Artificial Intelligence Review (2025) 58:197
https://doi.org/10.1007/s10462-025-11208-8

AI-driven job scheduling in cloud computing: a comprehensive review

Yousef Sanjalawe[1] · Salam Al-E'mari[2] · Salam Fraihat[3] · Sharif Makhadmeh[1]

Accepted: 24 March 2025

Abstract
The demand for efficient job scheduling in cloud computing has grown significantly with the rise of dynamic and heterogeneous cloud environments. While effective in simpler systems, traditional scheduling algorithms fail to meet the complex requirements of modern cloud infrastructures. These limitations motivate the need for AI-driven solutions that offer adaptability, scalability, and energy efficiency. This paper comprehensively reviews AI-based job scheduling techniques, addressing several key research gaps in current approaches. The existing methods face challenges such as resource heterogeneity, energy consumption, and real-time adaptability in multi-cloud systems. Accordingly, the support of AI-based job scheduling in cloud computing is summarized here toward machine learning, optimization techniques, heuristic techniques, and hybrid AI models. This paper pointedly underlines the strengths and weaknesses of various approaches through deep comparative analysis and focuses on how AI will overcome traditional algorithm shortcomings. Is worth noticing that several important improvements this kind of AI-driven model provides, for example, in resource allocation, cost efficiency, energy consumption, and complex dependencies between jobs and system faults. In the end, AI-driven job scheduling seems to be a promising avenue toward effectively responding to the booming demands of cloud infrastructures. Future research should concentrate on three major outlooks: scalability, better integration of AI with traditional scheduling methods, and the use of other emerging technologies like edge computing and blockchain for better optimization of cloud-based job scheduling. The paper underscores the need for more adaptive, secure, and energy-efficient scheduling frameworks to meet the evolving challenges of cloud environments.

Keywords AI-driven job scheduling · Cloud computing · Energy efficiency · Hybrid AI models · Machine learning · Resource allocation

Salam Al-E'mari, Salam Fraihat, and Sharif Makhadmeh have contributed equally to this work.

Extended author information available on the last page of the article

Published online: 11 April 2025

1 Introduction

Cloud computing has transformed the way computational resources are managed, distributed, and utilized, revolutionizing industries across the globe (Sanjalawe et al. 2021; Taha et al. 2024). One of the critical challenges that emerge in cloud environments is efficient job scheduling. Optimising task allocation and resource management becomes increasingly important as the cloud infrastructure grows and diversifies. Effective job scheduling ensures the smooth operation of cloud services by balancing workload distribution, minimizing processing delays, and enhancing overall performance. Job scheduling in cloud computing environments involves several critical factors, such as resource availability, task priority, energy consumption, and the heterogeneity of the resources (Gures et al. 2022; Shahid et al. 2020; Shafiq et al. 2022). Researchers have proposed numerous algorithms and approaches to address these challenges, each focusing on improving performance metrics like execution time, resource utilization, and energy efficiency. However, the growing demand for more complex and dynamic workloads in cloud infrastructures has created an urgent need for adaptive and intelligent job scheduling strategies. This review focuses on the role of AI in enhancing job scheduling mechanisms within cloud computing environments. AI-based scheduling has garnered attention for its potential to provide more flexible, dynamic, and accurate solutions to scheduling problems. By systematically analyzing existing literature, this paper aims to present a comprehensive overview of current trends, challenges, and advancements in AI-driven job scheduling.

The following subsections delve deeper into the critical aspects of AI-driven job scheduling in cloud computing. Section 1.1 provides a comprehensive background on the evolution of cloud computing and the challenges that arise with job scheduling in dynamic and heterogeneous environments. Section 1.2 discusses the motivation for exploring AI-based solutions, emphasizing the limitations of traditional scheduling methods and the potential benefits of AI in terms of adaptability, efficiency, and energy management. Finally, Sect. 1.3 outlines this review's key objectives, presenting the paper's significant goals in examining current research trends, challenges, and future directions for AI-based job scheduling in cloud environments.

The organization of the whole paper is summarized in Fig. 1

1.1 Background

Cloud computing is a transformative technology that enables the delivery of computing resources over the Internet on a pay-as-you-go basis. It eliminates the need for organizations to invest heavily in physical infrastructure and instead offers on-demand services (Sanjalawe et al. 2021; Taha et al. 2024). The flexibility and scalability provided by cloud services have made them indispensable to businesses and organizations worldwide. However, as cloud computing continues to grow, optimizing the allocation and scheduling of jobs becomes increasingly complex, requiring sophisticated strategies to ensure efficient resource utilization and performance. Job scheduling is a fundamental aspect of cloud computing that involves assigning jobs or tasks to available resources to maximise performance and minimise costs. In traditional computing systems, scheduling algorithms such as First-Come-First-Serve (FCFS), Round Robin, and Priority Scheduling have been widely used (Hamid et al. 2022; Putra 2022; Najm 2023). However, these algorithms are often

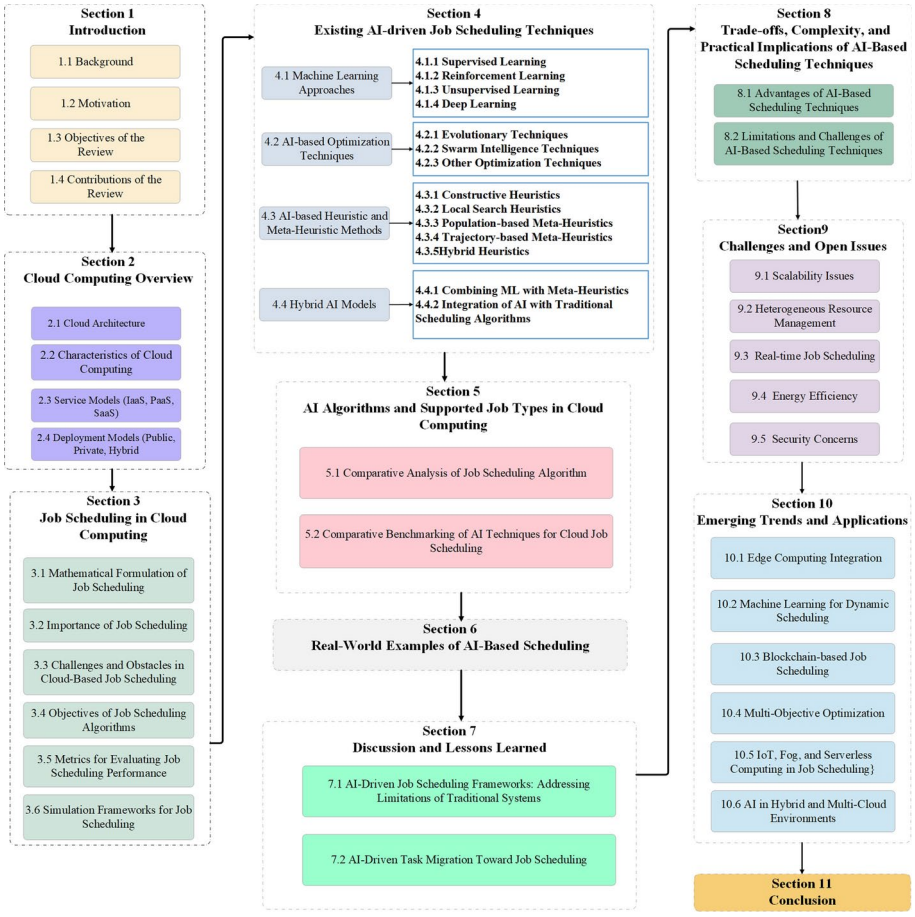

Fig. 1 Organization of paper

inadequate in cloud environments due to the sheer volume of tasks, the heterogeneity of resources, and the dynamic nature of workloads (Houssein et al. 2021; Pirozmand et al. 2023; Khan and Santhosh 2022). Cloud environments are highly dynamic, and resource demand can change rapidly and unpredictably. This volatility makes it difficult for static scheduling algorithms to perform optimally. Consequently, the need for adaptive scheduling approaches has become evident.

Researchers have turned themselves towards AI-based solutions that are adaptable to real-time conditions and their continuous optimization of resource allocations due to workloads and changing system requirements (Tuli et al. 2022; Kumar et al. 2022; Marahatta et al. 2020). Therefore, AI has tremendous potential for solving cloud computing job scheduling challenges. Specific AI techniques, such as machine learning, deep learning, and reinforcement learning, allow for the design of adaptive algorithms that can learn from past data and make intelligent decisions on task allotment. Most of these AI-driven techniques tend to have much more dynamic scheduling with efficiency, especially when it comes to large-scale heterogeneous cloud environments. Among many benefits, AI-based scheduling helps

to handle complex decision-making in real-time. Classic scheduling algorithms have mainly relied on predefined rules or static priorities, possibly not appropriate in the fluid nature of cloud computing (Marahatta et al. 2020; Rjoub et al. 2021). AI algorithms may analyze the characteristics of the tasks along with the availability of resources and system performance metrics to allow better and more conscious scheduling decisions (Kumar et al. 2022).

Another noticeable challenge in cloud-based job scheduling is heterogeneity in resources (Khallouli and Huang 2022; Singh et al. 2021). Usually, a cloud environment comprises different resources with variable processing capabilities, storage capacities, and energy consumption profiles. AI-based scheduling algorithms can consider heterogeneity in resources and optimize the allocation of resources based on these differences. Machine learning algorithms can, for example, predict different resources' performance for specific tasks to ensure jobs are assigned to the most appropriate resources. Energy efficiency is another crucial concern in cloud computing. A large amount of energy is consumed by data centres that power cloud infrastructures; this leads to high operational costs and environmental impacts. AI-based job scheduling can help reduce energy consumption by optimizing the reservations of resources and hence minimizing the wastage caused by their idling (Iftikhar et al. 2023a; Kumar et al. 2022). By predicting workloads and adjusting resource allocations dynamically, AI-driven algorithms should bring significant improvements in the energy efficiency of cloud systems. Apart from energy efficiency, AI-based scheduling also addresses the issue of prioritizing jobs. A cloud environment may contain tasks needing urgent attention or critical deadlines, while others may not be that rigid. AI algorithms can intelligently prioritize jobs based on their importance, urgency, and resource requirements to ensure high-priority tasks are completed within the required time frames while maintaining overall system performance. AI integration into data centre operations, exemplified by the DC-CFR framework (Sarkar et al. 2024), significantly enhances energy efficiency and reduces carbon emissions, aligning with green cloud computing objectives. The framework employs Multi-Agent Reinforcement Learning (MARL) to optimize workload distribution, HVAC systems, and battery operations in real-time (Sarkar et al. 2024). Key achievements include shifting delay-tolerant workloads to periods of lower carbon intensity, dynamically adjusting cooling systems to minimize energy consumption, and strategically managing battery usage to maximize renewable energy reliance. The DC-CFR framework outperforms traditional methods like ASHRAE, achieving a 14.46% reduction in carbon emissions, 14.35% in energy consumption, and 13.69% in energy costs over a year across diverse regions. Its real-time adaptability and scalability make it practical for dynamic and variable cloud environments, highlighting its potential for widespread adoption. The framework demonstrates a transformative path toward sustainable and energy-efficient cloud computing practices by incorporating intelligent AI-driven strategies.

Security and reliability are other imperative factors in cloud job scheduling (Sasubilli and Venkateswarlu 2021; Ghani et al. 2020). Immaculate cloud systems are an easy target for any cyber-attack; in this regard, ensuring the integrity of job scheduling is also very relevant to maintaining security in cloud systems (Abd Elaziz et al. 2021). AI-based scheduling algorithms could be developed to comprehend anomalies in resource utilization and job execution patterns that could pinpoint any probable security threat in real time for appropriate action. It works proactively to improve the general security of the cloud system. However, the development of edge computing with increased IoT devices has brought on new pressure related to job scheduling difficulty in cloud environments. Edge computing

refers to processing data closer to the source data generation device—like any IoT device—to minimize latency and improve real-time decisions. AI-based job scheduling is vital in managing task distribution for cloud and edge environments because it ensures that the right resources are used at the right time. Besides cloud computing, integrating blockchain with AI-based job scheduling is also gaining significant attention (Kurni et al. 2022; Yunlong and Jie 2024). Blockchain provides decentralized and transparent features; thus, it is suitable for secure job scheduling over distributed cloud infrastructures. AI algorithms could cooperate with blockchains to ensure job scheduling processes in cloud environments are untampered with and traceable, providing increased security and efficiency.

As the landscape of cloud computing evolves, AI-based job scheduling is expected to become even more critical in managing increasingly complex systems (Iftikhar et al. 2023a). The need for intelligent and adaptive scheduling algorithms will only increase with the rise of multi-cloud and hybrid cloud architectures, where multiple cloud service providers and deployment models are integrated. AI's ability to adapt, learn, and optimize resource allocation in real-time makes it an ideal solution for meeting the growing demands of modern cloud infrastructures.

1.2 Motivation

The ever-increasing demand for cloud computing services has created immense pressure on existing infrastructure to provide high performance, reliability, and scalability. While effective in earlier stages, traditional job scheduling approaches are no longer adequate to meet these growing needs (Umarani Srikanth and Geetha 2023; Prity et al. 2023). This mismatch between demand and existing technology has motivated researchers and practitioners to explore more sophisticated solutions, with AI emerging as a promising approach. The ability of AI algorithms to analyze large datasets, predict workloads, and make real-time decisions can significantly enhance job scheduling in cloud environments. One of the primary motivators for employing AI in job scheduling is its capacity to handle the complexity and heterogeneity of cloud resources (Zhou et al. 2024; Umarani Srikanth and Geetha 2023). Cloud infrastructures have various resources, each with different computational capabilities, energy requirements, and costs. AI-based algorithms can intelligently allocate jobs to the most suitable resources by learning from past data and making real-time adjustments. This results in improved resource utilization, faster job completion times, and reduced operational costs, making AI a powerful tool for optimizing cloud performance.

Energy efficiency has become another critical driving factor for adopting AI in cloud job scheduling (Katal et al. 2023; Verma et al. 2024; Rehan 2024). Data centres, the backbone of cloud computing, consume vast energy, contributing to environmental concerns and rising operational expenses. AI-driven job scheduling algorithms can optimize resource allocation to minimize energy usage by dynamically predicting workload trends and scaling resources up or down. By improving energy efficiency, AI can help cloud service providers reduce their carbon footprint and contribute to more sustainable cloud computing practices. In addition to performance and energy concerns, AI offers significant advantages in managing cloud workloads' increasingly complex and dynamic nature. Modern cloud environments often handle diverse and unpredictable tasks, ranging from routine computations to real-time data processing for critical applications such as financial transactions and medical diagnoses. Traditional job scheduling algorithms struggle to adapt to these dynamic workloads. Still,

AI-based algorithms can continuously learn and adapt, offering more responsive and flexible scheduling solutions to meet real-time processing demands (Iftikhar et al. 2023a).

Another motivating factor is the growing complexity of hybrid and multi-cloud environments, where cloud services are distributed across multiple cloud providers and deployment models (Mohammadzadeh and Masdari 2023; Zhu et al. 2021). Managing job scheduling across these fragmented systems presents a significant challenge, especially in maintaining performance, security, and cost-efficiency. AI algorithms offer the potential to harmonize scheduling across these diverse systems by making data-driven decisions that ensure tasks are executed efficiently and securely, regardless of the underlying cloud infrastructure. Moreover, AI's ability to enhance the security of job scheduling processes is a crucial motivation for its integration. As cloud systems become more vulnerable to cyberattacks, it is vital to have scheduling mechanisms that can detect and respond to anomalies in real-time (Aron and Abraham 2022). AI-based scheduling algorithms can monitor resource usage and task execution patterns to identify suspicious behaviour or potential threats, ensuring higher security in cloud environments. This proactive approach helps safeguard cloud infrastructures from internal and external threats, reinforcing the reliability and trustworthiness of cloud services.

Finally, the motivation for using AI in job scheduling stems from its potential to facilitate innovation in emerging cloud computing paradigms. For instance, edge computing and the Internet of Things (IoT) bring additional layers of complexity, requiring job scheduling that can manage both centralized cloud resources and decentralized edge devices (Wu et al. 2021; Huang et al. 2020). AI algorithms can seamlessly integrate scheduling across these distributed systems, improving the responsiveness and efficiency of cloud-edge ecosystems, which is essential for supporting real-time applications such as autonomous vehicles, smart cities, and industrial automation.

The comparative analysis highlights the unique contributions of the proposed paper relative to existing surveys. While many studies address specific aspects of job scheduling, such as heterogeneity handling, energy efficiency, or resource allocation, they often lack a holistic approach that integrates emerging technologies like blockchain or focuses on multi-cloud environments. For instance, works like Khan et al. (2023) and Iftikhar et al. (2023b) explore multi-cloud and AI-driven scheduling but fall short in addressing cross-cloud challenges and dynamic adaptability.

The proposed paper addresses the scalability and adaptability required in modern cloud infrastructures. Unlike prior surveys, it emphasizes hybrid AI models integrating machine learning and heuristic approaches for enhanced decision-making. In addition, it uniquely incorporates sustainability metrics, aligning scheduling practices with green computing goals. Existing studies largely lack security-aware scheduling and blockchain integration. The proposed paper bridges this gap by highlighting blockchain's role in ensuring transparent, traceable, and secure scheduling processes. Moreover, it emphasizes real-time adaptability, an underexplored feature in the current literature, to handle unpredictable workloads dynamically.

The proposed paper addresses the scalability and adaptability required in modern cloud infrastructures. Unlike prior surveys, it emphasizes hybrid AI models integrating machine learning and heuristic-based approaches to enhance scheduling decision-making. This hybridization provides the flexibility and robustness to manage dynamic and diverse workloads in real-world cloud environments. A significant shortcoming of many existing works

is the absence of sustainability metrics within their scheduling evaluations. With growing global attention to green computing, energy-aware scheduling is no longer desirable but necessary. This paper uniquely incorporates sustainability as a core criterion, aligning AI-based scheduling practices with carbon reduction goals and operational efficiency.

Moreover, lacking attention to security-aware scheduling and blockchain integration in most reviewed surveys limits their applicability in secure, transparent multi-tenant environments. The proposed work addresses this gap by exploring blockchain's role in reinforcing trust, integrity, and traceability in scheduling decisions, an aspect scarcely covered in prior literature. Another underexplored area in existing reviews is real-time adaptability. Traditional AI approaches often struggle to respond to sudden workload fluctuations or resource failures. By focusing on real-time decision-making, this work positions itself to meet the needs of mission-critical applications and time-sensitive workloads in modern cloud platforms.

The comparative Table 1 illustrates how this work surpasses prior surveys by incorporating multi-cloud support, heterogeneity management, security awareness, dynamic adaptability, sustainability, and blockchain integration. It demonstrates a broader, more holistic perspective for current and future cloud job scheduling paradigms.

In summary, this review paper bridges several critical gaps in the literature:

- Highlighting hybrid AI models that unify strengths from multiple approaches.
- Embedding sustainability and energy-efficiency as central evaluation metrics.
- Introducing blockchain-enabled scheduling frameworks for security and transparency.
- Stressing the importance of real-time adaptability for unpredictable cloud workloads.
- Providing an expanded comparative taxonomy that contextualizes its contributions.

This positions the paper as a comprehensive and forward-looking contribution that surveys existing methods and offers actionable insights for building resilient, adaptive, and sustainable job scheduling frameworks in AI-driven cloud systems.

Review methodology

Numerous articles have been published on AI-driven job scheduling in cloud computing. However, conducting a thorough literature review to identify and analyze relevant studies can be challenging for researchers. Scholarly databases and search engines depend heavily on carefully selected keywords to facilitate discovering pertinent research. Keywords are essential for indexing and retrieval, as they succinctly capture a paper's core focus and enhance its visibility and citation impact. We developed a keyword network visualization based on terms extracted from indexed research articles across various scholarly databases indexed by Scopus (from 2020 to 2025) to aid this process and contribute to the academic community. This visualization illustrates the relationships and prominence of key terms associated with AI-driven job scheduling in cloud computing. By providing a graphical representation of these terms, as depicted in Fig. 2, we aim to support researchers in identifying significant studies and emerging trends within this field. Furthermore, we offer an analysis of keyword frequencies, shedding light on the primary focus areas in AI-based job scheduling research.

This study synthesizes a comprehensive range of scholarly works to thoroughly review AI-driven job scheduling in cloud computing. To ensure broad coverage of the field, we relied on prominent scholarly databases, including but not limited to:

Table 1 Expanded comparative analysis with existing Surveys

Aspect	Houssein et al. (2021)	Khan et al. (2023)	Prity et al. (2023)	Hai et al. (2023)	Iftikhar et al. (2023b)	Rozehkhani et al. (2024)	Mahdizadeh et al. (2024)	Saidi and Bardou (2023)	Proposed
Multi-cloud integration		✓			✓				✓
Heterogeneity handling	✓	✓	✓	✓	✓				✓
Energy efficiency focus	✓			✓	✓	✓		✓	✓
Dynamic adaptability	✓				✓				✓
AI techniques utilized					✓				✓
Comprehensive taxonomy	✓	✓	✓		✓				✓
Scalability				✓	✓	✓			✓
Resource allocation	✓	✓		✓	✓	✓	✓	✓	✓
Hybrid AI models					✓				✓
Focus on edge integration									✓
Integration of blockchain									✓
Security-aware scheduling					✓				✓
Real-time decision making									✓
Job dependency management				✓					✓
Sustainability metrics									✓

- IEEE Xplore
- Springer
- Elsevier
- Wiley
- ACM Digital Library
- Tech Science Press
- Taylor and Francis
- Science Press

The methodology in Fig. 3 outlines a structured and systematic approach for reviewing AI-driven job scheduling in cloud computing. It begins with comprehensive literature exploration, encompassing prominent scholarly databases such as IEEE Xplore and Elsevier. It is followed by a rigorous selection and filtering process based on relevance and quality (Table 2 summarizes the inclusion and exclusion criteria). The final step involves synthesizing key findings and categorizing them into distinct themes or research gaps, aligning them with the review's objectives. This process ensures the integration of diverse perspectives and emerging trends, enabling a holistic domain analysis.

1.3 Objectives of the review

The primary objectives of this systematic literature review are as follows:

- To provide a comprehensive overview of the current state of research in AI-based job scheduling within cloud computing environments.
- To analyze and categorize existing job scheduling algorithms based on their performance, scalability, and adaptability in dynamic cloud systems.
- To identify key challenges and open issues in AI-driven job scheduling, including scalability, energy efficiency, and security concerns.
- To examine integrating emerging technologies such as edge computing, IoT, and blockchain into AI-based job scheduling frameworks.

1.4 Contributions of the review

This paper provides a comprehensive exploration of AI-driven job scheduling in cloud environments, with the following key contributions:

- Comprehensive analysis of AI-driven job scheduling challenges: The paper identifies and analyzes critical challenges in AI-driven job scheduling, including scalability, energy efficiency, security, heterogeneous resource management, and real-time adaptability. These challenges are foundational to addressing the complexities of dynamic and modern cloud environments.
- Optimization of energy efficiency and sustainability: It explores how AI techniques optimize energy usage in cloud data centres, emphasizing predictive scaling, renewable energy utilization, and carbon emission reduction. These approaches align with green computing and sustainability goals.
- Integration of advanced AI techniques: The paper benchmarks various AI approaches,

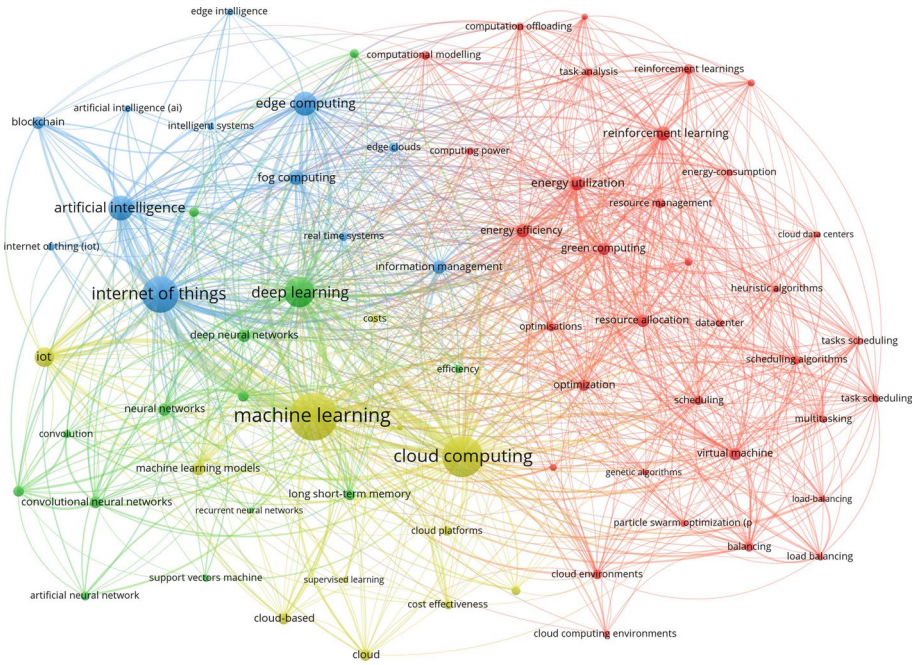

Fig. 2 Keywords network

including supervised learning, reinforcement learning, and hybrid models, highlighting their strengths, limitations, and applications. Special attention is given to their role in dynamic, real-time, and heterogeneous resource scheduling.

- Enhancing security and multi-cloud scheduling: The work examines AI-driven methods to enhance security by addressing data privacy, compliance, and protection against cyber threats. It also delves into the unique challenges of hybrid and multi-cloud environments, such as cross-cloud scheduling, latency reduction, and cost optimization.
- Future-oriented research directions: The paper proposes future research areas, including hybrid AI models for multi-objective optimization, collaborative multi-agent systems for resource management, and integrating sustainability metrics into AI-driven scheduling frameworks. These directions aim to advance the field further.

2 Cloud computing overview

Cloud computing has transformed the IT landscape by offering scalable, on-demand services over the Internet. The widespread adoption of cloud computing has been driven by the need for flexible, cost-effective, and efficient resource management solutions. With the rise of cloud infrastructures, organizations of all sizes have been able to access powerful computational resources without the need for heavy capital investment in hardware and software (Sanjalawe and Al-E'mari 2023; Althobaiti et al. 2023; Sanjalawe and Althobaiti 2023). This section provides a detailed overview of cloud computing, covering its architecture, key characteristics, service models, and deployment models.

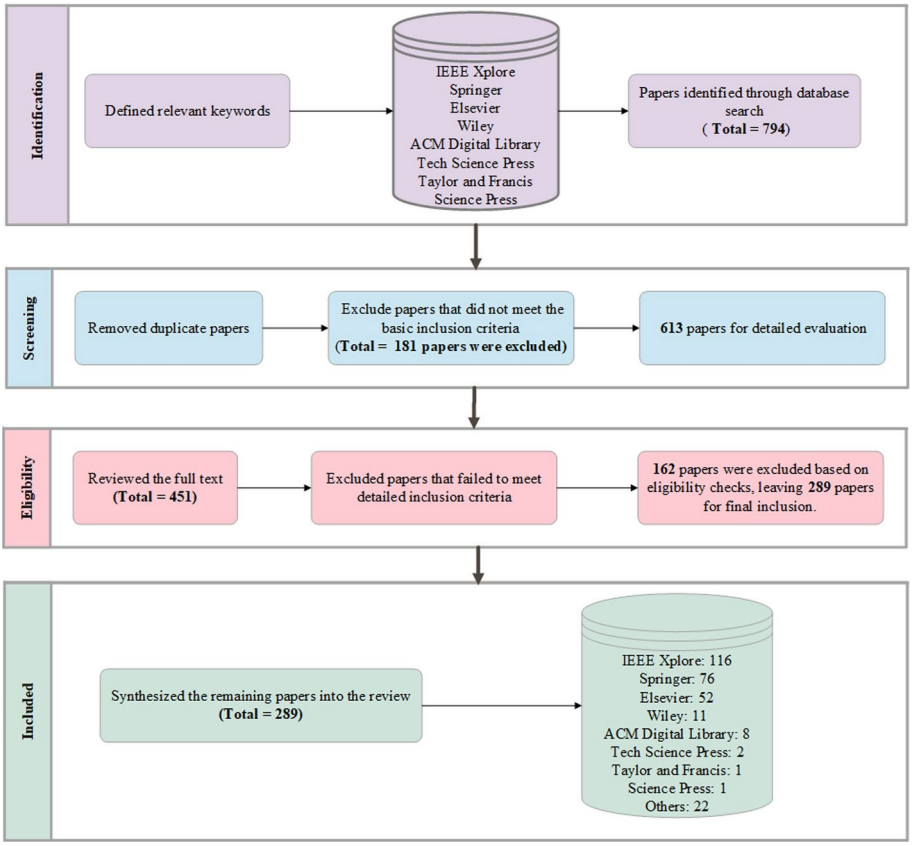

Fig. 3 Methodology of the paper

Cloud computing operates on a model that abstracts the underlying physical infrastructure and presents it as a set of services accessible over the Internet. The complexity of managing physical hardware, software, and network components is hidden from the end-user, who can access these resources seamlessly. This abstraction allows organizations to focus on their core competencies while the cloud provider manages the underlying IT infrastructure, ensuring performance, security, and availability.

This section is divided into four significant sub-sections to provide a structured understanding of cloud computing. First, we explore cloud architecture, outlining the components and layers of the cloud infrastructure in Subsect. 2.1. Second, in Subsect. 2.2, the characteristics of cloud computing are discussed, including scalability, elasticity, and multi-tenancy. Third, the Subsect. 2.3 examines the various cloud service models—Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS)—which offer different levels of abstraction and user control. Finally, we cover the deployment models of cloud computing—Public, Private, and Hybrid clouds—each of which provides unique benefits and challenges based on the user's specific requirements in Subsect. 2.4.

Table 2 Inclusion and exclusion criteria

Criteria type	Description
Inclusion criteria	
Relevance to AI-driven job scheduling	Studies addressing job scheduling challenges, techniques, or methodologies using AI, including machine learning, optimization algorithms, and hybrid approaches
Focus on cloud computing	Papers, which specifically examines job scheduling within cloud computing environments, including hybrid and multi-cloud setups
Publication year	Most of the articles were published between 2020 and 2025, ensuring the inclusion of recent advancements in the field
Publications	Peer-reviewed journal articles, conference proceedings, and white papers from reputable databases such as IEEE Xplore, Springer, Elsevier, and ACM
Metrics and objectives	Research that evaluates performance using relevant metrics like makespan, energy efficiency, load balancing, and system throughput
Language	Studies published in English for accessibility and consistency in review
Emerging technologies	Papers incorporating emerging technologies like edge computing, blockchain, or green computing in job scheduling
Exclusion criteria	
Non-AI approaches	Studies focus solely on traditional job scheduling algorithms without any integration of AI-based techniques
Irrelevant domains	Research unrelated to cloud computing, such as job scheduling in unrelated fields like manufacturing or logistics
Non-peer-reviewed sources	Articles from blogs, opinion pieces, or non-academic sources that lack validation and scientific rigour
Duplicate studies	Papers with overlapping content or methodology without offering new insights
Insufficient information	Studies that do not provide detailed methodologies, results, or discussions to allow comprehensive analysis
Non-English publications	Papers in languages other than English due to the limitation of translation and analysis

2.1 Cloud architecture

Cloud architecture refers to the structural design that enables the delivery of cloud services, comprising front-end platforms, back-end platforms, cloud-based delivery, and the network. The front end includes user interfaces and client devices (e.g., desktops, laptops, mobile devices), which communicate with back-end systems like servers, storage, and databases for processing and storing data. Virtualization is a key aspect, allowing multiple Virtual Machines (VMs) to run on a single physical server, maximizing resource utilization and efficiency (Al-E'mari et al. 2024; Birje et al. 2017; Yanamala 2024). By abstracting hardware resources into a pool, virtualization enhances scalability and flexibility. Networking infrastructure connects the components of cloud systems, enabling data transfer, load balancing, and redundancy. Geographically distributed data centres enhance network performance, providing low-latency access to services. The orchestration layer is another critical component, managing automation and coordination of services through resource provisioning, scaling, and monitoring tools (Alouffi et al. 2021; Ullah et al. 2022). Storage systems are central to cloud architecture, offering scalable, secure, and resilient solutions. Cloud storage is categorized into object, block, and file storage, tailored for different use cases (Wang and Zhang 2020; Ghani et al. 2020). For example, object storage is ideal for unstructured data, while block storage supports applications requiring low-latency access. Figure 4 illustrates

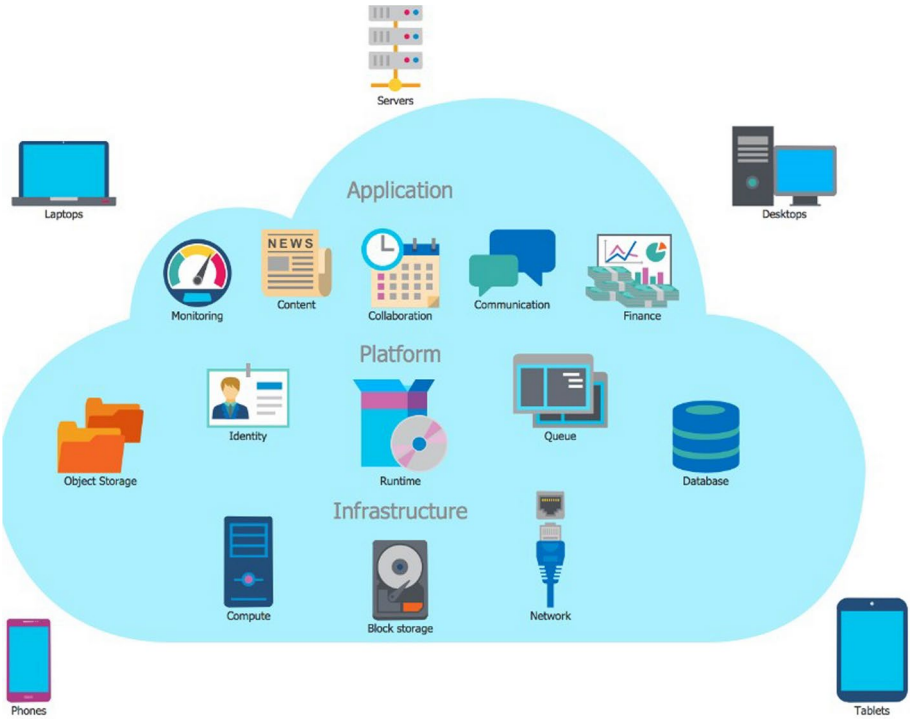

Fig. 4 Cloud computing architecture (ConceptDraw Software on Instagram n.d.)

the layered architecture of cloud computing, encompassing infrastructure, platform, and application services, along with their associated components and user devices.

Security is a fundamental aspect of cloud architecture, incorporating firewalls, encryption, identity management, and intrusion detection systems (Parast et al. 2022; Sasubilli and Venkateswarlu 2021). These tools work together to protect data as it is distributed across servers and accessed remotely. Cloud architecture is also designed to be resilient and fault-tolerant, with data centres employing redundancy through data replication, load balancing, and failover systems to maintain high availability even during hardware or network failures. Emerging technologies like edge computing and serverless computing are further shaping cloud architecture (Cao et al. 2020; Li et al. 2022). Edge computing reduces latency and bandwidth usage by processing data closer to the source (Cao et al. 2020; Luo et al. 2021), while serverless computing abstracts infrastructure management, allowing developers to focus on application code (Hassan et al. 2021; Li et al. 2022). Monitoring and management systems are also integral, providing real-time insights into resource performance and health. Cloud providers offer dashboards and APIs to enhance transparency and control over infrastructure. Cloud architecture integrates virtualization, networking, security, storage, and orchestration technologies to deliver scalable, reliable, and secure services that meet the demands of modern organizations. Its evolution ensures adaptability, efficiency, and resilience in dynamic computing environments.

2.2 Characteristics of cloud computing

Cloud computing is defined by several key characteristics. **On-demand self-service** allows users to provision resources like compute power and storage without human intervention, enabling flexibility and scalability. **Broad network access** ensures services are accessible from any location via devices like laptops and smartphones, enhancing mobility. **Resource pooling** supports multi-tenancy, with resources dynamically assigned while ensuring data isolation for security (Tang et al. 2021; Shahidinejad et al. 2021). **Scalability and elasticity** enable automatic adjustment of resources based on demand (Barnawi et al. 2020; Ngo et al. 2022), optimizing cost efficiency. **Measured service** provides a pay-per-use model, billing users for actual consumption (Zhang 2020; Vu et al. 2020). **Resilience and fault tolerance** ensure service availability through redundancy and data replication (Shahid et al. 2021; Rehman et al. 2022), while **automation** enables efficient resource management (Phasinam et al. 2022; Nasurudeen et al. 2021).

Security is embedded through encryption and access control, though it remains a shared responsibility (Sasubilli and Venkateswarlu 2021; Ghani et al. 2020). **High availability and reliability** are achieved via distributed infrastructures, load balancing, and failover mechanisms (Li et al. 2020a; Zhu et al. 2021). Lastly, **Service-Oriented Architecture (SOA)** supports modular, integrative cloud services, enabling customized solutions (Niknejad et al. 2020; Xia and Chen 2020).

2.3 Service models (IaaS, PaaS, SaaS)

Cloud computing is delivered through various service models, each offering different levels of abstraction and control. As illustrated in Fig. 5, the three primary service models are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) (Mohammed and Zeebaree 2021; Al-E'mari et al. 2024).

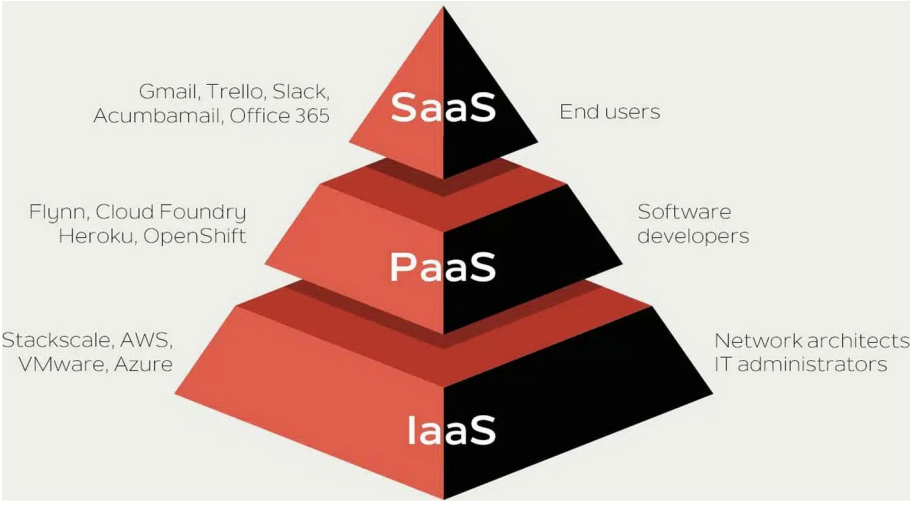

Fig. 5 Business models of cloud computing (bAmeer et al. 2022)

IaaS provides the most basic level of cloud services, offering virtualized computing resources such as virtual machines, storage, and networking. IaaS allows users to configure and manage their infrastructure, providing the most significant degree of flexibility (Al-E'mari et al. 2024; Rani and Ranjan 2014). However, it also requires users to operate and maintain operating systems, applications, and data. IaaS is ideal for businesses that require customizable and scalable infrastructure. Users can quickly scale their resources up or down based on demand, making IaaS a cost-effective solution for applications with fluctuating workloads. Additionally, IaaS provides disaster recovery and business continuity solutions, allowing organizations to back up their data and applications in the cloud for quick recovery in emergencies.

PaaS offers a higher level of abstraction by providing a platform that includes infrastructure, development tools, databases, and runtime environments. PaaS allows developers to build, test, and deploy applications without worrying about the underlying infrastructure (Nadeem 2022; Al-E'mari et al. 2024). This model is ideal for development teams focusing on writing code rather than managing hardware and software. PaaS platforms often include pre-configured environments for developing specific types of applications, such as web or mobile apps. These platforms also provide integrated services such as databases, message queues, and monitoring tools, simplifying development. By abstracting away the infrastructure layer, PaaS enables faster development cycles and reduces operational overhead.

SaaS is the most commonly used cloud service model and provides complete software applications over the internet. Users access SaaS applications via web browsers without the need to install or manage software on their local devices (Saraswat and Tripathi 2020; Vaquero 2011). Famous examples of SaaS applications include email services, Customer Relationship Management (CRM) tools (Limbasan and Rusu 2011), and Enterprise Resource Planning (ERP) systems (Seethamraju 2015). SaaS is ideal for businesses that require fully managed applications with minimal customization. Users benefit from automatic updates, maintenance, and security handled by the SaaS provider. This model benefits organizations that minimize their IT overhead and focus on using software to achieve business objectives.

2.4 Deployment models (public, private, hybrid)

Cloud computing can be deployed through various models, each catering to different organizational needs and preferences. As illustrated in Fig. 6, the primary deployment models are public cloud, private cloud, hybrid cloud, community cloud, and multi-cloud (Rai et al. 2021).

The **Public Cloud** is the most widely adopted model, where cloud resources are owned and operated by third-party cloud providers and are available to the general public. Public cloud services are delivered over the Internet and are typically billed on a pay-per-use basis (Khan et al. 2022; Nagarajan and Kumar 2021). This model is ideal for businesses without dedicated infrastructure and looking for a cost-effective, scalable solution. Public clouds offer several advantages, including reduced IT infrastructure costs, on-demand scalability, and access to a wide range of services. However, they pose potential security and compliance concerns, as data is stored on shared infrastructure. Organizations that handle sensitive data or require strict regulatory compliance may find public clouds less suitable for their needs.

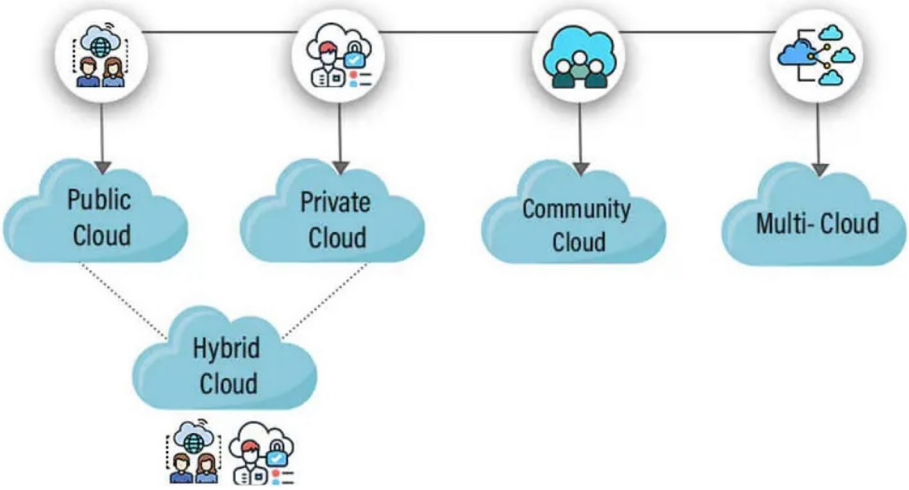

Fig. 6 Deployment models of cloud computing (Kamalakkannan n.d.)

In contrast, the **Private Cloud** is a cloud environment dedicated to a single organization. Private clouds can be hosted on-premises or by third-party providers, but the infrastructure is not shared with other organizations (Niranjanamurthy et al. 2020; Wagh et al. 2020). This model offers greater control, security, and customization, making it ideal for organizations with strict regulatory or security requirements. Private clouds allow organizations to tailor their infrastructure to specific needs, providing greater flexibility in data management, security policies, and application deployment. However, private clouds often require significant hardware, software, and expertise investment, making them more expensive than public clouds.

The **Hybrid Cloud** model combines elements of both public and private clouds, allowing organizations to take advantage of the benefits of both environments. In a hybrid cloud setup, workloads can be distributed between the public and private clouds based on security, performance, and cost. Hybrid clouds offer greater flexibility and scalability, enabling organizations to optimize their cloud usage (Saxena et al. 2021; Gundu et al. 2020). For instance, sensitive data can be stored in the private cloud while less-critical workloads run in the public cloud. Hybrid clouds also enable seamless integration between on-premises infrastructure and cloud services, allowing businesses to transition to the cloud at their own pace.

The **Community Cloud** is a shared cloud infrastructure for organizations with similar concerns, such as compliance, security, or performance goals (Aldahwan and Ramzan 2022). It is commonly used by healthcare, education, or government sectors, where shared governance frameworks exist. For instance, healthcare providers may use a community cloud to manage patient data securely while complying with HIPAA regulations. Managed by organizations or a third party, this model allows shared use of resources while maintaining control and customization (N'Goran et al. 2023; Qureshi and Sharma 2021). Benefits include cost savings, collaboration, and innovation through shared infrastructure. However, balancing diverse requirements among participants can be a challenge.

The **Multi-Cloud** strategy uses multiple cloud services from different vendors to optimize performance, reduce costs, and avoid dependency on a single provider (Zhu et al.

2021). By leveraging platforms like AWS, Azure, and Google Cloud, organizations can capitalize on their strengths—such as AWS for machine learning, Google Cloud for data analytics, and Azure for enterprise applications (Tomarchio et al. 2020; Khan 2020). This approach avoids vendor lock-in and enhances disaster recovery by spreading data and applications across multiple platforms, reducing downtime risks (Zhu et al. 2021; Tomarchio et al. 2020; Khan 2020). However, managing multi-cloud environments poses challenges, requiring smooth integration, robust security for access and compliance, and tools to monitor performance and costs. Despite these complexities, the strategy provides a more resilient and efficient cloud infrastructure.

3 Job scheduling in cloud computing

Cloud computing has transformed how computational tasks are handled, offering scalable and flexible resources. Job scheduling is critical in optimizing resource utilization and ensuring efficient task execution in a cloud environment. As cloud systems grow, scheduling algorithms have become more sophisticated, often leveraging Artificial Intelligence (AI) to improve decision-making processes (Kumar et al. 2022). In this section, we explore the concept of job scheduling within the context of cloud computing. Scheduling tasks effectively is essential to enhance system performance, reduce operational costs, and meet Service Level Agreements (SLAs) (Islam et al. 2021; Yakubu et al. 2020). By utilizing AI-based methods, cloud providers can achieve better results in scheduling by making dynamic, real-time decisions.

Several architectures have been proposed in the literature to enhance the job scheduling process in cloud computing, such as the one shown in Fig. 7, which illustrates the current Job Scheduling Technique. Research indicates that many factors contributing to effective job scheduling are not clearly defined, categorized, or modelled comprehensively that can be directly implemented in applications (Murad et al. 2022). Although research on job scheduling in cloud computing has become more advanced, significant gaps remain yet to be explored. After thoroughly reviewing existing research, Murad et al. (Murad et al. 2022) developed a novel approach for effective job scheduling in cloud computing environments.

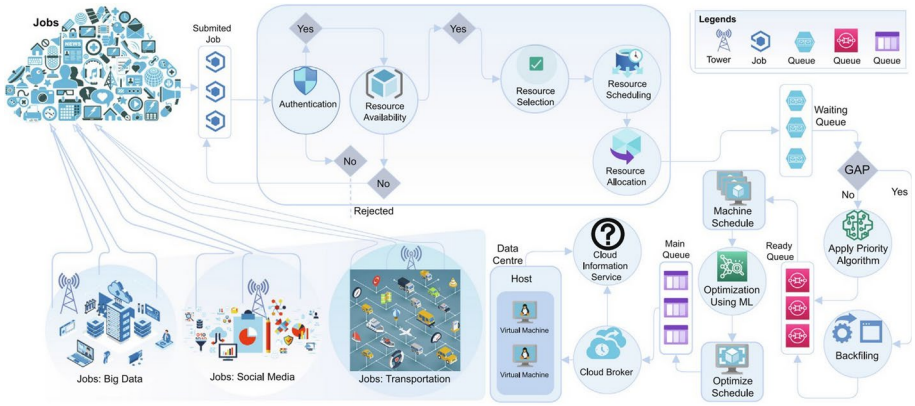

Fig. 7 Architecture of job scheduling in cloud computing (Murad et al. 2022)

Our proposed system is divided into several phases: resource allocation, backfilling, and optimization (using machine learning).

In the subsequent subsections, the discussion begins with an overview of job scheduling's importance, followed by a detailed examination of the critical challenges faced in cloud-based scheduling systems. We then outline the primary objectives of scheduling algorithms and conclude by introducing various metrics used to evaluate scheduling performance.

This section is organized as follows: Subsect. 3.2 presents the mathematical formulation of job scheduling, then Subsect. 3.2 discusses the importance of job scheduling in cloud environments. Subsection 3.3 focuses on cloud-based systems' challenges when scheduling jobs. In Subsect. 3.4, we present the objectives of job scheduling algorithms. Subsection 3.5 introduces the performance metrics used to evaluate job scheduling algorithms in cloud computing, providing explanations and equations for each metric. Finally, Subsect. 3.6 presents the simulation frameworks for job scheduling.

3.1 Mathematical formulation of job scheduling

Job scheduling in cloud computing is a complex optimization problem that aims to efficiently allocate tasks to resources while satisfying various objectives and constraints. This subsection provides a generic mathematical formulation to formalize the problem and its associated metrics.

Problem description

Let $J = \{j_1, j_2, \ldots, j_n\}$ represent the set of tasks (jobs) to be scheduled and $R = \{r_1, r_2, \ldots, r_m\}$ represent the set of available resources. Each task j_i has specific requirements such as execution time, precedence constraints, and resource needs. The goal is to assign tasks to resources in a manner that minimizes the overall execution time, optimizes resource utilization, and satisfies additional constraints like energy efficiency and fairness.

Decision variables

- $x_{ir} \in \{0, 1\}$: A binary variable indicating whether task j_i is assigned to resource r ($x_{ir} = 1$) or not ($x_{ir} = 0$).
- S_i: The start time of task j_i.
- C_i: The completion time of task j_i.
- U_r: Utilization of resource r.
- P_r: Power consumption of resource r.
- L_r: Load on resource r.

Objective function

The primary objective of job scheduling is to minimize the makespan, which is the total time required to complete all tasks. This can be formulated as:

$$\text{Minimize Makespan} = \max(C_i), \quad \forall i \in \{1, 2, \ldots, n\}. \tag{1}$$

Secondary objectives

Other objectives that may complement the primary goal include:

1. *Energy efficiency* Minimize the total energy consumption of resources during task execution:

$$\text{Minimize Energy\_Consumption} = \sum_{r=1}^{m} P_r \cdot U_r. \tag{2}$$

2. *Fairness* Ensure equitable resource allocation using Jain's fairness index:

$$\text{Fairness\_Index} = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \cdot \sum_{i=1}^{n} x_i^2}, \tag{3}$$

where x_i is the resource allocated to task j_i.

3. *Load balancing* Distribute workload evenly across all resources:

$$\text{Load\_Balancing\_Factor} = \frac{\sum_{r=1}^{m} L_r}{m}. \tag{4}$$

Constraints

The optimization process must adhere to the following constraints:

1. *Task assignment* Each task must be assigned to exactly one resource:

$$\sum_{r=1}^{m} x_{ir} = 1, \quad \forall i \in \{1, 2, \ldots, n\}. \tag{5}$$

2. *Resource capacity* The total load on each resource must not exceed its capacity:

$$\sum_{i=1}^{n} L_r \cdot x_{ir} \leq \text{Capacity}(r), \quad \forall r \in R. \tag{6}$$

3. *Task precedence* Tasks with dependencies must respect precedence constraints:

$$S_j \geq C_i, \quad \forall (i, j) \in \text{Dependencies}. \tag{7}$$

4. *Non-negative start times* All tasks must start at or after time zero:

$$S_i \geq 0, \quad \forall i \in \{1, 2, \ldots, n\}. \tag{8}$$

This formulation captures the essence of the job scheduling problem in cloud environments. The primary objective of minimizing makespan aligns with the need for efficiency, while secondary goals such as energy efficiency, fairness, and load balancing address other critical concerns. The constraints ensure that task assignments are feasible, respect resource limitations, and meet dependency requirements. By formalizing the problem mathematically, this

framework provides a unified perspective for analyzing various scheduling algorithms and their trade-offs. It also serves as a foundation for future work on advanced scheduling techniques, such as those leveraging machine learning or metaheuristic optimization.

3.2 Importance of job scheduling

Job scheduling is a crucial component of cloud computing that directly affects system performance, resource utilization, and overall user satisfaction. In cloud environments, job scheduling refers to assigning and managing tasks across distributed resources to ensure efficient execution. The complexity of cloud systems, with their dynamic and scalable nature, makes job scheduling essential for maximizing the potential of cloud infrastructure. The primary objective of job scheduling in cloud computing is to achieve optimal resource utilization. Cloud resources are typically shared among multiple users and applications, making it important to schedule jobs to minimise idle times and maximise throughput (Zubair et al. 2022; Gures et al. 2022; Tong et al. 2020). Effective job scheduling ensures that computational resources, such as CPU, memory, and network bandwidth, are used efficiently.

The following points outline the major objectives and considerations in job scheduling;

1. Optimal resource utilization: Job scheduling aims to maximize the utilization of computational resources such as CPU, memory, and network bandwidth. Efficient scheduling minimizes idle times and throughput in cloud environments where resources are shared among multiple users and applications (Zubair et al. 2022; Gures et al. 2022; Tong et al. 2020).
2. Cost reduction: Cloud providers charge users based on resource consumption. Optimized job scheduling minimizes task execution times and resource usage, thereby reducing operational costs for both providers and users (Shafiq et al. 2022; Iftikhar et al. 2023a).
3. Meeting Service Level Agreements (SLAs): Scheduling algorithms ensure jobs are completed within the performance parameters specified in SLAs, such as response times, availability, and resource guarantees. This helps avoid penalties and maintain customer satisfaction (Shafiq et al. 2022).
4. Fair resource allocation: Job scheduling ensures fairness by preventing resource monopolization by any single user or application. It allocates resources equitably, ensuring all users receive an appropriate share of the system's capacity (Zubair et al. 2022; Tong et al. 2020).
5. Energy efficiency: Data centres consume significant energy, and inefficient scheduling exacerbates this issue. Energy-aware job scheduling algorithms reduce power consumption by intelligently managing resource allocation and idle times, contributing to sustainable operations (Hussain et al. 2021; Ding et al. 2020).
6. Improved system throughput: Efficient scheduling minimizes job waiting times and reduces bottlenecks, enabling cloud systems to complete more tasks in a given period, thereby increasing throughput (Hussain et al. 2021; Khorsand and Ramezanpour 2020).
7. Scalability: As cloud systems grow in size and complexity, scalable scheduling algorithms ensure efficient operations by handling larger volumes of tasks and resources without performance degradation (Houssein et al. 2021; Rani and Suri 2020).

8. Latency reduction: Scheduling impacts the time it takes for a task to begin execution after submission. Low-latency scheduling is critical for real-time applications that require immediate processing (Memari et al. 2022).
9. Fault tolerance: Cloud environments are prone to failures. Fault-tolerant scheduling algorithms detect disruptions and reassign tasks to available resources, ensuring job completion despite crashes or network issues (Malik et al. 2022; Chawla and Kaur 2024).
10. Security and data privacy: In multi-tenant cloud environments, scheduling must preserve data isolation and prevent unauthorized access. Security-aware scheduling algorithms enforce policies that protect sensitive information while optimizing resource use (Malik et al. 2022; Chawla and Kaur 2024).
11. Adaptability to workload variability: Cloud workloads vary significantly, with some tasks requiring more computational power than others. Adaptive scheduling algorithms adjust resource allocation dynamically to meet changing workload conditions (Ebadifard and Babamir 2021).

3.3 Challenges and obstacles in cloud-based job scheduling

Cloud-based job scheduling presents several challenges due to cloud environments' dynamic, distributed, and heterogeneous nature. These challenges arise from allocating resources efficiently while balancing conflicting objectives, such as minimizing costs, optimizing performance, and meeting SLAs. In this section, we discuss some of the primary challenges faced by scheduling algorithms in cloud computing. One of the most significant challenges in cloud-based job scheduling is the dynamic nature of resource availability.

The following are the key challenges in cloud-based job scheduling:

1. Dynamic resource availability: In a cloud environment, resources are not static; they can be added or removed at any time due to changes in user demand, failures, or maintenance activities (Murad et al. 2022; Rashidifar et al. 2022). Scheduling algorithms must continuously adapt to these fluctuations to ensure efficient task allocation. This dynamic nature increases the complexity of scheduling, as algorithms must respond in real-time to resource changes.
2. Heterogeneous resources: The heterogeneity of cloud resources adds another layer of complexity to job scheduling (Loncar and Loncar 2022; Hamed et al. 2023). Cloud environments consist of various resources with differing capabilities, including CPU types, memory configurations, and network speeds. This heterogeneity makes it difficult for scheduling algorithms to match tasks to appropriate resources, as tasks may have specific requirements that not all resources can meet (Ibrahim et al. 2020; Krishnasamy et al. 2023). Algorithms must consider the particular characteristics of tasks and resources to ensure compatibility and efficiency.
3. Energy efficiency: Energy efficiency is a growing concern in cloud computing, particularly as data centres become more power-hungry. Inefficient scheduling can lead to excessive energy consumption, as idle resources consume power even when not used effectively. Energy-aware scheduling algorithms aim to minimize power consumption by ensuring that resources are only active when necessary and by dynamically turning off or reducing the power usage of idle resources.

4. Meeting SLAs: Meeting SLA requirements is another challenge in cloud-based job scheduling (Kaur et al. 2021; Lipsa and Dash 2024). SLAs define the expected performance and availability of cloud services, and failing to meet these agreements can result in penalties for cloud providers. Scheduling algorithms must ensure that tasks are completed within the timeframes specified in SLAs, which can be difficult in dynamic and heterogeneous environments. Achieving this requires careful prioritization of tasks and efficient resource allocation to avoid SLA violations.
5. Handling workload variability: Workloads in cloud environments are highly unpredictable and can fluctuate significantly depending on user demand (Kaur et al. 2021). Scheduling algorithms must handle sudden spikes or drops in workload without compromising system performance. This requires adaptive scheduling techniques that dynamically adjust to changing workloads and redistribute tasks to available resources as needed.
6. Job dependencies: Job dependency is another challenge in cloud scheduling. In many cases, jobs are not independent and may depend on other tasks that must be completed before they can begin (Kaur et al. 2022b). This creates additional complexity for scheduling algorithms, as they must ensure that tasks are scheduled in a way that respects these dependencies (Chung et al. 2020). If tasks are scheduled without considering dependencies, it can lead to delays and inefficient execution.
7. Fault tolerance: Fault tolerance is critical in cloud computing, as cloud environments are prone to failures such as hardware malfunctions, network outages, or software bugs. Scheduling algorithms must be able to detect and recover from these failures to ensure that tasks are completed successfully. Fault-tolerant scheduling techniques typically involve redundancy and task replication, allowing tasks to be rescheduled on different resources in case of failure.
8. Data locality: Data locality refers to the proximity of data to the computing resources that process it. In cloud environments, data may be stored in geographically distributed data centres (Cheng et al. 2021). Scheduling algorithms must consider data locality to minimize data transfer times and improve task execution efficiency. If a task is scheduled on a resource far from the data it needs to process, it can lead to significant delays and increased network costs (Rjoub et al. 2020).
9. Security and data privacy: Security is an emerging challenge in cloud-based job scheduling. In multi-tenant cloud environments, tasks from different users or organizations may run on the same physical infrastructure, raising concerns about data privacy and security (Rajasoundaran et al. 2021). Scheduling algorithms must ensure that tasks are scheduled to isolate them from each other, preventing unauthorized access to sensitive data and maintaining data privacy.
10. Multi-objective optimization: In cloud-based job scheduling, multiple objectives must be considered simultaneously, such as minimizing energy consumption, reducing makespan, and meeting SLAs (Cui et al. 2023; Malti et al. 2024). These objectives usually conflict, making it difficult to find an optimal solution that satisfies all requirements. Scheduling algorithms must use weighted objectives or trade-off analysis techniques to balance these competing goals.
11. Scalability: Scalability is another critical challenge in cloud environments. As the number of users and tasks increases, scheduling algorithms must scale to handle the growing workload. This requires algorithms to be both computationally efficient and capable

of handling large-scale distributed systems without significant performance degradation. Scalability becomes particularly challenging when combined with other issues, such as heterogeneity and dynamic resource availability.

12. Job prioritization: The challenge of job prioritization arises when tasks have different levels of importance or urgency. Scheduling algorithms must determine how tasks are executed to meet deadlines and ensure that high-priority tasks receive the needed resources. This is especially important in real-time cloud applications, where delays in task execution can have significant consequences.

Figure 8 provides a comprehensive view of the key challenges and obstacles encountered in job scheduling within cloud environments. Job scheduling is a critical process that aims to optimize resource allocation, task execution, and overall system performance. However, numerous factors introduce complexity to this process, making it difficult to achieve optimal outcomes. The challenges range from technical limitations to unpredictable workload fluctuations and energy efficiency concerns, each of which must be carefully managed to ensure smooth operations in cloud systems.

One significant challenge in job scheduling is **workload fluctuations**. Cloud environments often experience variable workloads due to the dynamic nature of user demands (Li et al. 2018). This fluctuation makes it difficult to predict the required resources, leading to either underutilization or overutilization. Schedulers must adapt in real-time, dynamically allocating resources to meet demand without causing delays or system bottlenecks. Additionally, the presence of **identical and diversified workloads** adds to the complexity, as

Fig. 8 Obstacles in job scheduling

different types of jobs may require varied processing power and execution strategies (Han et al. 2019). Another notable obstacle is **energy-efficient allocation**. Energy consumption is critical in cloud computing, especially in large data centres (Cao and Zhu 2013). Inefficient job scheduling can lead to excessive energy use, increasing operational costs and environmental impact. To address this, job scheduling algorithms must incorporate energy-saving techniques while balancing performance and availability.

VM migration, where virtual machines are transferred across servers, is another challenge that requires careful scheduling to avoid disruptions while optimizing the placement of VMs to reduce power consumption and improve resource utilization (Raghavendra et al. 2020). In addition, **Uncertainty** (Dong et al. 2021) and **scheduling tasks in parallel** (Hao et al. 2017) also introduce complications. Uncertainty can arise due to unpredictable workloads, network conditions, or system failures, which requires schedulers to be flexible and robust to handle unexpected events. Parallel task scheduling, while beneficial for improving performance, adds complexity regarding dependency management and resource allocation. Ensuring tasks execute efficiently and concurrently without conflict or delay is essential for maximizing system throughput. Together, these challenges highlight the multifaceted nature of job scheduling in cloud environments and the need for advanced, intelligent scheduling solutions to address them.

Cloud-based job scheduling involves navigating a complex landscape of challenges, including dynamic resource availability, heterogeneity, energy efficiency, SLA compliance, workload variability, job dependency, fault tolerance, data locality, security, multi-objective optimization, scalability, and job prioritization. These challenges require sophisticated scheduling algorithms capable of adapting to diverse and evolving cloud environments.

3.4 Objectives of job scheduling algorithms

Job scheduling algorithms in cloud computing aim to achieve objectives that ensure efficient and effective resource utilization, improved system performance, and optimized user satisfaction (Zubair et al. 2022; Gures et al. 2022; Tong et al. 2020). These objectives often vary depending on the specific requirements of the cloud environment and the nature of the scheduled tasks. In this subsection, we discuss the critical goals of job scheduling algorithms. Table 3 provides a comprehensive overview of the objectives in job scheduling, along with their descriptions. This structured representation highlights the importance of each objective and how it contributes to the overall efficiency and sustainability of cloud systems.

In conclusion, job scheduling algorithms in cloud computing have diverse and multifaceted objectives. From minimizing makespan and maximizing resource utilization to ensuring fairness and meeting SLAs, scheduling algorithms must balance competing goals while optimizing system performance. Their success is measured by their ability to meet these objectives in dynamic, heterogeneous, and large-scale cloud environments.

3.5 Metrics for evaluating job scheduling performance

Evaluating the performance of job scheduling algorithms in cloud computing requires using specific metrics that measure various aspects of system efficiency, resource utilization, and task execution. These metrics provide a quantitative basis for comparing different schedul-

Table 3 Objectives of job scheduling in cloud computing

Objectives	Description
Minimizing makespan	Minimizing makespan reduces the time required to complete all tasks, improving system throughput and ensuring prompt processing. This is particularly important in environments with high task volumes (Kaur et al. 2022c; Raju et al. 2013)
Maximizing resource utilization	Effective scheduling maximizes the usage of computational resources such as CPU, memory, and storage, minimizing idle times and reducing operational costs (Sharma and Sharma 2013; Loganathan and Mukherjee 2015)
Ensuring fairness in resource allocation	Fair scheduling prevents resource monopolization and ensures equitable distribution of resources among users, particularly in multi-tenant systems with varying priorities (Wang et al. 2014b; Tang et al. 2020)
Minimizing energy consumption	Energy-aware scheduling algorithms reduce power consumption by avoiding idle times and dynamically scaling resources, contributing to sustainability and cost savings (Hu et al. 2019, 2023)
Meeting SLAs	SLA-aware scheduling ensures that tasks are completed within agreed timeframes, avoiding penalties and maintaining user satisfaction (Nayak et al. 2015; Islam et al. 2021)
Reducing task latency	Scheduling algorithms that minimize latency improve system responsiveness, particularly for real-time applications like online gaming and financial trading (Memari et al. 2022; Aburukba et al. 2020)
Load balancing	Distributing tasks evenly across resources prevents overload and improves reliability in distributed systems with heterogeneous resources (Wang et al. 2014a; Paul et al. 2011)
Minimizing cost	Cost-aware scheduling reduces resource allocation and data transfer expenses for providers and users, achieving cost efficiency (Mansouri and Javidi 2020; Cheng et al. 2022a)
Ensuring scalability	Scalable scheduling algorithms handle increasing workloads without performance degradation, ensuring adaptability in growing cloud environments (Khallouli and Huang 2022; Alsaih et al. 2020)
Handling fault tolerance and failures	Fault-tolerant scheduling reschedules tasks on alternate resources in case of hardware, network, or software failures, ensuring system reliability (Xu et al. 2023; Indhumathi et al. 2023)
Supporting data locality	Data-locality-aware scheduling minimizes data transfer times by assigning tasks to resources close to the required data, reducing congestion and delays (Cheng et al. 2021; Mansouri and Javidi 2020)
Handling job dependencies	Dependency-aware scheduling respects task dependencies, ensuring correct execution order and efficient resource usage, particularly in workflows with complex relationships (Chen et al. 2022b; Chung et al. 2020)
Supporting multi-objective optimization	Multi-objective optimization balances conflicting goals, such as minimizing makespan while maximizing fairness, using trade-off techniques like Pareto optimization (Cui et al. 2023; Mohammadzadeh and Masdari 2023)

ing approaches and identifying areas for improvement. This subsection describes the most commonly used metrics for evaluating job scheduling performance in cloud environments. Table 4 provides a detailed overview of crucial job scheduling metrics, including their definitions and significance. This structured representation highlights the critical role of these metrics in optimizing cloud system performance and resource management.

In summary, these metrics provide a comprehensive framework for evaluating the performance of job scheduling algorithms in cloud computing. By analyzing metrics such as makespan, resource utilization, load balancing, and energy consumption, researchers and cloud providers can assess the effectiveness of their scheduling strategies and optimize them for better performance, scalability, and cost efficiency.

3.6 Simulation frameworks for job scheduling

Simulation frameworks are essential for evaluating job scheduling algorithms in cloud computing environments (Sanjalawe 2023). These frameworks provide a controlled and repeatable environment for testing, enabling researchers to model diverse scenarios and analyze the performance of scheduling strategies. By simulating resource allocation, workload distribution, and energy consumption, these tools help identify the weaknesses and strengths of various algorithms without the cost and complexity of real-world implementation. The wide variety of simulation frameworks available today reflects the diversity of cloud computing research. Some tools focus on energy efficiency, workflow management, or edge computing. Selecting an appropriate framework depends on the research goals, the desired level of detail, and the specific challenges being addressed, such as latency reduction, scalability, or cost optimization.

Table 5 compares prominent simulation frameworks for job scheduling in cloud environments.

4 Existing AI-driven job scheduling techniques

As depicted in Fig. 9, this section presents a comprehensive taxonomy of AI-driven job scheduling techniques, categorized into three main branches: Machine Learning-based, Optimization-based, and Hybrid AI-based approaches. Each category encompasses various methodologies tailored for different scheduling scenarios. The Machine Learning-based branch is further divided into reinforcement, deep learning, and supervised/unsupervised learning techniques, focusing on data-driven optimization. The Heuristic and Meta-Heuristic-based section highlights local search heuristics and population-based methods like evolutionary and bio-inspired algorithms. The Hybrid AI-based methods blend traditional AI techniques with modern scheduling algorithms, offering a multi-faceted approach to improving efficiency in job scheduling within cloud computing environments. These techniques are essential for addressing complex scheduling challenges and ensuring optimal resource utilization and performance.

4.1 Machine learning approaches

Machine learning has become a cornerstone in optimizing job scheduling processes, particularly in cloud computing environments. By leveraging large datasets, machine learning algorithms can identify patterns and make predictions that improve scheduling efficiency, resource allocation, and overall system performance (Rjoub et al. 2021). This section delves into various machine learning approaches that play a critical role in job scheduling, including supervised, reinforcement, unsupervised, and deep learning. Each of these techniques offers distinct advantages in handling scheduling challenges, from training models on labelled data to exploring new scheduling strategies through trial and error or identifying hidden patterns in data. As the demand for more efficient and scalable scheduling solutions grows, these machine-learning methodologies continue to push the boundaries of traditional job scheduling techniques.

4.1.1 Supervised learning

Supervised learning involves training a machine learning model on labelled datasets where the input and output are known. In job scheduling, supervised learning can be used to predict job execution times, resource needs, or potential bottlenecks, leading to more accurate and efficient scheduling decisions.

The authors of Onyema et al. (2024) proposed a task scheduling model for multi-cloud environments, presented in Fig. 10, to enhance performance by assigning tasks to appropriate resources. The proposed model operates in three stages. The first stage utilises a supervised classification technique to classify the features. The second and third stages focus on the execution and minimization of the completion time, average waiting time, turnaround time, and render duration. The execution time of tasks is modelled using exponential and normal distributions. The proposed model demonstrates improved outcomes based on the specified performance criteria. Another study proposed by Ranichandra et al. (2018) utilized supervised learning techniques for the task classification or resource allocation phases within the enhanced hyper-heuristic methodology. The classification method was used to predict execution times, classify tasks based on historical data, and select the most appropriate scheduling strategy based on learned patterns. By training models on historical scheduling data, the system improves its decision-making process for task scheduling, especially in environments with varying loads or resource demands. This enhances the adaptability and efficiency of the scheduling approach beyond traditional heuristic methods. Similarly, in Liu et al. (2019), the authors addressed the problem of inefficient resource allocation in hybrid cloud environments caused by the continuous arrival of diverse job types. The proposed solution is an adaptive job scheduling algorithm based on queuing theory. Jobs are classified using the logistic regression method, while cloud resources are categorized based on their utility, considering the heterogeneity within the private cloud. A queuing model is established, and an adaptive genetic algorithm is applied to manage job queue arrival rates, which drives resource allocation. Performance comparisons with existing algorithms show improvements in job response times and throughput.

Loganathan et al. (2017) designed a job scheduling mechanism to optimize energy consumption in cloud computing by efficiently assigning jobs to VMs. The approach focuses on reducing the number of active hosts, which minimizes energy use. A classification approach

Table 4 Metrics of job scheduling in cloud computing

Metric	Description and References
Makespan	Makespan refers to the time required to complete all tasks in a given schedule. It is defined as $\text{Makespan} = \max(T_i + C_i)\ \forall i \in \{1, 2, \ldots, N\}$, where T_i is the start time and C_i is the completion time of task *i*. Minimizing makespan is a key objective for improving system efficiency (Belgacem and Beghdad-Bey 2022)
Resource utilization	Resource utilization measures the percentage of available resources actively used during scheduling. It is calculated as $\text{Resource\_Utilization} = \frac{\sum_{i=1}^{N} U_i}{N}$, where U_i is the utilization of resource *i*. High utilization reflects effective resource allocation (Shu et al. 2021)
Load balancing	Load balancing ensures even distribution of workloads across resources to avoid overloading. The load balancing factor is defined as $\text{Load\_Balancing\_Factor} = \frac{\sum_{i=1}^{N} L_i}{N}$, where L_i is the load on resource *i*. An ideal factor close to 1 indicates balanced workloads (Li et al. 2020b)
Throughput	Throughput measures the number of tasks completed in a specific time frame. It is expressed as $\text{Throughput} = \frac{\text{Total Number of Jobs Completed}}{\text{Total Time}}$, reflecting the efficiency of the scheduling algorithm (Geng et al. 2020)
Energy consumption	Energy consumption refers to the total power used during task execution. It is calculated as $\text{Energy\_Consumption} = \sum_{i=1}^{N} P_i \cdot T_i$, where P_i is the power consumed and T_i is the active time of resource *i*. Energy-efficient scheduling minimizes this value (Chen et al. 2022b)
Job waiting time	Job waiting time is the duration a task remains in the queue before execution. The average waiting time is given as $\text{Average\_Waiting\_Time} = \frac{\sum_{i=1}^{N} W_i}{N}$, where W_i is the waiting time of task *i*. Lower waiting times improve system responsiveness (Mansouri and Javidi 2020)
Task completion time	Task completion time is the total duration from task submission to completion, calculated as $\text{Task\_Completion\_Time} = C_i - T_i$, where C_i is the completion time and T_i is the start time of task *i*. Lower completion times enhance performance (Ibrahim and Bassiouni 2020)
Fairness index	The fairness index measures equitable resource distribution. It is calculated using Jain's fairness index: $\text{Fairness\_Index} = \frac{\left(\sum_{i=1}^{N} x_i\right)^2}{N \cdot \sum_{i=1}^{N} x_i^2}$, where x_i is the resource allocated to task *i*. A value close to 1 indicates high fairness (Tang et al. 2020; Li et al. 2021b)
Scalability	Scalability evaluates how well the scheduling algorithm handles increasing tasks or resources. It is assessed by tracking performance metrics such as makespan, throughput, and resource utilization as the system grows (Ngo et al. 2022; Muniswamy and Vignesh 2022)
Reliability	Reliability reflects the algorithm's ability to handle failures, such as resource outages or network disruptions. It is measured by the success rate of task completion in environments with simulated failures (Tang 2021; Medara and Singh 2021)
Latency	Latency refers to the delay between task submission and execution start time. It is defined as $\text{Latency} = T_s - T_r$, where T_s is the submission time and T_r is the start time. Low latency is crucial for real-time applications (Tang et al. 2022)
Makespan variance	Makespan variance measures the consistency of scheduling under varying conditions. It is calculated as $\text{Makespan\_Variance} = \frac{\sum_{i=1}^{N} (M_i - \overline{M})^2}{N}$, where M_i is the makespan in scenario *i* and $\overline{M}$ is the average makespan (Chraibi et al. 2021)

Table 4 (continued)

Metric	Description and References
Cost efficiency	Cost efficiency measures the financial cost per job, calculated as $\text{Cost\_Efficiency} = \frac{\text{Total Cost}}{\text{Number of Jobs Completed}}$. Cost-aware scheduling minimizes expenses without compromising performance (Mansouri and Javidi 2020)

was utilized to classify jobs into three types and assign them based on a preemption policy, utilizing the earliest available VM. This approach increases the utilization of active hosts and reduces energy consumption. Simulations conducted with CloudSim demonstrate significant energy savings compared to non-energy-aware and energy-aware algorithms. The obtained results proved the high performance of the utilized approach in predicting the optimal job type based on historical data, achieving better scheduling and improving energy efficiency.

Baomin et al. (2011) introduced a job scheduling algorithm based on the Berger model, addressing the commercialization and virtualization features of cloud computing. The algorithm incorporates dual fairness constraints: first, it classifies user tasks according to QoS preferences, using a general expectation function to ensure fairness in resource selection. Second, it defines a fairness justice function to evaluate resource allocation. The algorithm was implemented in CloudSim, with experimental results demonstrating effective task execution and improved fairness. Another study by Li et al. (2021a) introduces a cloud-edge collaboration architecture. This method aimed at optimizing the placement and caching of streaming media. A cache-aware scheduling model, using neighbourhood search, is proposed to address four sub-problems: job classification, node resource allocation, node clustering, and cache-aware job scheduling. Jobs are first classified into three categories, with resources allocated based on job execution conditions. Nodes with similar capabilities are clustered, and jobs are cached using a delay-waiting strategy. For jobs lacking data locality, scheduling is adjusted according to neighbourhood search results. The proposed scheduling algorithm significantly reduces content transmission delay, lowers placement costs, shortens job execution time, and enhances the processing capacity of cloud data centres.

Kaur et al. (2022a) focused on addressing the rising energy consumption of cloud data centres by integrating renewable energy sources with conventional power grids. The proposed solution introduces a two-phase multi-objective optimization scheme for workload classification, job scheduling, and virtual machine placement in cloud DCs. In phase I, a random forest-based feature selection method (Boruta) is used to select relevant features for incoming workloads, followed by workload classification using a support vector machine model with locality-sensitive hashing. In phase II, a multi-objective optimization problem is formulated to manage job scheduling and VM placement, considering SLA, energy cost, carbon footprint, and RES availability. The problem is solved using an enhanced heuristic approach based on a greedy strategy. Experimental results demonstrate improvements in energy utilization, energy cost, and carbon footprint, with only a minor decrease in SLA performance. Another study addressed the energy consumption in cloud computing was proposed by Saroha and Rana (2019). This study proposed a multi-layer scheduling approach to optimize resource utilization and reduce energy usage. This approach involves a load balancer that manages multiple job queues across a cloud network. Client requests are classified by urgency, with high-priority tasks assigned to higher-configuration servers and lower-priority tasks to lower-configuration servers, contributing to energy savings. The

Table 5 Comparison of simulation frameworks for job scheduling

Framework	Platform	Language/script	Graphical support	Communication model support	Supported OS environment	Output result format
CloudSim (Goyal et al. 2012)	SimJava	Java	No	Limited	Windows/Linux	Text
iFogSim (Gupta et al. 2017)	SimJava	Java	Limited	Limited	Windows/Linux	Text
GreenCloud (Kliazovich et al. 2012)	NS2	C++/OTcl	Limited	Full	Linux	Dashboard plots
SimGrid (Casanova 2001)	POSIX	C	No	Full	Windows/Linux	Text
WorkflowSim (Chen and Deelman 2012)	SimJava	Java	Limited	Limited	Windows/Linux	Text
EdgeCloudSim (Sonmez et al. 2018)	SimJava	Java	No	Limited	Windows/Linux	Text
CloudSched (Byrne et al. 2017)	N/A	Java	No	Limited	Windows/Linux	Text
MDCSim (Lim et al. 2009)	Java-based	Java	No	Full	Windows/Linux	Text
NetCloudSim (Garg and Buyya 2011)	NS2	Java	No	Full	Windows/Linux	Text
ContainerCloudSim (Piraghaj et al. 2017)	SimJava	Java	No	Limited	Windows/Linux	Text
IoTSim (Zeng et al. 2017)	SimJava	Java	No	Limited	Windows/Linux	Text

performance of the proposed approach is evaluated against various scheduling and load balancing algorithms like Round Robin, Minimum Completion Time, and Opportunistic Load Balancing. Comparative results show better energy efficiency and improved resource utilization with the proposed method.

4.1.2 Reinforcement learning

Reinforcement learning focuses on decision-making by rewarding or penalizing actions based on their outcomes. In job scheduling, reinforcement learning models can autonomously explore various scheduling strategies, learning from their successes and failures to maximize performance and minimize resource consumption over time.

Many studies have utilized the Reinforcement Learning approach to address the job scheduling problem in cloud computing, most of which used its deep version. As illustrated in Fig. 11, Shi et al. (2022) utilized a Deep Reinforcement Learning (DRL)-based approach to optimize Spark job scheduling in hybrid cloud environments. The primary goal of this DRL-based scheduler is to enhance cluster performance and minimize the total cost of cluster usage. By employing a reinforcement learning agent, the system adaptively learns the characteristics of various jobs and the specifics of hybrid cloud environments. This learning allows the agent to make more informed decisions when scheduling jobs, ultimately reducing cluster usage costs and improving job execution efficiency.

In Cheng et al. (2022b), the authors introduced a DRL-based approach for scheduling real-time jobs in hybrid cloud environments, aiming to optimize monetary costs while maintaining high quality of service and low response times. Unlike other methods primarily focusing on batch jobs, this approach is tailored to handle real-time tasks. The DRL agent learns to make optimal decisions in real-time by selecting the most appropriate virtual machines for job execution, with the learning process driven by rewards based on performance. Experimental results show that this DRL-based method is more cost-efficient than existing scheduling techniques.

A novel Federated DRL (FDRL) approach was proposed by Wang et al. (2024) to address dynamic job scheduling challenges in a Cloud-Edge Collaborative Manufacturing System. The method aims to improve scheduling performance across distributed factories by sharing scheduling knowledge while preserving data privacy. Each factory aims to minimize makespan and energy consumption, accounting for machine warm-up and real-time dynamics. The FDRL model aggregates hidden parameters from heterogeneous factories using a two-phase algorithm of local training and global aggregation. The approach maintains privacy by avoiding direct data sharing and incorporates constraints in the loss functions to refine training. Experimental results show that the FDRL-based approach outperforms traditional methods, reducing makespan by up to 8.9% and energy consumption by 22.3%.

Recent advancements in DRL have proven highly effective in optimizing job scheduling within dynamic and complex cloud environments. Wei et al. (2018) proposed an intelligent QoS-aware job scheduling framework based on DRL to improve resource management and optimise performance in cloud-based applications. Their DRL-based job scheduler autonomously learns to make appropriate job-to-VM assignments directly from its experiences without requiring prior knowledge. By leveraging synthetic workloads and real-world NASA workload traces, the framework can significantly reduce average job response time by 40.4% while ensuring a high job success rate of over 93% even under varying workload

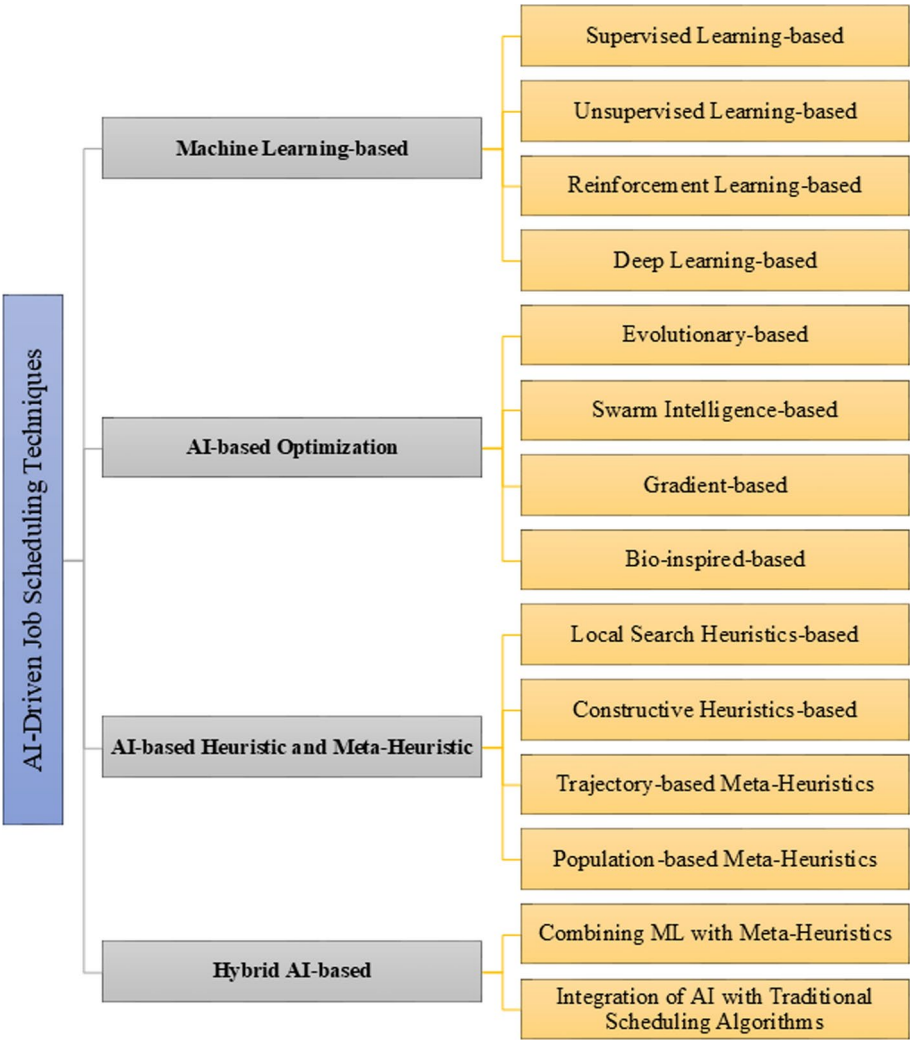

Fig. 9 Taxonomy of existing AI-driven job scheduling techniques

scenarios. Li and Hu (2019) introduced DeepJS, a DRL-based job scheduling algorithm built on the bin-packing problem framework. This method focuses on minimizing makespan and maximizing throughput in a cloud data centre. Unlike traditional heuristic approaches, which struggle to adapt to changing environments, DeepJS automatically learns an optimal fitness calculation method directly from its scheduling experiences. The results of trace-driven simulations highlight DeepJS's superior performance, surpassing heuristic-based algorithms in convergence speed and generalization.

In addressing real-time job scheduling problems, Cheng et al. (2022a) presented a DRL-based job scheduler specifically designed for dynamic cloud workloads where user requests arrive unpredictably. They employ a Deep Q-learning Network model to schedule jobs in real-time, ensuring high Quality of Service and cost efficiency. Their approach reduces the

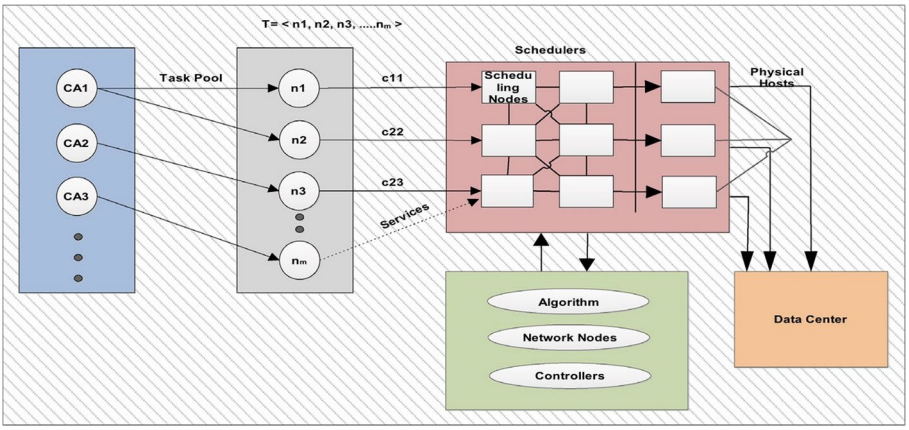

Fig. 10 Architecture of proposed model by Onyema et al. (2024)

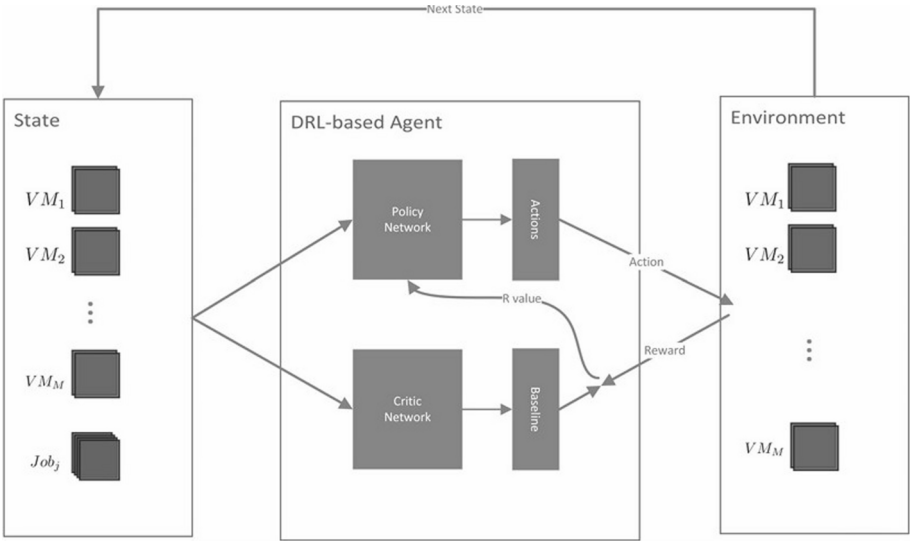

Fig. 11 Architecture of proposed approach by Shi et al. (2022)

monetary costs of executing jobs on virtual instances and outperforms conventional real-time scheduling algorithms. By dynamically adjusting to real-time user requests, the Deep Q-learning Network model-based scheduler proves its effectiveness in managing unpredictable workloads, improving scheduling accuracy, and optimizing resource utilization in cloud environments (Fig. 12).

Li and Peng (2022) proposed a DRL-based job scheduling approach tailored to complex cloud environments with multiple users, queues, and data centres. The study presents a system model that aims to minimize transmission, waiting, and execution times. Formulating the problem as a DRL-based optimization task enables the scheduling agent to learn from its environment and improve decision-making. Experiments demonstrate that their method

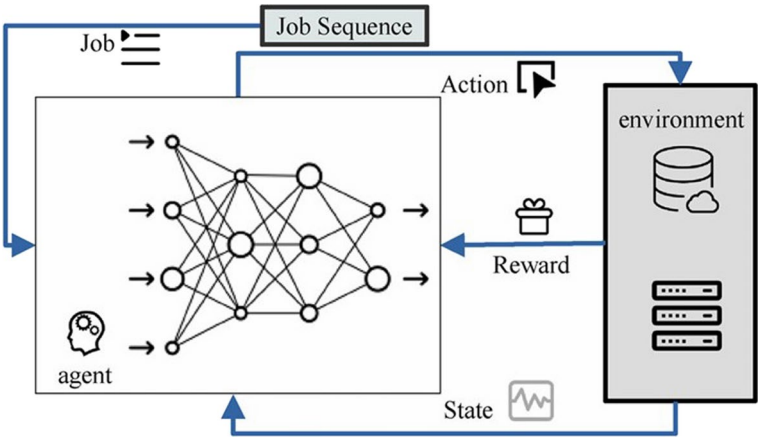

Fig. 12 Architecture of proposed approach by Cheng et al. (2022a)

effectively reduces overall job makespan compared to traditional scheduling algorithms, such as random scheduling, round-robin, first-fit, and optimal-fit. An advanced scheduling technique called Deep Adversarial Imitation Reinforcement Learning (AIRL) was proposed by Huang et al. (2022) to overcome limitations in existing DRL-based solutions for time-sensitive cloud job scheduling. In cloud computing, where job trajectories can be long and complex, standard DRL approaches may struggle to find optimal solutions due to delayed rewards. The AIRL framework improves on this by maximizing the job success rate and minimizing response times for real-time workloads. The AIRL framework outperforms conventional DRL methods through adversarial learning and imitation techniques and ensures better job scheduling efficiency, even under varied workloads and resource configurations.

In Balla et al. (2018), the authors address the problem of ensuring reliable task execution in cloud environments, particularly under resource constraints. They propose an adaptive RL-based scheduling method, combined with queuing theory, to handle dynamic changes in resource availability and task requirements. Their adaptive action-selection mechanism allows the system to choose suitable VMs based on queue buffer sizes and uncertainty values. The proposed method significantly improves task execution success rates, resource utilization, and response times compared to traditional scheduling policies like greedy and random approaches.

Yang et al. (2011) addressed job scheduling problems in utility-based cloud computing under overloading conditions and hardware/software failures. They introduce an RL-based job scheduling algorithm that is fault-tolerant and maximizes long-term utilities. The proposed approach handles system failures and recovery scenarios, ensuring that job scheduling remains reliable despite adverse conditions. The obtained experimental results, which compare their method with the Resource-constrained Utility Accrual algorithm, Utility Accrual Packet scheduling algorithm, and LBESA, show that the proposed RL-based algorithm performs favourably in maintaining system performance and utility under failure conditions. Similarly, Hu et al. (2012) tackled job scheduling problems in cloud-based power system computing environments by addressing the issue of hardware/software failure and recovery. The authors propose an RL-based scheduling algorithm that focuses on making job scheduling fault-tolerant in utility-based cloud systems. The method ensures that job

scheduling can adapt to failure and recovery events, providing a reliable solution for cloud systems supporting power system computations. The RL-based method demonstrates superior performance compared to other utility-based job scheduling algorithms.

Collectively, these studies underscore the growing importance of RL and DRL techniques in cloud job scheduling. They offer robust solutions to optimize job allocation, enhance system reliability, reduce resource costs, and handle real-time or fault-prone environments. RL enables these systems to learn and adapt to changing environments dynamically, improving upon traditional heuristic-based or static scheduling methods that often fail to account for modern cloud computing workloads' complex and transient nature.

4.1.3 Unsupervised learning

Unsupervised learning is applied when the dataset lacks labelled outputs. In job scheduling, unsupervised learning algorithms can discover hidden patterns, such as job clusters or resource usage trends, allowing schedulers to allocate resources more effectively without predefined outcomes.

Singhal et al. (2024) proposed a novel job scheduling approach using the Rock Hyrax model, where unsupervised learning is applied through clustering resources based on attributes such as processing power, storage, and network connectivity. By grouping similar resources, the scheduling algorithm can more efficiently map jobs to available resources, reducing makespan and energy consumption. Figure 13 summarizes the workflow of the proposed approach. This unsupervised clustering improves overall system performance

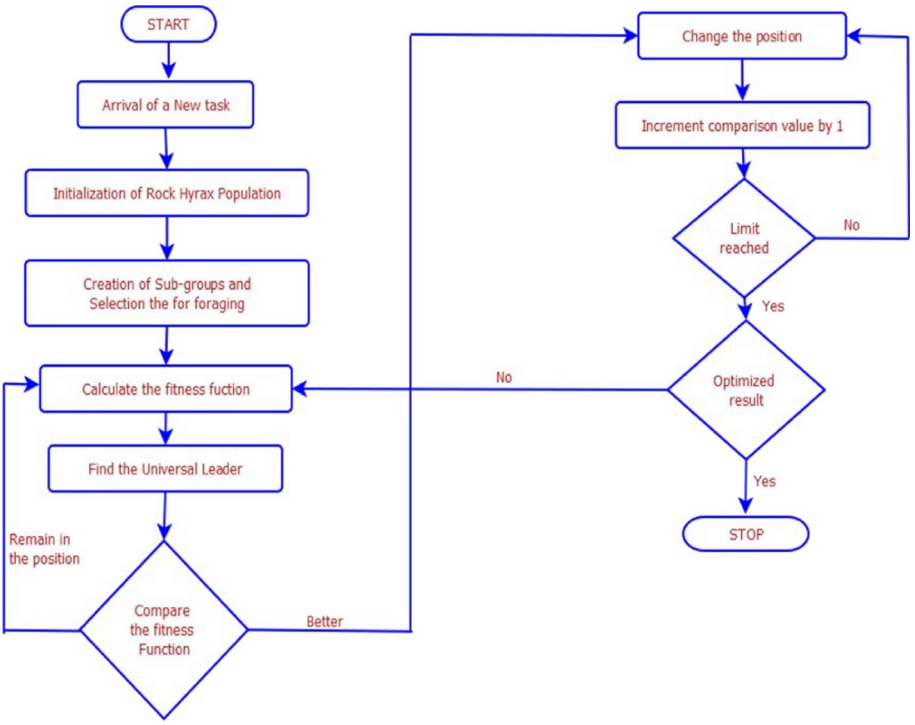

Fig. 13 Workflow of proposed approach by Singhal et al. (2024)

compared to existing scheduling algorithms like ACO and PSO, with a reduction in makespan by 2–9% and energy consumption by 7–23%.

Khaleel (2023) incorporate unsupervised learning in their dual-phase metaheuristic algorithm for IoT job scheduling. In the first phase, a clustering approach groups computing nodes based on their utilization levels and performance-to-power ratio. This clustering ensures that tasks are assigned to the most suitable VMs, effectively balancing loads, minimizing resource fragmentation, and reducing energy consumption. The method significantly improves task scheduling in IoT applications by leveraging unsupervised learning for clustering. Similarly, Kaur et al. (2023) utilize k-means clustering, a classic unsupervised learning algorithm, in their two-phase cloud scheduling architecture. The users and physical machines are clustered based on Quality of Service parameters such as response time and energy consumption. The clustering facilitates the efficient migration of users between physical machines, ensuring optimal resource usage and cost reduction in a multi-cloud environment. This proposed approach also incorporates CO2 emissions as an evaluation parameter, further enhancing the environmental efficiency of the cloud system.

A Dynamic Cluster Job Scheduling Optimization Algorithm (DCJS_DI) based on data irreversibility was proposed by Sun et al. (2019) to improve cluster rendering throughput. A clustering method is used in the rendering process to organize jobs and reduce resource fragmentation. The proposed method addresses job hunger and improves resource efficiency by better-dispatching tasks across the cluster. In Suganya et al. 2023, the authors proposed a novel resource clustering algorithm named Identicalness Split Up Periodic Node Size. This algorithm clusters cloud resources based on their characteristics, forming efficient resource clusters to improve job scheduling in the cloud environment. Resource clustering ensures better resource allocation, reduces job scheduling complexity, and improves overall system performance. The comparison with existing methods demonstrates that Identicalness Split Up Periodic Node Size produces near-optimal solutions, which enhance the cloud job scheduling process by forming appropriate resource clusters.

Karthick et al. (2014) present the Multi-Queue Scheduling algorithm, where jobs are clustered based on burst time. This unsupervised clustering method helps organise jobs more efficiently for scheduling, addressing the fragmentation problem common in traditional methods like First Come First Serve and Shortest Job First. By dynamically clustering jobs based on their execution times, the Multi-Queue Scheduling method reduces starvation and optimizes cloud resource utilization. Gouasmi et al. (2017) introduced a fully distributed scheduling algorithm called FedSCD, designed for MapReduce data-intensive applications across geo-distributed clusters in federated clouds. using the proposed approach, the tasks are clustered and scheduled closer to their data sources to reduce data transfer and VM costs. The proposed clustering approach enables significant cost reductions and efficient resource allocation, achieving a 40% cost reduction for MapReduce jobs.

4.1.4 Deep learning

Deep Learning (DL) has emerged as a pivotal technology in cloud computing and job scheduling, with various applications in resource optimization, performance improvement, and Big Data analytics. Several studies leverage the power of deep learning techniques, such as Artificial Neural Networks (ANNs), to tackle complex challenges in scheduling, data processing, and resource management.

Goga et al. (2019) evaluated the performance of ANNs with multiple hidden layers for job scheduling and data processing tasks in cloud environments. The study focuses on the trade-offs between processing time and data size, especially when training data is read from memory or disk. ANNs offer significant advantages for Big Data analytics, but their performance is heavily influenced by data dimensionality, sample size, and memory constraints. The authors demonstrate that effective scheduling of ANNs in Big Data environments can reduce processing time and optimize memory usage.

In Kim et al. (2024), the authors proposed an SLO-aware deep learning job scheduling model for FPGA-GPU edge cloud computing. The model dynamically updates accelerator configurations based on DL job requirements while considering Service-Level Objectives (SLOs). The scheduling algorithm optimizes the allocation of deep learning workloads between FPGAs and GPUs, improving energy consumption and maintaining SLO compliance. The results demonstrate that DL job scheduling can significantly enhance both performance and energy efficiency when optimised for hardware accelerators, reducing computation costs in cloud computing environments.

Lin et al. (2018) introduced a cloud job scheduling strategy based on the Deep Q-network (DQN) algorithm, a combination of reinforcement learning and deep learning techniques. The strategy aims to minimize the average job completion time and job slowdown in cloud environments. The deep learning-based scheduler improves decision-making for resource allocation, showing superior performance over standard policy gradient algorithms in job completion time and convergence speed. This highlights the capability of deep reinforcement learning to enhance cloud resource management by providing real-time scheduling decisions.

Overall, these studies illustrate the impact of Deep Learning in cloud job scheduling. Techniques such as ANNs and Deep Q-networks are leveraged to improve scheduling efficiency, reduce computational costs, and optimize resource allocation in complex cloud environments.

Table 6 compares machine learning-based approaches for solving specific problems. It organizes the information into four key dimensions: objectives, methods, findings, and weaknesses/shortcomings. Each reference included in the table is analyzed based on these dimensions, enabling a clear comparison of the strengths and limitations of each approach.

4.2 AI-based optimization techniques

Optimization techniques have long been central to solving complex job scheduling problems, ensuring that resources are efficiently utilized while minimizing processing time, costs, and other critical parameters. In AI-driven job scheduling, optimization techniques provide a structured approach to finding the best solution from a set of potential candidates. This section covers various optimization methodologies, including evolutionary techniques, swarm intelligence techniques, bio-inspired techniques, and gradient-based optimization. These methods, inspired by biological processes, collective behaviour, and mathematical modelling, offer innovative ways to tackle the inherent complexities of job scheduling in dynamic environments. By applying these approaches, schedulers can achieve superior performance, scalability, and robustness in decision-making processes.

Table 6 A comparative analysis of various machine learning-based approaches

References	Methods	Findings	Latency	Energy consumption	Cost	Pros	Cons
Onyema et al. (2024)	Supervised classification	Reduced completion and waiting times	Low	Moderate	Low	High accuracy in predictions	Limited scalability to large datasets
Ranichandra et al. (2018)	Supervised learning	Improved scheduling efficiency and adaptability	Moderate	Low	Moderate	Flexible for task variability	Requires significant training data
Liu et al. (2019)	Adaptive job scheduling algorithm with job classification	Better response times and throughput	Moderate	Moderate	Low	Adaptable to changing conditions	High computational overhead
Loganathan et al. (2017)	Classification-based	Improved energy efficiency	Moderate	Low	Low	Reduces idle energy usage	Limited for complex workloads
Kaur et al. (2022a)	Two-phase optimization with random forest	Reduced energy costs and carbon footprint	Moderate	Low	Moderate	Environmentally beneficial	Complex implementation
Shi et al. (2022)	DRL	Reduced cluster costs and better execution efficiency	Low	Moderate	High	Learns dynamically	High cost of initial setup
Cheng et al. (2022b)	DRL scheduler	Improved cost efficiency	Moderate	Moderate	Moderate	Balances multiple objectives	Long training times
Wang et al. (2024)	Two-phase FDRL model	Reduced makespan by 8.9% and energy by 22.3%	Moderate	Low	High	Effective in collaborative systems	Requires federated infrastructure
Singhal et al. (2024)	Clustering	Reduced makespan and energy by 7–23%	Low	Low	Moderate	Enhances resource grouping	Ineffective for unstructured data
Khaleel (2023)	Clustering computing nodes	Improved scheduling efficiency	Moderate	Low	Low	Simplifies node management	Limited adaptation to dynamic changes
Kaur et al. (2023)	k-means clustering	Enhanced efficiency and multi-cloud resource usage	Moderate	Moderate	Moderate	Efficient for specific patterns	Struggles with high-dimensional data
Suganya et al. (2023)	Identicalness split up periodic node size algorithm	Near-optimal solutions for resource allocation	Low	Moderate	Moderate	Precise allocations	Lacks flexibility in diverse scenarios
Kim et al. (2024)	Dynamic DL scheduling algorithm	Improved energy efficiency and reduced costs	Low	Low	Moderate	Suitable for real-time applications	Limited by training dependencies
Lin et al. (2018)	Deep Q-network	Improved job completion time	Moderate	Low	Moderate	Adaptive to task complexity	Slower initial training phase

4.2.1 Evolutionary techniques

Evolutionary techniques, such as Genetic Algorithms (GA), are inspired by natural selection. These algorithms iteratively evolve a population of potential solutions by selecting, crossing, and mutating candidates based on their fitness scores. In job scheduling, evolutionary techniques help optimize resource allocation and job sequencing, adapting to changing conditions while avoiding local optima.

Lane et al. (2022) investigated the role of optimization algorithms in enhancing cloud computing performance through job scheduling. The authors proposed a dynamic hierarchical connection system that optimizes the scheduling using artificial intelligence search algorithms. Specifically, they compare the effectiveness of GA and simulated annealing in optimizing the dynamic structure. The results demonstrate that these algorithms significantly reduce the makespan of heterogeneous tasks by efficiently allocating them to available resources, particularly in environments with limited resources.

A hybrid task scheduling approach for cloud computing that integrates GAs with multiple generic scheduling heuristics to enhance optimization potential was proposed in Remesh et al. (2023). Acknowledging the varied performance of different scheduling methods based on task characteristics and workload size, the proposed solution aims to optimize key performance metrics, including makespan, average flow time, throughput, and average waiting time. Implemented within the CloudSimPlus framework, experimental results show that the hybrid approach consistently outperforms individual heuristics across various workload scales, with its optimization potential increasing as the task volume grows. Focusing on flow time as the objective function also leads to complementary improvements in other performance metrics, further boosting overall system efficiency. Another study used the original GA proposed by Sahraei et al. (2019) to address the problem of optimizing time and cost in cloud job scheduling. The proposed model focuses on managing jobs to minimize makespan, reduce costs, and enhance resource utilization. Jobs are created in a genetic format, and various scenarios are analyzed based on different fitness functions and population structures. The model's performance is compared with other scheduling models, such as MAX-MIN and MIN–MIN, demonstrating significant cost, makespan, and utilization improvements. The results highlight the effectiveness of GAs in managing job scheduling in cloud environments.

Ali and Ali (2023) proposed a cloud-fog edge scheduling approach that integrates the Catastrophic GA (CGA) with a blockchain-based trust framework. The objective is to optimize job scheduling by reducing service call latency and achieving global optimization. The CGA is enhanced with mutation, crossover operators, roulette selection, and cataclysm to prevent premature convergence and expand the search area. Additionally, Berger's theory is applied to ensure trust-enabled interactions among nodes. Experimental findings reveal that this method reduces task completion time by over 97%, effectively resolving local optimization problems and substantially improving job scheduling efficiency.

Within the Google Cloud Platform (GCP), Kumar et al. (2024b) explored using GA to optimize job scheduling and resource allocation in cloud computing. The GA is designed to refine task-to-resource allocations based on cost, execution time, and resource consumption. By comparing GA with other methods like DBSCAN and Google Kubernetes Engine, the authors demonstrate GA's potential to address complex optimization problems, adapt to

constraints, and enhance scalability. The study provides valuable insights for researchers and practitioners in selecting efficient job scheduling strategies in cloud environments.

Frank Vijay (2019) focused on the role of data analysis in cloud computing environments and the importance of job scheduling to meet the needs of numerous users. The authors employ GA to improve the scheduling process, particularly in heterogeneous multiprocessor systems. Sharma and Lakra (2020) discussed the environmental impact of computing technologies, particularly the increasing carbon footprint associated with cloud computing. The authors highlighted the need for a shift toward green computing, which emphasizes minimizing environmental adverse effects. Though not a specific technology, green IT represents transitioning from traditional computing methods to more eco-friendly practices. The authors argue that this shift is essential for reducing energy consumption and mitigating the global warming effects caused by the large-scale use of computing resources. Wang et al. (2014c) proposed a novel multi-objective bi-level programming model for improving energy efficiency in cloud computing data centres. The model integrates an energy-aware data placement policy with a locality-aware multi-job scheduling scheme based on the MapReduce framework. The authors develop a multi-objective GA using MOEA/D to address the large-scale optimization problem of job scheduling in the cloud. As depicted in Fig. 14, numerical experiments demonstrate that the proposed model significantly improves

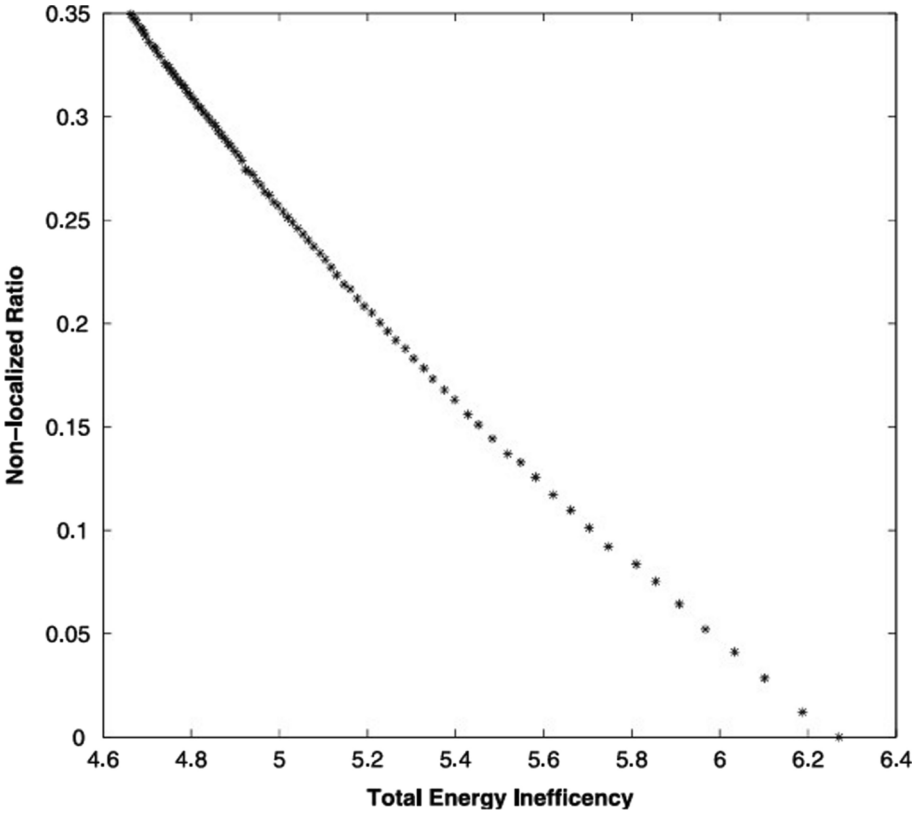

Fig. 14 Results obtained from the proposed approach in Wang et al. (2014c)

energy efficiency by dynamically adjusting data locality according to network conditions and optimizing task placement through a specially designed GA.

Ananth and Chandrasekaran (2016) address the issue of job scheduling in cloud computing, aiming to maximize resource utilization while adhering to Service Level Agreements and deadline constraints. The authors introduced a job scheduling technique based on cooperative game theory and the Non-dominated Sorting GA II (NSGA-II) to provide Pareto optimal solutions. The proposed approach focuses on minimizing deadline violations and makespan while maximizing profit for cloud service providers, offering a comprehensive solution for effective resource management in cloud environments.

4.2.2 Swarm intelligence techniques

The collective behaviour of decentralized, self-organized systems, such as flocks of birds or colonies of ants inspires swarm intelligence techniques. Algorithms like Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) mimic these natural systems to explore solution spaces efficiently. In job scheduling, swarm intelligence algorithms provide robust methods for handling large-scale, distributed scheduling problems with multiple constraints.

Kuppusamy et al. (2022) introduced a Dynamic Opposition Learning-based Social Spider Optimization algorithm to tackle job scheduling in fog computing. This algorithm enhances individual superiority and workflow scheduling by minimizing the makespan, reducing energy consumption, and improving resource allocation. The algorithm's performance was validated using the FogSim simulator, and the results showed that the proposed method achieved a 10–15% improvement in CPU utilization and a 5–10% reduction in energy consumption compared to five existing metaheuristic techniques. Khaleel (2024) addressed workload imbalance in cloud data centres by introducing the Regional Awareness Dynamic Scheduling Algorithm. This approach classifies tasks, estimates resource capacities, and uses a merge-and-split coalitional game-theoretic process to group nodes into clusters. The enhanced Sparrow Search Algorithm improves task placement, incorporating chaotic mapping and Cauchy mutation for fair load distribution. Simulations demonstrate that the proposed method reduced latency by 9%, processing time by 14%, workload imbalance by 15%, and energy consumption by 19%, while also increasing resource availability and service throughput.

The Cat Swarm Optimization (CSO) was adapted by Yousif et al. (2023) to enhance the job scheduling for IoT cloud computing. The mechanism depicted in Fig. 15 minimises job execution time by evaluating job-resource populations through fitness values. The approach was tested in CloudSim, showing significant improvements in execution time compared to the firefly algorithm and glowworm swarm optimization. Specifically, the CSO achieved an average execution time of 131, outperforming the firefly algorithm (237) and glowworm swarm optimization (220).

Abdullah et al. (2023) proposed an Adapting Fault-Tolerant Model (AFTM) based on PSO, Apache Spark, and ACO to address job scheduling failures in cloud environments. The model optimizes resource allocation through job selection and fault diagnosis, balancing CPU execution time and memory allocation. Empirical performance evaluations demonstrate that AFTM outperforms PSO and traditional GA regarding memory usage and CPU execution time, offering a more efficient solution for fault-tolerant job scheduling. Another

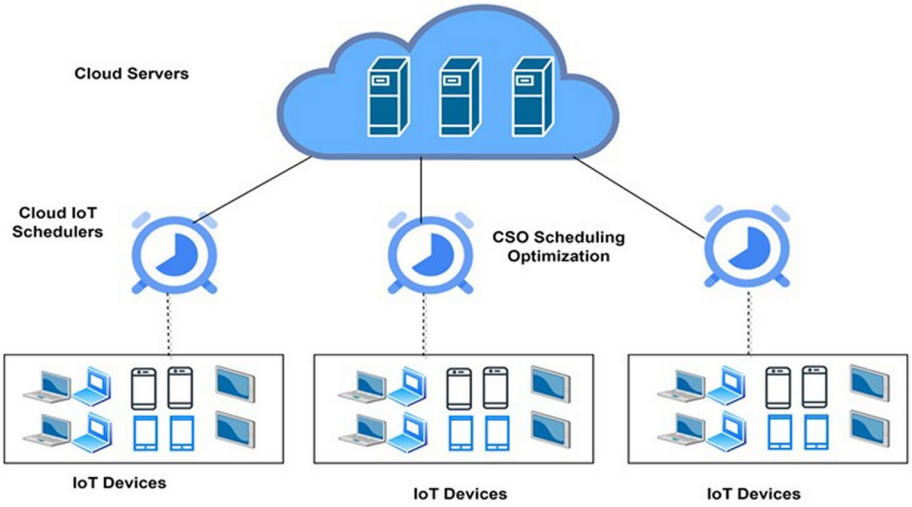

Fig. 15 Architecture of the proposed approach by Yousif et al. (2023)

enhanced optimization method, called a hybrid Whale and BAT Optimization algorithm (WBAT), by Hariharan and Paul Raj (2020). This method aims to minimize makespan and maximize job quality by iterating through job assignments and evaluating fitness functions. The WBAT approach outperforms existing scheduling methods, producing optimized job schedules with reduced makespan and improved job quality based on experimental results. Paulraj et al. (2023) proposed an Efficient Hybrid Job Scheduling Optimization (EHJSO) method combining Cuckoo Search Optimization and Grey Wolf Optimization (GWO). The EHJSO method optimises resource allocation and job scheduling in cloud environments by minimizing makespan, reducing computation time, and improving system performance. Experimental comparisons show that EHJSO outperforms existing methods in terms of fitness, success rate, and overall system efficiency. Also, Singh (2018) proposed a hybrid scheduling technique combining Variable Neighborhood Search (VNS), GA, and PSO to overcome limitations in traditional PSO-based scheduling in cloud computing. The hybrid approach (HGVP) enhances energy efficiency by improving the selection of initial particles for PSO, leading to better exploration and resource utilization. Simulations demonstrate that HGVP achieves superior results in energy consumption compared to existing scheduling techniques.

Sravanthi and Moparthi (2024) proposed a job scheduling technique using the Dual Interactive Wasserstein Generative Adversarial Network optimized with the Arithmetic Optimization Algorithm (AOA). The proposed method forecasts tasks in dynamic cloud environments while AOA optimizes the network's weight parameters. Tested with the Alibaba dataset, the approach reduces power consumption at cloud data centres and achieves lower prediction errors than current methods, improving task scheduling accuracy.

Zain and Yousif (2020) adapted a Chemical Reaction Optimization (CRO) algorithm for job scheduling in cloud computing. Simulation results conducted in CloudSim show that CRO significantly reduces execution time compared to other algorithms, such as glowworm swarm optimization, cat swarm optimization, and first-come-first-served scheduling, making it a more efficient choice for cloud job scheduling. Sutar et al. (2024) presented

a dual-objective job scheduling approach that minimizes energy usage and cost in cloud computing environments. The approach optimizes VM allocation using benchmark datasets and compares performance against methods such as Whale Optimization, ABC, and PSO. Simulation results show that the dual-objective method achieves better energy efficiency and cost reduction, outperforming existing scheduling techniques in cloud environments.

4.2.3 Other optimization techniques

Along with the previous optimization methods used to address the job scheduling problem in cloud computing, the literature has adapted and utilized many other methods belonging to other taxonomies.

Abdullah and Othman (2014) investigated task scheduling in cloud computing, which is an NP-hard optimization problem. The authors proposed a novel Simulated Annealing (SA) algorithm as an alternative to the commonly used GA for job scheduling. While GA has been effective in producing good results, its primary drawback is the time-consuming nature of the algorithm. The SA-based approach, in contrast, offers comparative results in a much shorter execution time, making it a promising solution for cloud task scheduling. Attiya et al. (2020a) proposed a modified Harris Hawks Optimization (HHO) algorithm enhanced with SA for job scheduling in cloud environments. The introduced HHOSA method utilizes SA as a local search to improve both convergence rate and solution quality. The performance of HHOSA is evaluated using the CloudSim toolkit on standard and synthetic workloads and is compared with other state-of-the-art job scheduling algorithms. Results show that HHOSA reduces makespan significantly compared to the standard HHO and different algorithms, making it suitable for large-scale scheduling problems due to its faster convergence in larger search spaces.

Hassan and Yousif (2020) introduced a new job scheduling mechanism based on the Ions Motion Optimization (IMO) algorithm for cloud computing environments. The proposed mechanism was tested in different scenarios using CloudSim, and its performance was compared with CSO, Glowworm Swarm Optimization (GSO), First Come, First Served, and random scheduling. The experimental results show that the IMO algorithm outperforms CSO and GSO in reducing execution time across all tested scenarios.

Suliman et al. (2019) proposed a job scheduling mechanism based on the Shark Smell Optimization (SSO) algorithm. The mechanism minimises job execution time by iterating through resources and blood points. The SSO-based method was evaluated through Java implementation and CloudSim simulation. Several experiments demonstrated that SSO outperformed the Firefly Algorithm regarding job execution time, proving to be a more efficient approach for job scheduling in cloud computing. The Rock Hyrax optimization was adapted by Singhal and Sharma (2021) for job scheduling in a dynamic and heterogeneous cloud environment. The method optimizes job scheduling by minimizing makespan and reducing energy consumption in data centres. Implemented in CloudSim, the algorithm's performance was evaluated qualitatively and quantitatively, showing a 5–15% reduction in makespan and a 4–12% decrease in energy consumption. Compared to other methods, the Rock Hyrax-based algorithm is an effective solution for dynamic job scheduling in cloud environments (Fig. 16).

Gasior and Seredynski (2015) proposed a decentralized multi-agent system for job scheduling in cloud computing based on the Sandpile cellular automaton. This system operates

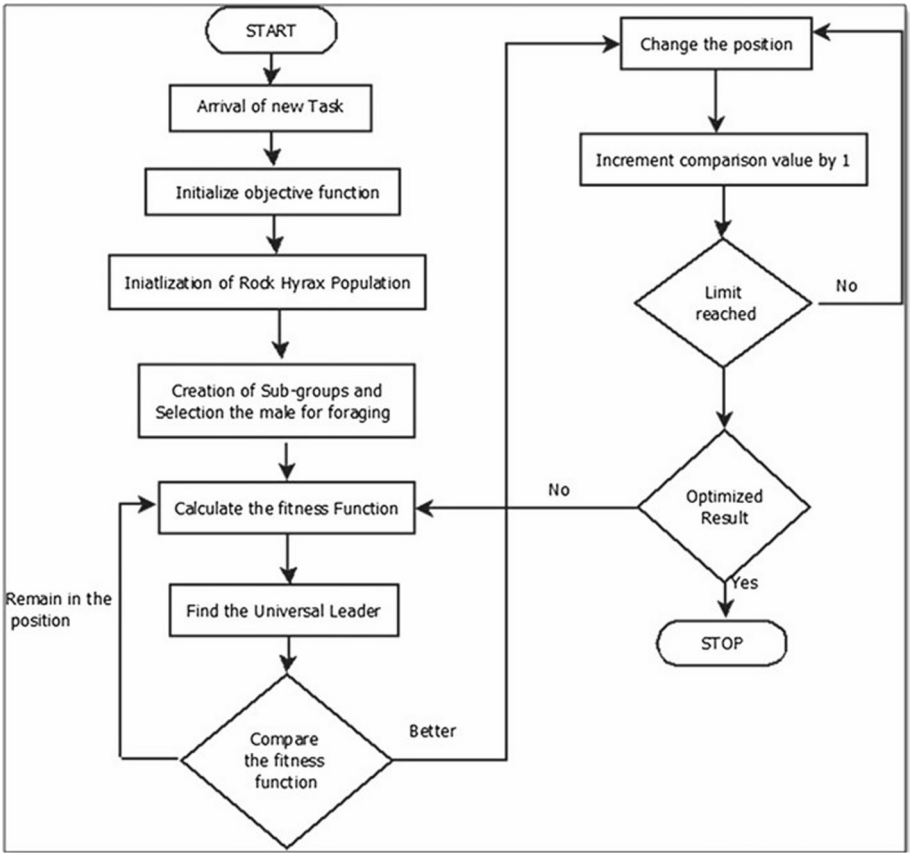

Fig. 16 Flowchart of the proposed approach (Singhal and Sharma 2021)

in a critical state at the edge of chaos, aiming to minimize job slowdown and dynamically reschedule tasks to the best resources. Experimental results demonstrate the scalability and effectiveness of the approach in managing large numbers of jobs and resources, as well as its ability to adapt to real-time changes in dynamic cloud environments.

Kim et al. (2014) addressed the job scheduling problem in cloud computing using Biogeography-Based Optimization (BBO). BBO migration is employed to adapt solutions, offering an adaptive process for binary integer job scheduling. Experimental results show that the proposed BBO-based method outperforms other considered methods, demonstrating better performance in optimizing job scheduling. Garg and Dhiman (2020) adapted the Seagull Optimization Algorithm (SOA) for job scheduling in cloud computing environments. SOA is implemented and evaluated using the CloudSim toolkit, where its performance is compared to state-of-the-art job scheduling algorithms. The results highlight the effectiveness of SOA in reducing the makespan of jobs, particularly in dynamic cloud environments with on-demand customer application requirements.

Khezri et al. (2024) focused on optimizing job scheduling in fog-cloud IoT systems, where latency can be problematic. The authors propose a Data-Locality Aware Job Scheduling in Fog-Cloud (DLJSF) algorithm based on an integer linear programming (ILP) model.

The DLJSF algorithm prioritizes local data processing to reduce data transfer delays. Experimental results show that the DLJSF algorithm performs close to optimal solutions, achieving 99.16% efficiency in processing data locally, making it a promising solution for latency-sensitive IoT applications. Kaur et al. (2021) presented an energy-aware and SLA-driven job scheduling framework for cloud data centres based on MapReduce. The framework explores task-to-slot/container mapping as a multi-objective problem formulated using Integer Linear Programming. The proposed scheduling strategy uses a Greedy heuristic and Hungarian algorithm to minimize energy consumption while adhering to SLA constraints. Validated on real-time data from OpenCloud, the results show that the framework significantly reduces energy consumption and makespan while minimizing SLA violations compared to existing schemes.

Table 7 provides a comparative analysis of various optimization-based approaches for solving specific problems. It organizes the information into four key dimensions: objectives, methods, findings, and weaknesses/shortcomings.

4.3 AI-based heuristic and meta-heuristic methods

Heuristic and meta-heuristic methods provide powerful techniques for solving complex optimization problems, mainly when the solution space is too ample for exhaustive search methods. These methods offer approximate solutions by following strategic rules or processes, making them highly suitable for job scheduling challenges that require quick and near-optimal solutions. This section explores the different categories within heuristic and meta-heuristic methods, such as constructive heuristics, local search heuristics, population-based meta-heuristics, trajectory-based meta-heuristics, and hybrid heuristics. These methods allow schedulers to explore solution spaces efficiently while balancing performance with computational costs, providing flexible, adaptive solutions in dynamic scheduling environments.

4.3.1 Constructive heuristics

Constructive heuristics build solutions incrementally, adding elements one at a time based on predefined rules. In job scheduling, these heuristics quickly generate an initial solution by prioritizing jobs according to processing time, deadlines, or resource availability. Although the solutions may not be optimal, constructive heuristics offer a good starting point for further refinement by other optimization techniques. Recent research on CPN-Local Execution Scheduling (CLES) introduced a dynamic programming method combined with an efficient heuristic algorithm to address task offloading and local execution in computing power networks. The algorithm considers strict energy limitations and task dependencies, offering near-optimal performance for small-scale problems. While the heuristic approach improves the initial solution iteratively, further exploration is needed to assess its performance in larger-scale or highly dynamic environments with more complex dependencies and energy constraints (Zhang and Cui 2023). Another study (Singh and Gupta 2017) proposed Efficient Multi-Queue Scheduling (MQS) as a heuristic approach to improve job scheduling in cloud computing. MQS organizes jobs into multiple queues to streamline resource utilization and reduce energy consumption. While the method improves energy efficiency and scheduling

time, it faces potential scalability and load-balancing challenges, particularly when handling highly diverse workloads or operating in dynamic cloud environments.

Abdulredha et al. reviewed heuristic and meta-heuristic techniques applied to task scheduling in cloud-fog environments. Among the constructive heuristics, the Min–Min and Max-Min algorithms were prominently reviewed. These algorithms are well-regarded for their efficiency in job scheduling tasks. Min–Min prioritizes tasks with the shortest execution time, thus reducing makespan, while Max-Min focuses on tasks with the longest execution time to balance load distribution across resources. The study highlights the adaptability and effectiveness of these methods in minimizing both makespan and cost, particularly for NP-hard problems in distributed systems (Abdulredha et al. 2020). In addition, the study in Hussain et al. (2019) evaluated ten constructive heuristics to minimise makespan and improve resource utilization. These heuristics, including Min–Min, Max-Min, Sufferage, and Task-Aware Scheduling Algorithm, were tested using the CloudSim simulator on synthetic and benchmark workloads. The results demonstrated that while Sufferage and TASA achieved the lowest makespan, they failed to exploit cloud virtual machines' computing capabilities fully, highlighting a need for more efficient load-balancing mechanisms to enhance overall resource utilization and throughput.

In the context of satellite scheduling, a recent study introduced an edge computing framework that uses a central node to filter tasks based on scheduling periods and visible time windows, while multiple edge nodes (each corresponding to a satellite) employ a Constructive Heuristic Algorithm based on the Density of Residual Tasks for task prioritization. The HADRT algorithm improved task completion and reward rates compared to other advanced methods while maintaining acceptable computational time, showcasing the effectiveness of constructive heuristics in addressing complex satellite scheduling challenges (He et al. 2019). Furthermore, Ghanbari proposed a multi-criteria priority-aware job scheduling algorithm in cloud computing, which incorporates various factors and local search heuristics to manage the complexity of scheduling in dynamic cloud environments. The method effectively improves task scheduling performance by considering job priorities (Ghanbari 2019).

4.3.2 Local search heuristics

Local search heuristics improve upon a given solution by making small changes to its components, typically focusing on neighbouring solutions. These heuristics explore the solution space by iteratively adjusting schedules and assessing whether the new solution is better. In job scheduling, local search heuristics are effective at fine-tuning schedules to minimize processing time or resource contention, making them ideal for real-time scheduling adjustments.

Ghanbari proposed a multi-criteria priority-aware job scheduling algorithm in cloud computing, which incorporates various factors and local search heuristics to manage the complexity of scheduling in dynamic cloud environments. The method effectively improves task scheduling performance by considering job priorities (Ghanbari 2019). Another study addressed the multi-stage job scheduling problem in hybrid cloud systems focused on minimizing the number of elastic computing instances while meeting job deadlines. The proposed approach combines simple dispatching rules to generate initial job sequences with a fast local search heuristic and a rescheduling process to refine the schedule iteratively. Experimental results demonstrated improved resource utilization and adherence to cloud

Table 7 A comparative analysis of various optimization-based approaches

References	Methods	Findings	Latency	Energy consumption	Cost	Pros	Cons
Lane et al. (2022)	GA and SA	Reduces makespan and improves resource allocation	Low	Moderate	Moderate	Efficient for resource allocation	Sensitive to parameter tuning
Remesh et al. (2023)	Hybrid GA-based approach	Enhanced task scheduling	Moderate	Moderate	High	Combines multiple optimization strategies	High computational cost
Sahraei et al. (2019)	GA	Improved cost, makespan, and resource utilization	Moderate	Moderate	Low	Balances multiple objectives	Limited scalability
Ali and Ali (2023)	Catastrophic GA (CGA)	Reduces task completion time by over 97%	Low	Moderate	High	Drastic reduction in completion time	Requires high energy consumption
Kumar et al. (2024b)	GA	Improved scalability	Moderate	High	Moderate	Scales well for large tasks	High energy demands
Frank Vijay (2019)	GA	Enhanced scheduling efficiency	Moderate	Low	Low	Simple implementation	Inefficient for complex tasks
Sharma and Lakra (2020)	Green IT practices	Reduced energy consumption	Low	Low	Low	Promotes sustainability	Limited to specific environments
Wang et al. (2014c)	Multi-objective GA	Improved energy efficiency by dynamically adjusting data locality and task placement	Moderate	Low	High	Effective for energy-aware applications	High setup costs
Ananth and Chandrasekaran (2016)	NSGA-II	Comprehensive solution for cloud resource management	Moderate	Moderate	High	Versatile for multi-objective problems	High computational demands
Kuppusamy et al. (2022)	Dynamic Opposition Learning-based Social Spider Optimization algorithm	Achieved a 10–15% improvement in CPU utilization and a 5–10% reduction in energy consumption	Low	Low	Moderate	Reduces energy costs	Limited flexibility for diverse workloads
Khaleel (2024)	Regional Awareness Dynamic Scheduling Algorithm	Reduces latency, processing time, workload imbalance, and energy consumption	Low	Low	High	Handles workload imbalance effectively	Expensive to implement
Yousif et al. (2023)	CSO	Improved execution time	Moderate	Moderate	Moderate	Faster execution	Complex implementation
Abdullah et al. (2023)	Adapting Fault-Tolerant Model	Optimized memory usage and CPU execution time	Moderate	Low	Low	Enhances fault tolerance	Limited to fault-prone environments

Table 7 (continued)

References	Methods	Findings	Latency	Energy consumption	Cost	Pros	Cons
Hariharan and Paul Raj (2020)	Hybrid Whale and BAT optimization algorithm	Reduced makespan and improved job quality	Moderate	Moderate	High	Combines strengths of two algorithms	Computationally intensive
Paulraj et al. (2023)	Efficient Hybrid Job Scheduling Optimization	Optimized fitness, success rate, and system efficiency	Moderate	Moderate	Moderate	High success rate	Time-consuming for larger datasets
Singh (2018)	HGVP	Reduced energy consumption	Moderate	Low	Low	Energy-efficient	Less adaptable to dynamic workloads
Sravanthi and Moparthi (2024)	Dual Interactive Wasserstein GAN and AOA	Reduced power consumption and prediction errors in cloud data centres	Low	Low	High	High accuracy in predictions	Resource-intensive
Zain and Yousif (2020)	Chemical Reaction Optimization	Reduced execution time	Moderate	Moderate	Moderate	Effective for execution speed	Limited scalability
Sutar et al. (2024)	Dual-objective job scheduling approach	Improved energy efficiency and cost reduction	Moderate	Low	High	Reduces overall costs	Complex implementation
Abdullah and Othman (2014)	Simulated Annealing	Short execution times	Low	Low	Low	Simple and effective	Not suitable for large datasets
Attiya et al. (2020a)	Modified HHO algorithm integrated with SA	Reduced makespan and improved convergence	Moderate	Moderate	High	Effective for convergence issues	Requires parameter fine-tuning
Hassan and Yousif (2020)	Ions Motion Optimization algorithm	Reduced execution time	Moderate	Low	Low	Low cost	Inefficient for complex scenarios
Suliman et al. (2019)	Shark Smell Optimization algorithm	Enhanced job execution time	Low	Low	Moderate	Suitable for small-scale tasks	Limited scalability
Singhal and Sharma (2021)	Rock Hyrax optimization	Reduced makespan by 5–15% and energy consumption by 4–12%	Moderate	Low	Moderate	Optimizes energy and time	Sensitive to initialization parameters
Gasior and Seredynski (2015)	Sandpile cellular automaton	Enhanced scalability in dynamic cloud environments	Moderate	Moderate	High	Excellent scalability	High computational cost
Kim et al. (2014)	Biogeography-based optimization	Optimized job scheduling	Moderate	Moderate	Moderate	Adapts to various scenarios	Limited to static environments
Garg and Dhiman (2020)	SOA	Reduced makespan	Low	Low	Moderate	Simple implementation	Limited flexibility

Table 7 (continued)

References	Methods	Findings	Latency	Energy consumption	Cost	Pros	Cons
Khezri et al. (2024)	DLJSF algorithm	Achieved 99.16% efficiency in reducing data transfer delays	Low	Low	High	Highly efficient	Expensive infrastructure
Kaur et al. (2021)	Greedy and Hungarian algorithms	Reduced energy consumption and makespan	Moderate	Moderate	High	Balances energy and time	High overhead in large-scale systems

service quality requirements. However, the study did not explore the algorithm's scalability in highly dynamic environments, leaving room for further research on its adaptability to real-time cloud systems (Zhu et al. 2016).

4.3.3 Population-based meta-heuristics

Population-based meta-heuristics, such as GA and PSO, operate on a population of potential solutions and evolve them over time. These techniques simultaneously explore diverse solutions by simulating natural processes such as selection, crossover, and mutation. In job scheduling, population-based meta-heuristics provide a robust approach to handling large-scale and complex scheduling problems, offering the potential to escape local optima and find globally optimal solutions. A comprehensive survey of cloud resource scheduling techniques was conducted in Aron and Abraham (2022), focusing on the role of meta-heuristic algorithms and Artificial Intelligence (AI) in optimizing resource management. The study examines several meta-heuristic approaches, such as GA, PSO, and ACO, which are crucial in addressing the NP-hard nature of cloud scheduling due to the heterogeneity of resources and unpredictability of workloads. Additionally, it explores heuristic methods for VM placement and Quality-of-Service (QoS) optimization. The survey emphasizes the growing importance of hybrid techniques that combine meta-heuristics with AI for enhanced performance. However, it notes the need for further real-world evaluations to assess scalability and efficiency in dynamic cloud environments. Recent work has implemented and compared ACO and PSO for dynamic virtual machine allocation to improve task scheduling and load balancing in cloud environments. These meta-heuristic algorithms outperform traditional methods, demonstrating better load balancing and scheduling efficiency. However, the study does not fully address how these algorithms perform under real-time dynamic conditions, where task demand and resource availability change unpredictably. Further research could explore the adaptability of these algorithms in such environments (Vidhya and Devi 2023).

To address the NP-complete task scheduling problem in cloud computing, researchers proposed a hybrid algorithm combining GWO and GA. By utilizing GA's crossover and mutation operators, the hybrid approach improves GWO's convergence speed and performance in large-scale scheduling problems, focusing on minimizing makespan, energy consumption, and cost. Evaluated using the CloudSim toolkit, the hybrid GWO-GA algorithm showed reductions of up to 23% in energy consumption, 22% in total cost, and 19% in makespan compared to traditional methods (Behera and Sobhanayak 2024). In addition, a study in Kumar et al. (2024a) explores the use of the GA on the Google Cloud Platform (GCP). GA is a population-based meta-heuristic that iteratively refines task-to-resource allocations based on principles of natural selection. The study compares GA's performance with other methods like DBSCAN, Google Cloud Dataproc, and Google Kubernetes Engine. It demonstrates GA's capability to optimize several parameters, including cost, execution time, and resource consumption. GA enhances system efficiency and scalability on GCP by efficiently exploring a broad solution space and adapting to complex optimization challenges, offering a viable solution for dynamic cloud environments.

Moreover, the authors proposed the Grouped Whale Optimization Algorithm (GWOA) for improving task scheduling in cloud computing environments. GWOA enhances WOA by dividing the whale population into groups, which helps balance global exploration and local exploitation, thereby improving the search process and preventing premature conver-

gence. The algorithm is applied to cloud computing schedulers to optimize critical metrics such as execution time, response time, and throughput under high workloads. Experimental results demonstrate that GWOA outperforms standard WOA, PSO, and Bat Algorithm regarding scheduling efficiency (Sangaiah et al. 2023). Another study has introduced two hybrid meta-heuristic algorithms based on Dynamic Dispatch Queues (TSDQ). The first algorithm, TSDQ-FLPSO, integrates Fuzzy Logic with PSO to enhance decision-making and optimization. In contrast, the second, TSDQ-SAPSO, combines SA with PSO to improve global search capability. Through extensive experimentation using the CloudSim simulator on both synthetic and natural datasets, the study demonstrates that TSDQ-FLPSO significantly outperforms TSDQ-SAPSO and other existing scheduling methods, particularly in high-dimensional tasks. The algorithm achieves better results regarding waiting time, queue length, makespan, cost, resource utilization, and load balancing, making it a promising approach for cloud resource optimization (Alla et al. 2018). In addition, the Inverted Ant Colony Optimization (IACO) algorithm has been proposed for workflow scheduling in cloud environments, aiming to improve resource utilization and minimize execution time and costs. By inverting the pheromone influence in the traditional ACO, IACO enhances exploration and prevents premature convergence. The algorithm outperformed existing simulation methods, providing better scheduling results regarding execution time and cost. However, further research is needed to assess its adaptability in real-time dynamic cloud environments with fluctuating VM availability (Hongwei and Zhang 2023).

In recent work on task scheduling in cloud computing, the Autodidactic Interactive School Optimization Algorithm (IASOA) was introduced to optimize load balancing and task allocation. By leveraging swarm intelligence-based meta-heuristics, the algorithm effectively balances local and global search strategies to reduce makespan and improve throughput. Comparative analysis with existing algorithms demonstrated improvements of up to 10% in makespan and 60% in throughput. However, the study did not explore how the algorithm performs under varying job complexities or resource constraints, leaving room for further investigation (Senthilkumar et al. 2024). Moreover, a multi-variant PSO algorithm has been proposed to enhance task scheduling in cloud computing environments. The algorithm leverages PSO's population-based meta-heuristic capabilities to improve load balancing, reduce execution time, and optimize resource allocation. The study demonstrated that multi-variant PSO can enhance cloud performance and cost efficiency by optimizing resource usage. However, further investigation is needed to understand its effectiveness in more complex and dynamic cloud environments (Krishna et al. 2024). Another study introduced an enhanced WOA for task scheduling in cloud computing, addressing common optimization challenges such as slow convergence and susceptibility to local optima. The improved WOA incorporates an adaptive adjustment method, differential variation, and a restart approach to balance exploration and exploitation dynamically during task scheduling. The evaluation demonstrated significant improvements in makespan, resource cost, and load balancing compared to existing algorithms. However, further research is needed to assess its performance in dynamic cloud environments with real-time scheduling demands (Pareek et al. 2024)

4.3.4 Trajectory-based meta-heuristics

Trajectory-based meta-heuristics focus on exploring the solution space by following a specific path or trajectory and adjusting parameters iteratively to reach better solutions. Examples include SA and Tabu Search (TS), which guide the search process by cooling or restricting certain moves. In job scheduling, trajectory-based meta-heuristics effectively balance exploration and exploitation, allowing schedulers to find high-quality solutions while avoiding premature convergence.

A recent study proposed a novel DAG scheduling model to optimise QoS parameters in cloud computing environments (CCE). The model efficiently schedules tasks with dependencies represented as Directed Acyclic Graphs (DAGs) to improve resource allocation and execution times. Extensive simulation results demonstrated that the proposed algorithm outperforms existing state-of-the-art scheduling techniques, particularly in optimizing QoS. However, potential scalability issues in larger cloud environments or under varying QoS demands were not extensively addressed (Rajak et al. 2023).

4.3.5 Hybrid heuristics

Hybrid heuristics combine multiple heuristic or meta-heuristic approaches to leverage the strengths of different methods. By integrating techniques like local search with population-based meta-heuristics or combining machine learning models with heuristic methods, hybrid heuristics can provide more flexible and efficient solutions to job scheduling problems. These hybrid approaches allow for adaptive scheduling strategies that respond dynamically to changes in workload, resource availability, and system constraints, resulting in more robust and scalable scheduling solutions.

In cloud computing, effectively scheduling job requests under uncertain capacity constraints are critical. A recent study proposed the Controlling under Uncertain Constraints (CUC) algorithm, which combines prediction and optimization to address this issue. By utilizing Bayesian optimization to predict available capacity and incorporating these predictions into the job scheduling process, CUC creates a feedback loop that improves future predictions based on scheduling outcomes. Experimental results on public datasets show that CUC significantly enhances scheduling performance and reliability under uncertain conditions (Dong et al. 2021). Furthermore, the hybrid approach has been proposed for task scheduling in cloud computing by combining PSO with heuristic-based initialization techniques, specifically Longest Job to Fastest Processor (LJFP) and Minimum Completion Time (MCT). The goal is to enhance the performance of PSO by providing more effective starting points for optimization, aiming to minimize key metrics such as makespan, total execution time, degree of imbalance, and total energy consumption. Simulation results show that the hybrid LJFP-PSO and MCT-PSO algorithms outperform conventional PSO and other recent task-scheduling methods. This demonstrates improved resource utilization and cost-effective task execution in cloud environments (Alsaidy et al. 2022).

Moreover, a hybrid algorithm combining GA and ACO was proposed to optimize task scheduling for IoT applications in a heterogeneous multiprocessor cloud environment. GAACO leverages GA's natural selection process alongside ACO's swarm intelligence to find the most efficient task schedules, minimizing makespan and adhering to task dependencies. Experimental results demonstrated that GAACO outperforms traditional GA and

ACO methods, especially when tested with varying task sizes and processor counts (Basu et al. 2018). Besides that, the study in Attiya et al. (2020a) introduced a hybrid algorithm combining HHO with SA, referred to as HHOSA. Integrating SA as a local search heuristic enhances the convergence rate and solution quality of the standard HHO algorithm, addressing the NP-complete nature of cloud job scheduling problems. Experimental results using the CloudSim toolkit demonstrated that HHOSA outperforms traditional algorithms by significantly reducing makespan and improving scalability in large-scale scheduling problems. This advancement is aligned with other studies in the field, emphasizing the importance of hybrid heuristic approaches for improving efficiency in complex cloud environments.

Furthermore, a novel cost optimization model has been proposed for Scientific Workflow Scheduling (SWFS) in cloud computing environments, focusing on minimizing makespan and computational costs. The study evaluated single-based heuristics, including GA, PSO, and Invasive Weed Optimization (IWO), as well as hybrid-based heuristics like Hybrid Invasive Weed Optimization (HIWO) and the Dynamic Hyper-Heuristic Algorithm (DHHA). The results demonstrated that the model efficiently optimized small and large dataset workflows. In particular, hybrid and hyper-based approaches outperformed single-based methods for medium-sized datasets, achieving better cost optimization and makespan results (Al-Khanak et al. 2021). A three-variant algorithm has been introduced to optimize workflow scheduling in cloud computing, focusing on minimizing both cost and makespan. The algorithm employs a combination of approaches, including local search heuristics, population-based meta-heuristics, and rule-based heuristics. Each variant addresses different aspects of the scheduling problem, allowing for a flexible and efficient solution across various cloud environments. The study demonstrated that the algorithm effectively reduces both makespan and cost while adapting to the complexity of cloud resource configurations (Kamanga et al. 2023). Also, a modified Symbiotic Organisms Search (SOS) algorithm, called G_SOS, was proposed to improve the efficiency of mapping heterogeneous tasks to cloud resources. By replacing the mutualism process's arithmetic mean with a geometric mean, G_SOS enhances species interactions, resulting in better task execution times, cost efficiency, and system balance. The algorithm demonstrated superior performance over classical SOS and PSO-SA, achieving up to 25.68% improvement in makespan for large-scale tasks. While promising, further studies are needed to assess its adaptability in real-time, dynamic cloud environments (Zubair et al. 2022).

Table 8 compares various heuristic and metaheuristic-based approaches for solving specific problems. It organizes the information into four key dimensions: objectives, methods, findings, and weaknesses/shortcomings.

4.4 Hybrid AI models

Hybrid AI models have emerged as powerful tools in job scheduling, combining the strengths of multiple AI approaches to achieve superior performance. These models leverage the adaptability and learning capabilities of machine learning, the search efficiency of meta-heuristic algorithms, and the reliability of traditional scheduling methods. The hybrid nature of these techniques allows for more flexible, scalable, and accurate job scheduling solutions, capable of handling both simple and complex tasks in dynamic cloud environments. This section explores the critical hybrid AI models, including combining machine learning with meta-heuristics and integrating AI with traditional scheduling algorithms.

Table 8 A comparative analysis of various heuristic and metaheuristic-based approaches

References	Methods	Findings	Latency	Energy consumption	Cost	Pros	Cons
Alsaidy et al. (2022)	LJFP-PSO, MCT-PSO	Superior performance compared to conventional PSO for task scheduling	Moderate	Low	Moderate	Improved task scheduling	Sensitive to parameter tuning
Behera and Sobhanayak (2024)	Hybrid GWO-GA	Outperforms other algorithms in makespan reduction and energy optimization	Low	Moderate	High	Efficient makespan reduction	High computational cost
Dong et al. (2021)	CUC	Efficiently manages capacity and schedules workloads for reliable cloud platforms	Moderate	Low	Moderate	Handles diverse workloads	Limited scalability
He et al. (2019)	HADRT	Higher task completion rate than other advanced algorithms	Low	Moderate	Low	High task completion rate	Limited to specific scenarios
Kumar et al. (2024a)	GA	Improves system efficiency and scalability on GCP	Low	Moderate	Moderate	Scales effectively	Moderate cost constraints
Alla et al. (2018)	TSDQ-FLPSO, TSDQ-SAPSO	Outperforms other scheduling algorithms in high-dimensional problems	Moderate	Moderate	Moderate	Effective in high dimensions	Computationally expensive
Sangaiah et al. (2023)	GWOA	Reduces execution and response times while increasing throughput	Low	Low	Moderate	High throughput	Implementation complexity
Aron and Abraham (2022)	GA, PSO, ACO	Handles NP-hard cloud scheduling problems efficiently	Moderate	Moderate	Moderate	Versatile for multiple tasks	Performance varies across algorithms
Singh and Gupta (2017)	MQS	Reduces energy consumption while maintaining system performance	Moderate	Low	Low	Energy-efficient	Limited dynamic adaptability
Basu et al. (2018)	GAACO	Improves makespan optimization and convergence efficiency	Moderate	Moderate	Moderate	High convergence efficiency	Limited scalability
Rajak et al. (2023)	DAG	Outperforms state-of-the-art algorithms in QoS and efficiency	Low	Moderate	High	Enhanced QoS	High computational cost
Al-Khanak et al. (2021)	GA, PSO, IWO	Outperforms single-based methods in cost and makespan optimization	Moderate	Moderate	Moderate	Balances cost and makespan	High complexity
Attiya et al. (2020a)	HHOSA	Reduces makespan significantly compared to other methods	Low	Moderate	Moderate	Drastic reduction in makespan	Parameter tuning required
Kamanga et al. (2023)	Three-variant algorithm	Effectively reduces makespan and monetary costs	Moderate	Moderate	High	Cost-efficient	Limited versatility

Table 8 (continued)

References	Methods	Findings	Latency	Energy consumption	Cost	Pros	Cons
Abdulredha et al. (2020)	Min–Min, Max-Min	Meta-heuristics provide near-optimal solutions for NP-hard problems	Moderate	Moderate	Moderate	Near-optimal solutions	Less adaptive to large changes
Hussain et al. (2019)	Min–Min, Max-Min, Sufferage	TASA and Sufferage achieve minimum makespan	Moderate	Low	Low	Minimum makespan	Limited scalability
Ghanbari (2019)	Multi-criteria	Handles multiple criteria for better resource allocation	Moderate	Moderate	Moderate	Better resource allocation	High overhead
Zhu et al. (2016)	Dispatching rules, local search heuristic	Improves resource utilization while meeting service quality	Moderate	Moderate	Moderate	High resource utilization	Limited flexibility
Senthilkumar et al. (2024)	IASOA	Improves makespan by 10% and throughput by 60%	Moderate	Moderate	High	High throughput improvement	High cost
Kumar et al. (2024a)	Multi-variant PSO	Improves execution time and resource allocation	Low	Low	Moderate	Efficient execution	Complex to configure
Pareek et al. (2024)	WOA	Improves makespan, resource cost, and load balancing efficiency	Low	Moderate	Moderate	Effective load balancing	Limited robustness
Vidhya and Devi (2023)	ACO, PSO	Outperforms traditional methods in load balancing and efficiency	Moderate	Low	Moderate	High efficiency	Limited scalability
Zhang and Cui (2023)	Dynamic programming, heuristic	Near-optimal performance for small-scale problems	Low	Low	High	Near-optimal for small scales	Poor scalability
Hongwei and Zhang (2023)	IACO	Demonstrates better execution time and cost results	Moderate	Moderate	High	Enhanced execution	High energy cost
Zubair et al. (2022)	G_SOS	Outperforms classical SOS and PSO-SA algorithms	Moderate	Moderate	Moderate	Versatile performance	Moderate complexity

These methods offer a comprehensive framework to optimize scheduling by blending different strategies and taking advantage of their complementary strengths.

4.4.1 Combining ML with meta-heuristics

Combining machine learning with meta-heuristics allows for a hybrid approach where machine learning models can guide the search process, providing valuable insights and predictions based on historical data. In job scheduling, this synergy enhances the ability of meta-heuristics, such as genetic algorithms or particle swarm optimization, to explore the solution space more effectively by narrowing the search to promising regions. Machine learning can predict job durations, resource requirements, or potential bottlenecks, while meta-heuristics refine the schedule to achieve optimal performance. This combination results in more efficient and intelligent scheduling, adapting to changing workloads and improving overall resource utilization.

A recent study introduced Energy-Efficient Task Scheduling (EETS), an enhanced task scheduling algorithm built on the DQN framework. By incorporating a Dueling-network architecture, Double DQN (DDQN), and Prioritized Experience Replay, EETS addresses limitations such as value overestimation and low sample utilization in traditional DQN-based scheduling. The algorithm demonstrated superior energy consumption, task response time, and machine working time compared to baseline methods, particularly when handling large task batches. However, further research is required to evaluate its adaptability in real-time dynamic environments (Hou and Ismail 2024). Moreover, the authors in Huang et al. (2024) introduced the MAGAIL-MCT algorithm, which applies Multi-Agent Generative Adversarial Imitation Learning to the task scheduling problem in an Ad Hoc cloud formed by battlefield equipment. The algorithm minimises system overhead and optimises task migration by utilizing idle computing resources from tanks, drones, and vehicles. The method incorporates behavioural cloning to mimic expert trajectories, simplifying the reward function for neural network training. While the results demonstrate reduced migration delay and energy consumption, further work is needed to assess the scheme's scalability and performance under dynamic network conditions.

A recent study introduced SafeTail to address the challenges of optimizing both median and tail latency in edge computing environments. This framework uses a reward-based deep learning model for optimal service placement. SafeTail selectively replicates services across edge servers to meet tail latency targets (beyond the 90th percentile) while minimizing additional resource usage. The framework demonstrated superior performance in trace-driven simulations, surpassing baseline strategies across diverse services (Shokhanda et al. 2024). Moreover, HPFL-CN has been introduced, a Hierarchical Personalized Federated Learning framework that enhances communication efficiency and model personalization in mobile edge computing for urban environmental prediction. The framework utilizes Privacy-preserving Feature Clustering (PFC) to cluster edge servers accurately and implements an Effective Hierarchical Scheduling (EHS) system to optimize federated learning at the cluster level. An adaptive extension, Ada-HPFL-CN, enables dynamic grouping and flexible model aggregation in real-time scenarios. The results showed significant improvements in communication efficiency and personalization compared to traditional FL methods, though further exploration is needed to evaluate its generalizability to other applications (Li et al. 2024).

Another study proposed TOMAC-PPO, a task-oriented multi-agent actor critique algorithm with a proximal policy optimization algorithm for task offloading and resource allocation. This decentralized scheme models the task offloading problem as a Markov decision process and uses a multi-agent reinforcement learning framework with edge servers as agents. The addition of the Transformer Neural Network enables adequate memory and prediction of network state information. Compared to baseline methods, the TOMAC-PPO algorithm demonstrated superior performance, reducing service costs, energy consumption, and task drop rates. However, its adaptability to dynamic real-time environments remains unexplored (Jiang et al. 2024). Besides that, the authors in Cao et al. (2024) have introduced a service migration strategy for smart rail systems that leverages intelligent agent group cooperative prediction and edge computing. The system dynamically adjusts task migration using deep reinforcement learning by implementing a cloud-edge-end collaborative scheduling network and an intelligent grouping collaborative migration strategy. This approach significantly reduces task delay and system overhead while ensuring service continuity and enhancing the reliability of smart rail services. However, further investigation is needed to assess the practical scalability of this approach in real-world applications.

4.4.2 Integration of AI with traditional scheduling algorithms

Integrating AI techniques with optimization algorithms bridges the gap between classical and modern AI-driven approaches. Traditional optimization algorithms, such as GA or PSO, are well-established and efficient in many scenarios but may lack the adaptability needed for complex, dynamic environments. The optimisation process can be significantly enhanced by incorporating AI techniques, such as reinforcement learning, deep learning, or novel hybrid models. These AI-driven models allow for better exploration of solution spaces, more intelligent decision-making, and real-time adjustments. In task scheduling, this integration enables handling unpredictable workloads, improved resource utilization, and faster convergence, resulting in more adaptive and efficient optimization solutions.

In recent research on cloud resource allocation for SaaS applications, the Deep-Hill algorithm was introduced to improve instance configuration prediction and resource optimization. The algorithm combines a 5-layer Deep Neural Network (DNN) with a Hill-Climbing optimization technique, achieving an impressive accuracy of 96.33% and reducing power consumption by 13.33%. This innovative approach demonstrates significant potential in optimizing cloud resources for SaaS applications. However, further investigation is required to assess its scalability and effectiveness in large-scale, real-world SaaS environments (Abouelyazid 2024). Furthermore, Geyser-inspired Jaya Algorithm (GIJA) is a novel optimization approach introduced for task scheduling in cloud computing environments. GIJA combines the eruptive dynamics of geysers with the Jaya algorithm, augmented by the Levy Flight mechanism, to enhance solution exploration and convergence. Comparative analysis against other task scheduling algorithms, such as AOA, RSA, and DMOA, demonstrated that GIJA significantly improves solution quality, convergence rate, and robustness, achieving a solution quality of 95% (Abualigah et al. 2024). In addition, Chi-squared Particle Swarm Optimization (CHPSO) has been introduced to address the challenges of workload balancing and resource utilization. This novel algorithm improves the distribution of computing workloads across available resources, resulting in enhanced system performance and cost-effectiveness. Comparative analysis showed that CHPSO outperforms traditional

algorithms like PSO and Cuckoo Search in optimizing resource usage. However, further research is needed to evaluate its scalability and adaptability in dynamic cloud environments with fluctuating workloads (Mikram et al. 2023).

Another research proposed the Green Computing Resource Scheduling (GCRS) model to tackle the challenge of high energy consumption in cloud data centres. By classifying servers into read and write servers based on their capacities, the model assigns tasks accordingly, ensuring full utilization of resources and reducing idle time. This approach leads to lower energy consumption and decreased carbon emissions while maintaining SLA compliance. Although the GCRS model demonstrates promise in optimizing energy use, further investigation is needed to assess its performance in dynamic cloud environments with varying workloads (Chitra and Getzi 2023). Furthermore, the study in Tuli et al. (2021) focused on addressing the challenge of workflow scheduling in edge-cloud computing environments, where interdependent tasks must be efficiently mapped across cloud and edge resources. These workflows, often modelled as Directed Acyclic Graphs (DAGs), are becoming increasingly resource-intensive with the rise of AI/ML/DL workloads. By leveraging edge devices for low-latency processing and cloud backends for compute-intensive tasks, edge-cloud environments can deliver significant cost-saving benefits. However, further exploration is needed to understand how these systems handle dynamic changes in workloads and real-time execution.

Table 9 compares various hybrid AI-based approaches for solving specific problems. It organizes the information into four key dimensions: objectives, methods, findings, and weaknesses/shortcomings.

Tradeoffs in hardware and software efficiency, compute learning needs, and overall effectiveness

Implementing AI-based approaches in cloud computing requires careful consideration of various tradeoffs between hardware efficiency, software optimization, compute learning needs, and overall system effectiveness. These tradeoffs influence the selection of algorithms, resource allocation strategies, and performance outcomes, particularly in dynamic and resource-constrained environments. This subsection delves into these tradeoffs, providing insights into how surveyed approaches balance competing priorities and achieve their objectives. By referencing the literature reviewed in this paper, we highlight the implications of these tradeoffs in real-world cloud computing scenarios. Table 10 summarizes these tradeoffs.

Implementing AI-based approaches in cloud computing involves significant trade-offs between hardware and software efficiency, compute learning requirements, and the overall effectiveness of these techniques. These trade offs are critical in determining the practicality, scalability, and cost-effectiveness of AI-based methods for resource allocation, scheduling, and optimization in cloud environments.

One of the critical challenges is balancing **hardware and software efficiency**. High-performance AI models, such as DRL and CNNs, often require specialized hardware like GPUs or TPUs to process large volumes of data and achieve high levels of accuracy and adaptability. As highlighted in Zhu et al. (2021), these models are particularly effective in handling complex tasks, such as multi-cloud job scheduling and dynamic resource allocation. However, this reliance on advanced hardware increases operational costs and energy consumption, which can be a significant limitation in large-scale or cost-sensitive cloud environments. On the other hand, lightweight methods, such as heuristic algorithms

Table 9 A comparative analysis of various hybrid AI-based approaches

References	Methods	Findings	Latency	Energy consumption	Cost	Pros	Cons
Hou and Ismail (2024)	EETS, DQN, DDQN	EETS shows faster convergence and higher rewards than DQN and DDQN	Low	Low	Moderate	High convergence speed	Sensitive to initial parameters
Huang et al. (2024)	MAGAIL-MCT	Reduces system overhead, enhances resource utilization, and minimizes migration delay and energy consumption	Low	Moderate	High	Efficient resource utilization	High computational cost
Shokhanda et al. (2024)	Reward-based DL	Achieves near-optimal performance in optimizing tail latency and resource management	Low	Moderate	High	Low latency	Requires substantial training data
Li et al. (2024)	PFC, EHS, Ada-HPFL-CN	Improves communication efficiency and model personalization compared to traditional FL	Low	Moderate	Moderate	Enhanced communication efficiency	Limited generalizability
Jiang et al. (2024)	TOMAC-PPO	Improves convergence speed, reduces service cost, and lowers task drop rates	Low	Low	Moderate	Reduces drop rates	Complex to implement
Cao et al. (2024)	RL	Reduces task delay and enhances reliability and system effectiveness	Low	Moderate	High	Reliable performance	Resource-intensive
Abualigah et al. (2024)	GIJA	Achieves 95% solution quality and outperforms existing scheduling algorithms	Moderate	Moderate	Moderate	High solution quality	Moderate adaptability
Abouelyazid (2024)	Deep-Hill	Reduces active hosts by four, lowering power consumption by 13.33%	Moderate	Low	High	Significant energy savings	High implementation cost
Mikram et al. (2023)	CHPSO	Outperforms PSO and CS in resource utilization and system performance	Moderate	Low	Moderate	Improved system performance	Limited scalability
Chitra and Getzi (2023)	GCRS	Optimizes server resource utilization, reducing energy consumption and emissions	Moderate	Low	Moderate	Environmentally friendly	Limited to specific use cases
Tuli et al. (2021)	DAGs	Achieves cost savings and low-latency service delivery for resource-intensive workflows	Low	Moderate	Moderate	Cost-effective	High complexity in implementation

and rule-based approaches, prioritize software efficiency and require minimal hardware resources. These methods are more suitable for environments with limited computational power or for applications where simplicity and speed are prioritized. For example, Tomarchio et al. (2020) demonstrated a heuristic-based optimization method that achieved efficient task scheduling with low computational overhead. However, these methods often struggle to adapt to complex or dynamic workloads, limiting their effectiveness in scenarios such as heterogeneous cloud systems.

Another critical tradeoff involves **compute learning needs**, which vary significantly across AI techniques. As discussed in Qureshi and Sharma (2021), supervised learning models require large labelled datasets to train effectively. While these models provide high accuracy in static environments, the cost and time associated with generating labelled data can be prohibitive. Reinforcement learning models, such as those explored in Zhu et al. (2021), are particularly well-suited to dynamic and uncertain environments due to their ability to learn from interaction with the system. However, these models demand substantial training time and computational resources, making them less ideal for real-time or resource-constrained applications. Unsupervised learning approaches, by contrast, do not require labelled data, as highlighted in Khan (2020), but often struggle with achieving the same precision as supervised methods when applied to structured or particular problems.

The overall **effectiveness of AI-based approaches** is another area of trade-off. High-accuracy methods, such as hybrid AI systems combining DRL and heuristic techniques, can deliver robust predictions and decision-making capabilities (Aldahwan and Ramzan 2022). However, these methods come with higher computational costs and increased complexity, requiring advanced hardware and significant expertise to implement effectively. Conversely, simpler algorithms, such as PSO or ACO, are better suited for real-time applications where low latency is critical (Zhu et al. 2021). While these methods provide faster results with lower resource demands, they may sacrifice accuracy or fail to scale efficiently in diverse or high-demand environments.

Practical examples from the literature illustrate these trade-offs. For instance, the reinforcement learning-based job scheduling approach described in Aldahwan and Ramzan (2022) demonstrated high adaptability to changing workloads but required significant computational resources during the training phase. Similarly, the lightweight blockchain-based resource allocation mechanism in N'Goran et al. (2023) was efficient in resource-constrained settings but faced challenges when scaling to larger workloads. These examples highlight the inherent trade-offs between cloud computing adaptability, computational efficiency, and scalability.

The tradeoff between hardware efficiency, compute learning needs, and overall effectiveness depends heavily on the application's specific requirements and constraints. Cost-sensitive environments may favour lightweight, heuristic-based methods, while accuracy-critical or dynamic scenarios may justify the investment in high-performance, hardware-intensive AI models. The surveyed literature underscores the importance of tailoring AI-based approaches to each cloud computing application's unique challenges and objectives, balancing competing priorities to optimize outcomes effectively.

Table 10 Tradeoffs in AI-based approaches for cloud computing

Aspect	Description
Hardware vs. software efficiency	High-performance AI models (e.g., deep neural networks) require specialized hardware, such as GPUs or TPUs, leading to increased costs and energy consumption. In contrast, lightweight methods like heuristic algorithms are resource-efficient but may lack adaptability in complex or dynamic scenarios
Compute learning needs	Supervised learning achieves high accuracy but requires labelled datasets, increasing setup costs. Reinforcement learning models excel in dynamic tasks but demand extensive training time and computational power. Unsupervised learning has lower data requirements but may struggle in structured domains
Overall effectiveness	High-accuracy methods (e.g., hybrid AI systems) provide better predictions but incur higher computational costs. Real-time applications prioritize low latency, favouring faster yet simpler models, often trading accuracy for speed
Energy-efficient scheduling	Optimization algorithms (e.g., Ant Colony Optimization, Particle Swarm Optimization) reduce energy usage but may face scalability challenges in large cloud environments
Security applications	AI models for security, such as anomaly detection, prioritize accuracy to minimize false positives. These models often require robust hardware to process data in real-time, increasing computational demands
Balancing tradeoffs	Cost-sensitive applications prefer simpler models to reduce expenses, while accuracy-critical tasks justify investment in high-performance systems. Dynamic environments require adaptive models like reinforcement learning despite their higher resource needs

5 AI algorithms and supported job types in cloud computing

In cloud computing, diverse job types require tailored scheduling approaches to achieve efficiency and scalability. AI algorithms have demonstrated significant potential in optimizing resource utilization, energy efficiency, and real-time adaptability. These jobs include:

Task classification

Supervised learning algorithms, such as Logistic Regression and SVM, excel in classifying incoming tasks based on predefined features. For instance:

- In large-scale cloud environments, a task classification model can predict whether a job is compute-intensive (e.g., image processing) or memory-intensive (e.g., database querying). This ensures that tasks are routed to the appropriate VMs.
- Companies like Amazon Web Services (AWS) use classification models to distinguish between high-priority tasks (e.g., transactional workloads) and low-priority tasks (e.g., batch processing) to optimize resource allocation.

Resource allocation

Supervised and hybrid AI algorithms play a pivotal role in predicting resource requirements. For example:

- Google's Kubernetes uses machine learning models to allocate computing resources dynamically, ensuring that workloads do not exceed available capacity while maintaining cost efficiency.
- In hybrid cloud setups, supervised learning models can identify underutilized resources and reassign tasks to improve overall resource utilization.

Workflow prediction

AI-based prediction models help detect bottlenecks and optimize workflows in distributed systems. Examples include:

- In scientific computing environments, such as CERN's particle physics experiments, workflow prediction algorithms schedule interdependent tasks to ensure timely execution without deadlocks.
- WorkflowSim, a simulation framework for cloud workflows, leverages AI to predict delays in Directed Acyclic Graph (DAG) workflows, enabling proactive rescheduling.

Real-time job scheduling

RL techniques, such as Deep Q-Learning, adapt dynamically to real-time workload changes. Real-world applications include:

- Autonomous vehicle systems rely on real-time scheduling to process sensor data in milliseconds, where delays can lead to safety hazards.
- Streaming platforms like Netflix use reinforcement learning to allocate bandwidth and computing resources during peak traffic periods dynamically.

Dynamic resource allocation

RL models, such as Federated DRL, excel in dynamically adjusting resources in distributed systems. Examples include:

- Alibaba Cloud uses RL-based systems to manage large-scale e-commerce events (e.g., Singles' Day), ensuring minimal latency and high availability.
- Edge computing platforms utilize RL to allocate nearby resources for latency-sensitive applications, such as real-time video analytics in smart cities.

Task dependency scheduling

Optimization techniques, like Genetic Algorithms, schedule tasks with dependencies. Examples include:

- In industrial automation, where robotic operations depend on sequential task execution, optimization algorithms ensure that workflows run without interruptions.
- Cloud platforms like Microsoft Azure use optimization algorithms to minimize delays in serverless workflows, such as those triggered by IoT devices.

Energy-efficient job assignment

Energy efficiency is critical for sustainable cloud computing. Optimization techniques and hybrid AI models are deployed in scenarios such as:

- Google's DeepMind AI optimizes energy consumption in data centres, reducing cooling costs while maintaining server performance.
- Smart grid systems utilize AI algorithms to dynamically assign computation-heavy tasks to low-energy periods, reducing peak power demand.

Multi-cloud job allocation

Hybrid AI models address the complexity of scheduling jobs across multiple cloud providers. For example:

- Enterprises using multi-cloud strategies, such as Netflix, distribute workloads across AWS, Google Cloud, and Azure for redundancy and cost optimization.
- Federated DRL systems enable collaborative scheduling in multi-cloud manufacturing and supply chain management architectures, improving reliability and resource sharing.

Energy-aware scheduling

Hybrid models that incorporate energy metrics ensure sustainability. For example:

- IBM Cloud uses energy-aware scheduling to prioritize jobs during off-peak hours, aligning with renewable energy availability.
- AI algorithms in green data centres allocate tasks to servers on renewable energy sources, reducing the carbon footprint.

SLA-based optimization

Hybrid AI models ensure critical jobs meet SLAs. Examples include:

- Financial institutions use SLA-aware scheduling to guarantee that high-priority financial transactions are processed within microseconds, ensuring compliance with regulatory standards.
- SLA-aware job scheduling is critical in healthcare, where AI-based systems prioritize urgent medical imaging tasks in cloud-based diagnostic platforms.

Table 11 categorizes AI-based algorithms according to their representative techniques, the job types they support, and their key features. This categorization highlights how each algorithm type uniquely addresses the challenges of cloud computing, such as dynamic resource allocation, energy efficiency, and real-time job scheduling.

5.1 Comparative analysis of job scheduling algorithms

The effectiveness of job scheduling algorithms can be evaluated based on various performance metrics and system objectives. Different algorithms prioritize specific aspects of scheduling, such as minimizing execution time, reducing energy consumption, or ensuring system reliability. This section compares job scheduling algorithms, focusing on crucial criteria like makespan minimization, cost efficiency, energy consumption, load balancing, and fault tolerance. This analysis helps identify the best-fit solutions for specific environments and use cases by comparing how different algorithms perform in these areas. Understanding the trade-offs between these metrics allows for more informed decision-making in selecting the optimal scheduling algorithm for cloud computing and other complex systems.

Makespan minimization

Makespan refers to the total time required to complete a set of jobs. Many job scheduling algorithms aim to minimize makespan to improve system efficiency and throughput. Algorithms prioritising makespan minimization, such as heuristic-based methods and hybrid AI models, focus on reducing job completion time, ensuring faster job turnaround. In job scheduling, minimizing makespan is particularly critical in environments with high workloads and tight deadlines, as it directly impacts overall system performance.

Cost efficiency

Cost efficiency in job scheduling involves minimizing the expenses of executing jobs, such as computing resources and energy consumption. Cloud-based job scheduling, for example, often includes a cost component, where reducing resource usage translates into

Table 11 AI-based algorithms and supported job types in cloud computing

Algorithm type	Representative algorithms	Supported job types	Key features
Supervised learning	Logistic regression, SVM	Task classification, resource allocation, workflow prediction	Predictive capabilities for labelled data, suitable for task prioritization and resource management
Reinforcement learning	Deep Q-learning, federated DRL	Real-time job scheduling, dynamic resource allocation, QoS optimization	Adaptive to dynamic environments, learns from feedback to optimize scheduling decisions
AI-based optimization techniques	Genetic algorithm, ant colony	Task dependency scheduling, energy-efficient job assignment	Solves multi-objective problems, balances energy consumption and execution efficiency
Hybrid AI models	DeepJS, federated DRL	Multi-cloud job allocation, energy-aware scheduling, SLA-based optimization	Combines strengths of multiple AI techniques for robust and versatile solutions

lower operational costs. Algorithms focusing on cost efficiency must balance job performance with resource utilization to optimize both aspects. Machine learning and optimization-based techniques are commonly used to predict and manage resources cost-effectively, ensuring that jobs are completed within budget constraints.

Energy consumption and green scheduling

Energy consumption has become critical in job scheduling, particularly in large data centres and cloud environments where reducing energy use can significantly lower operational costs and environmental impact. Green scheduling algorithms prioritize energy efficiency, minimizing power consumption while maintaining system performance. Techniques like dynamic voltage scaling, energy-aware scheduling, and AI-based methods optimize energy usage, making green scheduling a vital aspect of sustainable computing. Energy-efficient scheduling also helps reduce the carbon footprint of data centres and computing infrastructure.

Load balancing

Load balancing ensures that jobs are evenly distributed across resources to prevent bottlenecks and underutilization. Scheduling algorithms that prioritize load balancing aim to maximize resource utilization while avoiding overload on specific nodes. Heuristic and meta-heuristic methods are often used to distribute jobs efficiently across the network, ensuring that no single resource is overburdened. Effective load balancing improves system performance and enhances reliability by reducing the likelihood of system failures due to resource exhaustion.

Reliability and fault tolerance

Reliability and fault tolerance are critical metrics for ensuring continuous operation and resilience in job scheduling. Algorithms that focus on these aspects are designed to handle system failures, ensuring that jobs can still be completed even in the event of a malfunction. Redundancy, checkpointing, and AI-based recovery strategies enhance system reliability. Fault-tolerant scheduling algorithms ensure that jobs are rescheduled or rerouted in case of failure, minimizing downtime and maintaining consistent service levels. Reliability and fault tolerance are essential to effective job scheduling in environments where high availability is crucial, such as healthcare or finance.

Table 12 presents a comparative analysis of job scheduling algorithms based on several key performance metrics. It categorizes different scheduling approaches and evaluates them across critical dimensions such as makespan minimization, cost efficiency, energy consumption, load balancing, and reliability. This structured comparison allows for a clear understanding of how various algorithms perform relative to each other, highlighting their strengths and potential areas for improvement in addressing scheduling challenges in cloud environments. The table is a valuable reference for selecting the most appropriate scheduling techniques based on specific system requirements.

5.2 Comparative benchmarking of AI techniques for cloud job scheduling

The performance of AI-driven job scheduling models can vary significantly across cloud environments, depending on workload characteristics, resource heterogeneity, and dynamic demands. This section benchmarks commonly used AI techniques—supervised learning, reinforcement learning, unsupervised learning, and hybrid approaches—by analyzing their performance against scalability, energy efficiency, response time, and cost-effectiveness

Table 12 A comparative analysis of job scheduling approaches

References	Makespan minimization	Cost efficiency	Energy consumption and Green scheduling	Load balancing	Reliability and fault tolerance
Onyema et al. (2024)	✓	✓	✗	✓	✗
Ranichandra et al. (2018)	✓	✓	✗	✓	✗
Liu et al. (2019)	✓	✓	✗	✓	✗
Loganathan et al. (2017)	✓	✓	✓	✗	✗
Kaur et al. (2022a)	✓	✓	✓	✗	✗
Saroha and Rana (2019)	✗	✓	✗	✗	✗
Baomin et al. (2011)	✓	✓	✗	✓	✗
Li et al. (2021a)	✓	✓	✗	✗	✗
Shi et al. (2022)	✓	✓	✗	✓	✓
Cheng et al. (2022b)	✓	✓	✗	✗	✗
Wang et al. (2024)	✗	✓	✓	✗	✗
Wei et al. (2018)	✓	✗	✗	✗	✓
Li and Hu (2019)	✓	✓	✗	✓	✗
Cheng et al. (2022a)	✓	✓	✗	✗	✗
Li and Peng (2022)	✗	✓	✗	✗	✗
Huang et al. (2022)	✓	✗	✗	✗	✓
Balla et al. (2018)	✓	✓	✗	✗	✗
Yang et al. (2011)	✓	✓	✗	✓	✗
Hu et al. (2012)	✓	✓	✗	✗	✓
Singhal et al. (2024)	✗	✓	✓	✗	✗
Khaleel (2023)	✓	✓	✓	✓	✓
Kaur et al. (2023)	✓	✓	✓	✓	✗
Sun et al. (2019)	✓	✓	✓	✓	✗
Suganya et al. (2023)	✗	✗	✗	✓	✓
Karthick et al. (2014)	✓	✓	✗	✓	✗
Gouasmi et al. (2017)	✓	✓	✗	✗	✓
Goga et al. (2019)	✓	✓	✗	✗	✗
Kim et al. (2024)	✓	✗	✗	✓	✗
Lin et al. (2018)	✓	✓	✗	✓	✓
Hou and Ismail (2024)	✓	✓	✗	✓	✗
Huang et al. (2024)	✓	✓	✓	✓	✗
Shokhanda et al. (2024)	✓	✗	✓	✓	✗
Li et al. (2024)	✗	✓	✓	✗	✗
Jiang et al. (2024)	✓	✓	✓	✓	✓
Abualigah et al. (2024)	✓	✓	✓	✓	✗
Cao et al. (2024)	✓	✓	✗	✓	✗
Abouelyazid (2024)	✓	✓	✗	✓	✗
Mikram et al. (2023)	✓	✓	✗	✓	✗
Chitra and Getzi (2023)	✓	✓	✗	✗	✓
Tuli et al. (2021)	✓	✓	✗	✓	✓

metrics. The comparative discussion highlights each approach's strengths, limitations, and use cases, providing a deeper understanding of their suitability in different cloud scenarios. To better understand how various AI techniques perform in cloud job scheduling, a comparative analysis is presented in Table 13. The table highlights each technique's strengths, limitations, and best use cases, offering valuable insights into their applicability across various cloud environments. This structured comparison aids in identifying the most suitable approach for specific challenges in dynamic, heterogeneous, and large-scale cloud systems.

6 Real-world examples of AI-based scheduling

The practical implementation of AI-based scheduling techniques in real-world systems has revolutionized how cloud environments manage resources and handle dynamic workloads. Leading organizations across various industries have adopted AI-driven frameworks to optimize resource allocation, enhance system performance, and ensure reliability during high-demand scenarios. This section explores notable examples of AI-based scheduling, highlighting the technical methodologies and outcomes achieved by Google, Alibaba, Microsoft Azure, and Netflix. These real-world applications illustrate the transformative potential of AI in addressing the challenges of modern cloud computing environments, including scalability, efficiency, and fault tolerance.

Google's Borg system

Google's Borg system is a cluster management and Ato that efficiently manages workloads across tens of thousands of servers in the world's largest cloud infrastructures. It was designed to optimize resource allocation through machine learning and automation (Shahmirzadi et al. 2024).

- Borg operates using *containers* as its basic workload unit, encapsulating applications along with their dependencies and configurations to ensure portability and consistency.
- The system employs a *centralized scheduler* that makes resource allocation decisions based on real-time data and predictive analytics:
  - *Resource requests* Jobs specify the CPU, memory, and storage they require.
  - *Prioritization policies* Jobs are assigned priorities, enabling higher-priority jobs to preempt lower-priority ones if necessary.
- Borg leverages *machine learning models* to predict job resource needs and minimize idle resource usage, ensuring utilization rates often exceed 70–80%, a benchmark for large-scale systems.
- Fault tolerance is achieved through *replication and job rescheduling*, where Borg continuously monitors job health and automatically restarts jobs in case of node failures.
- These techniques significantly reduce operational costs while maintaining system reliability and efficiency.

Figure 17 illustrates the architecture of the Borg cluster manager, which is designed to manage large-scale clusters efficiently. At its core is the BorgMaster, which handles job scheduling and resource management. The scheduler allocates resources to jobs based on

Table 13 A comparative analysis of AI techniques for cloud job scheduling

AI technique	Strengths	Limitations	Best use cases
Supervised learning	High accuracy in static environments; suitable for predictable workloads; low computational overhead	Limited adaptability to dynamic workloads; requires large labelled datasets; less effective for real-time scenarios	Task classification and predictive scheduling in static or structured systems
RL	Highly adaptable to dynamic and real-time workloads; effective for multi-objective optimization; suitable for unpredictable environments	High training cost and computational resources; requires fine-tuning of hyperparameters; struggles with sparse rewards	Real-time workload balancing and energy-efficient scheduling in hybrid and multi-cloud systems
Unsupervised learning	Effective for detecting hidden patterns; works well with unlabeled data; useful for anomaly detection and clustering	Limited direct application to job scheduling; computationally expensive for large-scale clustering	Anomaly detection, resource optimization, and identifying resource bottlenecks
Hybrid AI approaches	Balances exploration and exploitation; robust to dynamic changes; capable of multi-objective optimization; suitable for heterogeneous cloud systems	Increased complexity in model design and implementation; high computational resource requirements	Multi-cloud resource allocation and complex scheduling scenarios in large-scale, heterogeneous environments

predefined priorities, balancing load across the cluster. The BorgMaster interacts with a persistent store (using Paxos for consistency) and divides its responsibilities across read/UI shards for user interactions and a link shard for inter-node communication. It manages multiple Borglets, which run on individual machines to execute tasks and report resource usage. Users interact with Borg through command-line tools, web interfaces, and configuration files, allowing seamless job submission and monitoring. This architecture ensures high scalability, fault tolerance, and efficient resource utilization, making it ideal for managing job scheduling in large distributed systems.

Alibaba's elastic scheduling framework

Alibaba's Elastic Scheduling Framework is an AI-powered solution to manage the company's extensive e-commerce operations (Minxian et al. 2024; Lu et al. 2023), particularly during high-demand periods like Singles' Day.

- The framework employs *deep reinforcement learning (DRL)* to enable adaptive and real-time resource allocation. The DRL agent learns workload dynamics by interacting with the cloud environment.
- *Resource elasticity* During peak demand, the system scales resources horizontally by spinning up additional VMs or containers and scales down during off-peak periods to conserve energy and reduce costs.
- *Load forecasting* Historical traffic patterns and real-time monitoring feed models such as *LSTM* neural networks to predict traffic and resource needs.
- *Priority-based scheduling* Critical tasks like transaction processing are prioritized, while non-urgent tasks such as analytics are deferred or assigned fewer resources.
- The system optimizes both *performance and cost-efficiency*, ensuring responsive services while minimizing overprovisioning.

Figure 18 presents the architecture with a highly available and scalable system built using Alibaba Cloud services. The system is designed for both dynamic and static content delivery. Users access the application through a domain name resolved by DNS, with content served via a Content Delivery Network for dynamic content and Object Storage Service for static content. Load Balancers distribute traffic to auto-scaling groups of Elastic Compute Service instances, divided into web server and application server groups across multiple availability zones for redundancy. A relational database service is configured with a master instance in one zone and a standby in another to ensure database failover and high availability. Auto-scaling ensures that resources dynamically adjust based on traffic demands, enhancing performance and reliability. Static content and backups are stored securely in OSS for easy access and disaster recovery.

Microsoft Azure's intelligent workload balancing

Microsoft Azure integrates AI into its workload balancing to meet the dynamic requirements of enterprise and consumer cloud applications, ensuring that resource allocation aligns with SLAs (Kaul 2019; Rajeshwari et al. 2022).

- Azure employs *machine learning models*, such as regression and decision trees, to predict resource demand based on historical usage patterns and real-time data.
- *Dynamic provisioning* Resources are allocated dynamically based on predictions. For instance, additional VMs are spun up during high demand, while underused VMs are

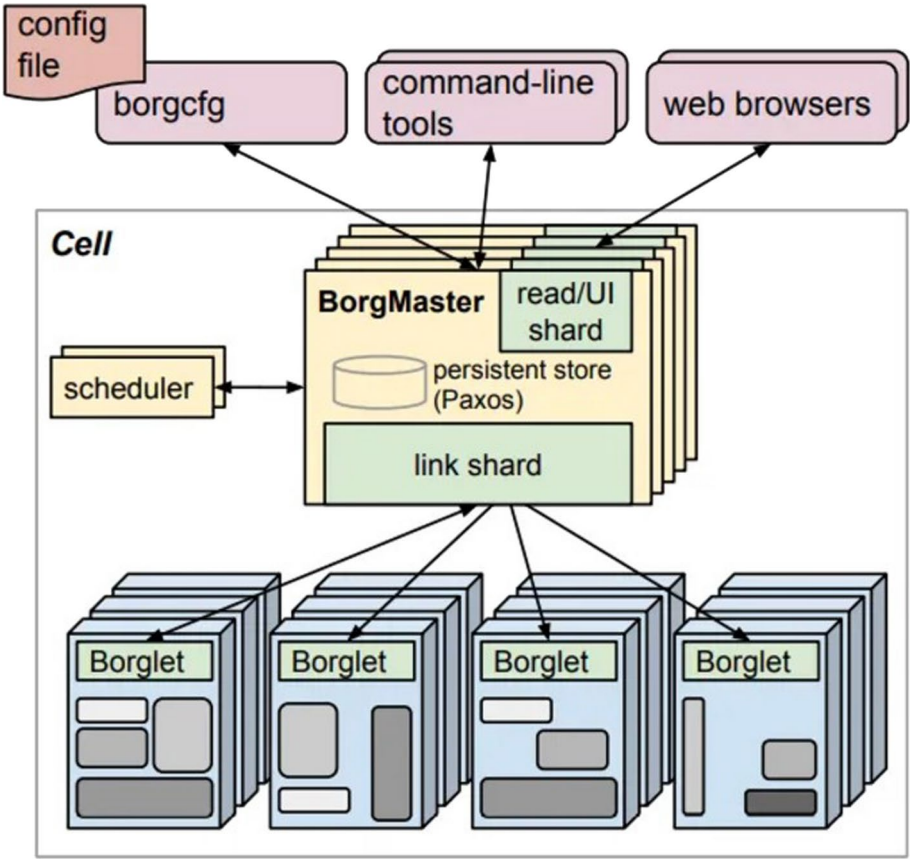

Fig. 17 Architecture of Borg cluster manger (Shete 2024)

deallocated to reduce costs.

- *Load distribution* AI balances workloads across geographically distributed data centres, minimizing latency and ensuring redundancy during localized failures.
- *Energy efficiency* Optimization algorithms consolidate workloads onto fewer servers during off-peak hours, reducing power consumption.
- *SLA compliance* Predictive scaling prevents performance degradation during traffic surges, ensuring critical workloads meet SLA guarantees.

For instance, Fig. 19 depicts the architecture of Azure workload balancer, demonstrating a high-availability, multi-tier Azure setup leveraging Traffic Manager, Application Gateway, and Load Balancer to optimize performance and reliability across two regions. Traffic Manager acts as a global load balancer, distributing incoming traffic between areas based on user-defined performance or geographic proximity policies. Within each region, the Application Gateway routes traffic at the web tier, directing specific requests (e.g., '/images/*') to designated server pools (Image or Default Server Pool). The Load Balancer evenly distributes requests among VMs to enhance fault tolerance and scalability at the database tier. This

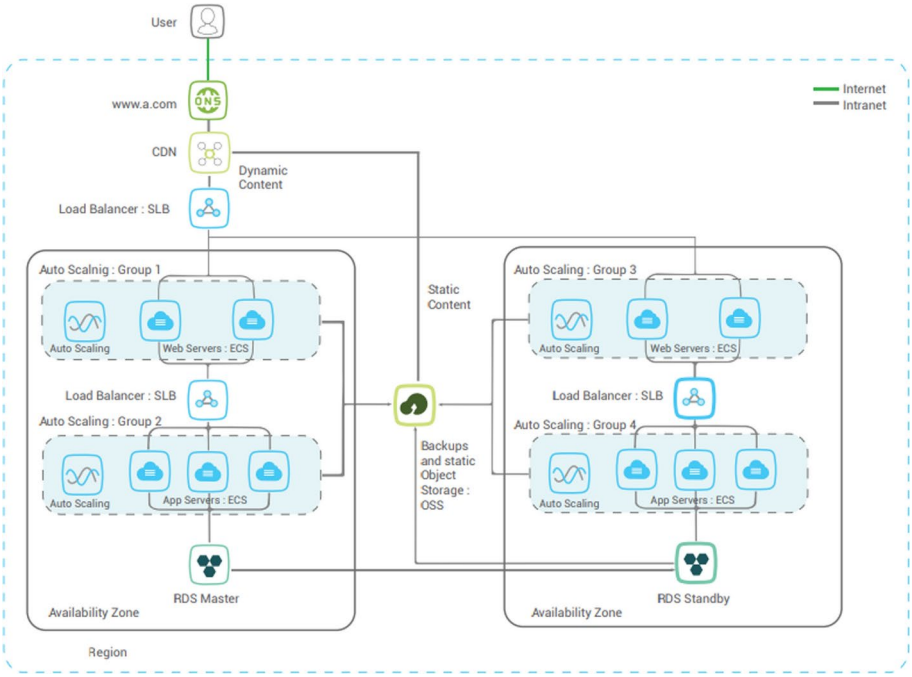

Fig. 18 Architecture of Alibaba workload balancer (Alibaba Cloud 2024)

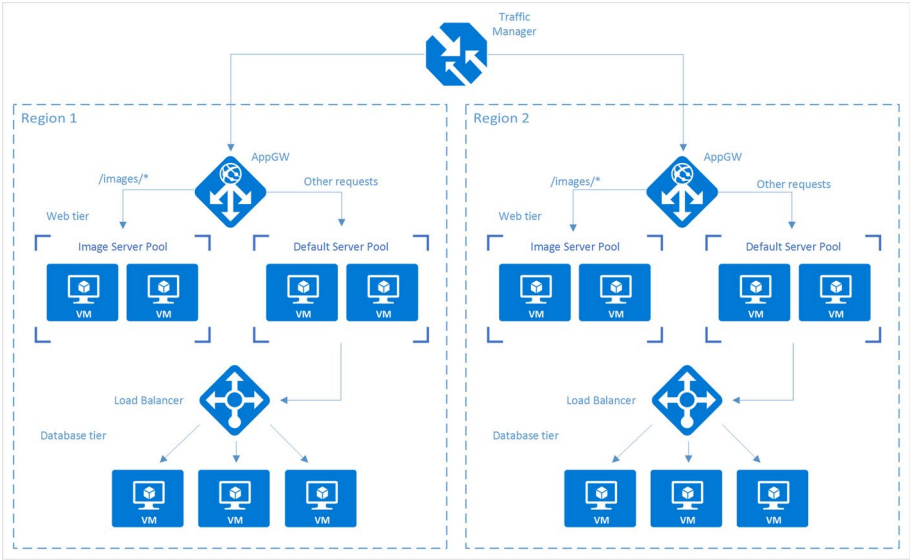

Fig. 19 Architecture of Azure workload balancer (Microsoft Learn 2024)

design ensures low latency, high availability, and resilience, making it suitable for global applications requiring redundancy, dynamic scalability, and efficient resource utilization.

Netflix's dynamic resource scaling

Netflix leverages AI to optimize cloud resource allocation, ensuring seamless streaming experiences, mainly during high-traffic periods like new show releases (Suleiman and Murtaza 2024; Rahumath et al. 2021).

- *Viewer demand prediction* Netflix employs AI models, such as Recurrent Neural Networks (RNNs) and LSTMs, to analyze historical viewing patterns and predict spikes in demand.
- *Auto-scaling policies* The system dynamically adjusts the number of cloud instances:
  - Additional servers are provisioned during major show releases.
  - Instances are deallocated when demand subsides to conserve resources.
- *Regional optimization* Netflix distributes content across its Content Delivery Network (CDN) using AI algorithms to minimize latency for users based on geographical location.
- *Energy and cost optimization* The system consolidates tasks and utilizes underutilized data centres during off-peak hours.
- *Fault tolerance* In case of resource failures, AI reroutes traffic to other servers in real-time to ensure uninterrupted streaming.

Figure 20 depicts Netflix's job scheduling workflow for content recommendations, highlighting the use of real-time and batch data processing. Millions of play-related signals are ingested via Kafka into a streaming layer for near-real-time processing. This data feeds into training pipelines powered by tools like Apache Spark to update machine learning models dynamically. The trained models predict user preferences, with outputs passed through a precompute/live compute layer to generate personalized rankings. These rankings are stored in online caches or datastores for quick access during user interactions. Job scheduling ensures the seamless execution of data ingestion, training, and computation tasks, enabling

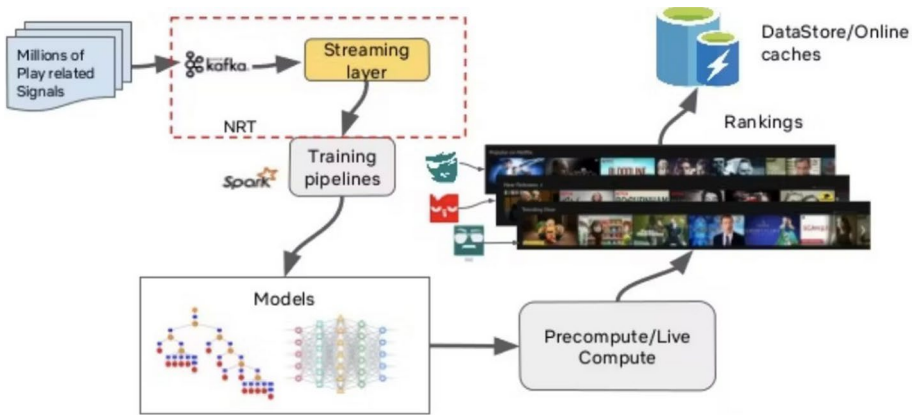

Fig. 20 Netflix's job scheduling workflow (Blog Jobins 2024)

efficient updates to recommendation systems and delivering a personalized experience to users.

These real-world examples demonstrate how AI-based scheduling transforms cloud computing environments by achieving:

- *Scalability* Adapting to dynamic workloads and ensuring system stability during traffic surges.
- *Efficiency* Reducing operational costs and energy consumption through predictive and dynamic resource allocation.
- *Reliability* Enhancing fault tolerance and SLA compliance through intelligent decision-making and real-time monitoring.

These practical applications underscore the transformative potential of AI-driven job scheduling in modern cloud infrastructures.

7 Discussion and lessons learned

Dynamic, multi-cloud environments present unique challenges, such as resource heterogeneity, energy consumption, real-time adaptability, scalability, and performance optimization under varying workloads. Resource heterogeneity arises from the diverse capabilities, configurations, and characteristics of resources provided by different cloud vendors. Addressing this requires intelligent strategies that dynamically allocate resources based on task requirements. Simultaneously, energy consumption in multi-cloud environments has become a critical concern due to the environmental impact and operational costs associated with large-scale cloud operations. Moreover, real-time adaptability is essential to handle fluctuating workloads, user demands, and unpredictable network conditions. Finally, scalability and performance must be ensured as workload complexities and system sizes increase, particularly in dynamic and highly distributed environments.

AI-driven job scheduling approaches are well-positioned to address these challenges by integrating machine learning, heuristic techniques, optimization algorithms, and hybrid AI models. Machine learning-based scheduling methods leverage historical and real-time data to optimize task placement and predict resource demand, improving resource utilization and minimizing latency. For instance, RL models, particularly DRL, adapt dynamically to environmental changes, enabling the efficient allocation of tasks to available resources. Real-world implementations of DRL in cloud computing have demonstrated reductions in task completion times and energy consumption, highlighting its potential to handle dynamic workloads effectively.

Heuristic techniques, such as Min–Min and Max-Min, offer fast, near-optimal solutions for job scheduling by prioritizing tasks based on their execution requirements. These approaches are particularly beneficial in scenarios with limited computational power or when time-sensitive decisions are required. While they may lack the robustness of more advanced optimization methods, heuristics remain valuable in AI-driven scheduling frameworks for small- to medium-scale cloud environments. Optimization algorithms, including GA, PSO, and ACO, further enhance scheduling efficiency by balancing resource utilization, minimizing makespan, and optimizing multiple objectives. Multi-objective optimiza-

tion approaches incorporate trade-offs between energy consumption and task performance, providing solutions tailored to diverse cloud environments. For example, hybrid models combining GA and PSO have been employed in cloud data centres to achieve energy-efficient task scheduling, reducing operational costs and carbon footprints while maintaining high performance.

Hybrid AI models represent an advanced solution by combining the strengths of ML, heuristics, and optimization algorithms. These models enable robust, scalable, and adaptive job scheduling in complex environments. For instance, hybrid approaches integrating DRL with heuristic techniques have been successfully applied to multi-cloud scenarios, optimizing workload distribution across heterogeneous cloud resources. These models utilize ML for demand forecasting, heuristics for initial task allocation, and optimization techniques for fine-tuning, ensuring consistent performance even under varying workloads. Resource heterogeneity is effectively addressed by leveraging AI models that dynamically allocate tasks based on resource characteristics. For example, optimization algorithms like ACO and GA ensure workload distribution across diverse resource types, improving compatibility and system efficiency. Energy consumption is minimized through energy-aware scheduling models, where multi-objective DRL strategies incorporate energy metrics into scheduling objectives, achieving significant power savings in practical applications.

Furthermore, hybrid models ensure real-time adaptability by dynamically adjusting scheduling policies in response to real-time workload and environmental changes. In edge-cloud environments, DRL combined with heuristic techniques ensures low-latency responses to variations in demand and system conditions. Integrating advanced AI techniques that maintain robust operation under high workload variability ensures scalability and performance. Hybrid models, such as PSO-GA, have demonstrated their ability to scale effectively in multi-cloud setups, delivering low latency and high throughput even in large-scale, dynamic environments. These models achieve scalability by balancing computational complexity with performance optimization, leveraging the strengths of each integrated technique. In conclusion, AI-driven job scheduling approaches provide a comprehensive solution to resource heterogeneity, energy consumption, real-time adaptability, scalability, and performance in dynamic multi-cloud environments. By integrating diverse AI techniques, these approaches offer scalable, adaptive, and efficient solutions, making them indispensable for modern cloud computing systems. This discussion highlights the practical applicability of these strategies and underscores their potential to transform resource management in multi-cloud environments.

Job scheduling in cloud computing has undergone substantial advancements, with numerous techniques developed to tackle challenges such as resource allocation, energy efficiency, and cost optimization. Despite these progressions, the practical implementation of these approaches often uncovers inherent limitations and trade-offs that warrant careful consideration. Analyzing the surveyed methods reveals several vital lessons, providing valuable insights into their strengths, specific application areas, and potential improvement avenues. Table 14 presents a structured summary of these findings, emphasizing the accomplishments achieved and the remaining challenges to address.

Table 14 Lessons learned and limitations of current job scheduling Techniques

Aspect	Lessons learned	Limitations	Future opportunities
Diverse objectives	Multi-objective optimization has advanced, balancing metrics like makespan, energy consumption, and fairness	Algorithms often prioritize one objective over others, leading to suboptimal outcomes for other metrics	Develop unified frameworks for balanced multi-objective optimization across diverse scenarios
Role of AI and machine learning	AI-driven approaches, especially reinforcement learning, adapt well to dynamic environments and unpredictable workloads	Computational resource demands and overfitting to specific scenarios limit broader applicability	Simplify AI models to balance adaptability with computational efficiency for real-time applications
Energy efficiency	Energy-aware scheduling has shown significant power savings through resource scaling and intelligent allocation	Trade-offs with execution time and fairness remain challenges in real-world deployments	Integrate renewable energy sources and fine-tune trade-offs in energy-efficient algorithms
Scalability	Hierarchical and distributed mechanisms ensure consistent performance under increasing workload scales	Small-scale algorithms often fail in large distributed systems, creating bottlenecks	Design modular algorithms to handle large-scale systems dynamically
Heterogeneity of resources	Heterogeneity-aware techniques improve task-to-resource mapping in diverse systems	Extreme heterogeneity in hybrid and multi-cloud systems poses significant challenges	Develop universal schedulers adaptable to varied resource configurations
Workflow scheduling	Workflow-aware algorithms manage interdependent tasks effectively, leveraging tools like WorkflowSim	High overhead and inability to handle dynamic workflow changes hinder scalability	Implement dynamic scheduling policies that adjust to real-time workflow changes
Security and privacy	Emerging techniques address data isolation and access control in multi-tenant cloud environments	Integration of robust security measures into scheduling remains limited	Design scheduling algorithms with built-in security and privacy protocols
Real-world validation	Simulation frameworks like CloudSim provide valuable environments for testing scheduling techniques	Simulation environments do not fully capture real-world variability, limiting practical validation	Enhance real-world deployment and testing to improve algorithm robustness
Cost optimization	Cost-aware algorithms optimize financial expenses for providers and users by fine-tuning resource allocation	Trade-offs with execution time and energy efficiency are common challenges	Incorporate dynamic pricing models to improve cost-effectiveness across cloud systems
Latency reduction	Latency-sensitive scheduling is critical for real-time applications like IoT and edge computing	Meeting stringent latency requirements is challenging in dynamic and large-scale systems	Focus on hybrid scheduling models combining edge and cloud to minimize latency
Generalizability	Specialized techniques effectively address specific challenges, such as edge computing or high-performance tasks	Most algorithms lack generalizability across diverse cloud scenarios	Develop adaptive scheduling algorithms applicable across varied environments and use cases

7.1 AI-driven job scheduling frameworks: addressing limitations of traditional systems

Traditional scheduling systems often face significant limitations in dynamic and complex cloud environments, particularly when addressing complex job dependencies, system faults, and energy efficiency in multi-cloud infrastructures. AI-driven job scheduling frameworks present a transformative solution to these challenges by leveraging intelligent algorithms capable of dynamic decision-making and real-time adaptability.

Handling complex dependencies between jobs

Traditional scheduling systems often rely on static algorithms that are not well-equipped to handle complex dependencies between jobs. Dependencies can arise from workflows requiring tasks to be executed in a specific order or sharing resources, which, if not managed efficiently, can lead to bottlenecks and delays. AI-driven frameworks address these issues through reinforcement learning and dependency graph analysis. By learning from historical data, these frameworks can optimize task prioritization and allocate resources dynamically to ensure that dependencies are respected while minimizing delays. Machine learning models, such as Graph Neural Networks (GNNs), can also model and resolve intricate job relationships, ensuring efficient scheduling even in complex workflows.

Mitigating system faults

Fault tolerance is a critical challenge in large-scale cloud environments, where failures in hardware, software, or network components can disrupt job execution. Traditional systems often use reactive fault recovery mechanisms, which may result in significant downtime. In contrast, AI-driven frameworks adopt proactive and adaptive strategies to handle faults. For instance, predictive analytics can anticipate failures based on system health metrics and initiate preemptive resource reallocation. Reinforcement learning-based models can learn optimal fault recovery strategies by simulating various failure scenarios, enabling quick recovery with minimal disruption. Additionally, multi-agent systems can enhance fault tolerance by distributing tasks across multiple nodes, ensuring redundancy and minimizing single points of failure.

Achieving energy efficiency and cost-effectiveness

Energy efficiency and cost-effectiveness are paramount in multi-cloud environments, where resources are shared across numerous applications and users. Traditional systems often fail to adapt resource allocation dynamically, leading to energy waste and increased costs. AI-driven scheduling frameworks overcome these limitations through predictive and adaptive resource management. For example:

- *Energy prediction models* Machine learning models can predict energy consumption patterns based on workload characteristics and adjust resource allocation to minimize energy use.
- *Dynamic scaling* AI frameworks can implement dynamic scaling strategies, such as turning off idle resources or shifting workloads to energy-efficient nodes.
- *Multi-objective optimization* AI-driven systems use optimization techniques, such as genetic algorithms or PSO, to balance trade-offs between energy consumption, execution time, and cost. These techniques ensure that resources are utilized efficiently while meeting performance and budget constraints.

Scalability in multi-cloud environments

Large-scale multi-cloud environments introduce additional challenges due to their heterogeneity and distributed nature. Traditional systems struggle to maintain performance consistency and efficient resource utilization across multiple clouds. AI-driven frameworks excel in these scenarios by integrating real-time data analysis and adaptive scheduling. Federated learning techniques, for instance, allow scheduling algorithms to learn and optimize across multiple cloud environments without centralizing data, preserving privacy and reducing overhead. Furthermore, meta-learning approaches enable the scheduler to adapt quickly to new environments by transferring knowledge from previously learned tasks.

By addressing these limitations, AI-driven job scheduling frameworks enhance system performance and efficiency and ensure robust and scalable operations in multi-cloud environments. Incorporating intelligent algorithms for dependency resolution, fault tolerance, and energy optimization positions AI-driven systems as the future of job scheduling in dynamic cloud infrastructures.

7.2 AI-driven task migration toward job scheduling

Task migration is a critical mechanism in cloud computing, enabling the dynamic relocation of tasks between computational resources to optimize performance, reduce latency, and balance workloads (Priyadarshini et al. 2024; Zhu et al. 2024). AI-driven task migration leverages artificial intelligence algorithms to make these decisions intelligently, ensuring tasks are allocated to the most suitable resources based on real-time system states. By analyzing metrics such as resource availability, task priority, energy consumption, and network latency, AI can predict the most efficient migration paths, minimizing overheads and improving the overall performance of cloud environments. This ability to adapt to changing workloads and resource conditions makes AI-driven task migration indispensable in modern cloud infrastructure.

One of the primary benefits of AI-driven task migration is its contribution to efficient job scheduling in dynamic and heterogeneous cloud environments. Traditional job scheduling algorithms often struggle to address the complexities of resource allocation in real-time, mainly when workloads fluctuate or unexpected failures occur. Job scheduling systems can adapt to these challenges by integrating task migration with AI capabilities (Zhu et al. 2024; Pu et al. 2024). For instance, if a resource becomes overloaded or underperforms, AI algorithms can identify optimal migration strategies to redistribute tasks, ensuring SLAs are met without compromising system reliability. AI-driven task migration is also essential for achieving energy efficiency in cloud computing. Data centers are notorious for their high energy consumption; inefficient task scheduling can exacerbate this issue (Zhu et al. 2024). By leveraging predictive analytics and machine learning models, AI can anticipate resource demand and migrate tasks to energy-efficient servers or underutilized nodes. This dynamic allocation helps reduce idle power consumption and ensures that only the necessary resources are active, contributing to greener cloud operations. Moreover, by migrating tasks away from resources with high power usage or low energy efficiency, cloud providers can optimize their operational costs while maintaining high performance.

For instance, Fig. 21 illustrates an AI-driven cloud-edge architecture for task migration and job scheduling. It features three layers: the Cloud Layer, responsible for centralized resource management, task scheduling, and load balancing; the Edge Layer, consisting of

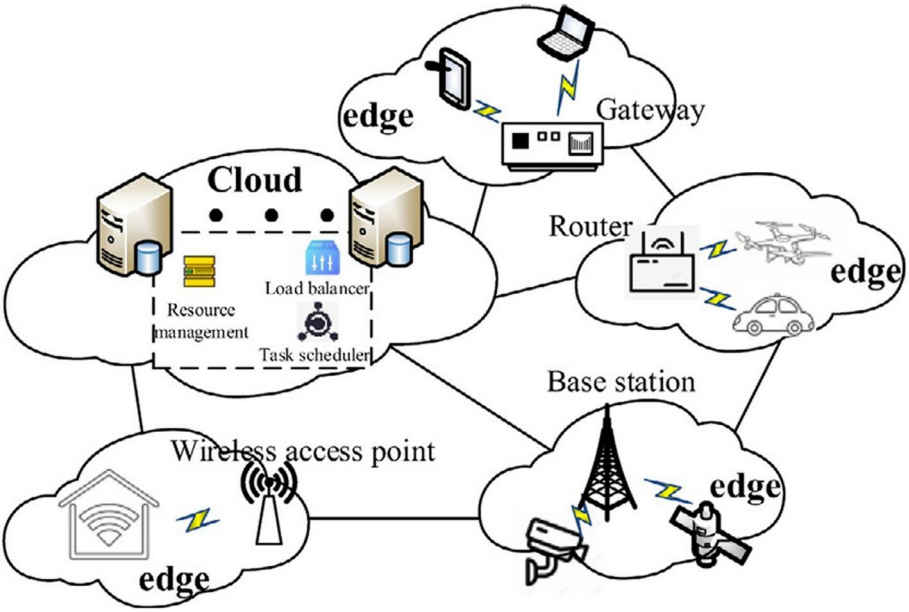

Fig. 21 Task migration toward job scheduling in cloud computing (Zhu et al. 2024)

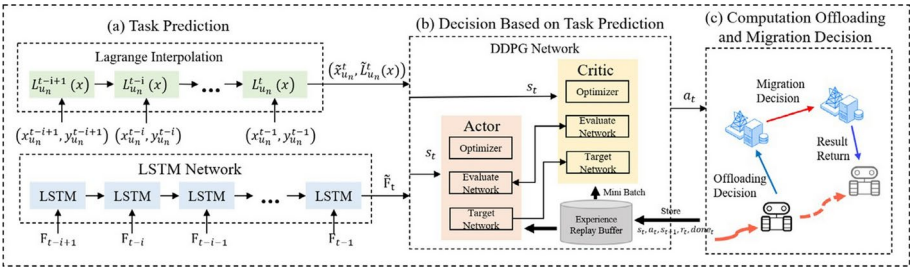

Fig. 22 AI-driven task migration toward job scheduling in cloud computing (Qin et al. 2024)

edge nodes such as gateways and base stations that process tasks closer to end devices to minimize latency; and the Device Layer, where tasks originate from end-user devices like smartphones and IoT systems. Arrows represent task migration and interaction across these layers, enabling dynamic allocation of workloads (Zhu et al. 2024). AI algorithms optimize resource utilization by migrating tasks between overloaded and underutilized nodes, reducing latency, balancing loads, and ensuring efficient job scheduling throughout the system.

Also, Fig. 22 demonstrates a three-step process for mobility-aware computation offloading and task migration in industrial IoT (IIoT) environments. In (a) Task Prediction, Lagrange interpolation predicts mobile device trajectories, while an LSTM network anticipates system resource availability, forming a foundation for subsequent decisions. (b) Decision-Based on Task Prediction uses a Deep Deterministic Policy Gradient (DDPG) network, incorporating actor and critic networks, to evaluate system states and determine optimal offloading and migration strategies. Lastly, (c) Computation Offloading and Migration

Decision dynamically allocate tasks between edge servers and devices, balancing resource usage while reducing latency and energy consumption. This architecture ensures adaptive and efficient management of IIoT tasks under mobility constraints, optimizing turnaround time and resource utilization.

The role of task migration in fault tolerance and reliability further underscores its relevance to job scheduling. Cloud environments are susceptible to resource failures, network issues, and hardware malfunctions, which can disrupt task execution and degrade service quality. AI-driven task migration enables systems to proactively detect potential failures and relocate tasks to healthy nodes before disruptions occur (Cui et al. 2024). By incorporating real-time monitoring and anomaly detection, AI enhances the resilience of cloud systems, ensuring that job scheduling processes remain uninterrupted even in adverse conditions. In multi-cloud and edge-cloud environments, AI-driven task migration is becoming increasingly significant. These architectures involve distributed resources across multiple providers or geographically dispersed locations, creating additional complexities in resource management (Hosseini Shirvani 2024). AI algorithms can analyze the characteristics of tasks and the capabilities of diverse resources to determine optimal migration strategies, ensuring seamless task execution across clouds. Furthermore, AI-driven task migration supports latency-sensitive applications by enabling tasks to move closer to the edge, reducing communication delays and improving user experiences. Thus, integrating AI in task migration enhances job scheduling efficiency and drives innovation in emerging cloud paradigms.

8 Trade-offs, complexity, and practical implications of AI-based scheduling techniques

AI has revolutionized job scheduling in cloud computing, offering intelligent and adaptive solutions for handling complex, dynamic, and heterogeneous workloads. As explored in Sect. 1, AI-based scheduling approaches such as supervised learning, RL, DL, and hybrid models significantly enhance cloud resource allocation. However, these advancements are accompanied by trade-offs that influence their computational cost, integration complexity, fairness, transparency, and applicability. This section explores these trade-offs in-depth and summarizes the comparative strengths and weaknesses of the reviewed techniques.

8.1 Advantages of AI-based scheduling techniques

Integrating AI into job scheduling frameworks has ushered in a new era of optimization and adaptability in cloud computing environments. Unlike traditional scheduling algorithms that rely on static rules or heuristic-based decision-making, AI-driven approaches offer dynamic, context-aware solutions capable of learning from historical patterns and adapting to real-time system states. These intelligent techniques harness the power of machine learning, reinforcement learning, and hybrid models to enhance resource utilization, reduce energy consumption, minimize task latency, and improve system scalability. Moreover, AI-based scheduling methods demonstrate superior performance in handling the heterogeneity and unpredictability of modern cloud infrastructures. This section outlines the key advantages of AI-based scheduling, emphasizing their role in achieving operational efficiency, cost-effectiveness, and sustainability in cloud environments.

- **Improved prediction and real-time adaptability**
  - AI models enhance scheduling flexibility and responsiveness compared to traditional algorithms.
  - Supervised learning predicts resource demands and execution times using historical data.
  - RL and DRL dynamically adjust to workload changes in hybrid and multi-cloud systems.
  - DRL agents learn optimal scheduling policies via trial-and-error, improving over time and adapting to unpredictable conditions.
  - These models support proactive and SLA-sensitive scheduling through continual learning from the environment.

- **Energy efficiency and SLA compliance**
  - AI-based models contribute significantly to energy-efficient scheduling in data centres.
  - They reduce power consumption by minimizing idle resource times using predictive analytics.
  - The DC-CFR model, for example, employs Multi-Agent RL to optimize workload distribution and HVAC control, reducing energy and carbon output.
  - Supervised learning models predict task urgency and prioritize job execution based on SLA requirements.
  - These techniques ensure SLA compliance, which is crucial for maintaining contractual guarantees in commercial cloud platforms.

- **Enhanced fault tolerance and security**
  - AI schedulers improve system resilience by reallocating jobs during hardware, software, or network failures.
  - RL and DRL models learn fault recovery strategies through environmental interaction.
  - Anomaly detection algorithms embedded in AI scheduling monitor and detect abnormal job behaviours in real-time.
  - Integrating blockchain technologies enhances data integrity and traceability in multi-tenant cloud environments.

8.2 Limitations and challenges of AI-based scheduling techniques

While AI-based scheduling techniques have demonstrated significant potential in optimizing job allocation and resource management in cloud computing, they are not without limitations. The complexity of implementing intelligent algorithms, particularly those involving deep learning or reinforcement learning, often requires substantial computational overhead and extensive training data, which may not be readily available in all scenarios. Additionally, AI models may struggle to generalize across diverse cloud environments due to variations in workload patterns, infrastructure configurations, and application require-

ments. Interpretability, model transparency, and real-time responsiveness pose challenges, particularly in mission-critical applications where scheduling decisions must be explainable and deterministic. Moreover, the integration of AI with legacy systems and the continuous need for model retraining to adapt to evolving conditions introduce additional overhead. This section delves into the technical, operational, and ethical challenges associated with AI-based scheduling techniques, providing a balanced perspective on their current limitations and areas requiring further research.

- **High computational and data requirements**
  - AI models like DRL and deep neural networks require extensive training on large-scale datasets.
  - The training process demands high computational power, especially unsuitable for edge or low-resource environments.
  - Supervised models depend on large, high-quality labelled datasets, often unavailable or noisy in real-world logs.
  - Frequent retraining is required in dynamic environments to maintain model relevance and accuracy.

- **Potential for bias and unfairness**
  - AI models may replicate biases in historical job logs, such as preference for certain job types.
  - Unchecked bias can lead to unfair resource distribution and SLA violations for underrepresented workloads.
  - RL agents, if not carefully reward-engineered, may optimize throughput at the cost of fairness or prioritization balance.

- **Integration and hyperparameter tuning complexity**
  - Integrating AI into existing cloud scheduling systems introduces added system complexity.
  - Requires dedicated infrastructure for model deployment, monitoring, rollback mechanisms, and fault tolerance.
  - Hyperparameter tuning (e.g., learning rate, architecture depth) is time-consuming and computationally intensive.
  - Poorly tuned models may overfit or underperform, making their behaviour unpredictable.

- **Limited generalizability and transparency**
  - AI scheduling models are often tailored for specific cloud environments or workload types.
  - Limited generalization prevents their reuse across different cloud setups with varying resource configurations.

- Deep learning and DRL models act as black boxes, reducing transparency in scheduling decisions.
- Lack of explainability hampers trust, especially when scheduling outcomes impact SLA or cost-sensitive operations.
- Explainable AI (XAI) is still emerging and not yet fully applicable to job scheduling contexts.

Table 15 provides a comparative summary of the major AI-based scheduling techniques reviewed in this paper, highlighting their strengths, weaknesses, and recommended use cases.

AI-based job scheduling techniques provide unprecedented capabilities in managing complex cloud computing environments. Their ability to dynamically adapt, predict workloads, and enhance energy efficiency positions them as leading solutions for modern cloud challenges. However, the computational cost, integration challenges, potential for bias, and generalizability issues must be carefully balanced. Future work should focus on developing lightweight, interpretable, and generalizable AI models, possibly leveraging federated and transfer learning to enable broader applicability in heterogeneous cloud ecosystems.

9 Challenges and open issues

Cloud computing has revolutionized how computing resources are accessed and utilized, enabling flexibility, scalability, and cost-efficiency. However, several challenges remain in job scheduling within cloud environments, amplified by the increasing complexity of cloud infrastructure and user requirements. This section discusses the most significant challenges and open issues in cloud-based job scheduling.

9.1 Scalability issues

Scalability is one of the primary challenges in cloud job scheduling, especially as the number of users and tasks grows. Cloud environments must be capable of handling massive workloads while maintaining performance efficiency. However, many traditional scheduling algorithms struggle to scale in large, dynamic cloud environments. The core issue is that the number of tasks and resources increases as cloud systems expand, which demands more computational power from the scheduling algorithms. As the size of the task pool grows, scheduling decisions become more complex, often leading to increased overhead and degraded performance. Algorithms that work efficiently for smaller systems may not scale well in more extensive cloud infrastructures, resulting in bottlenecks and inefficient resource allocation.

A scalable scheduling algorithm must manage thousands or even millions of tasks in real-time while maintaining fairness and minimizing makespan. This requires designing algorithms that distribute the computational scheduling load across multiple nodes or use decentralized methods to improve scalability. One potential solution is AI-driven, self-adaptive scheduling techniques that dynamically adjust to the cloud's growth and workload patterns. However, developing scalable, decentralized, and low-latency scheduling solutions remains an open research problem.

Table 15 Summary of AI-based scheduling techniques and trade-offs

Aspect	Lessons learned	Limitations	Future opportunities
Supervised learning	Offers predictive scheduling using historical data and enhances SLA-awareness	Requires labeled datasets and struggles to adapt to unseen or dynamic workloads	Incorporate semi-supervised learning and continual learning methods for better adaptability
Reinforcement learning	Enables adaptive, fault-tolerant scheduling by learning optimal strategies through feedback	Demands long training cycles and complex reward design, limiting immediate deployment	Design lightweight RL models and modular reward functions tailored to cloud environments
Deep reinforcement learning	Excels in complex scheduling scenarios with large state and action spaces	Computationally intensive and lacks interpretability, hindering trust in critical applications	Focus on explainable DRL models and model compression for real-time applicability
Federated deep reinforcement learning	Preserves data privacy across distributed sources while enabling shared learning	High communication overhead and synchronization challenges reduce performance	Optimize communication protocols and adopt hierarchical learning strategies for scalability
Unsupervised learning	Detects unknown patterns and clusters tasks/resources without needing labelled data	Limited use in direct scheduling decisions and lacks control mechanisms	Integrate with supervised or reinforcement methods to enhance decision-making power
Hybrid AI models	Achieve robust performance by combining strengths of multiple AI paradigms	Often complex in architecture and hard to tune or integrate effectively	Develop modular, flexible hybrid frameworks with automatic tuning and explainability features

9.2 Heterogeneous resource management

Cloud environments consist of heterogeneous resources, meaning that the computational power, memory capacity, storage, and network bandwidth available across different nodes can vary significantly. This heterogeneity challenges job scheduling algorithms, which must allocate tasks to the most suitable resources. A vital issue in heterogeneous resource management is matching the right task to the appropriate resource. Some tasks may require high-performance CPUs or large amounts of memory, while others may be I/O-bound or network-intensive. Scheduling algorithms must be capable of identifying the specific needs of each task and assigning it to a resource that can handle it efficiently. Failure to do so can result in underutilization of resources or overloading specific nodes, leading to poor performance.

Heterogeneous environments also introduce complexities in terms of load balancing. The variability in resource capabilities makes it challenging to evenly distribute tasks across all nodes, as some may be more powerful than others. This creates a need for sophisticated load-balancing algorithms that can account for the differences in resource performance and optimize task placement accordingly. Open issues in this area include the development of more advanced resource characterization techniques that can assess the specific capabilities of cloud resources and improve task-resource matching. Additionally, AI-driven resource management systems that can dynamically assess and predict resource availability and task requirements in real-time are needed to tackle this challenge.

Developing AI-based scheduling systems in heterogeneous cloud infrastructures necessitates balancing scalability, adaptability, and energy efficiency while addressing challenges such as reliability, cost efficiency, and integration with existing services. Multi-objective optimization techniques, including Pareto-based approaches and machine learning models like reinforcement learning, offer dynamic solutions by prioritizing different objectives based on workload conditions and system requirements. Predictive analytics and fault-tolerant mechanisms enhance system reliability by anticipating failures and enabling proactive resource reallocation. Additionally, multi-agent systems and federated learning improve redundancy and interoperability, ensuring robust and reliable cloud operations. To achieve cost efficiency and seamless integration, AI-driven frameworks leverage adaptive resource management strategies, such as dynamic scaling and predictive billing models, to minimize costs and optimize resource utilization. APIs, containerization tools like Docker and Kubernetes, and cross-cloud compatibility technologies facilitate integration with existing cloud services. These approaches collectively enable AI-based scheduling systems to address the complexities of modern cloud environments while optimizing performance, energy efficiency, and operational costs. Future research should focus on hybrid frameworks that combine these methodologies to create intelligent, scalable, and resilient cloud scheduling solutions.

Several advancements in AI-driven job scheduling techniques are necessary to address critical issues such as cost efficiency, real-time adaptability, and managing complex dependencies in heterogeneous cloud computing environments. Cost efficiency can be improved by integrating dynamic pricing models, where algorithms like Dynamic Time Warping (DTW) and RL optimize resource allocation based on real-time cost variations offered by cloud providers. Energy-aware scheduling techniques, such as Green Scheduler and Power-Aware Resource Allocation (PARA), leverage historical and real-time energy consumption

data to consolidate workloads and minimize operational expenses. Additionally, multi-objective optimization frameworks, such as the Non-dominated Sorting Genetic Algorithm II (NSGA-II), can balance cost efficiency with other critical metrics like makespan and throughput. Real-time adaptability is a vital feature in dynamic cloud environments, and advanced RL methods, such as DQN and Federated RL, enable schedulers to adapt dynamically to fluctuating workloads and resource availability. Streaming data integration through techniques like Online Gradient Descent and Adaptive Boosting allows models to update in real-time without retraining from scratch. Edge-AI solutions, such as Federated Edge Learning (FEL), decentralize scheduling decisions, reduce latency, and enhance responsiveness by processing tasks closer to the data source, which is especially crucial for latency-sensitive applications.

Managing complex dependencies between jobs and system faults requires sophisticated modelling and fault-tolerant mechanisms. Graph-based models, including GNNs and DAG-based scheduling algorithms, effectively capture task precedence and resource interdependencies. Hybrid AI approaches that combine GA, and PSO has shown promise in navigating complex dependency constraints while optimizing scheduling objectives. Fault-aware scheduling techniques like adaptive fault-tolerant job scheduling and Anomaly Detection using LSTM can predict and mitigate system faults, ensuring the continuity of job execution and minimizing downtime. Innovations in job scheduling also extend to integrating advanced technologies like blockchain and Explainable AI (XAI). Blockchain-based scheduling systems, such as secure job allocation using blockchain (SJAB), provide transparency and security, ensuring tamper-proof scheduling records and improved trust. Explainable AI techniques, including SHapley Additive exPlanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME), enhance the interpretability of AI-driven decisions, allowing stakeholders to understand and validate the reasoning behind scheduling outcomes. AI-driven load-balancing algorithms, like Load Balancing Neural Networks (LBNNs) and Deep Reinforcement Learning-based Load Balancers (DRL-LB), further contribute to efficient resource utilization and prevent bottlenecks in heterogeneous cloud systems.

Scalability and collaborative models are crucial for handling the growing complexity of cloud computing. Federated Learning frameworks like federated averaging allow AI models to learn collaboratively across multiple cloud environments without centralized data collection, preserving privacy and enabling scalability. Multi-agent systems, like the Cooperative Scheduling Multi-Agent Reinforcement Learning (CS-MARL), empower independent agents to manage sub-tasks collaboratively, ensuring efficient scheduling in multi-cloud and hybrid-cloud scenarios. Finally, simulation frameworks and validation tools enhance the reliability of scheduling strategies. When integrated with AI models, tools like CloudSim, WorkflowSim, and iFogSim provide a robust environment for evaluating scheduling algorithms under diverse scenarios. Digital twins, such as Azure Digital Twins, serve as virtual replicas of cloud systems, enabling fine-tuning of AI-driven scheduling methods in simulated environments. These advancements collectively enable AI-driven job scheduling techniques to address the challenges of cost efficiency, real-time adaptability, and managing complex dependencies, ensuring robust, scalable, and efficient cloud computing systems.

9.3 Real-time job scheduling

Real-time job scheduling presents a significant challenge in cloud computing, particularly for applications requiring immediate processing or strict deadlines. Such applications include financial trading systems, healthcare monitoring, and autonomous vehicle control. In these cases, the scheduling algorithm must ensure that jobs are executed within a specific time window, often with little delay tolerance. The difficulty with real-time scheduling lies in the unpredictability of cloud environments. Network delays, resource contention, and fluctuations in workload can all impact the ability to schedule jobs within the required timeframes. Traditional scheduling algorithms, which focus on optimizing for metrics such as makespan or resource utilization, are often ill-suited for real-time applications that demand low latency and deterministic execution times.

Another challenge is prioritizing real-time tasks without negatively affecting the performance of non-real-time tasks. Real-time job scheduling algorithms must be designed to balance the need for prompt task execution with the requirement to maintain overall system performance. This can involve preemptive scheduling, pausing lower-priority tasks to allow real-time jobs to execute, or dynamic prioritization based on task deadlines. Research into real-time job scheduling for cloud environments is still evolving, with open issues including developing algorithms that can handle unpredictable workloads, provide guaranteed execution times, and prioritize real-time tasks effectively. Additionally, integrating real-time scheduling with other objectives, such as energy efficiency and scalability, remains a challenging research problem.

9.4 Energy efficiency

Energy consumption is a growing concern in cloud computing, particularly as data centres become more resource-intensive. Job scheduling plays a critical role in energy efficiency, as inefficient scheduling can result in higher power usage due to idle resources, frequent task migrations, and poorly optimized task placement. One of the main challenges in energy-efficient scheduling is balancing performance with power consumption. For example, shutting down idle resources or dynamically scaling down active resources can save energy, but it may also impact task execution times and system throughput. Algorithms must be able to make intelligent decisions about when and how to adjust resource power states without significantly affecting overall performance.

In addition to resource management, energy-efficient scheduling algorithms must also consider cooling costs and the physical location of data centres. Tasks that require large amounts of computational power may cause certain resources to overheat, leading to increased cooling costs. Scheduling tasks in a way that minimizes heat generation and spreads the workload across resources with optimal thermal conditions is a critical factor in reducing energy usage. Open research questions in this area include the development of algorithms that can dynamically optimize energy efficiency while still meeting performance requirements and SLAs. AI and machine learning techniques offer potential solutions by enabling predictive energy management, but their implementation in real-world cloud systems is still in its early stages.

9.5 Security concerns

Security is an ever-present challenge in cloud computing, and job scheduling is no exception. In multi-tenant environments, tasks from different users or organizations may run on the same physical infrastructure, raising concerns about data privacy, resource isolation, and attack vulnerability. One of the leading security concerns in cloud-based job scheduling is ensuring that tasks are isolated from each other to prevent data leakage or unauthorized access. This becomes particularly important in environments where sensitive data is being processed, such as healthcare or financial applications. Scheduling algorithms must ensure that tasks from different tenants are kept separate and that no information is shared directly or through side channels.

Another security issue is the risk of Denial of Service (DoS) attacks, where malicious users flood the system with an excessive number of tasks, overwhelming the scheduling algorithm and degrading system performance. Scheduling algorithms must be robust enough to detect and mitigate such attacks, ensuring that legitimate tasks are not affected by the malicious behaviour of others. Job scheduling algorithms must also account for the potential vulnerabilities in the communication between cloud resources and users. Secure communication protocols and encryption techniques are necessary to protect data during transmission and ensure external attackers do not manipulate scheduling decisions.

AI-based job scheduling minimizes the attack surface by effectively managing and allocating resources. Idle or underutilized resources in cloud environments are often more susceptible to exploitation, such as unauthorized access or attacks targeting unmonitored systems. By dynamically scheduling tasks and reallocating resources based on real-time requirements, AI-driven schedulers ensure that resources are actively utilized, leaving fewer opportunities for malicious actors to exploit. Furthermore, these algorithms can prevent over-provisioning, often leading to resource contention and potential denial-of-service (DoS) vulnerabilities. In addition, AI-based job scheduling contributes to securing resource allocation by integrating security policies into the scheduling process. For example, sensitive tasks requiring higher data privacy or encryption levels can be allocated to secure nodes or private cloud environments, ensuring compliance with data protection standards and minimizing the risk of breaches. Simultaneously, less critical or non-sensitive tasks can be offloaded to public cloud nodes, balancing security and cost-efficiency. This intelligent allocation helps protect sensitive information while maintaining operational performance. Another crucial aspect is the role of job scheduling in detecting and responding to malicious activities. AI-driven schedulers monitor resource usage patterns and can identify anomalies indicative of potential security threats. For instance, unusual spikes in resource consumption may signal an ongoing insider threat or an external attack. By detecting such anomalies, schedulers can prioritize security-related jobs, such as real-time intrusion detection or log analysis, to mitigate potential risks. In hybrid cloud environments, AI-based job scheduling also supports layered security. Tasks that involve critical security functions, such as encryption, access log management, or vulnerability scans, can be prioritized and scheduled on private cloud nodes where security measures are more robust. Meanwhile, non-sensitive tasks can be distributed to public clouds, optimizing overall resource utilization without compromising security. By integrating security considerations into task scheduling, AI-based job scheduling indirectly enhances the resilience and trustworthiness of cloud systems. This approach ensures operational efficiency and aligns with maintaining secure and reliable

cloud infrastructures. Through its adaptive, real-time nature, AI-based job scheduling supports proactive and dynamic responses to evolving security challenges in the cloud.

As cloud computing infrastructures scale and diversify, integrating security mechanisms within job scheduling becomes critical. Security-aware scheduling involves making informed decisions about task placement, prioritization, and isolation while proactively mitigating threats such as resource abuse, side-channel attacks, and data leakage. With its capacity for learning and inference, AI is a potent tool to embed intelligence into the job scheduling layer, enabling dynamic anomaly detection, adaptive access control, and privacy-preserving decision-making. This section presents a technical deep dive into AI's role in enhancing the security posture of scheduling systems.

9.5.1 AI-driven anomaly detection in scheduling pipelines

AI-powered anomaly detection techniques focus on modelling the normal behaviour of scheduling operations across various metrics, including CPU load distribution, job execution latency, VM migration frequency, memory access patterns, and network I/O traffic. Anomalies may signify the presence of insider threats, denial-of-service attempts, or stealthy lateral movement attacks.

- Autoencoder-Based Detection: Deep autoencoders are trained on normal workload traces to learn compact latent representations of typical scheduling behaviours. Reconstruction error is computed for each new job or scheduling event during deployment. Significant deviations from the reconstruction baseline trigger an anomaly alert. These models are particularly effective in capturing non-linear relationships among high-dimensional metrics.
- Temporal Deep Learning Models: Recurrent Neural Networks (RNNs), especially LSTM and GRU architectures, capture temporal dependencies across scheduling events. They can detect time-series anomalies such as gradual increase in job starvation or covert timing channels. A model trained on inter-job arrival times and task durations can, for instance, detect timing jitter indicative of scheduling abuse or cryptomining attacks.
- Graph-Based Anomaly Detection: Cloud environments can be modeled as dynamic graphs where nodes represent virtual machines, jobs, or users, and edges signify resource dependencies. Graph Neural Networks (GNNs) can identify structural anomalies such as unauthorized job hops or anomalous communication paths among nodes. Graph embeddings can also reveal abnormal scheduling trajectories or misuse of shared resources.
- Ensemble Learning for Attack Classification: Combining multiple classifiers (e.g., Random Forest, XGBoost, LightGBM) within a stacked or voting ensemble can improve the robustness and accuracy of intrusion detection. Each base model learns different aspects of scheduling behaviours, and the ensemble aggregates predictions to flag malicious patterns in real-time.

9.5.2 Privacy-preserving scheduling via AI

In multi-tenant or federated clouds, job metadata, execution context, and performance logs often contain sensitive information. AI-based scheduling must, therefore, be equipped with mechanisms to preserve confidentiality while enabling data-driven decision-making.

- Federated Learning (FL): FL enables collaborative model training across cloud regions or organizational boundaries without centralizing raw job data. Each node trains a local model on its private job history, and shares encrypted gradients with a global model aggregator. Techniques like Secure Aggregation and homomorphic encryption further enhance the confidentiality of model updates. FL-based schedulers can learn global resource allocation policies while adhering to strict data privacy constraints.
- Differential Privacy (DP): AI models embedded in scheduling pipelines can employ differential privacy to ensure that the inclusion or exclusion of any single job trace does not significantly affect the model's output. This is achieved by injecting calibrated Laplace or Gaussian noise into model gradients or outputs. DP guarantees provide mathematical bounds on privacy leakage, making them suitable for regulated environments such as healthcare or finance clouds.
- Split Learning and Secure Multi-Party Computation (SMPC): In split learning, the AI model is partitioned between the client and the scheduler. Only intermediate activations are exchanged, ensuring that raw job features are never revealed. SMPC allows multiple scheduling entities to collaboratively compute placement decisions without disclosing their private inputs. This is crucial for inter-cloud scheduling in federated scenarios.

9.5.3 Policy-aware AI for secure scheduling decisions

Beyond detection and privacy, AI models can enforce complex security policies during job scheduling:

- Reinforcement Learning for Policy Compliance: Deep RL agents can learn to map system states (e.g., current workload distribution, job trust scores, network load) to scheduling actions (e.g., defer, replicate, isolate) under predefined reward structures that encode security policies. For example, a policy may penalize the collocation of sensitive jobs with untrusted VMs, prompting the agent to isolate workloads dynamically.
- Trust-Aware Scheduling Models: AI models can maintain dynamic trust scores using behavioural profiling, past violations, and SLA adherence. Trust levels can then guide scheduling strategies, prioritizing compliant jobs and deprioritizing or sandboxing low-trust workloads. Bayesian networks or recurrent models can update trust levels over time.
- Adaptive Access Control and Role Learning: Machine learning can augment traditional RBAC/ABAC by learning implicit roles or contextual access patterns. For example, clustering techniques can identify job access behavior outliers within tenant groups. Combined with anomaly detection, this enables automatic access control policies and proactive job quarantine refinement.

Despite its efficacy, integrating AI into secure scheduling pipelines presents several technical challenges:

- Adversarial Robustness: Deep models are vulnerable to adversarial scheduling inputs, where carefully crafted job parameters evade detection. Adversarial training, input sanitization, and certified defences are necessary to harden scheduling models.
- Latency Constraints: High-frequency, low-latency scheduling decisions cannot accommodate complex model inference without optimization. Model pruning, quantization, and edge deployment strategies must be investigated to meet real-time constraints.
- Joint Optimization Objectives: Balancing security, performance, cost, and SLA compliance in a unified AI model remains an open research question. Multi-objective reinforcement learning and Pareto-efficient scheduling frameworks offer promising directions.
- Auditable and Explainable AI (XAI): Scheduling decisions that affect security must be interpretable for audit and compliance purposes. Incorporating SHAP values, LIME explanations, or attention-based interpretability into scheduling models is essential for adoption in sensitive domains.

In summary, the application of AI in security-aware scheduling introduces intelligent automation for threat mitigation, secure data handling, and dynamic policy enforcement. Integrating deep anomaly detection, privacy-preserving learning, and trust-aware placement engines marks a significant step toward building resilient, secure, and autonomous cloud computing environments.

Open issues in cloud scheduling security include the development of more secure multi-tenant scheduling mechanisms, robust detection methods for resource-based attacks, and the integration of advanced cryptographic techniques into scheduling algorithms to ensure data integrity and confidentiality. In summary, while job scheduling in cloud computing has made significant strides, several key challenges remain. Scalability, heterogeneous resource management, real-time job scheduling, energy efficiency, and security concerns present ongoing research challenges and open issues. Addressing these challenges will require the development of new algorithms that leverage advanced AI techniques, dynamic resource management, and robust security protocols to meet the evolving demands of cloud computing environments.

10 Emerging trends and applications

The field of job scheduling in cloud computing is continuously evolving, driven by advancements in technology and the growing complexity of cloud infrastructures. Several emerging trends are shaping the future of job scheduling, aiming to address current challenges and enhance the efficiency, security, and scalability of cloud environments. This section explores the most promising emerging trends and future directions in cloud-based job scheduling.

Figure 23 illustrates the future scope and applications of task scheduling, highlighting advancements in edge computing integration, machine learning for dynamic scheduling, blockchain-based job scheduling, multi-objective optimization, IoT, fog, and serverless computing, as well as AI in hybrid and multi-cloud environments. The figure emphasizes

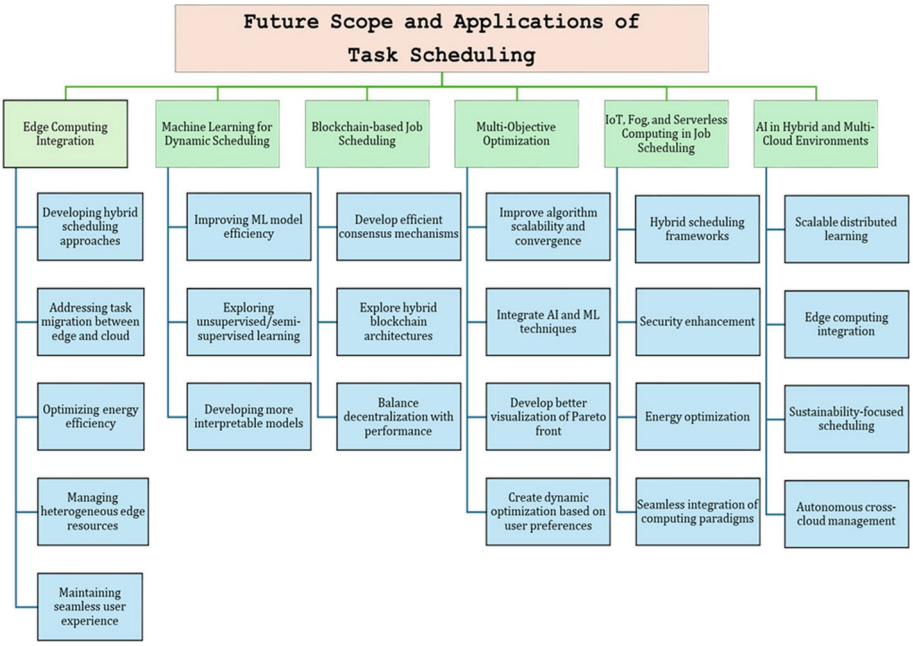

Fig. 23 Future scope and applications of task scheduling

key areas of innovation and research opportunities in improving scheduling efficiency, scalability, security, and sustainability.

10.1 Edge computing integration

Edge computing is an emerging paradigm that brings computational resources closer to the data source, reducing latency and improving real-time processing capabilities. The integration of edge computing into cloud environments has introduced new opportunities and challenges for job scheduling (Khan et al. 2019; Shaji George et al. 2023; Ahmad et al. 2023). Traditional cloud-based scheduling algorithms, which focus on centralized resource allocation, must be adapted to handle edge environments' decentralized and distributed nature.

One of the primary benefits of edge computing integration is the reduction in data transmission times (Shaji George et al. 2023). By processing data closer to the edge, scheduling algorithms can improve the performance of latency-sensitive applications, such as the IoT (Shukla et al. 2023), augmented reality (Qian and Coutinho 2024), and autonomous vehicles (Trabelsi et al. 2024). However, this also introduces the challenge of coordinating tasks between edge and cloud resources, requiring scheduling algorithms that dynamically allocate jobs based on resource availability and proximity to data sources.

The future of job scheduling lies in hybrid approaches that combine the strengths of cloud and edge environments. These algorithms must account for the heterogeneity of edge resources, the dynamic nature of edge nodes, and the varying requirements of different applications. Research into edge-aware scheduling is still in its early stages (Weng et al. 2024), with open questions regarding task migration between edge and cloud, energy effi-

ciency at the edge, and maintaining a seamless user experience across different infrastructure layers.

10.2 Machine learning for dynamic scheduling

Applying ML techniques to job scheduling in cloud environments is a rapidly growing area of research (Zhang et al. 2024). ML algorithms can make data-driven decisions, predict future workloads, and optimize scheduling policies in real-time. This dynamic scheduling approach can significantly enhance cloud systems' flexibility and efficiency, particularly in environments with highly variable workloads and resource demands (Umarani Srikanth and Geetha 2023; Zhang et al. 2024). One of the critical advantages of ML-based scheduling is the ability to predict future resource requirements based on historical data. By analyzing patterns in workload fluctuations, ML models can anticipate spikes in demand and proactively allocate resources to avoid performance degradation. Additionally, reinforcement learning approaches can continuously improve scheduling decisions over time by learning from the outcomes of previous scheduling actions.

Another promising direction is using deep learning models to handle complex scheduling problems, such as job prioritization, resource allocation, and load balancing. These models can learn to optimize multiple objectives simultaneously, making them well-suited for multi-objective optimization in cloud environments (discussed in Sect. 9.3). Despite the potential of ML-based scheduling, several challenges remain, including the computational overhead of training models, the need for large amounts of training data, and the difficulty of applying these models in real-time cloud environments. Future research in this area will likely focus on improving the efficiency of ML models for scheduling, exploring unsupervised and semi-supervised learning techniques, and developing more interpretable models that can provide insights into scheduling decisions.

10.2.1 Hybrid AI models

Hybrid AI models represent one of the most promising future directions in job scheduling research. These models combine the strengths of multiple artificial intelligence paradigms—such as supervised learning, reinforcement learning, deep learning, fuzzy logic, and evolutionary algorithms—to overcome the limitations of individual techniques and adapt more effectively to cloud environments' complex, dynamic nature. From a technical perspective, hybrid models can be designed to balance exploration and exploitation more effectively. For example, RL agents can be augmented with supervised learning models trained on historical job data to provide a warm start for policy learning, thereby accelerating convergence and improving early-stage performance. Similarly, integrating fuzzy logic with RL can enable better decision-making under uncertainty, particularly in environments with imprecise or noisy input data.

Incorporating metaheuristic algorithms (e.g., Genetic Algorithms, Particle Swarm Optimization, Ant Colony Optimization) into hybrid frameworks allows for global search and fine-tuning of scheduling parameters. These algorithms can optimize hyperparameters in deep neural networks or dynamically adjust reward functions in RL agents, leading to more robust and context-aware scheduling policies. Several studies have explored hybridization strategies with promising results. For instance, hybrid Genetic Algorithm and Neural

Network (GA-NN) models leverage the global search capabilities of GAs for optimal job sequence generation while using neural networks for workload prediction. Another example is the combination of Deep Q-Networks (DQN) with Ant Colony Optimization (ACO), where DQN learns task prioritization and ACO dynamically determines the most efficient resource paths. Such hybridizations improve scheduling accuracy, adaptability to unforeseen workloads, and dynamic resource availability.

A key area of exploration is the development of multi-agent hybrid systems. Agents employing different AI strategies can cooperate or compete to manage resources across distributed nodes in such architectures. For example, a deep learning agent may handle workload prediction, while an RL agent governs task allocation, and a rule-based expert system enforces policy constraints. This modular, agent-based design improves scalability, maintainability, and specialization. Moreover, hybrid AI models can facilitate context-aware and SLA-driven scheduling by fusing multiple data modalities—such as workload characteristics, energy profiles, user priorities, and security levels—into a unified decision-making framework. Multimodal deep learning architectures can process this heterogeneous input and feed insights into adaptive scheduling engines.

The benefits of hybridization include:

- Improved generalization: Hybrid systems can generalise better across diverse workloads by integrating models trained on different objectives or data representations.
- Enhanced fault tolerance: Different AI components can back each other up or recover gracefully in case of failure, improving system resilience.
- Higher optimization performance: Metaheuristics and learning-based agents can collaboratively reach global optima more efficiently than isolated models.
- Greater flexibility and modularity: Hybrid systems can be customized for specific environments by mixing and matching different AI techniques.

However, hybridization introduces its challenges. These include increased architectural complexity, the need for cross-model compatibility, and the difficulty of achieving real-time responsiveness. Future research should therefore focus on:

- Designing lightweight and modular hybrid models that can be deployed at the edge or in resource-constrained environments.
- Developing standardized frameworks and APIs to facilitate the orchestration and interaction of heterogeneous AI components.
- Exploring knowledge distillation and model compression techniques to reduce the overhead of deep or ensemble-based hybrid models.
- Integrating hybrid AI systems with explainable AI (XAI) techniques to improve transparency and facilitate debugging in production environments.
- Creating datasets and benchmarks to evaluate hybrid AI scheduling strategies under diverse and realistic cloud scenarios.

In conclusion, hybrid AI models offer a pathway to building intelligent, resilient, and adaptable job scheduling systems. Their ability to integrate multiple perspectives and learning mechanisms makes them well-suited to the heterogeneous and evolving demands of cloud

computing. As research matures, these models are expected to play a central role in the next generation of AI-driven scheduling frameworks.

10.3 Blockchain-based job scheduling

Blockchain technology has gained significant attention recently for its potential to enhance security, transparency, and decentralization in various fields, including cloud computing (Javadpour et al. 2024; Tarannum and Abidin 2023; Mohammad Alshinwan et al. 2023). In the context of job scheduling, blockchain offers a novel approach to managing task allocation and resource sharing in multi-tenant cloud environments (Javadpour et al. 2024). By leveraging decentralized ledger technology, blockchain-based scheduling algorithms can ensure transparency and trust in resource allocation decisions (Baranwal et al. 2023; Javadpour et al. 2024).

One critical advantage of blockchain-based job scheduling is the ability to create tamper-proof records of scheduling decisions. This enhances accountability in cloud environments, particularly in multi-tenant settings where users share resources with varying priorities. Smart contracts, a key feature of blockchain technology, can be used to automate the execution of scheduling policies, ensuring that tasks are allocated based on predefined rules and conditions. Blockchain-based scheduling also has the potential to improve security by preventing malicious users from manipulating scheduling decisions (Cao et al. 2023; Baranwal et al. 2023). Since blockchain operates on a decentralized network, it resists single points of failure, reducing the risk of denial-of-service attacks and other security threats.

From a technical standpoint, blockchain enables each node in the network to maintain an identical copy of the distributed ledger, which records every scheduling transaction in a cryptographically secure and verifiable format. When a scheduling request is initiated, it is broadcast to the network, where nodes validate the request against consensus rules before adding it to the blockchain. This mechanism ensures that only legitimate, consensus-approved scheduling actions are executed, eliminating the possibility of tampering or unauthorized changes. Smart contracts can be programmed to encode job scheduling policies such as priority queues, fair resource allocation, and access control mechanisms. These contracts execute autonomously and deterministically once their conditions are met, reducing latency and eliminating the need for manual intervention. For example, a smart contract could be designed to trigger a job migration if a node's resource utilization crosses a threshold or reassign a job if an SLA is violated. This ensures fairness and compliance in job allocation across decentralized infrastructure.

Furthermore, blockchain's suitability feature allows any stakeholder to trace the history of scheduling decisions and verify compliance with agreed policies. This is especially crucial in collaborative cloud environments or inter-cloud federations where multiple organizations share resources and require verifiable proof of fair treatment. Despite its potential, blockchain-based scheduling is still in its infancy and faces several technical challenges. The scalability of public blockchain networks remains a concern due to limited transaction throughput and high latency. The consensus protocols (e.g., Proof of Work or Proof of Stake) that govern these systems can be computationally intensive and may not meet the real-time responsiveness required for high-frequency job scheduling.

To address these challenges, future research may explore hybrid blockchain architectures that combine permissioned and public blockchains to balance decentralization with

performance. Lightweight consensus mechanisms like Practical Byzantine Fault Tolerance (PBFT) or Delegated Proof of Stake (DPoS) can reduce computational overhead while maintaining security guarantees. Additionally, off-chain scheduling computations with on-chain verification could be leveraged to decouple the scheduling logic from the blockchain's data layer, thus improving scalability. Integrating blockchain into existing cloud platforms also presents architectural hurdles, such as interoperability, compatibility with cloud orchestration tools (e.g., Kubernetes, OpenStack), and the need for secure APIs to interface with smart contracts. Robust middleware frameworks are required to abstract blockchain complexity from cloud users and developers, allowing seamless integration without compromising usability or efficiency.

In summary, blockchain-based job scheduling offers a paradigm shift in how trust, transparency, and automation can be embedded into cloud resource management. While its limitations pose practical deployment challenges, ongoing advancements in consensus algorithms, hybrid systems, and blockchain-cloud integration frameworks pave the way for widespread adoption.

10.4 Multi-objective optimization

Job scheduling in cloud environments often involves balancing multiple, frequently conflicting objectives, such as minimizing makespan, maximizing resource utilization, reducing energy consumption, and meeting SLAs. Multi-objective optimization (MOO) is an emerging trend in cloud scheduling that seeks to address this challenge by developing algorithms capable of optimizing multiple objectives simultaneously (Kruekaew and Kimpan 2022; Malti et al. 2024). Traditional scheduling algorithms typically focus on a single objective, such as minimizing makespan or maximizing throughput. However, in real-world cloud environments, a single-objective approach is often insufficient to meet the diverse requirements of different applications and users. MOO algorithms address this by providing a set of trade-off solutions, known as the Pareto front, where no single solution dominates the others across all objectives (Malti et al. 2024; Mangalampalli et al. 2023).

One of the main challenges in MOO is finding an optimal balance between competing objectives. For example, minimizing makespan may lead to higher energy consumption, while optimizing energy efficiency may result in longer task completion times. MOO algorithms must be designed to handle these trade-offs and provide solutions acceptable to cloud providers and users (Malti et al. 2024; Mangalampalli et al. 2023). Genetic algorithms (Mangalampalli et al. 2023), particle swarm optimization (Singh and Chaturvedi 2024), and multi-objective evolutionary algorithms (Wang et al. 2023) are some techniques applied to MOO in job scheduling. These algorithms explore an ample search space to find a set of non-dominated solutions representing different trade-offs between objectives. Future research in MOO for cloud scheduling will likely focus on improving the scalability and convergence speed of optimization algorithms, integrating AI and ML techniques to guide the search process, and developing more efficient ways to visualize and interpret the Pareto front. Additionally, there is a need for more practical applications of MOO in real-world cloud systems, where the trade-offs between objectives can be dynamically adjusted based on user preferences or system conditions.

10.5 IoT, fog, and serverless computing in job scheduling

The rapid evolution of cloud computing has brought forth several emerging paradigms, including the IoT, Fog Computing, and Serverless Computing. These advancements are reshaping the landscape of job scheduling by introducing unique challenges and opportunities that demand innovative solutions, particularly leveraging AI-based algorithms. Integrating IoT devices into cloud ecosystems has significantly changed job scheduling mechanisms (Khezri et al. 2024; Alsamarai et al. 2023). IoT devices generate large volumes of data in real-time, requiring immediate processing and decision-making capabilities. Traditional centralized cloud models face challenges in handling such demands due to latency and bandwidth constraints. AI-driven job scheduling can bridge this gap by dynamically assigning tasks to edge or cloud resources based on workload characteristics, ensuring timely execution and optimal resource utilization. Furthermore, IoT-driven scheduling requires algorithms to prioritize energy-efficient operations, given the battery-powered nature of many IoT devices.

Fog Computing extends cloud functionalities closer to the data source, reducing latency and bandwidth usage. This paradigm decentralizes job scheduling by distributing computation across multiple network layers, including edge devices and intermediate fog nodes (Iftikhar et al. 2023a; Vispute and Vashisht 2023). AI algorithms tailored for fog environments must address unique challenges such as limited resource availability, dynamic topology, and the heterogeneity of devices. Job scheduling in fog computing emphasizes real-time adaptability, as tasks are often time-sensitive and localized. By leveraging AI techniques such as reinforcement learning and predictive analytics, job schedulers can optimize task allocation across fog nodes to ensure minimal delay and efficient use of resources.

Serverless Computing abstracts infrastructure management, allowing developers to focus solely on application logic. This paradigm operates on an event-driven model, executing tasks as functions responding to specific triggers (Pan et al. 2023; Ghorbian et al. 2024). While this approach enhances scalability and cost efficiency, it introduces new complexities in job scheduling. AI-based schedulers can optimize the execution of serverless functions by analyzing workload patterns, predicting trigger frequencies, and managing resource allocation dynamically. Serverless architectures also require schedulers to ensure high concurrency and fault tolerance, as multiple functions may execute simultaneously under unpredictable workloads. The convergence of IoT, Fog Computing, and Serverless Computing with AI-driven scheduling presents immense potential for enhancing efficiency and scalability. However, it also brings challenges such as security concerns, energy optimization, and consistency across distributed environments. Future research should focus on hybrid scheduling frameworks seamlessly integrating these paradigms, leveraging AI to balance trade-offs between latency, cost, and resource utilization. By embracing these emerging trends, job scheduling mechanisms can evolve to meet the demands of increasingly complex and decentralized computing environments.

In summary, job scheduling in cloud computing is undergoing significant transformations, driven by emerging trends such as edge computing integration, machine learning for dynamic scheduling, blockchain-based scheduling, multi-objective optimization, and IoT, fog, and serverless computing in job scheduling. These trends can address many challenges in cloud environments, such as scalability, security, and resource management. However, they also introduce new research questions and technical challenges that must be addressed

to realize their full potential. Future research in these areas will be crucial for advancing the state of cloud computing and developing more efficient, secure, and adaptable scheduling algorithms.

10.6 AI in hybrid and multi-cloud environments

Hybrid and multi-cloud environments, where workloads are distributed across multiple cloud providers or a combination of public and private clouds, introduce unique challenges and opportunities for AI-based scheduling. This subsection explores these complexities and highlights AI's transformative potential in addressing them.

Challenges in hybrid and multi-cloud scheduling

Hybrid and multi-cloud environments exhibit several inherent complexities that make efficient scheduling challenging:

- Resource heterogeneity: Cloud providers offer varied hardware configurations, virtualization technologies, and performance benchmarks. Scheduling across such diverse resources requires sophisticated algorithms to match workload requirements with available resources optimally (Krishnasamy et al. 2023).
- Latency and interconnectivity: Data transfer between clouds incur latency and costs, which must be minimized to maintain system performance and operational budgets (Merseedi and Zeebaree 2024). This is particularly challenging for latency-sensitive applications.
- Dynamic workload distribution: The fluctuating nature of workloads necessitates real-time decisions to allocate resources effectively, balancing load across multiple clouds while minimizing underutilization (Rajeshwari et al. 2022).
- Security and compliance: Different clouds may have distinct security protocols and compliance requirements. Scheduling must ensure that workloads adhere to these constraints without sacrificing efficiency (Julakanti et al. 2022).
- Cost optimization: Cloud providers have varying pricing models. Efficient scheduling must factor in cost variations, such as on-demand, reserved, or spot instances, to achieve economic efficiency (George 2022).

The potential of AI in hybrid and multi-cloud scheduling

AI offers transformative solutions to these challenges through advanced techniques that enable intelligent and adaptive scheduling:

- Reinforcement learning for dynamic scheduling: AI-based RL algorithms learn optimal policies by interacting with the hybrid or multi-cloud environment (Mangalampalli et al. 2024). RL agents can dynamically allocate resources and prioritize tasks, responding in real-time to workload fluctuations and system state changes.
- Prediction models for proactive scaling: Machine learning models, such as LSTM networks, can predict workload demands and resource utilization patterns (Sharma et al. 2024). These predictions enable proactive resource scaling to prevent bottlenecks and ensure performance consistency.
- Collaborative multi-Agent systems: Multi-agent AI systems facilitate cloud coordination, optimizing resource allocation and workload balancing (Chen et al. 2022a). Each

agent focuses on a specific cloud or workload, and collaborative reward mechanisms ensure global optimization.

- Hybrid optimization techniques: AI-powered hybrid optimization approaches combine heuristic methods (e.g., genetic algorithms) with machine learning to address multi-objective challenges (Mohammadzadeh and Masdari 2023). These include balancing cost, performance, energy efficiency, and data transfer latency in multi-cloud settings.
- Energy and carbon efficiency: AI models can schedule workloads to utilize renewable energy sources preferentially or during low-carbon-intensity periods (Merseedi and Zeebaree 2024). This supports sustainability goals while optimizing operational costs.

Cross-cloud scheduling and coordination

AI plays a pivotal role in addressing the complexities of cross-cloud scheduling:

- Latency-aware scheduling: AI models can incorporate network latency metrics into their optimization processes to minimize delays in inter-cloud communication (Tamoor-ul Hassan et al. 2023).
- Data placement and replication: Intelligent AI algorithms can decide where to place or replicate data to reduce cloud access latency and storage costs (Aldailamy et al. 2024).
- Fault tolerance and redundancy: AI-enabled scheduling ensures critical workloads are replicated or reallocated in real-time to maintain fault tolerance and high availability (Suresh et al. 2024).
- *Cost-aware scheduling* AI systems dynamically evaluate cloud pricing models to minimize operational costs by utilizing spot instances or shifting workloads to lower-cost regions (Merseedi and Zeebaree 2024).

Implications and future directions

The integration of AI in hybrid and multi-cloud environments has the potential to redefine resource management and workload scheduling:

- Scalability and adaptability: AI techniques enable scaling operations across diverse and geographically distributed clouds, adapting to changing workloads and system conditions. Scaling AI in large, distributed cloud environments involves leveraging advanced techniques to manage resources effectively and maintain performance (Merseedi and Zeebaree 2024). Distributed learning approaches like federated learning and decentralized reinforcement learning enable AI models to train collaboratively across multiple nodes without centralizing data, reducing bandwidth requirements and preserving privacy (Merseedi and Zeebaree 2024). Hierarchical scheduling models enhance scalability by deploying local AI agents for regional optimization while a global controller coordinates across clouds to balance workloads (Tang 2021). Containerization and orchestration platforms like Kubernetes play a critical role by enabling elastic scaling of AI workloads. Microservices architectures can complement this by breaking down AI functionalities into independently scalable components (Waseem et al. 2024). Additionally, parallelism techniques, such as model parallelism (splitting large models across nodes) and data parallelism (distributing datasets), accelerate training and inference processes for resource-intensive tasks (Mithila et al. 2023). Integrating edge computing allows lightweight AI models to process data closer to the source, reducing latency and

central cloud dependency. Powered by AI models, predictive scaling forecasts resource demands and dynamically allocates resources to ensure readiness during peak usage. Incorporating carbon-intensity data into scaling decisions supports sustainability goals, optimizing energy usage and performance (Miao et al. 2024).

- Fault tolerance: Fault tolerance is another crucial aspect, with AI monitoring resource health to predict failures and reroute workloads as needed (Suresh et al. 2024). Real-world systems like Google's Borg and Microsoft Azure's intelligent scaling are successful examples of scalable AI in action, demonstrating the potential of adaptive resource management. Multi-agent AI systems, where agents manage infrastructure segments while collaborating globally, enhance scalability in complex, distributed settings (Merseedi and Zeebaree 2024).
- Sustainability goals: By prioritizing low-carbon-intensity periods and renewable energy sources, AI contributes to green computing objectives (Miao et al. 2024).
- Automation and efficiency: AI reduces the need for manual intervention, enabling fully autonomous cross-cloud scheduling with improved efficiency and reliability (Rajeshwari et al. 2022).

This discussion underscores AI's critical role in addressing the challenges of hybrid and multi-cloud environments, highlighting its potential to optimize performance, reduce costs, and meet sustainability goals. Further research and development will likely lead to more robust and scalable cloud computing solutions.

11 Conclusions

AI-driven job scheduling has demonstrated its potential to improve the efficiency and performance of cloud computing environments significantly. Traditional scheduling methods often rely on static rules and predefined priorities and are no longer sufficient in increasingly complex and dynamic cloud systems. AI-based techniques, including machine learning, optimization algorithms, heuristic methods, and deep learning, provide a more flexible, adaptive, and scalable solution to the challenges of job scheduling. Integrating these intelligent approaches has improved resource allocation, energy consumption, and system reliability, making them an essential component of modern cloud computing infrastructures. One of the significant advantages of AI-driven job scheduling lies in its ability to handle the heterogeneous nature of cloud resources. As cloud environments become more diverse, with various computing nodes, memory configurations, and network conditions, AI-based algorithms can dynamically adapt to these changes. By learning from historical data and making real-time decisions, AI models ensure that tasks are assigned to the most suitable resources, leading to better system performance and reduced processing delays. Furthermore, AI-driven methods enhance energy efficiency by predicting workloads and adjusting resource usage accordingly, critical for reducing operational costs and minimizing environmental impact.

Despite these advancements, several challenges remain in fully optimizing AI-driven job scheduling in cloud computing. Scalability remains a significant concern, particularly as the number of tasks and users in cloud systems grows. The complexity of scheduling increases as the system expands, requiring more sophisticated and computationally effi-

cient algorithms. Additionally, integrating AI with traditional scheduling methods remains an area that needs further exploration. While hybrid AI models have shown promise in combining the strengths of various approaches, achieving the right balance between AI-driven optimization and traditional scheduling efficiency is still an open question. Future research should focus on enhancing the scalability of AI-driven job scheduling algorithms, especially in multi-cloud and hybrid-cloud environments. As cloud computing evolves to include edge computing and IoT devices, scheduling solutions must be able to manage distributed resources across different network layers. Furthermore, security will be increasingly crucial in job scheduling, particularly in multi-tenant cloud systems where data privacy and integrity are paramount. AI-based scheduling algorithms could incorporate anomaly detection and predictive analytics to identify potential security threats, improving the overall trustworthiness of cloud infrastructures. Finally, integrating emerging technologies like blockchain with AI-driven scheduling offers exciting new possibilities. Blockchain's decentralized nature can enhance the security and transparency of scheduling processes, ensuring that tasks are executed reliably and without interference. By combining AI's adaptability with blockchain's security features, future scheduling algorithms could offer more robust, efficient, and secure solutions for cloud computing environments. These advancements will pave the way for more intelligent, sustainable, and secure cloud systems capable of meeting the growing demands of modern applications and industries.

Author contributions The authors of this paper have contributed equally to its research, writing, and revision processes and are therefore to be considered as co-equal contributors.

Funding The authors declare that no funding was received for conducting this study and preparing this manuscript.

Data availability Not applicable.

Materials availability Not applicable.

Code availability Not applicable.

Declarations

Conflict of interest All authors declare no conflict of interest related to this study.

Ethical approval All authors with this confirm that they have reviewed and accepted the ethics approval associated with this study.

Consent to participate All authors consent to participate in this research and confirm that they have contributed significantly to the work reported in this manuscript.

Consent for publication All authors consent to the publication of the paper in the Multimedia Tools and Applications journal.

References

Abd 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

Abdullah M, Othman M (2014) Simulated annealing approach to cost-based multi-quality of service job scheduling in cloud computing environment. Am J Appl Sci 11(6):872–877

Abdullah SH, Ayad A-H, Mohammed NM, Saad RMA (2023) Adaptive fault-tolerance during job scheduling in cloud services based on swarm intelligence and apache spark. Int J Intell Syst Appl Eng 11(2):74–81

Abdulredha MN, Bara'a AA, Jabir AJ (2020) Heuristic and meta-heuristic optimization models for task scheduling in cloud-fog systems: a review. Iraqi J Electr Electron Eng 16(2):103–112

Abouelyazid M (2024) Deep-hill: an innovative cloud resource optimization algorithm by predicting SaaS instance configuration using deep learning. IEEE Access. https://doi.org/10.1109/ACCESS.2024.3423339

Abualigah L, Hussein AMA, Almomani MH, Zitar RA, Daoud MS, Migdady H, Alzahrani AI, Alwadain A (2024) GIJA: enhanced geyser-inspired Jaya algorithm for task scheduling optimization in cloud computing. Trans Emerg Telecommun Technol 35(7):e5019

Aburukba RO, AliKarrar M, Landolsi T, El-Fakih K (2020) Scheduling internet of things requests to minimize latency in hybrid fog-cloud computing. Future Gener Comput Syst 111:539–551

Ahmad S, Shakeel I, Mehfuz S, Ahmad J (2023) Deep learning models for cloud, edge, fog, and IoT computing paradigms: survey, recent advances, and future directions. Comput Sci Rev 49:100568

Alam KMS, Santhosh R (2022) Task scheduling in cloud computing using hybrid optimization algorithm. Soft Comput 26(23):13069–13079

Aldahwan NS (2022) Ramzan MS (2022) Quadruple theories based determinants and their causal relationships affecting the adoption of community cloud in Saudi HEI. Biomed Res Int 1:2382535

Aldailamy AY, Muhammed A, Latip R, Hamid NAWA, Ismail W (2024) Online dynamic replication and placement algorithms for cost optimization of online social networks in two-tier multi-cloud. J Netw Comput Appl 224:103827

Al-E'mari S, Sanjalawe Y, Al-Daraiseh A, Taha MB, Aladaileh M (2024) Cloud datacenter selection using service broker policies: a survey. Comput Model Eng Sci 139(1):1–41

Ali NAM, Ali FAM (2023) Optimizing cloud-fog-edge job scheduling using catastrophic genetic algorithm and block chain-based trust: a collaborative approach. J Appl Eng Technol Sci 5(1):569–580

Alibaba Cloud (2024) Excellence in deliverance with elastic computing: Alibaba Cloud elastic compute service (part 2). https://www.alibabacloud.com/blog/excellence-in-deliverance-with-elastic-computing-alibaba-cloud-elastic-compute-service-part-2_597311. Accessed 24 Nov 2024

Al-Khanak EN, Lee SP, Khan SUR, Behboodian N, Khalaf OI, Verbraeck A, van Lint H (2021) A heuristics-based cost model for scientific workflow scheduling in cloud. Comput Mater Contin 67(3):3265–3282

Alla HB, Alla SB, Touhafi A, Ezzati A (2018) A novel task scheduling approach based on dynamic queues and hybrid meta-heuristic algorithms for cloud computing environment. Clust Comput 21(4):1797–1820

Alouffi B, Hasnain M, Alharbi A, Alosaimi W, Alyami H, Ayaz M (2021) A systematic literature review on cloud computing security: threats and mitigation strategies. IEEE Access 9:57792–57807

Alsaidy SA, Abbood AD, Sahib MA (2022) Heuristic initialization of PSO task scheduling algorithm in cloud computing. J King Saud Univ Comput Inf Sci 34(6):2370–2382

Alsaih MA, Latip R, Abdullah A, Subramaniam SK, Ali Alezabi K (2020) Dynamic job scheduling strategy using jobs characteristics in cloud computing. Symmetry 12(10):1638

Alsamarai NA, Uçan ON, Khalaf OF (2023) Bandwidth-deadline IoT task scheduling in fog-cloud computing environment based on the task bandwidth. Wirel Pers Commun. https://doi.org/10.1007/s11277-023-10567-1

Althobaiti T, Sanjalawe Y, Ramzan N (2023) Securing cloud computing from flash crowd attack using ensemble intrusion detection system. Comput Syst Sci Eng 47(1):453–469

Ananth A, Chandrasekaran K (2016) Cooperative game theoretic approach for job scheduling in cloud computing. Institute of Electrical and Electronics Engineers Inc., pp 147– 156

Aron R, Abraham A (2022) Resource scheduling methods for cloud computing environment: the role of meta-heuristics and artificial intelligence. Eng Appl Artif Intell 116:105345

Attiya I, Elaziz MA, Xiong S (2020) Job scheduling in cloud computing using a modified Harris hawks optimization and simulated annealing algorithm. Comput Intell Neurosci. https://doi.org/10.1155/2020/3504642
Balla HAM, Sheng CG, Weipeng J (2018) Reliability enhancement in cloud computing via optimized job scheduling implementing reinforcement learning algorithm and queuing theory. Institute of Electrical and Electronics Engineers Inc., pp 127–130
bAmeer HA, Mohammedali MA, Abdul-Rahaim LA, Al-Kharsan IH (2022) Design of surgical arm robot based on cloud computing. In: 2022 5th International Conference on Engineering Technology and its Applications (IICETA). IEEE, pp 289–293
Baomin X, Zhao C, Enzhao H, Bin H (2011) Job scheduling algorithm based on Berger model in cloud environment. Adv Eng Softw 42(7):419–425
Baranwal G, Kumar D, Vidyarthi DP (2023) Blockchain based resource allocation in cloud and distributed edge computing: a survey. Comput Commun. https://doi.org/10.1016/j.comcom.2023.07.023
Barnawi A, Sakr S, Xiao W, Al-Barakati A (2020) The views, measurements and challenges of elasticity in the cloud: a review. Comput Commun 154:111–117
Basu S, Karuppiah M, Selvakumar K, Li K-C, Islam SKH, Hassan MM, Bhuiyan MZA (2018) An intelligent/cognitive model of task scheduling for IoT applications in cloud computing environment. Future Gener Comput Syst 88:254–261
Behera I, Sobhanayak S (2024) Task scheduling optimization in heterogeneous cloud computing environments: A hybrid GA-GWO approach. J Parallel Distrib Comput 183:104766
Belgacem A, Beghdad-Bey K (2022) Multi-objective workflow scheduling in cloud computing: trade-off between makespan and cost. Clust Comput 25(1):579–595
Birje MN, Challagidad PS, Goudar RH, Tapale MT (2017) Cloud computing review: concepts, technology, challenges and security. Int J Cloud Comput 6(1):32–57
Blog Jobins (2024) Designing for scale: a deep dive into the Netflix system. https://blog.jobins.jp/designing-for-scale-a-deep-dive-into-the-netflix-system. Accessed 24 Nov 2024
Byrne J, Svorobej S, Giannoutakis KM, Tzovaras D, Byrne PJ, Östberg PO, Lynn T (2017) A review of cloud computing simulation platforms and related environments. In: International conference on cloud computing and services science, vol 2. SciTePress, pp 679–691
Cao F, Zhu MM (2013) Energy efficient workflow job scheduling for green cloud. In: 2013 IEEE international symposium on parallel & distributed processing, workshops and PhD forum. IEEE, pp 2218–2221
Cao K, Liu Y, Meng G, Sun Q (2020) An overview on edge computing research. IEEE Access 8:85714–85728
Cao S, Zhan Z, Dai C, Chen S, Zhang W, Han Z (2023) Delay-aware and energy-efficient IoT task scheduling algorithm with double blockchain enabled in cloud-fog collaborative networks. IEEE Internet Things J. https://doi.org/10.1109/JIOT.2023.3296478
Cao J, Yu Z, Yang J (2024) A migration strategy based on cluster collaboration predictions for mobile edge computing-enabled smart rail system. J Supercomput. https://doi.org/10.1007/s11227-024-06058-0
Casanova H (2001) Simgrid: a toolkit for the simulation of application scheduling. In: Proceedings First IEEE/ACM international symposium on cluster computing and the grid. IEEE, pp. 430–437
Chawla S, Kaur A (2024) Thorough understanding of existing fault-tolerant techniques for task scheduling in cloud computing. In: International conference on innovative computing and communication. Springer Nature Singapore, pp 511–525
Chen W, Deelman E (2012) WorkflowSim: a toolkit for simulating scientific workflows in distributed environments. In: 2012 IEEE 8th international conference on e-science. IEEE, pp 1–8
Chen J, Chen P, Niu X, Zongling W, Xiong L, Shi C (2022a) Task offloading in hybrid-decision-based multi-cloud computing network: a cooperative multi-agent deep reinforcement learning. J Cloud Comput 11(1):90
Chen R, Chen X, Yang C (2022b) Using a task dependency job-scheduling method to make energy savings in a cloud computing environment. J Supercomput 78(3):4550–4573
Cheng L, Wang Y, Liu Q, Epema DH, Liu C, Mao Y, Murphy J (2021) Network-aware locality scheduling for distributed data operators in data centers. IEEE Trans Parallel Distrib Syst 32(6):1494–1510
Cheng F, Huang Y, Tanpure B, Sawalani P, Cheng L, Liu C (2022a) Cost-aware job scheduling for cloud instances using deep reinforcement learning. Clust Comput. https://doi.org/10.1007/s10586-021-03436-8
Cheng L, Kalapgar A, Jain A, Wang Y, Qin Y, Li Y, Liu C (2022b) Cost-aware real-time job scheduling for hybrid cloud using deep reinforcement learning. Neural Comput Appl 34(21):18579–18593
Chitra K, Getzi M(2023) GCRS—green computing resource scheduler: an optimized energy efficient cloud data centers to scale down carbon emission. In: 2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS). IEEE, pp 1091–1098
Chraibi A, Ben Alla S (2021) Ezzati A (2021) Makespan optimisation in cloudlet scheduling with improved DQN algorithm in cloud computing. Sci Program 1:7216795

Chung A, Krishnan S, Karanasos K, Curino C, Ganger GR (2020) Unearthing inter-job dependencies for better cluster scheduling. In: 14th USENIX symposium on Operating Systems Design and Implementation (OSDI 20). USENIX Association, pp 1205–1223
ConceptDraw Software on Instagram (2023) Cloud computing diagrams - cloud applications. Accessed 23 Nov 2024. https://in.pinterest.com/pin/conceptdraw-software-on-instagram-cloud-computing-diagrams-cloud-applications-cloudcomputing-diagr--551550285596665597/
Cui Z, Zhao T, Wu L, Qin AK, Li J (2023) Multi-objective cloud task scheduling optimization based on evolutionary multi-factor algorithm. IEEE Trans Cloud Comput. https://doi.org/10.1109/TCC.2023.3234567
Cui Y, Zhang D, Zhang J, Zhang T, Cao L, Chen L (2024) Multi-user reinforcement learning based task migration in mobile edge computing. Front Comput Sci 18(4):184504
Ding D, Fan X, Zhao Y, Kang K, Yin Q, Zeng J (2020) Q-learning based dynamic task scheduling for energy-efficient cloud computing. Future Gener Comput Syst 108:361–371
Dong H, Wang B, Qiao B, Xing W, Luo C, Qin S, Moscibroda T (2021) Predictive job scheduling under uncertain constraints in cloud computing. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI). pp 3627–3634
Ebadifard F, Babamir SM (2021) Autonomic task scheduling algorithm for dynamic workloads through a load balancing technique for the cloud-computing environment. Clust Comput 24:1075–1101
Frank Vijay J (2019) Cloud data analysis using a genetic algorithm-based job scheduling process. Expert Syst 36(5):e12436
Garg SK, Buyya R (2011) NetworkCloudSim: modelling parallel applications in cloud simulations. In: 2011 fourth IEEE international conference on utility and cloud computing. IEEE, pp 105–113
Garg M, Dhiman G (2020) Job scheduling in cloud using seagull optimization algorithm. IGI Global
Gasior J, Seredynski F (2015) Decentralized job scheduling in the cloud based on a spatially generalized prisoner's dilemma game. Int J Appl Math Comput Sci 25(4):737
Geng S, Wu D, Wang P, Cai X (2020) Many-objective cloud task scheduling. IEEE Access 8:79079–79088
George J (2022) Optimizing hybrid and multi-cloud architectures for real-time data streaming and analytics: strategies for scalability and integration. World J Adv Eng Technol Sci 7(1):10–30574
Ghanbari S (2019) Priority-aware job scheduling algorithm in cloud computing: a multi-criteria approach. Azerb J High Perform Comput 2(1):29–38
Ghani A, Badshah A, Jan S, Alshdadi AA, Daud A (2020) Issues and challenges in cloud storage architecture: a survey. arXiv Preprint http://arxiv.org/abs/2004.06809
Ghorbian M, Ghobaei-Arani M, Esmaeili L (2024) A survey on the scheduling mechanisms in serverless computing: a taxonomy, challenges, and trends. Clust Comput. https://doi.org/10.1007/s10586-023-04264-8
Goga K, Xhafa F, Terzo O (2019) An evaluation of neural networks performance for job scheduling in a public cloud environment. Adv Intell Syst Comput 772:760–769
Gouasmi T, Louati W, Kacem AH (2017) Cost-efficient distributed MapReduce job scheduling across cloud federation. Institute of Electrical and Electronics Engineers Inc., pp 289–296
Goyal T, Singh A, Agrawal A (2012) CloudSim: simulator for cloud computing infrastructure and modeling. Procedia Eng 38:3566–3572
Gundu SR, Panem CA, Thimmapuram A (2020) Hybrid it and multi cloud: an emerging trend and improved performance in cloud computing. SN Comput Sci 1(5):256
Gupta H, Vahid Dastjerdi A, Ghosh SK, Buyya R (2017) iFogSim: a toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. Softw Pract Exp 47(9):1275–1296
Gures E, Shayea I, Ergen M, Azmi MH, El-Saleh AA (2022) Machine learning-based load balancing algorithms in future heterogeneous networks: a survey. IEEE Access 10:37689–37717
Hai T, Zhou J, Jawawi D, Wang D, Oduah U, Biamba C, Jain SK (2023) Task scheduling in cloud environment: optimization, security prioritization and processor selection schemes. J Cloud Comput 12(1):15
Hamed AY, Elnahary MK, Alsubaei FS, El-Sayed HH (2023) Optimization task scheduling using cooperation search algorithm for heterogeneous cloud computing systems. Comput Mater Contin 74(1):183–199
Hamid L, Jadoon A, Asghar H (2022) Comparative analysis of task level heuristic scheduling algorithms in cloud computing. J Supercomput 78(11):12931–12949
Han R, Liu CH, Zong Z, Chen LY, Liu W, Wang S, Zhan J (2019) Workload-adaptive configuration tuning for hierarchical cloud schedulers. IEEE Trans Parallel Distrib Syst 30(12):2879–2895
Hao Y, Xia M, Wen N, Hou R, Deng H, Wang L, Wang Q (2017) Parallel task scheduling under multi-clouds. KSII Trans Internet Inf Syst 11(1):39–60
Hariharan B, Paul Raj D (2020) WBAT job scheduler: a multi-objective approach for job scheduling problem on cloud computing. J Circuits Syst Comput 29(6):1
Hassan ME, Yousif A (2020) Cloud job scheduling with ions motion optimization algorithm. Eng Technol Appl Sci Res 10(2):5459–5465
Hassan HB, Barakat SA, Sarhan QI (2021) Survey on serverless computing. J Cloud Comput 10:1–29

He Y, Chen Y, Jimin L, Chen C, Guohua W (2019) Scheduling multiple agile earth observation satellites with an edge computing framework and a constructive heuristic algorithm. J Syst Archit 95:55–66
Hongwei D, Zhang Y (2023) Efficient cloud workflow scheduling with inverted ant colony optimization algorithm. Int J Adv Comput Sci Appl 14(10):1
Hosseini Shirvani M (2024) A survey study on task scheduling schemes for workflow executions in cloud computing environment: classification and challenges. J Supercomput 80(7):9384–9437
Hou H, Ismail A (2024) EETS: an energy-efficient task scheduler in cloud computing based on improved DQN algorithm. J King Saud Univ Comput Inf Sci 36:102177
Houssein EH, Gad AG, Wazery YM, Suganthan PN (2021) Task scheduling in cloud computing based on meta-heuristics: review, taxonomy, open challenges, and future trends. Swarm Evol Comput 62:100841
Hu Z, Wu K, Huang J (2012) An utility-based job scheduling algorithm for current computing cloud considering reliability factor. pp 296–299, Beijing, China: I EEE.
Hu B, Cao Z, Zhou M (2019) Scheduling real-time parallel applications in cloud to minimize energy consumption. IEEE Trans Cloud Comput 10(1):662–674
Hu B, Shi Y, Chen G, Cao Z, Zhou M (2023) Workload-aware scheduling of real-time jobs in cloud computing to minimize energy consumption. IEEE Internet Things J 11(1):638–652
Huang J, Li S, Chen Y (2020) Revenue-optimal task scheduling and resource management for IoT batch jobs in mobile edge computing. Peer Peer Netw Appl 13(5):1776–1787
Huang Y, Cheng L, Xue L, Liu C, Li Y, Li J, Ward T (2022) Deep adversarial imitation reinforcement learning for QoS-aware cloud job scheduling. IEEE Syst J 16(3):4232–4242
Huang H, Li J, Lu F, Li J (2024) Computational task transfer scheme based on multi-agent generative adversarial imitation learning. Research Square
Hussain A, Aleem M, Iqbal MA, Islam MA (2019) Investigation of cloud scheduling algorithms for resource utilization using CloudSim. Comput Inform 38(3):525–554
Hussain M, Wei LF, Lakhan A, Wali S, Ali S, Hussain A (2021) Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing. Sustain Comput Inform Syst 30:100517
Ibrahim IA, Bassiouni M (2020) Improvement of job completion time in data-intensive cloud computing applications. J Cloud Comput 9(1):8
Ibrahim M, Nabi S, Baz A, Naveed N, Alhakami H (2020) Towards a task and resource aware task scheduling in cloud computing: an experimental comparative evaluation. Int J Netw Distrib Comput 8(3):131–138
Iftikhar S, Ahmad Mirza MM, Tuli S, Chowdhury D, Xu M, Gill SS, Uhlig S (2023a) HunterPlus: AI based energy-efficient task scheduling for cloud-fog computing environments. Internet Things 21:100667
Iftikhar S, Gill SS, Song C, Minxian X, Aslanpour MS, Toosi AN, Junhui D, Huaming W, Ghosh S, Chowdhury D et al (2023b) AI-based fog and edge computing: a systematic review, taxonomy and future directions. Internet Things 21:100674
Indhumathi R, Amuthabala K, Kiruthiga G, Yuvaraj N, Pandey A (2023) Design of task scheduling and fault tolerance mechanism based on GWO algorithm for attaining better QoS in cloud system. Wireless Pers Commun 128(4):2811–2829
Islam MT, Wu H, Karunasekera S, Buyya R (2021) SLA-based scheduling of spark jobs in hybrid cloud computing environments. IEEE Trans Comput 71(5):1117–1132
Javadpour A, Sangaiah AK, Zhang W, Vidyarthi A, Ahmadi HR (2024) Decentralized AI-based task distribution on blockchain for cloud industrial internet of things. J Grid Comput 22(1):33
Jiang G, Huang R, Bao Z, Wang G (2024) A task offloading and resource allocation strategy based on multi-agent reinforcement learning in mobile edge computing. Future Internet 16(9):1. https://doi.org/10.3390/fi16090333
Julakanti SR, Sattiraju N, Julakanti R (2022) Multi-cloud security: strategies for managing hybrid environments. NeuroQuantology 20(11):10063–10074
Kamalakkannan R (n.d.) Cloud deployment models. https://www.linkedin.com/pulse/cloud-deployment-models-kamalakkannan-r-zquuc. Accessed 23 Nov 2024
Kamanga CT, Bugingo E, Badibanga SN, Mukendi EM (2023) A multi-criteria decision making heuristic for workflow scheduling in cloud computing environment. J Supercomput 79(1):243–264
Karthick AV, Ramaraj E, Ganapathy Subramanian R (2014) An efficient multi queue job scheduling for cloud computing. IEEE Computer Society, pp 164–166
Katal A, Dahiya S, Choudhury T (2023) Energy efficiency in cloud computing data centers: a survey on software technologies. Clust Comput 26(3):1845–1875
Kaul D (2019) Optimizing resource allocation in multi-cloud environments with artificial intelligence: balancing cost, performance, and security. J Big-Data Anal Cloud Comput 4(5):26–50
Kaur K, Garg S, Kaddoum G, Kumar N (2021) Energy and SLA-driven MapReduce job scheduling framework for cloud-based cyber-physical systems. ACM Trans Internet Technol 21(2):1–24
Kaur K, Garg S, Aujla GS, Kumar N, Zomaya AY (2022a) A multi-objective optimization scheme for job scheduling in sustainable cloud data centers. IEEE Trans Cloud Comput 10(1):172–186

Kaur M, Kadam S, Hannoon N (2022b) Multi-level parallel scheduling of dependent-tasks using graph-partitioning and hybrid approaches over edge-cloud. Soft Comput 26(11):5347–5362
Kaur R, Laxmi V, Balkrishan (2022c) Performance evaluation of task scheduling algorithms in virtual cloud environment to minimize makespan. Int J Inf Technol. https://doi.org/10.1007/s41870-021-00753-4
Kaur R, Anand D, Kaur U, Verma S, Kavita Park SW, Hosen AS, Ra IH (2023) An advanced job scheduling algorithmic architecture to reduce energy consumption and CO2 emissions in multi-cloud. Electronics (Switzerland) 12(8):1810
Khaleel MI (2023) Efficient job scheduling paradigm based on hybrid sparrow search algorithm and differential evolution optimization for heterogeneous cloud computing platforms. Internet Things (Netherlands) 22:100697
Khaleel MI (2024) Region-aware dynamic job scheduling and resource efficiency for load balancing based on adaptive chaotic sparrow search optimization and coalitional game in cloud computing environments. J Netw Comput Appl 221:103788
Khallouli W, Huang J (2022) Cluster resource scheduling in cloud computing: literature review and research challenges. J Supercomput 78(5):6898–6943
Khan MA (2020) Optimized hybrid service brokering for multi-cloud architectures. J Supercomput 76(1):666–687
Khan WZ, Ahmed E, Hakak S, Yaqoob I, Ahmed A (2019) Edge computing: a survey. Future Gener Comput Syst 97:219–235
Khan MJ, Ullah F, Imran M, Khan J, Khan A, AlGhamdi AS, Alshamrani SS (2022) Identifying challenges for clients in adopting sustainable public cloud computing. Sustainability 14(16):9809
Khan ZA, Aziz IA, Osman NA et al (2023) A review on task scheduling techniques in cloud and fog computing: taxonomy, tools, open issues, challenges, and future directions. IEEE Access. https://doi.org/10.1109/ACCESS.2023.3343877
Khezri E, Yahya RO, Hassanzadeh H, Mohaidat M, Ahmadi S, Trik M (2024) DLJSF: data-locality aware job scheduling IoT tasks in fog-cloud computing environments. Results Eng 21:101780
Khorsand R, Ramezanpour M (2020) An energy-efficient task-scheduling algorithm based on a multi-criteria decision-making method in cloud computing. Int J Commun Syst 33(9):e4379
Kim S-S, Byeon J-H, Hong Yu, Liu H (2014) Biogeography-based optimization for optimal job scheduling in cloud computing. Appl Math Comput 247:266–280
Kim T, Jeon M, Lee C, Kim SH, AL-Hazemi F, Youn C-H (2024) SLO-aware DL job scheduling for efficient FPGA-GPU edge cloud computing. Commun Comput Inf Sci 1898:19–29
Kliazovich D, Bouvry P, Khan SU (2012) GreenCloud: a packet-level simulator of energy-aware cloud computing data centers. J Supercomput 62:1263–1283
Krishna BSR, Krishna TM, Sri JB, Priya BV (2024) Task scheduling using TVIW-PSO in cloud computing. In: 2024 international conference on Advances in Computing, Communication and Applied Informatics (ACCAI). IEEE, pp 1–7
Krishnasamy KG, Periasamy S, Periasamy K, Moorthy VP, Thangavel G, Lamba R, Muthusamy S (2023) A pair-task heuristic for scheduling tasks in heterogeneous multi-cloud environment. Wireless Pers Commun 131(2):773–804
Kruekaew B, Kimpan W (2022) Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning. IEEE Access 10:17803–17818
Kumar Y, Kaul S, Yu-Chen H (2022) Machine learning for energy-resource allocation, workflow scheduling and live migration in cloud computing: state-of-the-art survey. Sustain Comput Inform Syst 36:100780
Kumar P, Karthikeyan S et al (2024a) Using genetic algorithms to optimize job scheduling in Google cloud platform. In: 2024 2nd International Conference on Networking and Communications (ICNWC). IEEE, pp 1–6
Kumar P, Pandi SS, Karthikeyan S (2024b) Using genetic algorithms to optimize job scheduling in Google cloud platform. Institute of Electrical and Electronics Engineers Inc
Kuppusamy P, Kumari NMJ, Alghamdi WY, Alyami H, Ramalingam R, Javed AR, Rashid MM (2022) Job scheduling problem in fog-cloud-based environment using reinforced social spider optimization. J Cloud Comput 11(1):1
Kurni M, Saritha K, Nagadevi D, Reddy KS (2022) A forefront insight into the integration of AI and blockchain technologies. In: Blockchain technology for emerging applications. Academic Press, pp 297–320
Lane P, Helian N, Bodla MH, Zheng M, Moggridge P (2022) Dynamic hierarchical structure optimisation for cloud computing job scheduling. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), 13224 LNCS:301–316
Li F, Hu B (2019) DeepJS: job scheduling based on deep reinforcement learning in cloud data center. Association for Computing Machinery, pp 48–53

Li Q, Peng X (2022) Job scheduling and simulation in cloud based on deep reinforcement learning. Xitong Fangzhen Xuebao 34(2):258–268
Li W, Xia Y, Zhou M, Sun X, Zhu Q (2018) Fluctuation-aware and predictive workflow scheduling in cost-effective infrastructure-as-a-service clouds. IEEE Access 6:61488–61502
Li C, Song M, Zhang M, Luo Y (2020a) Effective replica management for improving reliability and availability in edge-cloud computing environment. J Parallel Distrib Comput 143:107–128
Li C, Tang J, Ma T, Yang X, Luo Y (2020b) Load balance based workflow job scheduling algorithm in distributed cloud. J Netw Comput Appl 152:102518
Li C, Zhang YH, Luo Y (2021a) Neighborhood search-based job scheduling for IoT big data real-time processing in distributed edge-cloud computing environment. J Supercomput 77(2):1853–1878
Li W, Cao S, Hu K, Cao J, Buyya R (2021b) Blockchain-enhanced fair task scheduling for cloud-fog-edge coordination environments: model and algorithm. Secur Commun Netw 2021(1):5563312
Li Y, Lin Y, Wang Y, Ye K, Xu C (2022) Serverless computing: state-of-the-art, challenges and opportunities. IEEE Trans Serv Comput 16(2):1522–1539
Li Z, Chen Z, Wei X, Gao S, Yue H, Xu Z, Quek Tony QS (2024) Exploiting complex network-based clustering for personalization-enhanced hierarchical federated edge learning. IEEE Trans Mob Comput. https://doi.org/10.1109/TMC.2024.3449129
Lim SH, Sharma B, Nam G, Kim EK, Das CR (2009) MDCSim: a multi-tier data center simulation platform. In: 2009 IEEE international conference on cluster computing and workshops. IEEE, pp 1–9
Limbasan A, Rusu L (2011) Implementing SaaS solution for CRM. Inform Econ 15(2):175
Lin J, Peng Z, Cui D (2018) Deep reinforcement learning for multi-resource cloud job scheduling. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), 11303 LNCS:289–302
Lipsa S, Dash RK (2024) Sla-based task scheduling in cloud computing using randomized PSO algorithm. In: SCI. Springer, pp 206–217
Liu Y, Chen X, Ying H, Cai Q (2019) Adaptive job-scheduling algorithm based on queuing theory in a hybrid cloud environment. Int J Perform Eng 15(6):1580–1590
Loganathan S, Mukherjee S (2015) Job scheduling with efficient resource monitoring in cloud datacenter. Sci World J 2015(1):983018
Loganathan S, Saravanan RD, Mukherjee S (2017) Energy aware resource management and job scheduling in cloud datacenter. Int J Intell Eng Syst 10(4):175–184
Loncar P, Loncar P (2022) Scalable management of heterogeneous cloud resources based on evolution strategies algorithm. IEEE Access 10:68778–68791
Lu C, Xu H, Ye K, Xu G, Zhang L, Yang G, Xu C (2023) Understanding and optimizing workloads for unified resource management in large cloud platforms. In: Proceedings of the eighteenth European conference on computer systems. pp 416–432
Luo Q, Hu S, Li C, Li G, Shi W (2021) Resource scheduling in edge computing: a survey. IEEE Commun Surv Tutor 23(4):2131–2165
Mahdizadeh M, Montazerolghaem A, Jamshidi K (2024) Task scheduling and load balancing in SDN-based cloud computing: a review of relevant research. J Eng Res. https://doi.org/10.1016/j.jer.2024.11.002
Malik MK, Singh A, Swaroop A (2022) A planned scheduling process of cloud computing by an effective job allocation and fault-tolerant mechanism. J Ambient Intell Humaniz Comput 13(2):1153–1171
Malti AN, Hakem M, Benmammar B (2024) A new hybrid multi-objective optimization algorithm for task scheduling in cloud systems. Clust Comput 27(3):2525–2548
Mangalampalli S, Ganesh RK, Kumar M (2023) Multi objective task scheduling algorithm in cloud computing using grey wolf optimization. Clust Comput 26(6):3803–3822
Mangalampalli S, Karri GR, Ratnamani MV, Mohanty SN, Jabr BA, Ali YA, Ali S, Abdullaeva BS (2024) Efficient deep reinforcement learning based task scheduler in multi cloud environment. Sci Rep 14(1):21850
Mansouri N, Javidi MM (2020) Cost-based job scheduling strategy in cloud computing environments. Distrib Parallel Databases 38(2):365–400
Marahatta A, Xin Q, Chi C, Zhang F, Liu Z (2020) PEFS: AI-driven prediction based energy-aware fault-tolerant scheduling scheme for cloud data center. IEEE Trans Sustain Comput 6(4):655–666
Medara R, Singh RS (2021) Energy efficient and reliability aware workflow task scheduling in cloud environment. Wirel Pers Commun 119(2):1301–1320
Memari P, Mohammadi SS, Jolai F, Tavakkoli-Moghaddam R (2022) A latency-aware task scheduling algorithm for allocating virtual machines in a cost-effective and time-sensitive fog-cloud architecture. J Supercomput 78(1):93–122
Merseedi KJ, Zeebaree SRM (2024) The cloud architectures for distributed multi-cloud computing: a review of hybrid and federated cloud environment. Indones J Comput Sci 13(2):1

Miao Z, Liu L, Nan H, Li W, Pan X, Yang X, Mi Yu, Chen H, Zhao Y (2024) Energy and carbon-aware distributed machine learning tasks scheduling scheme for the multi-renewable energy-based edge-cloud continuum. Sci Technol Energy Transit 79:82

Microsoft Learn (2024) Traffic manager load-balancing methods—Azure traffic manager. https://learn.microsoft.com/en-us/azure/traffic-manager/traffic-manager-load-balancing-azure. Accessed 24 Nov 2024

Mikram H, El Kafhali S, Saadi Y (2023) A hybrid algorithm based on PSO algorithm and chi-squared distribution for tasks consolidation in cloud computing environment. In: 2023 IEEE 6th international conference on cloud computing and artificial intelligence: technologies and applications (CloudTech). IEEE, pp 1–6

Minxian X, Yang L, Wang Y, Gao C, Wen L, Guoyao X, Zhang L, Ye K, Chengzhong X (2024) Practice of Alibaba cloud on elastic resource provisioning for large-scale microservices cluster. Softw Pract Exp 54(1):39–57

Mithila SP, Franz P, Baumgartner G (2023) Scheduling many-task applications on multi-clouds and hybrid clouds. In: Workshop on asynchronous many-task systems and applications. Springer, pp 65–78

Mohammad Alshinwan A, Shdefat NM, AlSokkar A, Alsarhan T, Almajali D (2023) Integrated cloud computing and blockchain systems: a review. Int J Data Netw Sci 7(2):941–956

Mohammadzadeh A, Masdari M (2023) Scientific workflow scheduling in multi-cloud computing using a hybrid multi-objective optimization algorithm. J Ambient Intell Humaniz Comput 14(4):3509–3529

Mohammed CM, Zeebaree SR (2021) Sufficient comparison among cloud computing services: IaaS, PaaS, and SaaS: a review. Int J Sci Bus 5(2):17–30

Muniswamy S, Vignesh R (2022) DSTS: a hybrid optimal and deep learning for dynamic scalable task scheduling on container cloud environment. J Cloud Comput 11(1):33

Murad SA, Muzahid A, Azmi Z, Hoque MI, Kowsher M (2022) A review on job scheduling technique in cloud computing and priority rule based intelligent framework. J King Saud Univ Comput Inf Sci 34(6):2309–2331

Nadeem F (2022) Evaluating and ranking cloud IaaS, PaaS and SaaS models based on functional and non-functional key performance indicators. IEEE Access 10:63245–63257

Nagarajan G, Kumar KS (2021) Security threats and challenges in public cloud storage. In: 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE). IEEE, pp 97–100

Najm NMAM (2023) New hybrid priority scheduling algorithm based on a round robin with dynamic time quantum. In: AIP conference proceedings, vol 2787. AIP Publishing

Nasurudeen TFK, Shukla VK, Gupta S (2021) Automation of disaster recovery and security in cloud computing. In: 2021 International Conference on Communication Information and Computing Technology (ICCICT). IEEE, pp 1–6

Nayak D, Martha VS, Threm D, Ramaswamy S, Prince S, Fahrnberger G (2015) Adaptive scheduling in the cloud-SLA for Hadoop job scheduling. In: 2015 Science and Information Conference (SAI). IEEE, pp 832–837

Ngo KL, Mukherjee J, Jiang ZM, Litoiu M (2022) Evaluating the scalability and elasticity of function as a service platform. In: Proceedings of the 2022 ACM/SPEC on international conference on performance engineering. pp 117–124

N'Goran KR, Tetchueng JL, Kermarrec Y, Brou APB, Asseu O (2023) Blockchain-based identity and access management in a community cloud. In: 2023 international conference on software, telecommunications and computer networks (SoftCOM). IEEE, pp 1–6

Niknejad N, Ismail W, Ghani I, Nazari B, Bahari M (2020) Understanding service-oriented architecture (SOA): a systematic literature review and directions for further investigation. Inf Syst 91:101491

Niranjanamurthy M, Amulya MP, Niveditha NM, Dayananda P (2020) Creating a custom virtual private cloud and launch an elastic compute cloud (EC2) instance in your virtual private cloud. J Comput Theor Nanosci 17(9–10):4509–4514

Onyema EM, Gude V, Bhatt A, Aggarwal A, Kumar S, Benson-Emenike ME, Nwobodo LO (2024) Smart job scheduling model for cloud computing network application. SN Comput Sci 5(1):1

Pan S, Zhao H, Cai Z, Li D, Ma R, Guan H (2023) Sustainable serverless computing with cold-start optimization and automatic workflow resource scheduling. IEEE Trans Sustain Comput. https://doi.org/10.1109/TSUSC.2023.3311197

Parast FK, Sindhav C, Nikam S, Yekta HI, Kent KB, Hakak S (2022) Cloud computing security: a survey of service-based models. Comput Secur 114:102580

Pareek PK, Nithyashree GD, Rajashekhar SA, Zanke Pankaj, Navyashree KS (2024) Scheduling the tasks in cloud computing using adaptive adjustment in whale optimization algorithm. In: 2024 second international conference on data science and information system (ICDSIS). IEEE, pp 1–8

Paul M, Samanta D, Sanyal G (2011) Dynamic job scheduling in cloud computing based on horizontal load balancing. Int J Comput Technol Appl 2(5):1552–1556

Paulraj D, Sethukarasi T, Neelakandan S, Prakash M, Baburaj E (2023) An efficient hybrid job scheduling optimization (EHJSO) approach to enhance resource search using cuckoo and grey wolf job optimization for cloud environment. PLoS ONE 18(3):e0282600
Phasinam K, Kassanuk T, Shinde PP, Thakar CM, Sharma DK, Mohiddin MK (2022) Rahmani AW (2022) Application of IoT and cloud computing in automation of agriculture irrigation. J Food Qual 1:8285969
Piraghaj SF, Dastjerdi AV, Calheiros RN, Buyya R (2017) ContainerCloudSim: an environment for modeling and simulation of containers in cloud data centers. Softw Pract Exp 47(4):505–521
Pirozmand P, Jalalinejad H, Hosseinabadi AAR, Mirkamali S, Li Y (2023) An improved particle swarm optimization algorithm for task scheduling in cloud computing. J Ambient Intell Humaniz Comput 14(4):4313–4327
Prity FS, Gazi MH, Aslam Uddin KM (2023) A review of task scheduling in cloud computing based on nature-inspired optimization algorithm. Clust Comput 26(5):3037–3067
Priyadarshini A, Pradhan SK, Laha SR, Nayak S, Pattanaik BC (2024) Dynamic load balancing with task migration: a genetic algorithm approach for optimizing cloud computing infrastructure. In: 2024 international conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC). IEEE, pp 1–6
Putra TD (2022) Analysis of priority preemptive scheduling algorithm: case study. Int J Adv Res Comput Commun Eng 11(1):27–30
Qian W, Coutinho RWL (2024) Load-aware orchestrator for edge computing-aided wireless augmented reality. IEEE Internet Things J. https://doi.org/10.1109/JIOT.2024.3494533
Qin W, Chen H, Wang L, Xia Y, Nascita A, Pescapè A (2024) MCOTM: mobility-aware computation offloading and task migration for edge computing in industrial IoT. Future Gener Comput Syst 151:232–241
Qureshi A, Sharma A (2021) Cloud computing: the new world of technology. In: Proceedings of second international conference on smart energy and communication: ICSEC 2020. Springer Singapore, pp 55–60
Raghavendra SN, Jogendra KM, Smitha CC (2020) A secured and effective load monitoring and scheduling migration VM in cloud computing. In: IOP conference series: materials science and engineering, vol 981. IOP Publishing, p 022069
Rahumath AS, Mohanasundaram N, Malangai AR (2021) Resource scalability and security using entropy based adaptive krill herd optimization for auto scaling in cloud. Wirel Pers Commun 119:791–813
Rai HP, Ogeti P, Fadnavis NS, Patil GB, Padyana UK (2021) Integrating public and private clouds: the future of hybrid cloud solutions. Univ Res Rep 8(2):143–153
Rajak R, Kumar S, Prakash S, Rajak N, Dixit P (2023) A novel technique to optimize quality of service for directed acyclic graph (DAG) scheduling in cloud computing environment using heuristic approach. J Supercomput 79(2):1956–1979
Rajasoundaran S, Prabu AV, Routray S, Kumar SS, Malla PP, Maloji S, Ghosh U (2021) Machine learning based deep job exploration and secure transactions in virtual private cloud systems. Comput Secur 109:102379
Rajeshwari BS, Dakshayini M, Guruprasad HS (2022) Workload balancing in a multi-cloud environment: challenges and research directions. In: Operationalizing multi-cloud environments: technologies, tools and use cases. pp 129–144
Raju R, Babukarthik RG, Chandramohan D, Dhavachelvan P, Vengattaraman T (2013) Minimizing the makespan using hybrid algorithm for cloud computing. In: 2013 3rd IEEE International Advance Computing Conference (IACC). IEEE, pp 957–962
Rani D, Ranjan RK (2014) A comparative study of SaaS, PaaS and IaaS in cloud computing. Int J Adv Res Comput Sci Softw Eng 4(6):1
Rani S, Suri PK (2020) An efficient and scalable hybrid task scheduling approach for cloud environment. Int J Inf Technol 12(4):1451–1457
Ranichandra S, Vaneeswari V, Chinnasamy R, Dinesh S (2018) An enhanced job scheduling policy for the cloud environment to achieve optimal solution for low task low resource classification. Eurasian J Anal Chem 13(2):429–434
Rashidifar R, Bouzary H, Chen FF (2022) Resource scheduling in cloud-based manufacturing system: a comprehensive survey. Int J Adv Manuf Technol 122(11):4201–4219
Rehan H (2024) Revolutionizing America's cloud computing the pivotal role of AI in driving innovation and security. J Artif Intell Gener Sci 2(1):239–240
Rehman AU, Aguiar RL, Barraca JP (2022) Fault-tolerance in the scope of cloud computing. IEEE Access 10:63422–63441
Remesh A, Nahhas A, Kharitonov A, Turowski K (2023) A hybrid job scheduling approach for cloud computing environments: on the usage of heuristic and metaheuristic methods, vol 2023-January. IEEE Computer Society, pp 1580–1589
Rjoub G, Bentahar J, Wahab OA (2020) BigTrustScheduling: trust-aware big data task scheduling approach in cloud computing environments. Future Gener Comput Syst 110:1079–1097

Rjoub G, Bentahar J, Wahab OA, Bataineh AS (2021) Deep and reinforcement learning for automated task scheduling in large-scale cloud computing systems. Concurr Comput Pract Exp 33(23):e5919
Rozehkhani SM, Mahan F, Pedrycz W (2024) VM consolidation steps in cloud computing: a perspective review. Simul Model Pract Theory 138:103034
Sahraei SH, Kashani MMR, Rezazadeh J, Farahbakhsh R (2019) Efficient job scheduling in cloud computing based on genetic algorithm. Int J Commun Netw Distrib Syst 22(4):447–467
Saidi K, Bardou D (2023) Task scheduling and VM placement to resource allocation in cloud computing: challenges and opportunities. Clust Comput 26(5):3069–3087
Sangaiah AK, Javadpour A, Pinto P, Rezaei S, Zhang W (2023) Enhanced resource allocation in distributed cloud using fuzzy meta-heuristics optimization. Comput Commun 209:14–25
Sanjalawe Y (2023) Cloud computing simulators: a review. In: 2023 24th international Arab Conference on Information Technology (ACIT). IEEE, pp 1–14
Sanjalawe Y, Al-E'mari S (2023) Cloud computing simulators: a review. In: 2023 24th international Arab Conference on Information Technology (ACIT). pp 1–14
Sanjalawe Y, Althobaiti T (2023) DDoS attack detection in cloud computing based on ensemble feature selection and deep learning. Comput Mater Contin 75(2):1
Sanjalawe Y, Anbar M, Al-E'mari S, Abdullah R, Hasbullah I, Aladaileh M (2021) Cloud data center selection using a modified differential evolution. Comput Mater Contin 69(3):1
Saraswat M, Tripathi RC (2020) Cloud computing: analysis of top 5 CSPs in SaaS, PaaS and IaaS platforms. In: 2020 9th international conference System Modeling and Advancement in Research Trends (SMART). IEEE, pp 300–305
Sarkar S, Naug A, Luna R, Guillen A, Gundecha V, Ghorbanpour S, Mousavi S, Markovikj D, Babu AR (2024) Carbon footprint reduction for sustainable data centers in real-time. Proceedings of the AAAI conference on artificial intelligence 38:22322–22330
Saroha VK, Rana S (2019) Performance evaluation in implementing a multi-layer job scheduling approach with energy efficient resource utilization over a cloud. Int J Eng Adv Technol 8(3):28–33
Sasubilli MK, Venkateswarlu R (2021) Cloud computing security challenges, threats and vulnerabilities. In: 2021 6th International Conference on Inventive Computation Technologies (ICICT). IEEE, pp 476–480
Saxena S, Yagyasen D, Saranya CN, Boddu RSK, Sharma AK, Gupta SK (2021) Hybrid cloud computing for data security system. In: 2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA). IEEE, pp 1–8
Seethamraju R (2015) Adoption of software as a service (SaaS) enterprise resource planning (ERP) systems in small and medium sized enterprises (SMEs). Inf Syst Front 17:475–492
Senthilkumar G, Suvarnamukhi B, Lekashri S, Mohammed Thaha M (2024) Effective task scheduling based on interactive autodidactic school algorithm for cloud computing. Automatika 65(1):159–166
Shafiq DA, Jhanjhi NZ, Abdullah A (2022) Load balancing techniques in cloud computing environment: a review. J King Saud Univ Comput Inf Sci 34(7):3910–3933
Shahid MA, Islam N, Alam MM, Su'ud MM, Musa S (2020) A comprehensive study of load balancing approaches in the cloud computing environment and a novel fault tolerance approach. IEEE Access 8:130500–130526
Shahid MA, Islam N, Alam MM, Mazliham MS, Musa S (2021) Towards resilient method: an exhaustive survey of fault tolerance methods in the cloud computing environment. Comput Sci Rev 40:100398
Shahidinejad A, Ghobaei-Arani M, Masdari M (2021) Resource provisioning using workload clustering in cloud computing environment: a hybrid approach. Clust Comput 24(1):319–342
Shahmirzadi D, Khaledian N, Rahmani AM (2024) Analyzing the impact of various parameters on job scheduling in the Google cluster dataset. Clust Comput. https://doi.org/10.1007/s10586-024-04377-8
Shaji George A, Hovan George AS, Baskar T (2023) Edge computing and the future of cloud computing: a survey of industry perspectives and predictions. Partn Univers Int Res J 2(2):19–44
Sharma Y, Lakra S (2020) Green cloud job scheduling and load balancing using hybrid biogeography based optimization and genetic algorithm: a proposed approach. Lecture Notes in Networks and Systems 106:185–195
Sharma RK, Sharma N (2013) A dynamic optimization algorithm for task scheduling in cloud computing with resource utilization. Int J Sci Eng Technol 2(10):1062–1068
Sharma S, Kumar N, Dash Y, Dubey A, Devi K (2024) Intelligent multi-cloud orchestration for AI workloads: enhancing performance and reliability. In: 2024 7th International Conference on Contemporary Computing and Informatics (IC3I), vol 7. IEEE, pp 1421–1426
Shete A (2024) Borg: large-scale cluster management system. https://medium.com/@adityashete009/borg-large-scale-cluster-management-system-cbdcc4f8eb91. Accessed 24 Nov 2024
Shi W, Li H, Zeng H (2022) DRL-based and BSLD-aware job scheduling for apache spark cluster in hybrid cloud computing environments. J Grid Comput 20(4):1

Shokhanda J, Pal U, Kumar A, Chattopadhyay S, Bhattacharya A (2024) SafeTail: efficient tail latency optimization in edge service scheduling via computational redundancy management. arXiv Preprint. http://arxiv.org/abs/2408.17171

Shu W, Cai K, Xiong NN (2021) Research on strong agile response task scheduling optimization enhancement with optimal resource usage in green cloud computing. Future Gener Comput Syst 124:12–20

Shukla S, Hassan MF, Tran DC, Akbar R, Paputungan IV, Khan MK (2023) Improving latency in internet-of-things and cloud computing for real-time data transmission: a systematic literature review (SLR). Clust Comput. https://doi.org/10.1007/s10586-021-03279-3

Singh R (2018) Hybrid genetic, variable neighbourhood search and particle swarm optimisation-based job scheduling for cloud computing. Int J Comput Sci Eng 17(2):184–191

Singh G, Chaturvedi AK (2024) Hybrid modified particle swarm optimization with genetic algorithm (GA) based workflow scheduling in cloud-fog environment for multi-objective optimization. Clust Comput 27(2):1947–1964

Singh J, Gupta D (2017) Energy efficient heuristic base job scheduling algorithms in cloud computing. IOSR J Comput Eng e-ISSN, pp 2278–0661

Singh H, Tyagi S, Kumar P, Gill SS, Buyya R (2021) Metaheuristics for scheduling of heterogeneous tasks in cloud computing environments: analysis, performance evaluation, and future directions. Simul Model Pract Theory 111:102353

Singhal S, Sharma A (2021) A job scheduling algorithm based on rock hyrax optimization in cloud computing. Computing 103(9):2115–2142

Singhal S, Ali S, Awasthy M, Shukla DK, Tiwari R (2024) Rock-hyrax: an energy efficient job scheduling using cluster of resources in cloud computing environment. Sustain Comput Inform Syst 42:100985

Sonmez C, Ozgovde A, Ersoy C (2018) EdgeCloudSim: an environment for performance evaluation of edge computing systems. Trans Emerg Telecommun Technol 29(11):e3493

Sravanthi G, Moparthi NR (2024) Dual interactive Wasserstein generative adversarial network optimized with arithmetic optimization algorithm-based job scheduling in cloud-based IoT. Clust Comput 27(1):931–944

Suganya R, Joseph Niju P, Rajadevi R, Ramamoorthy S (2023) Enhancing the job scheduling procedure to develop an efficient cloud environment using near optimal clustering algorithm. Int J Cloud Comput 12(2–4):134–147

Suleiman N, Murtaza Y (2024) Scaling microservices for enterprise applications: comprehensive strategies for achieving high availability, performance optimization, resilience, and seamless integration in large-scale distributed systems and complex cloud environments. Appl Res Artif Intell Cloud Comput 7(6):46–82

Suliman YM, Yousif A, Bashir MB (2019) Shark smell optimization (SSO) algorithm for cloud jobs scheduling. Commun Comput Inf Sci 1098 CCIS:71–80

Sun Z, Liu J, Xing X, Li C, Pan X (2019) A dynamic cluster job scheduling optimisation algorithm based on data irreversibility in sensor cloud. Int J Embed Syst 11(5):551–561

Suresh P, Keerthika P, Manjula Devi R, Kamalam GK, Logeswaran K, Sadasivuni KK, Devendran K (2024) Optimized task scheduling approach with fault tolerant load balancing using multi-objective cat swarm optimization for multi-cloud environment. Appl Soft Comput 165:112129

Sutar S, Byranahallieraiah M, Shivashankaraiah K (2024) A dual-objective approach for allocation of virtual machine with improved job scheduling in cloud computing. Int Arab J Inf Technol 21(1):46–56

Taha MB, Sanjalawe Y, Al-Daraiseh A, Fraihat S et al (2024) Proactive auto-scaling for service function chains in cloud computing based on deep learning. IEEE Access, 12:38575–38593.

Tamoor-ul Hassan S, Samarakoon S, Bennis M, Latva-Aho M (2023) Latency-aware radio resource optimization in learning-based cloud-aided small cell wireless networks. IEEE Trans Green Commun Netw

Tang X (2021) Reliability-aware cost-efficient scientific workflows scheduling strategy on multi-cloud systems. IEEE Trans Cloud Comput 10(4):2909–2919

Tang S, Yu C, Li Y (2020) Fairness-efficiency scheduling for cloud computing with soft fairness guarantees. IEEE Trans Cloud Comput 10(3):1806–1818

Tang C, Xia S, Li Q, Chen W, Fang W (2021) Resource pooling in vehicular fog computing. J Cloud Comput 10:1–14

Tang J, Jalalzai MM, Feng C, Xiong Z, Zhang Y (2022) Latency-aware task scheduling in software-defined edge and cloud computing with erasure-coded storage systems. IEEE Trans Cloud Comput 11(2):1575–1590

Tarannum W, Abidin S (2023) Integration of blockchain and cloud computing: a review. In: 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom). IEEE, pp 1623–1628

Tomarchio O, Calcaterra D, Modica GD (2020) Cloud resource orchestration in the multi-cloud landscape: a systematic review of existing frameworks. J Cloud Comput 9(1):49

Tong Z, Chen H, Deng X, Li K, Li K (2020) A scheduling scheme in the cloud computing environment using deep q-learning. Inf Sci 512:1170–1191
Trabelsi Z, Ali M, Qayyum T (2024) Fuzzy-based task offloading in Internet of Vehicles (IoV) edge computing for latency-sensitive applications. Internet Things 28:101392
Tuli S, Casale G, Jennings NR (2021) MCDS: AI augmented workflow scheduling in mobile edge cloud computing systems. IEEE Trans Parallel Distrib Syst 33(11):2794–2807
Tuli S, Gill SS, Minxian X, Garraghan P, Bahsoon R, Dustdar S, Sakellariou R, Rana O, Buyya R, Casale G et al (2022) Hunter: AI based holistic resource management for sustainable cloud computing. J Syst Softw 184:111124
Ullah A, Nawi NM, Ouhame S (2022) Recent advancement in VM task allocation system for cloud computing: review from 2015 to 2021. Artif Intell Rev 55(3):2529–2573
Umarani Srikanth G, Geetha R (2023) Effectiveness review of the machine learning algorithms for scheduling in cloud environment. Arch Comput Methods Eng 30(6):3769–3789
Vaquero LM (2011) EduCloud: PaaS versus IaaS cloud usage for an advanced computer science course. IEEE Trans Educ 54(4):590–598
Verma P, Maurya AK, Yadav RS (2024) A survey on energy-efficient workflow scheduling algorithms in cloud computing. Softw Pract Exp 54(5):637–682
Vidhya M, Devi R (2023) Analysis of optimization and conventional algorithms using CloudSim in cloud. In: 2023 12th international conference on System Modeling & Advancement in Research Trends (SMART). IEEE, pp 403–408
Vispute SD, Vashisht P (2023) Energy-efficient task scheduling in fog computing based on particle swarm optimization. SN Comput Sci 4(4):391
Vu K, Hartley K, Kankanhalli A (2020) Predictors of cloud computing adoption: a cross-country study. Telemat Inform 52:101426
Wagh N, Pawar V, Kharat K (2020) Educational cloud framework—a literature review on finding better private cloud framework for educational hub. In: Microservices in Big Data analytics: second international, ICETCE 2019, Rajasthan, India, February 1st-2nd 2019, revised selected papers. Springer, pp 13–27
Wang M, Zhang Q (2020) Optimized data storage algorithm of IoT based on cloud computing in distributed system. Comput Commun 157:124–131
Wang T, Liu Z, Chen Y, Xu Y, Dai X (2014a) Load balancing task scheduling based on genetic algorithm in cloud computing. In: 2014 IEEE 12th international conference on dependable, autonomic and secure computing. IEEE, pp 146–152
Wang W, Liang B, Li B (2014b) Multi-resource fair allocation in heterogeneous cloud computing systems. IEEE Trans Parallel Distrib Syst 26(10):2822–2835
Wang X, Wang Y, Cui Y (2014c) A new multi-objective bi-level programming model for energy and locality aware multi-job scheduling in cloud computing. Future Gener Comput Syst 36:91–101
Wang X, Lou H, Dong Z, Chentao Yu, Renquan L (2023) Decomposition-based multi-objective evolutionary algorithm for virtual machine and task joint scheduling of cloud computing in data space. Swarm Evol Comput 77:101230
Wang X, Zhang L, Wang L, Vincent Wang X, Liu Y (2024) Federated deep reinforcement learning for dynamic job scheduling in cloud-edge collaborative manufacturing systems. Int J Product Res. https://doi.org/10.1080/00207543.2024.2328116
Waseem M, Ahmad A, Liang P, Akbar MA, Khan AA, Ahmad I, Setälä M, Mikkonen T (2024) Containerization in multi-cloud environment: roles, strategies, challenges, and solutions for effective implementation. arXiv Preprint. http://arxiv.org/abs/2403.12980
Wei Y, Pan L, Liu S, Lei W, Meng X (2018) DRL-scheduling: an intelligent QoS-aware job scheduling framework for applications in clouds. IEEE Access 6:55112–55125
Weng C, Wan Y, Xie H (2024) Deepws: dynamic workflow scheduling in heterogeneous cloud clusters with edge-aware reinforcement learning and graph convolution networks. In: 2024 4th international conference on Neural Networks, Information and Communication (NNICE). IEEE, pp 410–414
Wu J, Zhang G, Nie J, Peng Y (2021) Zhang Y (2021) Deep reinforcement learning for scheduling in an edge computing-based industrial internet of things. Wirel Commun Mob Comput 1:8017334
Xia X, Chen L (2020) Elastic optical network service-oriented architecture (SOA) used for cloud computing and its resource mapping optimization scheme. J Nanoelectron Optoelectron 15(4):442–449
Xu H, Xu S, Wei W, Guo N (2023) Fault tolerance and quality of service aware virtual machine scheduling algorithm in cloud data centers. J Supercomput 79(3):2603–2625
Yakubu IZ, Musa ZA, Muhammed L, Ja'afaru B, Shittu F, Matinja ZI (2020) Service level agreement violation preventive task scheduling for quality of service delivery in cloud computing environment. Procedia Comput Sci 178:375–385

Yanamala A (2024) Emerging challenges in cloud computing security: a comprehensive review. Int J Adv Eng Technol Innov 1(4):448–479
Yang B, Xu X, Tan F, Park DH (2011) An utility-based job scheduling algorithm for cloud computing considering reliability factor. pp 95–102
Yonglin P, Li Z, Jiong Yu, Liang L, Guo B (2024) An elastic framework construction method based on task migration in edge computing. Softw Pract Exp 54(9):1811–1830
Yousif A, Shohdy M, Hassan A, Ali A (2023) Internet of things data cloud jobs scheduling using modified distance cat swarm optimization. Electronics (Switzerland) 12(23):1
Yunlong F, Jie L (2024) Incentive approaches for cloud computing: challenges and solutions. J Eng Appl Sci 71(1):51
Zain AM, Yousif A (2020) Chemical reaction optimization (CRO) for cloud job scheduling. SN Appl Sci 2(1):1
Zeng X, Garg SK, Strazdins P, Jayaraman PP, Georgakopoulos D, Ranjan R (2017) IoTSim: a simulator for analysing IoT applications. J Syst Archit 72:93–107
Zhang R (2020) The impacts of cloud computing architecture on cloud service performance. J Comput Inf Syst
Zhang W, Cui H (2023) Offloading dependency mobile tasks to computing power network. In: 2023 eleventh international conference on advanced Cloud and Big Data (CBD). IEEE, pp 19–26
Zhang Y, Liu B, Gong Y, Huang J, Xu J, Wan W (2024) Application of machine learning optimization in cloud computing resource scheduling and management. In: Proceedings of the 5th international conference on computer information and big data applications. pp 171–175
Zhou G, Tian W, Buyya R, Xue R, Song L (2024) Deep reinforcement learning-based methods for resource scheduling in cloud computing: a review and future directions. Artif Intell Rev 57(5):124
Zhu J, Li X, én RR, Xu X, Zhang Y (2016) Scheduling stochastic multi-stage jobs on elastic computing services in hybrid clouds. In 2016 IEEE International Conference on Web Services (ICWS). IEEE, pp 678–681
Zhu QH, Tang H, Huang JJ, Hou Y (2021) Task scheduling for multi-cloud computing subject to security and reliability constraints. IEEE/CAA J Autom Sin 8(4):848–865
Zhu X, Yao W, Wang W (2024) Load-aware task migration algorithm toward adaptive load balancing in edge computing. Future Gener Comput Syst 157:303–312
Zubair AA, Razak SA, Ngadi MA, Al-Dhaqm A, Yafooz WMS, Emara A-HM, Saad A, Al-Aqrabi H (2022) A cloud computing-based modified symbiotic organisms search algorithm (AI) for optimal task scheduling. Sensors 22(4):1674

Authors and Affiliations

Yousef Sanjalawe[1] · Salam Al-E'mari[2] · Salam Fraihat[3] · Sharif Makhadmeh[1]

✉ Salam Fraihat
s.fraihat@ajman.ac.ae

Yousef Sanjalawe
y.sanjalawe@ju.edu.jo

Salam Al-E'mari
salam.ammari@uop.edu.jo

Sharif Makhadmeh
s_makhadmeh@ju.edu.jo

[1] Department of Information Technology, King Abdullah II School for Information Technology, University of Jordan (UJ), Amman 11942, Jordan

[2] Information Security Department, Faculty of Information Technology, University of Petra, Amman 11196, Jordan

[3] Artificial Intelligence Research Center (AIRC), College of Engineering and Information Technology, Ajman University, 346 Ajman, United Arab Emirates