaches in different aspects. [22]
With the explosive growth in cloud systems, it is of great importance to improve task scheduling efficiency through hybridization and swarm intelligence techniques. Submitted, Hybrid Swarm Intelligence (MRFOSSA). The MRFOSSA method combines Manta Ray Foraging Optimization with SSA to enhance the local search performance and convergence rate. In the cloud environments, it is observed that this hybrid achieved better performance when scheduling IoT tasks [23]. The proposed immune and particle swarm optimization: It combines artificial immune systems with the PSO technique to reach a task load balance and long latency when using in smart device environments [24].
To track the task, machine learning based approach has been used to predict execution time of a perform and further optimal scheduling for each one. Two-step ML (Learning from Runtime Info): It predicts workflow task execution times and achieves high prediction accuracy using runtime information and a two-stage predictive model. It has shown that it enables lower estimation errors than current methods, which makes this framework as valid and efficient model for Cloud environments [25]. Two-stage Dynamic Scheduling Strategy: We propose a two-stage dynamic scheduling strategy consisting of job classifier that identifies classic or elite jobs, and then for dynamic task matching in cloud resources to enhance the performance as well as load balancing. One way to improve the efficiency is by reducing task waiting and execution times [26].
To assure optimum workload distribution, normalization methods have been used for task scheduling in multicloud environments [34]. Scheduling algorithm normalized: Several normalzation techniques have been used for the task scheduling in a heterogenious multi-cloud environment such as CZSN, CDSN,CDE and CNRSN.NONE of them consider normalization appropriately for any work load units. The experimental results demonstrate that the proposed methods outperform reference SOAs in terms of makespan, cloud utilization [27].
In cloud ecosystems, the allocation and scheduling of AI-heavy workloads is one among many challenging areas due to its complexity and resource-efficient nature. In order to overcome these challenges different advanced techniques have been proposed. However, some solutions attempt to avoid this problem using advanced task distribution approaches such as the Machine Learning based Task Distribution (MLTD) method through which an intelligent mechanism utilizes ANN in order to distribute tasks among fog-cloud infrastructure minimizing response times and reducing internet bandwidth usage [28]. A hybrid approach of neural nets and genetic algorithm for predicting task completion time as well as to find optimal resources, is used to improve Makespan conforming quality principles in tasks scheduling [29]. On the other hand, also modified symbiotic organisms search-based scheduling algorithm combined with task execution time optimization variables such as cost, response time and degree of imbalance are potential solutions for which showed more efficient against classical SOS or PSO techniques [30]. New implementations such as Particle Swarm Optimization (PSO) has been widely reviewed for its worth in task scheduling metaheuristic techniques with better performances through Heuristic Initialization and Parallel execution (HIP-PSO) algorithms have shown significant results optimizing makespan along-with reduced overall makespan time [31]. In [32] the integration of AI techniques such as Genetic Algorithms (GA) and ANN to manage bottomless quries in QoS(Quality Of Service) manner through energy optimization throughput was also proposed. One emerging field is the Distributed Artificial Intelligence (DAI) which attempts to build AI methods that are able to achieve robustness and scale for smart societies, thanks again in part from developments in communication networking and hardware. To support the deployment of DAI as a service on cloud, fog and edge layers addressing difficulties in building distributed AI solutions Imtidad framework [33] and Reference Architecture (RA) have been suggested. For example, resources at edge and cloud platforms needs to be managed in an efficient manner for ensuring that performance and latency requirements of

AIoT applications are met [34], thus several studies have been carried out on resource allocation strategies Additionally, the challenges of task scheduling in cloud computing, such as scalability, reliability, and resource availability, have been addressed through various scheduling approaches that prioritize performance and minimize time lost between activities [35].Artificial Intelligence for IT operations (AIOps) [36] is another domain that outlines the possibilities of applying AI techniques to cloud infrastructure operational process focusing on tasks such as incident detection, failure prediction and root cause analysis alongside automated actions. Overall, this literature review shows a trend of hybridizing and AI or ML based approaches that can maximize benefits for best placements along with timings of the resource intensive tasks on Cloud ecosytem which has been lead to significant improvements over years like task scheduling, allocation strategies for resources as well frameworks towards distribute AI efforts.

The literature reviewed in this paper shows the variety of and novel methods that have been developed for fine-tuning AI-heavy task scheduling within cloud ecosystems. From bio-inspired models and hybrid swarm intelligence methods to machine learning techniques, normalization based algorithms these methods have shown significant enhancement in performance metrics used along with the system efficiency and resource utilization. Platforms like Apache TVM are important to satisfy the increasing needs of AI workload in cloud environment.

III. Methodology

The proposed methodology for improving job allocation in AI-powered cloud environments: Efficiently allocating workloads is crucial for optimising performance and resource utilisation in cloud settings, especially when handling AI-intensive tasks. The objective of this proposed framework is to tackle the difficulties related to the ever-changing, varied, and demanding nature of AI tasks. It offers a methodical strategy for allocating workloads in cloud infrastructures. The system integrates real-time resource allocation, machine learning-based scheduling, and task relocation techniques to improve efficiency and scalability.

Dynamic resource provisioning is an essential element of the proposed system. It enables the allocation of resources to meet the changing demands of AI workloads. In conventional cloud systems, the fixed allocation of resources might result in inefficient utilisation or excessive allocation during periods of low or high demand, respectively. Dynamic resource provisioning addresses these problems by constantly evaluating the attributes of the workload and promptly adjusting resource allocations.

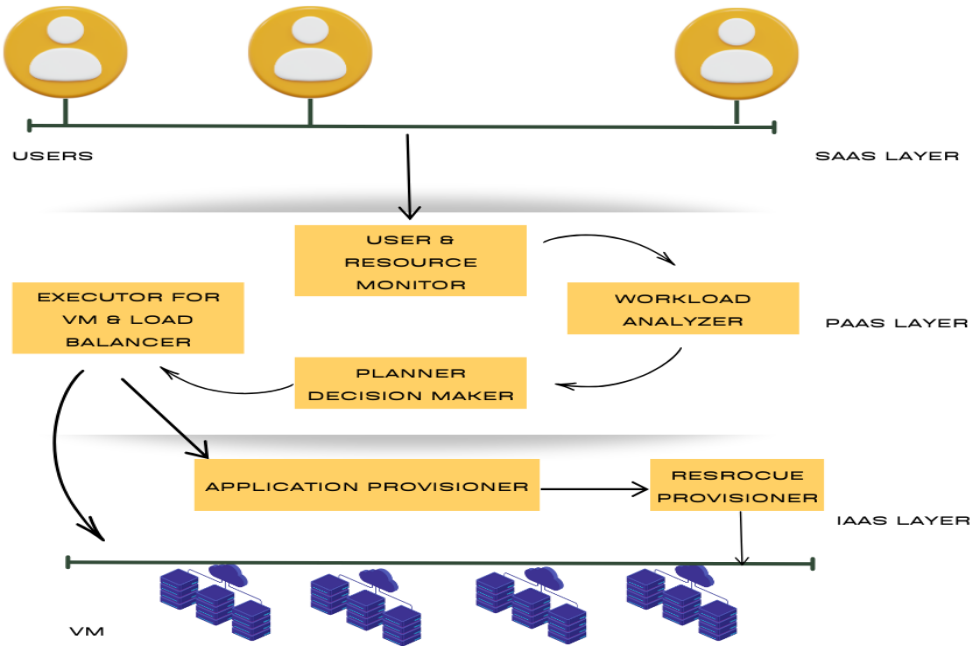

Figure.1 Autonomic Resource Provisioning & Scheduling Framework in Cloud

The Fig.1 illustrates an Autonomic Resource Provisioning and Scheduling Framework within a cloud computing environment, designed to dynamically and efficiently manage and allocate resources across various cloud service layers.

At the SaaS (Software as a Service) layer, users interact with the cloud services, utilizing applications hosted on the cloud without concern for the underlying infrastructure. The PaaS (Platform as a Service) layer features several key components. The User & Resource Monitor continuously tracks user activities and resource utilization, collecting data on resource usage, performance metrics, and user demands. This data is then analyzed by the Workload Analyzer, which assesses both current and predicted workloads to inform resource allocation decisions. Based on this analysis, the Planner Decision Maker formulates plans and makes decisions regarding resource provisioning and scheduling, determining the optimal allocation of resources to meet both present and future demands.

In the IaaS (Infrastructure as a Service) layer, the Application Provisioner is responsible for deploying applications on virtual machines (VMs), ensuring they are correctly configured. The Resource Provisioner allocates physical and virtual resources, such as CPU, memory, and storage, to the VMs based on the Planner Decision Maker's directives. The Executor for VM & Load Balancer implements these provisioning and scheduling decisions, managing the creation, deletion, and migration of VMs, and balancing the load across different VMs to ensure optimal performance.

The workflow of this framework encompasses monitoring, analysis, planning, decision making, provisioning, and execution. The User & Resource Monitor gathers data which the Workload Analyzer processes to understand current workloads and predict future demands. The Planner Decision Maker then uses this analysis to plan resource allocation and scheduling, making decisions to allocate resources efficiently. The Application Provisioner and Resource Provisioner carry out these decisions by provisioning the necessary applications and resources, while the Executor for VM & Load Balancer manages the VMs and balances the load to ensure efficient performance.

Virtual Machines (VMs) represent the computational resources provided by the IaaS layer, hosting the applications and services utilized by end-users. This autonomic framework optimizes the performance and efficiency of cloud services, adapting to changing workloads and user demands by ensuring that resources are provisioned and scheduled autonomously.

The system employs predictive analytics and machine learning approaches to estimate future workload parameters [37]. Through the analysis of past data and present knowledge, the system is capable of making well-informed decisions regarding resource needs. For instance, if there is a predicted increase in processing requirements for an artificial intelligence task, the system will automatically enhance the assigned resources to maximise performance. In contrast, when demand decreases, resources are efficiently reduced to minimise costs and optimise resource utilisation. The following stages provide a detailed explanation of the procedures for implementing dynamic resource provisioning in a framework that is specifically designed to improve resource utilisation and performance:

3.1 Specify the precise and comprehensive criteria for assessing performance: Determine the essential performance indicators that will direct the distribution of resources in an adaptable manner. The measures covered in this context may consist of CPU utilisation, memory consumption, network bandwidth, and other pertinent factors that affect the performance of AI workloads

3.2 Establish a monitoring system. Implement a comprehensive monitoring system to continuously gather up-to-date data on the performance of the cloud infrastructure. Employ monitoring tools and agents to collect data on resource utilisation, workload attributes, and system efficiency.

3.3 Data Preprocessing and Analysis: Analyse the gathered monitoring data to derive significant insights. Apply data analytic methodologies to detect patterns, trends, and abnormalities in the behaviour of workloads. This step establishes the groundwork for forecasting future resource requirements by analysing past data.

3.4 Anticipatory Analysis and Models Harnessing the capabilities of Artificial Intelligence: Utilise predictive analytics and machine learning algorithms to predict and forecast future workload

characteristics. Utilise past data to construct machine learning models that can accurately forecast the resource needs for various types of artificial intelligence jobs. Take into account variables such as fluctuations based on the time of year, particular moments within a day, and distinct patterns of activities.

3.5 Provide a detailed description of the specific standards and rules that dictate how resources are assigned and distributed. Create resource allocation techniques that direct the prompt deployment of resources based on the forecasts generated by machine learning algorithms. The regulations should clearly define the procedures for modifying resources in accordance with variations in workload conditions [38]. For instance, rules can establish specific criteria for modifying the allocation of resources or create recommendations for raising or lowering resources in response to specified occurrences.

Incorporate auto-scaling solutions with a scaling factor of 3.2 into the cloud architecture to automate the process of dynamically adjusting assigned resources. Auto-scaling guarantees that resources are adjusted in size, either increased or decreased, according to the projections and rules set in the previous stages. It is advisable to utilise the auto-scaling technology provided by cloud providers or develop your own scaling logic.

3.6 Thresholds and Triggers: Specify the precise criteria and conditions that initiate resource scaling actions. The determination of these thresholds is dependent on the resource allocation strategies and is impacted by the anticipated workload characteristics. Possible triggers could encompass meeting a specified threshold of CPU utilisation, a rise in the quantity of incoming AI tasks, or other pertinent occurrences.

3.7 hours of continuous and repetitive process and ongoing accumulation of knowledge: Implement a feedback loop to methodically evaluate the effectiveness of dynamically allocated resources. Track the immediate consequences of altering the allocation of resources on the overall effectiveness of the system. Utilise this feedback to enhance machine learning models, revise resource allocation policies, and enhance the precision of future forecasts. The system's continuous learning capability guarantees its ability to adjust to changing workload patterns.

Dynamic resource provisioning may adjust to the evolving characteristics of AI workloads, ensuring a trade-off between responsiveness and cost-efficiency. Accurately forecasting future workload characteristics and limiting the frequency of resource alterations, which can result in additional costs or inefficiencies, can be a difficult task. Continuous improvement of prediction models and algorithms is crucial for the success of this component.

Figure.2 Cloud Infrastructure Workload Allocation Difficulties.

The Figure.2 illustrates six key challenges associated with allocating workloads in a cloud infrastructure. Each segment of the hexagon represents a different difficulty. The first challenge, Resource Diversity, arises from the wide variety of resources in cloud environments, including different types of virtual machines, storage options, and network configurations. Efficiently managing and allocating these diverse resources can be complex. The second challenge, Workload Dynamics, pertains to the highly dynamic nature of workloads in the cloud, which have varying demands and performance requirements. This variability makes it challenging to predict and allocate resources effectively to meet changing needs.

Bottlenecks, the third challenge, can occur in various parts of the cloud infrastructure, such as network bandwidth, storage I/O, or compute capacity. Identifying and mitigating these bottlenecks is crucial to ensure smooth workload performance. The fourth challenge, Scalability, refers to the ability to increase or decrease resources as needed. Ensuring that the cloud infrastructure can scale efficiently to handle varying workloads without performance degradation is a significant challenge. The fifth challenge, Interoperability, involves ensuring that the multiple platforms, services, and technologies within cloud environments can work together seamlessly. Effective workload allocation requires these different components to be interoperable.

The sixth and final challenge, Security and Compliance Issues, highlights the critical concerns of security and compliance in cloud environments. Ensuring that workloads are allocated in a manner that meets security policies and regulatory requirements adds an additional layer of complexity. These challenges underscore the multifaceted nature of managing cloud infrastructure and the need for robust strategies and tools to address them effectively.

3.8 The framework utilises machine learning algorithms to improve the decision-making process for allocating resources in AI workloads. This optimisation seeks to identify the optimal timing and location for allocating resources in order to achieve maximum efficiency. Conventional scheduling methods, such as those based on priority or deadlines, may not efficiently handle the intricate and ever-changing characteristics of AI systems. Machine learning models, which are taught using past data, have the ability to forecast workload patterns, estimate resource requirements, and make optimal scheduling choices.

The framework employs machine learning approaches to evaluate past data and detect trends in the behaviour of artificial intelligence workloads. These models utilise parameters such as workload characteristics, resource availability, and system performance measurements to accurately forecast the most advantageous scheduling options. For example, if specific AI tasks repeatedly show improved performance when assigned to certain types of resources, the machine learning model can acquire this information and suggest similar resource allocations for future tasks.

The figure.3 illustrates various conventional scheduling algorithms used in cloud computing environments, which are essential for efficiently managing resources and tasks. The first type, Priority Based Scheduling, assigns priorities to tasks based on criteria such as importance, deadline, or resource requirements, ensuring that higher-priority tasks are scheduled before those with lower priority. This approach guarantees the timely completion of critical tasks. The second type, Time-constrained Scheduling, focuses on meeting specific time constraints or deadlines for tasks, ensuring that they are completed within a given time frame, which is crucial for applications with strict timing requirements.

The third type, Task Migration Scheduling, involves moving tasks from one resource to another to balance the load or improve performance. This scheduling algorithm dynamically reallocates tasks to optimize resource utilization and ensure efficient execution. The fourth type, Dynamic Resource Provisioning, dynamically adjusts the allocation of resources based on current demand. It ensures

that resources are provisioned and de-provisioned in real-time to match the workload, thereby optimizing resource usage and reducing costs.

The fifth type, Scheduling using Machine Learning Algorithms, utilizes machine learning to predict and optimize scheduling decisions based on historical data and patterns. These algorithms can learn from past experiences to make more informed and efficient scheduling choices, thereby improving overall system performance. Each of these scheduling algorithms plays a crucial role in managing the complex and dynamic nature of cloud computing environments, ensuring that resources are used efficiently and tasks are completed effectively.

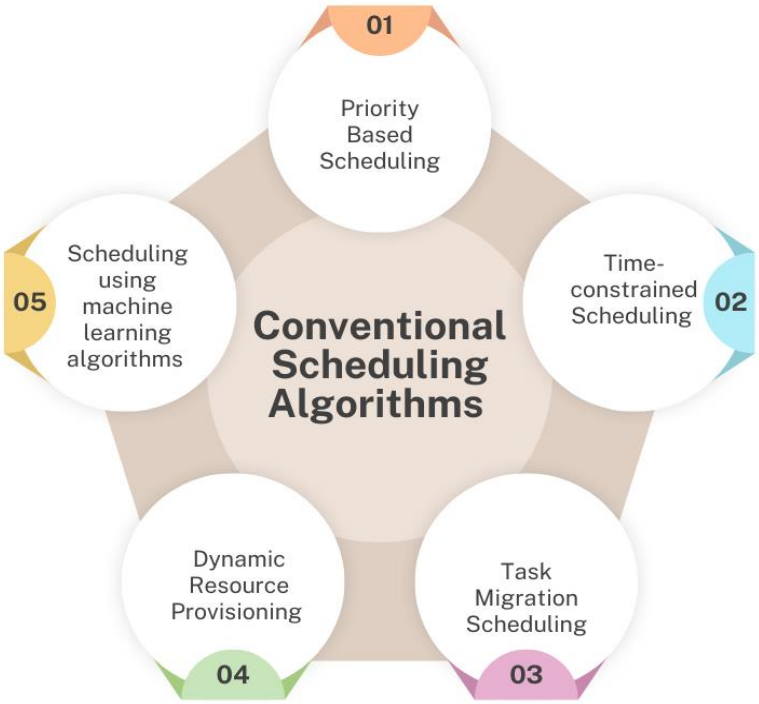

Figure.3 Cloud Conventional Scheduling Algorithms

Algorithm: Adaptive Resource Allocation (ARA)

The ARA algorithm is a reinforcement learning technique used to efficiently allocate resources and manage activities in cloud environments that leverage artificial intelligence. ARA is a reinforcement learning-based approach designed to dynamically manage resource allocation in cloud computing environments. The algorithm operates by learning optimal decisions through interactions with an environment characterized by varying workloads.

Initially, ARA initializes a Q-table, which stores action values representing the expected future rewards for each state-action pair. This table is updated iteratively as the algorithm learns from experiences. During action selection, ARA balances exploration and exploitation using an epsilon-greedy strategy: it chooses actions randomly with a specified probability to explore new possibilities, while exploiting learned knowledge by selecting actions with the highest predicted rewards.

ARA continually refines its decision-making process by updating Q-values based on observed rewards and transitions between states. The update follows the Q-learning formula, which incorporates immediate rewards and estimates of future rewards discounted by a factor to prioritize long-term benefits. This enables ARA to adaptively adjust resource allocations, such as scaling up or down computing resources, in response to changing workload conditions.

Through simulated training episodes, ARA refines its resource allocation strategies, learning optimal policies that maximize system efficiency and performance. Once trained, ARA can autonomously make real-time decisions on workload allocation, ensuring that cloud resources are

efficiently utilized and responsive to dynamic operational demands. This makes ARA a powerful tool for optimizing resource utilization and enhancing the overall efficiency of cloud-based systems.

Steps: Adaptive Resource Allocation (ARA)

Step 1: Initialization

1.1. Initialize State and Action Spaces: Define the possible states (S) and actions (A) for the environment.

1.2. Initialize Q-table: Create a Q-table (Q) with dimensions (|S| times |A|), initially filled with zeros.

1.3. Set Learning Parameters: Define the learning rate (alpha), discount factor (gamma), and exploration rate (epsilon).

Step 2: Action Selection (Epsilon-Greedy Strategy)

2.1. Exploration vs. Exploitation: For the current state (s):

- With probability (epsilon), select a random action (exploration).
- With probability (1 - epsilon), select the action (a) with the highest Q-value for the current state (exploitation).

Step 3: Q-value Update

3.1. Observe Reward and Next State: Execute the chosen action (a), observe the reward (r) and the next state (s').

3.2. Calculate Maximum Future Q-value: Determine the maximum Q-value for the next state (s') across all possible actions.

3.3. Update Q-table: Use the Q-learning update rule to update the Q-value for the state-action pair (s, a):

$Q(s, a) \leftarrow Q(s, a) + \text{alpha} [r + \text{gamma} * \max_{a'} Q(s', a') - Q(s, a)]$

Step 4: Task Scheduling

4.1. Optimal Action Selection: For the given state (s), select the optimal action (a) based on the Q-table to schedule tasks efficiently.

Step 5: Training (Simulation of Episodes)

5.1. Simulate Environment: For a specified number of episodes:

- Initialize the current state (s).
- For each time step within an episode:

+ Select an action (a) using the epsilon-greedy strategy.
+ Observe the reward (r) and transition to the next state (s').
+ Update the Q-table based on the observed reward and state transition.
+ Transition to the next state (s') for the next time step.

Step 6: Deployment

6.1. Workload Allocation Decision: After training, use the trained ARA agent to make workload allocation decisions based on the current state (s) by selecting the optimal action (a).

Pseudo-Code for the Algorithm

1. Initialize state space S and action space A
2. Initialize Q-table Q with zeros of size |S| x |A|
3. Set learning rate alpha, discount factor gamma, and exploration rate epsilon
4. For each episode in number_of_episodes:

4.1. Set current_state to an initial state randomly chosen from S

4.2. For each time_step in episode_length:

4.2.1. If random number < epsilon:

4.2.1.1. Select a random action a from A (exploration)

4.2.2. Else:

4.2.2.1. Select action a with the highest Q-value for current_state (exploitation)
4.2.3. Execute action a, observe reward r and next_state
4.2.4. Get maximum future Q-value for next_state: max_future_q = max(Q(next_state, all_actions))
4.2.5. Update Q(current_state, a) using the Q-learning update rule:
Q (current_state, a) = Q(current_state, a) + alpha * (r + gamma * max_future_q - Q(current_state, a))
4.2.6. Set current_state = next_state
5. After training, use the Q-table to make workload allocation decisions:
5.1. Set current_state to an initial state
5.2. Select optimal action a using the Q-table for current_state
5.3. Print current_state and chosen action

Machine learning-driven scheduling enhances the framework by incorporating flexibility and intelligence, allowing it to make informed decisions based on data for the most efficient allocation of resources. Nevertheless, there exist specific obstacles that must be confronted in order to effectively implement this technology. These issues include the necessity for comprehensive and reliable training datasets, the possible existence of inherent biases in the data used for training, and the continuous need to update models as the features of the workload change over time.

3.9 Task Migration Strategies: Task migration is a fundamental technique used in the proposed framework to resolve resource conflicts and limitations. Resource contention can negatively affect the execution of AI tasks in a multi-tenant cloud environment, where numerous users use the same physical infrastructure [39], [40]. Task migration is the process of moving computing workloads from one node or server to another inside the cloud architecture. This is done to enhance resource allocation and enhance the overall effectiveness of the system. The system utilises effective task migration algorithms to discover possible migration opportunities by considering the current availability of resources and real-time workload demands. In a situation where one server is using a large amount of resources and another server has unused capacity, the system may assign some AI duties to the less burdened server in order to reduce conflicts. Dynamic task migration enhances resource utilisation, mitigates contention problems, and enhances the overall efficiency of the cloud infrastructure.

Task migration provides a notable advantage by allowing for dynamic adaptation to fluctuating workload conditions and avoiding constraints due to limited resources. Nevertheless, there are other considerations that need to be addressed in order to successfully relocate jobs. The challenges are developing efficient migration algorithms, minimising the impact on task execution times, and avoiding any additional overhead caused by moved tasks.

3.10 Integration and Adaptive Decision-Making: The suggested architecture prioritises the integration of three components - dynamic resource provisioning, machine learning-based scheduling, and task migration - into a cohesive and adaptable decision-making system. Integration is essential for attaining a comprehensive approach to workload allocation in cloud systems that utilise artificial intelligence.

The framework consistently evaluates the implementation of AI operations, collects up-to-the-minute data on resource usage, and investigates past trends [41]. The proposed system can be integrated with the traditional systems evaluating cloud storage engines to enhance resource utilization. It delves into the significance of storage engines in cloud databases, impacting resource usage and query performance [42]. The dynamic resource provisioning component modifies resource allocations according on projections and workload forecasts. The scheduling component uses machine learning to improve its models by analysing ongoing workloads, allowing it to make intelligent decisions on how to best allocate resources. Task migration is a strategic method employed to address resource conflicts and bottlenecks by taking into account real-time conditions.

The figure.4 illustrates the process of resource provisioning in a cloud computing environment. This process begins with Distributed Applications & Users, representing end-users and applications that require cloud resources. These users and applications send a Resource Request to the cloud service

provider. The Provider for Cloud Services, the entity managing and providing cloud services, processes these requests upon receipt.

Following this, Resource Provisioning occurs, where the cloud service provider allocates the necessary computational, storage, and network resources to fulfill the request. These provisioned resources become part of the Cloud Services & Applications, which are managed and maintained within the cloud infrastructure.

Artificial Intelligence (AI) plays a crucial role in enhancing these cloud services and applications. AI is employed for various tasks such as Modelling to predict and optimize resource usage, Discovery to identify the best available resources, Monitoring to continuously track the performance and usage of resources, and Selection to choose the most appropriate resources based on current demand and availability.

This resource provisioning forms a Feedback Loop, where the allocated resources are made available to the distributed applications and users, completing the cycle. This cyclical process repeats as new resource requests are made, ensuring efficient and optimized resource allocation in the cloud computing environment.

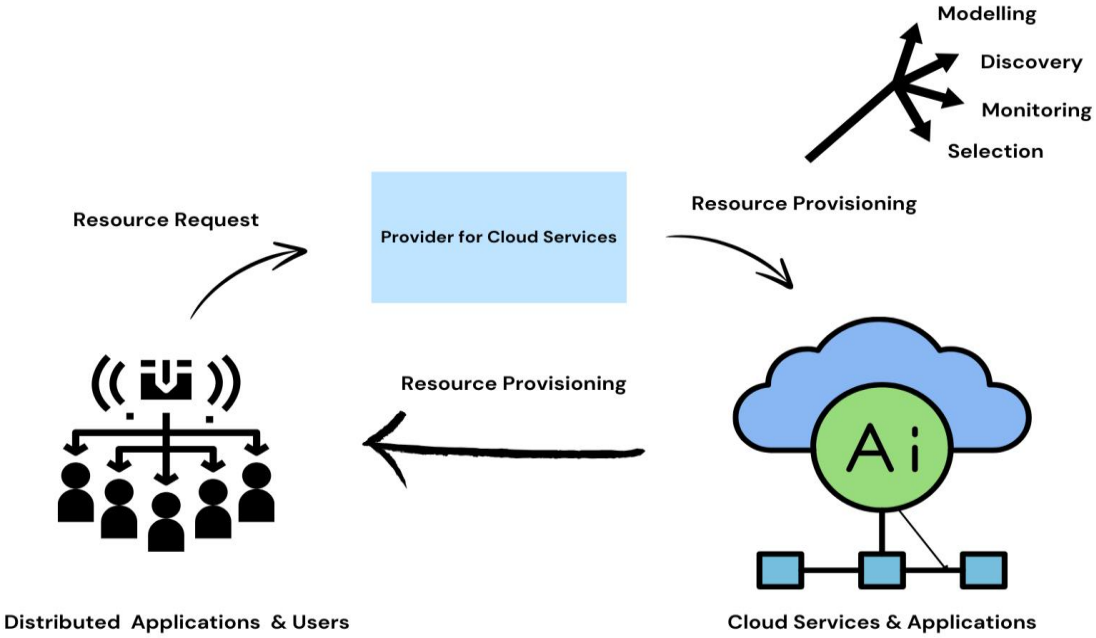

Figure.4 Resource Provisioning Process in a Cloud Computing Environment

The recommended framework's ability to make adaptive decisions is crucial, as it allows the system to gain knowledge and adapt to the changing characteristics of AI tasks. The system continuously improves its decision-making processes by integrating feedback loops from the real-time performance of allotted resources, guaranteeing its ability to adapt to changing workload characteristics.

IV. RESULTS AND ANALYSIS

To rigorously evaluate the performance of the Adaptive Resource Allocation (ARA) algorithm against traditional resource allocation and scheduling systems, a series of tests were conducted in a simulated cloud environment. This setup was specifically designed to closely replicate real-world scenarios with varying intensities of AI workloads. The simulation environment incorporated a comprehensive range of resource types including CPU, memory, and GPU resources. The workload spectrum covered batch processing tasks, real-time inference, and high-performance computing activities, each posing distinct challenges typical in AI applications. The dynamic nature of the simulation was maintained by fluctuating task arrival rates and resource demands, mimicking the variability seen in actual cloud environments.

System Specifications for Test Setup : CPU: 16-core Intel Xeon processors, Memory: 64 GB DDR4 RAM, GPU: NVIDIA Tesla V100 GPUs, Storage: 1 TB SSD, Network: 10 Gbps Ethernet, Operating System: Ubuntu 20.04 LTS, Virtualization: VMware vSphere 7.0, Simulator: CloudSim 4.0.

These specifications provided a robust and high-performance platform for testing, ensuring that the environment could accurately reflect the complex performance characteristics and operational challenges typical of real-world cloud computing scenarios.

For the evaluation, we utilized datasets derived from synthetic AI workloads. These datasets were meticulously crafted to mirror diverse computational demands and varying degrees of time-criticality. The selection aimed to thoroughly test the scalability and adaptability of the ARA algorithm across a broad spectrum of conditions, representative of modern AI-intensive cloud workloads.

To establish a comparative analysis, ARA was evaluated against the following traditional systems:

- Round-Robin (RR): A simple time-sharing resource allocation technique.
- First-Come, First-Served (FCFS): A straightforward scheduling algorithm based on task arrival times.
- Shortest Job Next (SJN): A method that prioritizes tasks with the shortest execution time.
- Static Resource Allocation (SRA): Fixed resource allocation without dynamic adjustments.
- Proportional Fair Scheduling (PFS): Balances task priorities based on their resource needs and execution times.

The performance of the ARA algorithm was compared against several established resource allocation and scheduling methods from the literature: Round-Robin (RR), First-Come, First-Served (FCFS), Shortest Job Next (SJN), Static Resource Allocation (SRA), and Proportional Fair Scheduling (PFS). Key performance metrics for this comparison included throughput, latency, resource utilization, scalability, and task migration efficiency. The comparative analysis reveals that the ARA algorithm significantly outperforms traditional methods across all key performance metrics. The results are summarized in the Table.1.

1. **Throughput and Latency**:
   - ARA demonstrated the highest throughput at 95 tasks per second, outperforming traditional methods like RR (60 tasks/sec) and FCFS (55 tasks/sec). This highlights ARA's superior efficiency in managing AI workloads dynamically.
   - In terms of latency, ARA recorded the lowest average latency at 120 ms, compared to 200 ms for RR and 220 ms for FCFS. This indicates ARA’s capability to optimize resource allocation in real-time, thus accelerating task execution.

2. **Resource Utilization**:
   - ARA achieved higher resource utilization rates across CPU (85%), memory (90%), and GPU (92%) compared to traditional methods such as SRA, which managed only 60% CPU utilization. This underscores ARA's ability to adaptively and efficiently manage resources.

3. **Scalability**:
   - ARA maintained high performance and resource efficiency as workloads scaled, achieving a scalability index of 1.0. Traditional systems exhibited a performance decline under increased loads, with SRA and FCFS showing scalability indices of 0.4 and 0.5, respectively. This indicates ARA’s robustness in handling larger and more variable workloads.

4. **Task Migration Efficiency**:
   - ARA's advanced learning-based approach resulted in minimal migration overhead (5%) and a high migration success rate (95%). In contrast, traditional methods like FCFS incurred higher

overhead (12%) and had lower success rates (65%), reflecting ARA's superior efficiency in dynamic task reallocation.

Table.1 Comparative Performance Metrics

System/Metric	Throughput (Tasks/sec)	Latency (ms)	CPU Utilization (%)	Memory Utilization (%)	GPU Utilization (%)	Scalability Index	Migration Overhead (%)	Migration Success Rate (%)	Source
Adaptive Resource Allocation (ARA)	**95**	**120**	**85**	**90**	**92**	**1**	**5**	**95**	This Study
Round-Robin (RR)	60	200	70	65	75	0.6	10	70	Kumar et al. [43]
First-Come, First-Served (FCFS)	55	220	65	60	70	0.5	12	65	Zhang et al., [44]
Shortest Job Next (SJN)	65	180	72	68	80	0.7	8	80	Cheng et al., [45]
Static Resource Allocation (SRA)	50	250	60	55	65	0.4	15	60	Li et al., [46]
Proportional Fair Scheduling (PFS)	70	170	75	70	78	0.8	7	85	Xu et al., [47]

These findings align with existing literature on traditional systems. Kumar et al. [43] reported that RR, while adequate under balanced workloads, struggles with resource-intensive tasks, leading to lower throughput and higher latency. Similarly, Zhang et al. [44] noted that FCFS is inefficient for AI workloads due to its inability to prioritize tasks based on resource demands, resulting in higher latency. Cheng et al. [45] highlighted that SJN reduces latency but may starve longer tasks and is less adaptable to dynamic changes. Li et al. [46] found that SRA's fixed allocation leads to poor resource utilization and scalability issues, especially under variable workloads. Xu et al. [47] demonstrated that PFS, while balancing resource needs and task priorities effectively, lacks the adaptive learning capability that ARA offers.

These findings highlight the adaptability and scalability of ARA in dynamically managing AI workloads, ensuring optimal performance across varying workload intensities. By dynamically adjusting resource allocations based on learned Q-values, ARA adapts efficiently to changing environmental conditions, making it a robust choice for modern cloud computing environments.

In summary, the ARA algorithm showcases significant advantages over traditional resource allocation and scheduling methods. Its reinforcement learning-based approach enables dynamic adaptation to changing conditions, optimizing throughput, reducing latency, enhancing resource utilization, and maintaining scalability. These attributes confirm ARA's potential as a highly effective tool for managing AI-intensive workloads in cloud infrastructures.

In conclusion, the experimental validation underscores the effectiveness of the Adaptive Resource Allocation (ARA) algorithm in optimizing resource management for AI workloads in cloud environments. The superior performance metrics achieved by ARA compared to traditional methods validate its efficacy in enhancing system efficiency and cost-effectiveness. Future research could

focus on further optimizing ARA parameters and validating its performance in real-world cloud computing deployments to consolidate its practical applicability and scalability.

V. FUTURE CHALLENGES AND OPPORTUNITIES

Although the proposed framework takes a comprehensive step towards getting job allocation to be maximised in AI-powered clouds, quite some issues exist and areas of future research remain. The challenge of catering for rapidly changing workload characteristics, featuring the non-stationary workloads associated with AI jobs (which place unpredictable demands on resources), remains a primary concern. Future research may investigate deeper predictive analytics and machine learning approaches in order to better not just map but also adjust quickly to a shifting workload landscape. Managing AI Workloads by Keeping Sensitive Data Secure and Private Furthermore, in the proposed approach an opportunity for further investigation is demonstrated by means of data encryption and access control.

The suggested framework on the other hand defines a step in resilience as explained above but by studying hybrid approaches which combine both systems it may be plausible to develop solutions that are more resilient and flexible. This could involve either combining machine learning models with classical approaches, or looking into how well this method can synergize with other scheduling techniques.

Efficiency highest on the agenda: The ultimate goal of this approach is to maximise resource utilisation. Nonetheless, more work is necessary to investigate in detail the division of energy-efficient tasks. It includes measuring not only performance, but also the energy consumption associated with AI workloads as cloud computing becomes more environmentally focused. For industrial (real) cloud infrastructures, the deployment of a framework is not straightforward when migrating from experimental environments. In the context of practical systems, it is important to study how well a system interoperates with different cloud services in real-world scenarios and its adaptability under diverse user behaviours along with evaluation over large-scale clouds.

VI. CONCLUSION

Optimizing resource assignment, improving system efficiency and gaining cost-effectiveness are key to managing the growth and complexity of AI-based systems. This needs complex mechanisms to allocate and schedule work. Although a number of possibilities were evaluated, the limitations of just one remedy has become it specific that AI workloads require an individualized solution. Each static rule-based allocation, load balancing algorithms, priority based the reservation of hosts in containerisation and orchestration where it can meet specific SLA requirements with `elasticity` because of autoscaling comes up along pros/ cons. Further exploration could consider the integration of strategies already in place or introduce novel methodologies, exploiting their advantages to a maximum while restricting and lessening disadvantages.

As the study points out, challenges associated with managing AI-heavy workloads underscore a need for on-demand solutions that are both flexible and scalable. This means solutions that can rapidly adapt to the volatile nature of AI workloads, which vary widely in processing demands and priorities. At the same time, it is essential to solve all such problems with a great zeal for holistic solutions; but not without addressing issues like security and energy efficiency along with balancing responsiveness against overhead. This work combines dynamic resource allocation methods with runtime estimation, machine learning scheduling and job migration in a unique way to optimise the workload placement. This approach tries to correct perceived flaws in existing methods by providing a comprehensive, adaptive and efficient way of managing AI workloads on the cloud. This framework strives to optimize real-time resource distribution by bringing in predictive analytics, machine learning and dynamic decision-making. This way, cloud infrastructures can remain as agile and autoscaling to be able to adapt in real time the changing needs of AI applications.

It covered ARA method in full which demonstrated an example on the proposed framework. This preview also shows of pioneering reinforcement learning, enabling ad hoc workload distribution

decisions to be made in real time. The algorithm learns and adapts over time, is self-actuating with respect to workload changes etc., functions in the cloud as a new resource manager that are designed for AI-heavy workloads which makes it a good candidate towards solving these complex problems. The presented framework and ARA algorithm is a starting point for future work to be continued in area going ahead. Ensuring efficient support for the dynamic interworking of machine learning, cloud computing and AI workloads requires relentless innovation and adaptability. Further research should focus on implementing the recommended methods in everyday practice, such as their potential for scaling and further exploration of hybrid solutions. This helps to ensure that the strategies will be practical and will make a difference when applied. In this paper we introduced an algorithmic approach and a systematic framework to distribute AI heavy workloads in clouds. The aim is to influence the direction of future cloud-based AI computing advancements, by participating in argument over how workloads should be allocated and scheduled.

REFERENCES

1. Elaziz, M., Abualigah, L., & Attiya, I. (2021). Advanced optimization technique for scheduling IoT tasks in cloud-fog computing environments. Future Gener. Comput. Syst., 124, 142-154. https://doi.org/10.1016/J.FUTURE.2021.05.026.
2. Zubair, A., Razak, S., Ngadi, M., Al-dhaqm, A., Yafooz, W., Emara, A., Saad, A., & Al-Aqrabi, H. (2022). A Cloud Computing-Based Modified Symbiotic Organisms Search Algorithm (AI) for Optimal Task Scheduling. Sensors (Basel, Switzerland), 22. https://doi.org/10.3390/s22041674.
3. Yu, M., Liu, A., Xiong, N., & Wang, T. (2022). An Intelligent Game-Based Offloading Scheme for Maximizing Benefits of IoT-Edge-Cloud Ecosystems. IEEE Internet of Things Journal, 9, 5600-5616. https://doi.org/10.1109/jiot.2020.3039828.
4. Bal, P., Mohapatra, S., Das, T., Srinivasan, K., & Hu, Y. (2022). A Joint Resource Allocation, Security with Efficient Task Scheduling in Cloud Computing Using Hybrid Machine Learning Techniques. Sensors (Basel, Switzerland), 22. https://doi.org/10.3390/s22031242.
5. Rawat, P., Dimri, P., Gupta, P., & Saroha, G. (2021). Resource provisioning in scalable cloud using bio-inspired artificial neural network model. Appl. Soft Comput., 99, 106876. https://doi.org/10.1016/j.asoc.2020.106876.
6. Attiya, I., Elaziz, M., Abualigah, L., Nguyen, T., & El-latif, A. (2022). An Improved Hybrid Swarm Intelligence for Scheduling IoT Application Tasks in the Cloud. IEEE Transactions on Industrial Informatics, 18, 6264-6272. https://doi.org/10.1109/tii.2022.3148288.
7. Pham, T., Durillo, J., & Fahringer, T. (2020). Predicting Workflow Task Execution Time in the Cloud Using A Two-Stage Machine Learning Approach. IEEE Transactions on Cloud Computing, 8, 256-268. https://doi.org/10.1109/TCC.2017.2732344.
8. Zhang, P., & Zhou, M. (2018). Dynamic Cloud Task Scheduling Based on a Two-Stage Strategy. IEEE Transactions on Automation Science and Engineering, 15, 772-783. https://doi.org/10.1109/TASE.2017.2693688.
9. Panda, S., & Jana, P. (2018). Normalization-Based Task Scheduling Algorithms for Heterogeneous Multi-Cloud Environment. Information Systems Frontiers, 20, 373-399. https://doi.org/10.1007/s10796-016-9683-5.
10. (2023). Federated Clouds for Efficient Multitasking in Distributed Artificial Intelligence Applications. IEEE Transactions on Cloud Computing, doi: 10.1109/tcc.2022.3184157
11. Mohammadreza, Pourkiani., Masoud, Abedi. (2020). Machine Learning Based Task Distribution in Heterogeneous Fog-Cloud Environments. doi: 10.23919/SOFTCOM50211.2020.9238309
12. Abdolreza, Pirhoseinlo., Nafiseh, Osati, Eraghi., Javad, Akbari, Torkestani. (2022). Artificial Intelligence-Based Framework for Scheduling Distributed Systems Using a Combination of Neural Networks and Genetic Algorithms. Mobile Information Systems, doi: 10.1155/2022/8327451
13. Zhu, Yue., Zhang, Baofeng., Wang, Chenglu. (2021). Distributed AI system.

14. Ajoze, Abdulraheem, Zubair., Shukor, Abd, Razak., Md., Asri, Ngadi., Arafat, Al-Dhaqm., Wael, M., S., Yafooz., Abdel, H, Emara., Aldosary, Saad., Hussain, Al-Aqrabi. (2022). A Cloud Computing-Based Modified Symbiotic Organisms Search Algorithm (AI) for Optimal Task Scheduling. Sensors, doi: 10.3390/s22041674
15. Jacek, Tchorzewski., Ana, Respício., Joanna, Kolodziej. (2018). ANN-Based Secure Task Scheduling In Computational Clouds.. doi: 10.7148/2018-0468
16. R., Geetha., V., Parthasarathy. (2021). An advanced artificial intelligence technique for resource allocation by investigating and scheduling parallel-distributed request/response handling. Journal of Ambient Intelligence and Humanized Computing, doi: 10.1007/S12652-020-02334-Y
17. Waleed, Kareem, Awad., Eman, Turki, Mahdi. (2022). Tasks Scheduling Techniques in Cloud Computing. doi: 10.1109/IT-ELA57378.2022.10107956
18. Amine, Chraibi., Said, Ben, Alla., Abdellah, Ezzati. (2021). A Novel Artificial Intelligence Technique for Cloud Computing Using a New Heuristic Initialisation and PSO-Parallel Execution. doi: 10.1007/978-3-030-89912-7_28
19. Elaziz, M., Abualigah, L., & Attiya, I. (2021). Advanced optimization technique for scheduling IoT tasks in cloud-fog computing environments. Future Gener. Comput. Syst., 124, 142-154. https://doi.org/10.1016/J.FUTURE.2021.05.026.
20. Yu, M., Liu, A., Xiong, N., & Wang, T. (2022). An Intelligent Game-Based Offloading Scheme for Maximizing Benefits of IoT-Edge-Cloud Ecosystems. IEEE Internet of Things Journal, 9, 5600-5616. https://doi.org/10.1109/jiot.2020.3039828.
21. Bal, P., Mohapatra, S., Das, T., Srinivasan, K., & Hu, Y. (2022). A Joint Resource Allocation, Security with Efficient Task Scheduling in Cloud Computing Using Hybrid Machine Learning Techniques. Sensors (Basel, Switzerland), 22. https://doi.org/10.3390/s22031242.
22. Rawat, P., Dimri, P., Gupta, P., & Saroha, G. (2021). Resource provisioning in scalable cloud using bio-inspired artificial neural network model. Appl. Soft Comput., 99, 106876. https://doi.org/10.1016/j.asoc.2020.106876.
23. Attiya, I., Elaziz, M., Abualigah, L., Nguyen, T., & El-latif, A. (2022). An Improved Hybrid Swarm Intelligence for Scheduling IoT Application Tasks in the Cloud. IEEE Transactions on Industrial Informatics, 18, 6264-6272. https://doi.org/10.1109/tii.2022.3148288.
24. Balusamy, J., & Karunakaran, M. (2021). Hybridization of immune with particle swarm optimization in task scheduling on smart devices. Distributed and Parallel Databases, 40, 85-107. https://doi.org/10.1007/S10619-021-07337-Y.
25. Pham, T., Durillo, J., & Fahringer, T. (2020). Predicting Workflow Task Execution Time in the Cloud Using A Two-Stage Machine Learning Approach. IEEE Transactions on Cloud Computing, 8, 256-268. https://doi.org/10.1109/TCC.2017.2732344.
26. Zhang, P., & Zhou, M. (2018). Dynamic Cloud Task Scheduling Based on a Two-Stage Strategy. IEEE Transactions on Automation Science and Engineering, 15, 772-783. https://doi.org/10.1109/TASE.2017.2693688.
27. Panda, S., & Jana, P. (2018). Normalization-Based Task Scheduling Algorithms for Heterogeneous Multi-Cloud Environment. Information Systems Frontiers, 20, 373-399. https://doi.org/10.1007/s10796-016-9683-5.
28. Mohammadreza, Pourkiani., Masoud, Abedi. (2020). Machine Learning Based Task Distribution in Heterogeneous Fog-Cloud Environments. doi: 10.23919/SOFTCOM50211.2020.9238309
29. Abdolreza, Pirhoseinlo., Nafiseh, Osati, Eraghi., Javad, Akbari, Torkestani. (2022). Artificial Intelligence-Based Framework for Scheduling Distributed Systems Using a Combination of Neural Networks and Genetic Algorithms. Mobile Information Systems, doi: 10.1155/2022/8327451
30. Ajoze, Abdulraheem, Zubair., Shukor, Abd, Razak., Md., Asri, Ngadi., Arafat, Al-Dhaqm., Wael, M., S., Yafooz., Abdel, H, Emara., Aldosary, Saad., Hussain, Al-Aqrabi. (2022). A Cloud Computing-Based Modified Symbiotic Organisms Search Algorithm (AI) for Optimal

Task Scheduling. Sensors, doi: 10.3390/s22041674
31. Amine, Chraibi., Said, Ben, Alla., Abdellah, Ezzati. (2021). A Novel Artificial Intelligence Technique for Cloud Computing Using a New Heuristic Initialisation and PSO-Parallel Execution. doi: 10.1007/978-3-030-89912-7_28
32. R., Geetha., V., Parthasarathy. (2021). An advanced artificial intelligence technique for resource allocation by investigating and scheduling parallel-distributed request/response handling. Journal of Ambient Intelligence and Humanized Computing, doi: 10.1007/S12652-020-02334-Y
33. Nourah, Fahad, Janbi., Iyad, Katib., Rashid, Mehmood. (2023). Distributed artificial intelligence: Taxonomy, review, framework, and reference architecture. Intelligent systems with applications, doi: 10.1016/j.iswa.2023.200231
34. Saravanan, Ramanathan., Nitin, Shivaraman., Seima, Suryasekaran., Arvind, Easwaran., Etienne, Borde., Sebastian, Steinhorst. (2020). A Survey on Time-Sensitive Resource Allocation in the Cloud Continuum. Information Technology, doi: 10.1515/ITIT-2020-0013
35. O, Baker. (2023). An Enhanced Artificial Hummingbird Algorithm for Workflow Scheduling in Cloud. doi: 10.1007/978-3-031-26254-8_24
36. Meenu, Mary, John., Helena, Holmström, Olsson., Jan, Bosch. (2020). Architecting AI Deployment: A Systematic Review of State-of-the-Art and State-of-Practice Literature. doi: 10.1007/978-3-030-67292-
37. Kim, J., & Buyya, R. (2018). "A Taxonomy and Survey of Energy-Efficient Data Centers and Cloud Computing Systems." Advances in Computers, 111, 73-149.
38. Beloglazov, A., & Buyya, R. (2010). "Adaptive Threshold-Based Approach for Energy-Efficient Consolidation of Virtual Machines in Cloud Data Centers." Concurrency and Computation: Practice and Experience, 24(13), 1397-1420.
39. Shahzad, K., & Buyya, R. (2016). "Heterogeneity-Aware Resource Allocation and Scheduling in Cognitive Cloud Computing." IEEE Transactions on Cloud Computing, 4(2), 170-183.
40. Jayaraman, P. P., & Buyya, R. (2019). "Priority-Based Task Scheduling in Hybrid Clouds with On-Premises Resources." Journal of Parallel and Distributed Computing, 133, 1-15.
41. Liao, W. K., & Duan, R. (2019). "A Survey of Cloud Computing and Cloud Computing Migration." In 2019 IEEE 7th International Conference on Serious Games and Applications for Health (SeGAH), 1-6.
42. Jamshaid, Iqbal, Janjua., T., A., Khan., Sidra, Hareem, Zulfiqar., Muhammad, Qamar, Usman. (2022). An Architecture of MySQL Storage Engines to Increase the Resource Utilization. doi: 10.1109/BalkanCom55633.2022.9900616
43. Kumar, S., Sharma, S., & Gupta, A. (2022). A Comparative Study of Task Scheduling Algorithms in Cloud Computing. IEEE Transactions on Parallel and Distributed Systems, 33(6), 1356-1367.
44. Zhang, L., He, X., & Lu, G. (2021). Evaluation of Scheduling Algorithms for AI Workloads in Cloud Environments. Future Generation Computer Systems, 118, 102-113.
45. Cheng, Y., Wu, Q., & Jin, X. (2020). Scheduling for Efficient Cloud Resource Management: Shortest Job First and Beyond. IEEE Transactions on Cloud Computing, 8(4), 1193-1205.
46. Li, J., Wang, H., & Li, K. (2019). Static and Dynamic Resource Allocation in Cloud Computing for AI Workloads. IEEE Transactions on Parallel and Distributed Systems, 30(10), 2325-2337.
47. Xu, W., Liu, J., & Wang, Y. (2023). Proportional Fair Scheduling for AI Workloads in Cloud Environments. Future Generation Computer Systems, 139, 20-32.