Simulation Modelling Practice and Theory 111 (2021) 102353

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Metaheuristics for scheduling of heterogeneous tasks in cloud computing environments: Analysis, performance evaluation, and future directions

Harvinder Singh [a,*], Sanjay Tyagi [b,1], Pardeep Kumar [b,1], Sukhpal Singh Gill [c,2], Rajkumar Buyya [d,3]

[a] Department of Virtualization, School of Computer Science, University of Petroleum and Energy Studies, India
[b] Department of Computer Science and Applications, Kurukshetra University, India
[c] School of Electronic Engineering and Computer Science, Queen Mary University of London, United Kingdom
[d] Cloud Computing and Distributed Systems (CLOUDS) Laboratory, School of Computing and Information Systems, The University of Melbourne, Australia

ARTICLE INFO

Keywords:
Cloud computing
Scheduling
Metaheuristic scheduling
Multi-objective optimization

ABSTRACT

In cloud computing environments, when a client wants to access any resources, hardware components, or application services, he needs to get a subscription for the same from service providers. The usages of each client are monitored over a network by service providers and later on user will be charged for the services used. Cloud service provider is responsible for providing Quality of Service to clients. As the number of client request increases in cloud environment, cloud service providers face various issues such as scheduling and allocation of resources, security, privacy and virtual machine migration. Swarm intelligence, biological systems, physical and chemical systems based metaheuristic algorithms have proved to be efficient and used to solve real world scheduling optimization problems. This review focused on the insight view of various nature-inspired metaheuristic algorithms and their comparisons on the basis of certain parameters that affects the efficiency and effectiveness of their applicability in order to schedule different tasks in cloud environment. This work facilitates comparative analysis of six metaheuristic techniques quantitatively based on scheduling parameters like makespan and resource utilization cost. The objective of this systematic review is to find the most optimal scheduling technique for solving multi criteria scheduling problem. After evaluating and comparing Ant Colony Optimization, Particle Swarm Optimization, Genetic Algorithm, Artificial Bee Colony algorithm, Crow Search Algorithm and Penguin Swarm Optimization Algorithm, it has been identified that Crow Search algorithm is the most optimal technique in terms of makespan and resource utilization cost parameters with significant improvement over others. Finally, the promising research directions has been identified.

* Correspondence to: Department of Virtualization, School of Computer Science, University of Petroleum and Energy Studies, Dehradun, Uttarakhand, India.
E-mail addresses: hsingh@ddn.upes.ac.in (H. Singh), tyagikuk@gmail.com (S. Tyagi), mittalkuk@gmail.com (P. Kumar), s.s.gill@qmul.ac.uk (S.S. Gill), rbuyya@unimelb.edu.au (R. Buyya).

[1] Department of Computer Science and Applications, Kurukshetra University, Kurukshetra, Haryana, India.

[2] School of Electronic Engineering and Computer Science, Queen Mary University of London, London, United Kingdom.

[3] Cloud Computing and Distributed Systems (CLOUDS) Laboratory, School of Computing and Information Systems, The University of Melbourne, Melbourne, Australia.

https://doi.org/10.1016/j.simpat.2021.102353
Received 22 June 2020; Received in revised form 24 March 2021; Accepted 13 May 2021
Available online 17 May 2021

1. Introduction

A system is considered to be an efficient if all of the applications executing over it gives their best performance. So for efficient performance of any system, there is a need for effective management of resources and scheduling of tasks on them. Same is the case in cloud environment, if users want an efficient performance, they need to have effective scheduling techniques [1]. Scheduling is a process of provisioning the available resources to the submitted applications in a given time period in such a way that each application can utilize the resources effectively which leads to maximization of Quality of Service (QoS). Therefore, multiple jobs are allocated to different types of resources respecting constraints given by the cloud consumers and cloud service providers.

From the cloud user's perspective, the constraints can be a deadline or a given budget i.e. they want to get their jobs done in a given time period and within a limited budget. On the other hand, there can be constraints from cloud service provider's side as they want to maximize both resource utilization and profits. Thus, keeping in mind the constraints of both the stakeholders and considering the scalability feature of the cloud environment, which means the user's requirements can change dynamically, the applications have to be scheduled onto given resources while maintaining QoS, this makes scheduling in cloud environment a NP-hard problem.

In cloud environment, the scheduling problem is a NP-hard optimization problem which means that these problems cannot be solved in polynomial time and for such problems no polynomial-time algorithms are known. Scheduling is a challenging task in cloud environment, for which optimization techniques such as heuristic techniques are often considered as a feasible solution [2]. Heuristic is a strategic way of hit and trial organized by set of rules. At times, when the complexity in the problem increases, heuristic has very limited success recorded among various applications. This limitation of heuristic approach is due to the delay caused in reaching an optimal solution. Hence, heuristic is considered as a time consuming and least optimal solution based approach. On the other hand, metaheuristic approach is expected to overcome these limitations and provide a most optimal solution with less time [3]. In cloud environment, the following challenges related to scheduling of tasks to limited number of resources have been identified from the literature:

- In cloud environment, there exists a task interference problem during the scale-in or scale-out of the resources dynamically as per the demand.
- The poor admission control mechanism can result in executing multiple tasks simultaneously which further can result in sudden exhaustion of resources.
- The variations in the QoS requirements at runtime can results in poor management of the provisioned resources.
- The execution of newly submitted tasks (unhandled requests) at runtime should be taken care off immediately.

1.1. Motivation

The motivation of this systematic review is to analyse the problem of optimal task-resource mapping in depth through a comparison among the standard versions of metaheuristic algorithms in cloud computing environment. This article is particularly motivated from the following requirements:

- In cloud environment, there is a need and demand for deeply understanding algorithms of task-resource mapping.
- The algorithms examined in this systematic review were initially evaluated in varying scenarios and configurations, thus their advantage and disadvantage are not carefully investigated.
- The need for selecting the best suitable algorithm based on different cloud consumer's and provider's requirements.
- There is a need to find the best simulator for a particular scenario such as to implement makespan and cost aware task to resource mapping approaches based on metaheuristic techniques.
- There is a need to identify the future research directions and open challenges in this area.

In cloud environment, the solution to task scheduling problem based on exhaustive search are impractical. In contrast, various metaheuristic techniques like Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Artificial Bee Colony (ABC), Crow Search Algorithm (CSA) and Penguins Search Optimization Algorithm (PeSOA) can be applied to solve scheduling problems in cloud computing. Moreover, analysis of different metaheuristic techniques: PeSOA, GA, PSO, ACO, CSA and ABC has to be performed quantitatively depending on multiple parameters. The performance constraints like load, resource utilization cost, makespan and QoS should be considered for determining the optimal metaheuristic technique for a particular set of cloud environment's simulation conditions [4,5].

In this systematic review, various well-established task-resource metaheuristic algorithms are chosen and the experiments are performed using CloudSim toolkit. The evaluated algorithms are selected from state-of-the-art algorithms based on their efficient performance in their individual implementations on different simulation parameters.

1.2. Our contributions

The main contributions of this work are as follows:

- Offering a cross-sectional view of the investigated metaheuristic-based task-resource mapping algorithms, which presents outstanding performance in cloud computing area.

Table 1
Comparison between present survey and other survey articles.

Criteria	1	2	3	4	5	6	7	8	9	10	11
Based on performance constraints	✓	✓	✓	✓	✓	✓	×	×	✓	✓	✓
Based on control parameters	✓	✓	✓	×	✓	×	✓	✓	×	×	×
Based on problems applied	✓	×	×	✓	×	×	×	✓	×	×	✓
Based on demerits	✓	×	✓	✓	✓	×	×	×	×	✓	×
Based on open research challenges for every technique	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	×
Based on objective	✓	×	✓	✓	✓	✓	✓	✓	×	×	×
Proposed an analytical model for quantitative analysis	✓	×	×	×	×	×	×	×	×	×	×
Complexity based analysis of each algorithm	✓	×	×	×	×	×	×	×	×	×	×

1—Present Survey, 2—Kalra and Singh 2015 [6]. 3—Xu and Buyya 2019 [7], 4—Singh and Chana 2016 [8], 5—Gill and Buyya 2019 [9], 6—Hameed et al. 2016 [10], 7—Boussaid et al. 2013 [12], 8—Tsai and Rodrigues 2014 [13], 9—Buyya et al. 2019 [14], 10—Kumar et al. 2019 [15], 11—Ting et al. 2015 [16].

- Presenting a unified simulation-based analysis framework based on CloudSim that allows evaluation and comparison of task-resource mapping algorithms in a unified and unbiased way.
- Discussing the merits and demerits of the investigated algorithms to suggest optimal algorithms for different set of conditions.
- Presenting a comparative analysis among various cloud simulators based on different parameters to provide the prospective readers a clear understanding in decision making while choosing the simulator for their applications.
- Presenting the promising future research directions for prospective readers.

1.3. Related surveys

In the past, there are some research articles that have comprehensively reviewed the metaheuristic techniques in cloud environment. Authors, Kalra and Singh [6] explored three standard metaheuristic techniques and provides the performance-based critical analysis of them, to achieve the near-optimal solution for the problem in hand. Xu and Buyya [7] has introduced Brownout approach and defined its taxonomies for handling resources and applications in cloud environment. Singh and Chana [8] conducted methodical analysis to explore vital characteristics of resource scheduling techniques so that future researchers will not face any trouble while choosing appropriate resource scheduling technique for specific workload. In 2019, Gill and Buyya [9] presented a conceptual model which helps cloud service providers in ensuring sustainability of the cloud services with the feature of maintaining energy efficiency. Hameed et al. [10] investigated the existing energy efficient resource allocation approaches based on suggested multi-dimension taxonomies. Beloglazov et al. [11] proposed different taxonomies for energy-efficient framework of computing system. These taxonomies can help to overcome the issues and challenges related to high energy consumption.

The comparison between our survey and other survey articles based on various criterion is presented in Table 1. It can be clearly understood from the comparative analysis that till now there is no existing survey or review that discuss about the quantitative comparative analysis of the six standard metaheuristic techniques for scheduling of heterogeneous tasks to provisioned resources. This work enhances the previous surveys and focuses on the analytical comparisons among the standard metaheuristic algorithms so that their applicability can be justified for scheduling optimization problem. Further, the critical analysis and observations are identified and proposed as future research directions.

1.4. Article organization

The rest of the paper content is organized as follows: The state-of-the-art on metaheuristic scheduling techniques in cloud environment is presented in Section 2. The investigated algorithms architecture and modelling is introduced in Section 3. In Section 4, the pseudocode of the investigated algorithms is presented, whereas the performance metrics are summarized in Section 5. Section 6 shows the performance comparisons of the investigated algorithms. Section 7 proposes future research directions. Finally, the paper is concluded in Section 8.

2. Review methodology

The review work presented in this paper is accomplished by following the systematic steps which includes the formulation of research questions based on various aspects of the topic in hand, executing the survey by searching for the topic in various information sources, examining the outcomes of the review by applying a review technique for selecting the appropriate review articles, managing the outcomes of the review by applying search criteria and performing quality assessment and at last extracting relevant data by crosschecking the results from selected articles using random samples. The review methodology have been represented diagrammatically in Fig. 1.

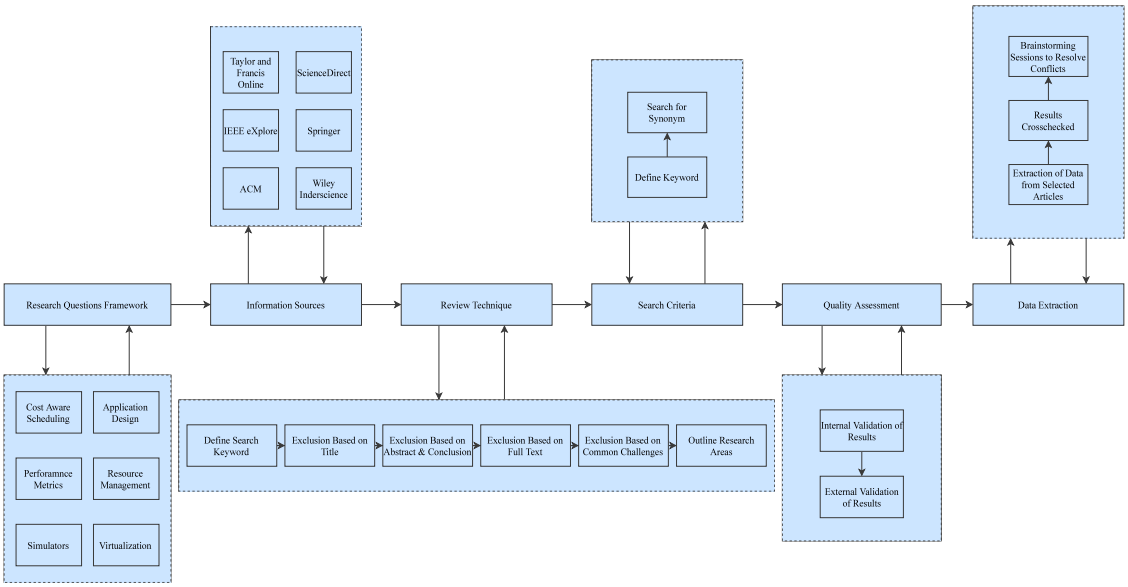

Fig. 1. Review methodology.

3. State-of-the-art metaheuristic techniques

In cloud environment, generation of optimal schedules for task scheduling on available resources is very tedious task [4], for the same, various techniques have been proposed and tested like greedy, genetic, heuristic and metaheuristic techniques [12,17]. Nowadays, global optimization problems are solved by using nature-inspired metaheuristic algorithms that came into existence since last two decades. Some of these standard algorithms are reviewed by Singh et al. [18] in terms of their representations, operators and application areas.

In systems with limited number of resources on cloud service provider side, to provide required quality of service to customers, additional cost will be charged by cloud service providers. So scheduling of tasks to resources must be done effectively to provide cloud consumers with required number of resources at reduced cost [19]. Scheduling problem is related to two types of users. First is cloud consumers, who wants to execute their tasks of varying complexity and size. Second is cloud providers, they want to contribute resources for executing consumer's jobs [20].

Both cloud consumers and cloud providers have different objectives. Cloud consumers benefit by selecting resources wisely from the provisioned resources with a focus to reduce time and cost [21]. For an application, if a user wants to minimize time period, he has to hire more resource from cloud providers which results in spending more money. In contrast, cloud service providers incur lot of money in building resources and providing infrastructure so as to maximize the resource utilization to earn profit [22]. Therefore, there exists a trade-off between cost and time [23]. For minimizing both cost and time, there has to be an algorithm which will balance between these two parameters [24]. The metaheuristic technique in the cloud environment is broadly classified into five categories based on structure namely flow control, memory, diversification, intensification and solution state. The solution state is further classified into singular metaheuristic and population metaheuristic. The iterated and guided local search, variable neighbourhood search, tabu search and simulated annealing belongs to the class of single-solution based metaheuristic however the swarm intelligence and evolutionary computation belongs to the class of population-based metaheuristic. The family of bio-inspired algorithms like ACO, ABC, CSA, PSO and PeSOA comes under swarm intelligence category while differential evolution, evolutionary strategy, GA, genetic programming and evolutionary programming comes under the category of evolutionary computation. The taxonomy of metaheuristic techniques is represented diagrammatically in Fig. 2.

In cloud computing environment, for ensuring QoS requirements of cloud consumers, a load balancing mechanism has been suggested by Singh et al. [25], Singh et al. [26] and Ye et al. [27]. The optimum resource utilization and energy consumption are the two major constraints considered for evaluating the performance of the proposed algorithm. The autonomic cloud computing has been comprehensively and critically reviewed by Kumar and Kumar [28] and Singh and Chana [29] to find out the research gaps that exists and further future research directions have been identified and quoted. The MOSACO i.e. "a Multi-objective scheduling method based on ant colony optimization" technique have been proposed by Zuo et al. [30] to optimally schedule the computing resources with an intent to minimize the execution cost and task completion times while maximizing the QoS and profit of the cloud service providers. The BULLET i.e. a "PSO based resource scheduling technique" have been proposed by the Gill et al. [31]. The suggested technique was compared with its counterparts [32–34] and found to have reduced the execution times, energy consumption and execution cost. The Table 2 shows comprehensive study of various state-of-the-art metaheuristic strategies on the basis of performance constraints.

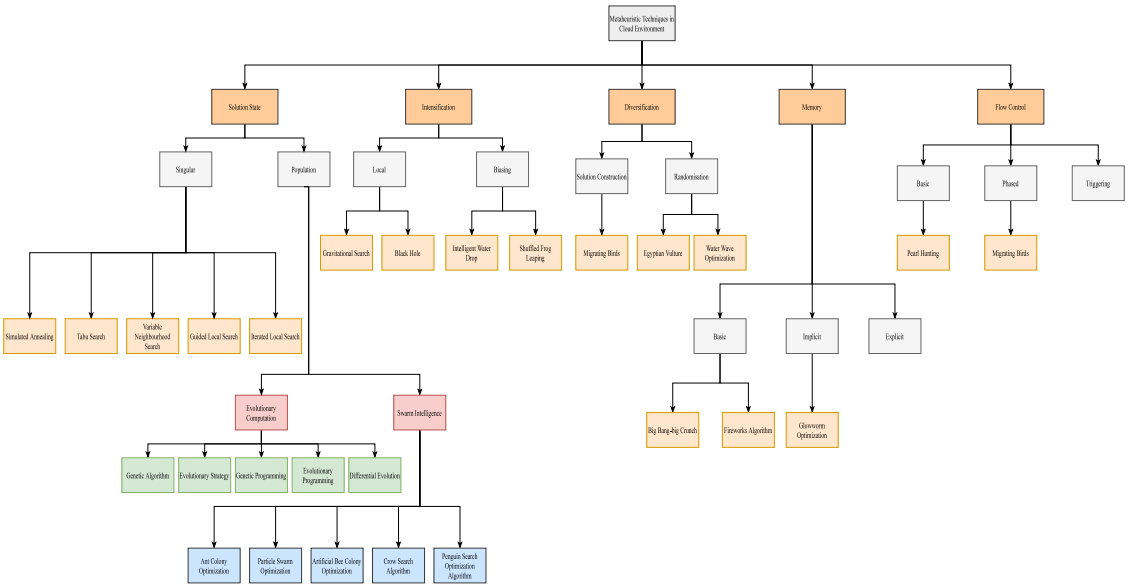

Fig. 2. A taxonomy of metaheuristic techniques for task scheduling in cloud computing.

Table 2
Review of metaheuristic algorithms based on Performance Constraints.

Author, Citation	Makespan time	Turnaround time	Execution time	Execution cost	Throughput	Resource utilization	Energy consumption
Wang et al. [35]	✓	×	×	✓	×	×	×
Salimi et al. [36]	✓	✓	✓	×	×	✓	×
Gill and Buyya [37]	×	×	✓	✓	✓	✓	✓
Abd Elaziz et al. [38]	✓	×	✓	×	✓	×	×
Adhikari et al. [39]	✓	×	✓	✓	✓	✓	×
Gill et al. [31]	×	×	✓	✓	×	✓	✓
Kumar et al. [15]	✓	✓	✓	✓	✓	✓	✓
Almezeini and Hafez [40]	✓	×	✓	✓	×	✓	×
Jena [41]	✓	×	✓	✓	×	✓	✓
Zuo et al. [30]	✓	×	✓	✓	×	✓	×
Mirjalili and Lewis [42]	✓	×	✓	×	×	×	×
Pacini et al. [43]	✓	×	✓	×	✓	✓	×
Zhao [44]	✓	×	✓	✓	×	✓	×
Singh and Chana [29]	×	×	✓	✓	✓	✓	✓
Ramezani et al. [45]	✓	×	✓	✓	✓	✓	×
Babu and Krishna [46]	✓	×	✓	×	✓	✓	×
Ye et al. [27]	×	×	×	×	×	✓	✓
Tsai and Rodrigues [13]	✓	×	✓	✓	×	×	×
Dasgupta et al. [47]	✓	×	✓	×	×	×	×
Yassa et al. [34]	✓	×	✓	✓	×	×	✓
Venters and Whitley [48]	×	×	✓	×	×	×	×

In the next decade the technological advancements like Internet of Things, server-less computing and edge computing is going to change the method of implementation and usage of cloud computing. Buyya et al. [14] has portrayed different challenges posed by the advances in technologies. They identified that there is a need of developing new methodologies which should be embedded with the existing infrastructure of cloud environment so that QoS requirements of all the associated stakeholders will be taken care off. The SCOOTER i.e. "self management of cloud services for execution of clustered workload" have been suggested by Gill and Buyya [37] to effectively and efficiently map the resources of cloud service providers to the consumer applications. So that execution cost, execution time, service level agreement violation rate, throughput, and resource utilization parameters are optimized as compared to its counterparts [49,50].

For scheduling optimization problem, a near-optimal solution has been proposed by Salimi et al. [36], where they have suggested a technique that can be analysed on the basis of number of processors and finish time required to accomplish the tasks. The trade-off between makespan and the cost of executing the tasks by cloud resources have been optimally managed by an IMPSO algorithm [35]. This algorithm minimizes the probability of plunging into local optimization and improves the convergence rate of PSO algorithm. The PSO algorithm can be further combined with ANN ("Artificial Neural Network") algorithm to make a hybridized ANN-PSO

Table 3
High-level Comparison of metaheuristic algorithms under investigation.

Technique	Inspiration	Control parameters	Objective function	Optimization parameters	Problems applied	Merits	Demerits	Open research challenges
Ant Colony Optimization [52]	It is based on the behaviour of ants searching for food	Quantity of trail laid by ants, trail persistence, relative importance of trail, relative importance of visibility	Reduce deadline violation rate and minimize cost	Deadline violation rate and cost	Multi-objective scheduling optimization problem	Optimal solution obtained through feedback network	Deadline violation is high	Different heuristic or metaheuristic techniques can be merged with standard ACO algorithm to improve its performance and convergence rate
Particle Swarm Optimization [53]	It is inspired by the movements of birds in flocks	Inertia weight, range of particles, dimensions of particles, number of particles, population size, number of iterations	Maximize convergence speed	Sphere function, rosenbrock function, de Jong's function, foxholes function, shaffer's function, griewank function, rostrigin function	Bench mark functions	The real strength of the particle swarm derives from the interaction among particles as they search space collaboratively	Trade-off between different parameters depends on the optimization problem in hand	The analysis of social influence aspect of the algorithm is a topic for future research
Genetic Algorithm [54]	It is inspired by the natural process of evolution	Crossover rate, mutation rate, generation gap	Maximize convergence speed	Weighted sum approaches, altering objective functions, pareto ranking approaches	Multiple fitness functions	The solution for problems with multimodal, non-convex and discontinuous solution space can be found easily	More difficult to implement as compared to others	Technique for determining optimal trade-off values of different control parameters is required
Artificial Bee Colony Algorithm [55]	It is inspired by the intelligent behaviour of honey bee swarm	Maximum number of cycle, colony size, maximum number of iterations, population size	Maximize convergence speed	Griewank function, rostrigin function, rosenbrock function, ackley function, schwefel function	High dimensional bench mark functions having multi-modality	It can be efficiently used for multi-variable and multi-modal function optimization problems	The convergence rate to get the solution of independent task scheduling needs improvement	The effect of changes in control parameters on the performance of ABC algorithm can be determined, performance of ABC algorithm can be improved by integrating useful heuristics
Crow Search Algorithm [56]	It is inspired by the intelligent behaviour of crows in hiding & retrieving its food when required	Flight length, awareness probability	Maximize convergence speed	Sphere function, rosenbrock function, griewank function, ackley function, schwefel function	Engineering optimization problem, pressure vessel design problem, tension/ compression spring design problem, welded beam design problem, bench mark functions	Convergence rate is good and fewer parameters to adjust	Trade-off between control parameters has to balanced in order to get the improved convergence rate	CSA algorithm can be applied for solving scheduling optimization problem in future, better results can be obtained by balancing trade-off between different control parameters of CSA algorithm
Penguin Search Optimization Algorithm [57]	It is inspired by the collaborative hunting strategy of penguins	Probability distribution of holes, probability distribution of levels	Maximize convergence speed	Rostrigin function, de Jong function, rosenbrock function, schwefel function	Bench mark functions	It is more robust & efficient	The number of penguins should be large for searching all local & global minima in search space	principle of reproduction and migration can be introduced to enhance search mechanism

algorithm in order to apply it for harmonic estimation [51]. The metaheuristic algorithms in cloud systems can be compared on the basis of common characteristics they possess to solve scheduling optimization problem. The inspiration, objectives, control parameters, problems on which they are applied, optimization parameters, merits, demerits and future research challenges are some of the basic characteristics that should be investigated for each of the metaheuristic algorithm as quoted in Table 4. The high level comparison of metaheuristic algorithms based on criteria mentioned in Table 4 is given in Table 3.

3.1. Discussions

The analysis of different metaheuristic scheduling algorithms based on various performance constraints has been done and quoted in Tables 2 and 3, the following perceptions and difficulties are recognized:

- The selection of appropriate scheduling technique depends on how efficient a chosen technique can address the trade-off between customer necessities and asset utilization. The different requirements possessed by customer tasks may include varied processing time, memory space, information traffic, response time, etc.
- The genuine need in the present cloud environment is to serve the customers as per their desired QoS-level expectations. For this, cloud facilitators needs to ensure that sufficient proportion of resources are provisioned to cloud buyers.
- In multi criteria task scheduling optimization, the schedulers would be efficient enough to handle the issues of high load, genuine asset contention, and inefficient participation to meet the needs of stakeholders.
- The strategies used diverse goals for task-resource mapping however the use of various objectives like makespan and resource utilization cost inside one calculation was not considered.

Table 4
Traits of metaheuristic algorithms.

Trait	Description
Technique	The different population based metaheuristic algorithms have been portrayed in this section.
Inspiration	The nature-inspired motivation source of metaheuristic algorithms have been depicted in this section.
Control Parameters	The control parameters of metaheuristic algorithms and their count plays a vital role in the selection of algorithm for solving a given problem.
Objective Function	The objective function of the metaheuristic algorithm has been designed for a specific purpose i.e. either minimizing/maximizing the given function value.
Optimization Parameters	For performing comparative analysis between various metaheuristic algorithms, the performance criterion have been designed using different optimization parameters.
Problems Applied	The problems applied define the environment where the metaheuristic algorithms can be applied and executed.
Merits	The advantages of metaheuristic algorithms have been described in this section.
Demerits	The disadvantages of metaheuristic algorithms have been described in this section.
Open Research Challenges	The research challenges related to different metaheuristic algorithms have been outlined in this section for the purpose of providing appropriate solution in future.

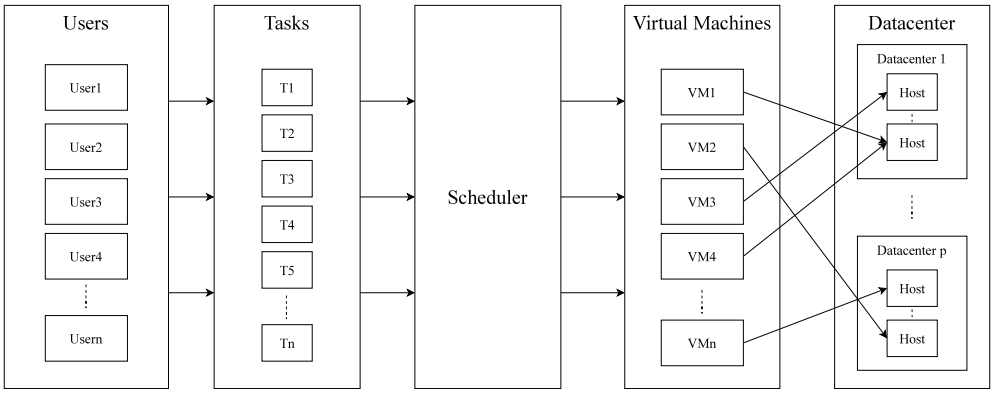

Fig. 3. Scheduling methodology.

Depending on following criteria, we carefully select 6 metaheuristic based state-of-the-art task-resource mapping techniques for our comparisons and evaluations:

- To make the comparison more persuasive, the algorithms were published in prominent journals or conferences, and the algorithms can be representative of a category of algorithms.
- To ensure the evaluation results reproducible, the algorithms were implemented in CloudSim or can be easily evaluated in CloudSim.
- To make the algorithm comparable, the algorithms should have been evaluated with the same baseline.

4. Scheduling model

Scheduling is a mechanism of allocating user tasks to available number of resources, with a condition of satisfying some constraints imposed on both tasks and resources by their respective cloud users and cloud service providers. The flow diagram of mapping different user tasks to the available resources i.e VMs in data centre is shown in Fig. 3.

Tasks submitted by different users are independent to each other with varied requirements. It is shown in Fig. 3 that cloud environment can consists of n number of users denoted as $User_1, User_2, ..User_n$ who can submit n number of tasks denoted as $T_1, T_2, ..T_n$

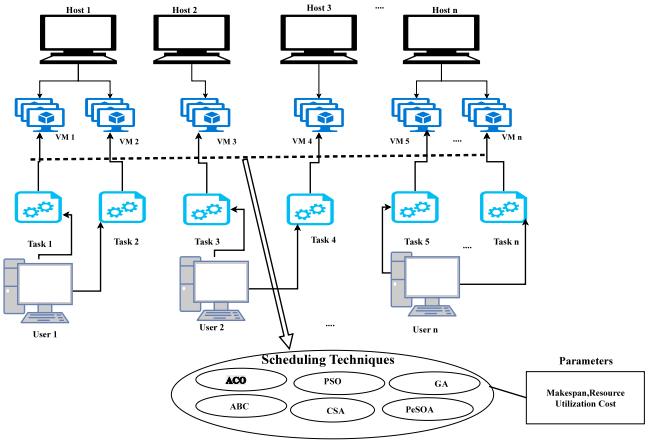

Fig. 4. Schematic diagram for implementing scheduling techniques.

that are allocated to the n number of VM's denoted as $VM_1, VM_2, ..VM_n$ of p number of data centres $Datacenter_1, Datacenter_2, ..Datacenter_p$.

The schematic diagram for implementing scheduling techniques is shown in Fig. 4. As portrayed in Fig. 4, at a particular instant of time, in a cloud environment there can be n number of users labelled as $User_1, User_2, ..User_n$. These users can either work dependently or independently of each other. The multiple applications submitted by the users are represented as n number of tasks labelled as $Task_1, Task_2, ..Task_n$. These tasks need resources for their execution. The resources in cloud environment is represented by Hosts labelled as $Host_1, Host_2, ..Host_n$. Further, each Host can consists of single or multiple VMs labelled as $VM_1, VM_2, ..VM_n$. The mapping of multiple tasks to available resources (VMs) is termed as scheduling problem [58]. The optimal scheduling algorithm is required in order to map the tasks to limited number of VMs keeping in view various performance constraints. In this paper, the six different metaheuristic scheduling algorithms from the state of the art literature has been identified and applied to solve multi-objective scheduling optimization problem considering makespan and resource utilization cost as parameters.

4.1. Objective function formulation

When a scheduling algorithm is designed and jobs are to be mapped on resources, the restriction or the optimization criteria specified by the stakeholders should be kept in mind [59]. Which further helps in designing the objective function [60,61]. An objective function is designed for scheduling algorithm and the main objective is to minimize or maximize this objective function according to the criteria specified by the user. So it is basically the solution for scheduling problem. The metrics used for the experimentation are makespan and resource utilization cost.

4.1.1. Makespan

Makespan is the cumulative execution time of aggregate tasks in a job queue. An optimal scheduling algorithm aimed at reducing the makespan [62]. Makespan (MS) can be calculated from Eq. (1)

$$MS = \sum_{r=1}^{s} max\ ET_r \tag{1}$$

where, s is the total number of tasks and ET_r refers to the execution time of the rth task which is calculated from Eq. (2)

$$ET_r = \frac{W}{\frac{1}{3} * \left[\frac{M^{ta} * MI_p}{c_1} + \frac{M^{ta} * PR_p}{c_2} + \frac{M^{ta} * MEM_p}{c_3}\right]} \tag{2}$$

with condition $1 \leq p \leq g$, where $c_1 = max\ MI_p, c_2 = max\ PR_p, c_3 = max\ MEM_p$ are the constants which represent the maximum value of MIPS, processor, memory of the VMs. M^{ta} refers to the task assignment matrix which is calculated on the tasks allocated to the VMs. W refers to the number of tasks waiting in the queue for execution.

The calculation of makespan relies on number of tasks, execution time, assignment matrix, MIPS, processor and memory of the VM where tasks are being executed.

4.1.2. Resource Utilization cost (RUC)

RUC is the total cost that client needs to pay for asset usage to service provider. The resource utilization of the system can be determined based on the task time matrix (M^{tt}) and the parameters of the VMs. The (M^{tt}) can be calculated from Eq. (3) [63].

$$M^{tt} = M^{ta} * ET_r \tag{3}$$

where M^{ta} is the task assignment matrix, ET_r is the execution time required to execute the rth task.

The resource utilization (RU) can be calculated by Eq. (4), given below.

$$RU = \frac{1}{s * N_v}\left\{\sum_{p=1}^{q}\sum_{r=1}^{s} M_{pr}^{tt} * \left[\frac{MI_p}{c_1} + \frac{PR_p}{c_2} + \frac{MEM_p}{c_3}\right]\right\} \tag{4}$$

where N_v is the normalized value, s is the total number of tasks, M_{pr}^{tt} is the task time matrix of rth task executed in the pth VM, MI_p is the MIPS of the pth VM, PR_p is the processor of the pth VM, MEM_p is the memory of the pth VM.

The resource utilization cost (RUC) can be calculated by using equation (5).

$$RUC = 1 - RU \tag{5}$$

4.1.3. Objective function

The objective function (F_n) for the comparative analysis of different metaheuristic scheduling algorithms based on makespan and resource utilization cost is defined in Eq. (6). The objective function aims at offering the minimum value.

$$F_n = \frac{MS}{R_{MS}} + \left[1 - RU\right] \tag{6}$$

In the further section, the metaheuristic based state-of-the-art task-resource mapping techniques have been studied in detail in terms of their applicability in solving the task scheduling optimization problem in cloud environment.

5. An overview of investigated algorithms

A metaheuristic is a more elevated amount of heuristics and is pertinent in situations where data given is inadequate or computational limit is deficient. These types of metaheuristic techniques are utilized to solve the complex scheduling optimization problems [13,64–66] . Some of them are discussed in the following sections.

5.1. Optimization in scheduling techniques

Optimization criteria is illustrated by the case in which user has specified that the job should be finished in some minimum amount of time period and minimum cost should be incurred, but has not specified any deadline or any budget specifications. So scheduling techniques are developed such that the constraints specified by the users should be satisfied. Like a job should be finished in a given amount of time period within specified budget constraints [67,68]. Optimization criteria can be a combination of various constraints, used when making scheduling decisions and it represents the goal of scheduling process.

5.2. Ant Colony Optimization (ACO) algorithm

ACO is aimed at searching for an optimal solution for an scheduling optimization problem. It simulates the searching behaviour of ants.

When a group of ants have to search for the shortest optimal path between their home and sustenance source, even without the usage of senses such as the sense of sight, ants are able to do so quite effortlessly. The answer to this is pheromones. Ants have a large number of pheromones that their body secretes where each pheromone symbolizes a different thing. Initially the ants choose a random path and once they successfully reach their desired target, the path of fitness is calculated and the pheromones are secreted and set on that particular path. As more and more ants follow this path the amount of pheromones increases and making it clearer for the other ants as to which path to choose through pheromone updating. The path having highest pheromone value represents the shorter path and the path having lowest pheromone value represents longest path. Initially, the ants are placed randomly at a point. The ants move from one VM to another, until the task is completely executed. The rate of evaporation of the pheromone is also another deciding factor as to which path to choose [65]. The same mechanism can be applied in the field of computer sciences, where the necessary parameters like number of tasks and resources, task deadline, incurred costs, and other related constraints are taken as input.

5.2.1. Pseudocode of ACO algorithm

The pseudocode of Ant Colony Optimization Algorithm is shown in Algorithm 1. The transition probability of kth ant on tth instance over edge e is given in Eq. (7)

$$P_e^k(t) = \begin{cases} \frac{\tau_e(t)^{\alpha}.\eta_e^{\beta}}{\sum_{k\in Nontraversed_k}(\tau_{e_1^k}(t)^{\alpha}).\eta_{e_2^k}^{\beta}} & if\ j \in Nontraversed_k \\ 0 & otherwise \end{cases} \tag{7}$$

where, if $e = (i, j)$ then $e_1^k = (i, k)$ and $e_2^k = (k, j)$

$P_e^k(t)$ represents transition probability of kth ant on tth instance over edge e and

$Nontraversed_k = V - traversed_k$,

α and β parameters are used to control relative importance of trail vs visibility

V is a set of vertices.

$$\Delta\tau_e^k = \begin{cases} \frac{Q}{tour_k} & if\ \tau \in traversed_k \\ 0 & otherwise \end{cases} \tag{8}$$

where, τ_e^k represent evaporation for kth ant on edge e
$tour_k$ is tour length for k_{th} ant
Q is the constant

Algorithm 1 Ant Colony Optimization (ACO)

Input: $G = (V, E)$, m = number of ants, c = small positive number, NC_{max} = maximum number of iterations
Output: Shortest path

Set $t \leftarrow 0, NC \leftarrow 0$, $\tau_e(t) \leftarrow \ c \ \forall \ e \in E \ \ \& \ \ \Delta t_E = 0$
randomly map m ants on V nodes as a initial loc
$S \leftarrow 1$
for $k \leftarrow 1$ *to* m **do**
    $traversed_k[m] \leftarrow starting\ town\ of\ ant\ k$
end for
while $(S < |V|)$ **do**
    $S \leftarrow S + 1$
    for $k \leftarrow 1$ *to* m **do**
        Compute $P_e(t)$ with Eq. (7)
        Select edge e to traverse with the probability $P_e(t)$
        Let ant k traverse with edge e
        $e = (i, j)$, insert j into $traversed_k$
    end for
end while
$min_k \leftarrow$ minimum tour among all ants
for $(each\ e \in E)$ **do**
    for $(k \leftarrow 1\ to\ m)$ **do**
        update $\Delta\tau_e^k$ according to Eq. (8)
        $\Delta\tau_e \leftarrow \Delta\tau_e + \Delta\tau_e^k$
        Compute $\tau_e(t + n) = \rho\tau_e(t) + \Delta\tau_e$
        $t \leftarrow t + n,\ NC \leftarrow NC + 1$
    end for
    for $(each\ e \in E)$ **do**
        $\tau_e \leftarrow 0$
        if $((NC < NC_{max})$ & not stagnation behaviour) **then**
            for $(k \leftarrow 1\ to\ m)$ **do**
                $traversed_k \leftarrow \phi$
            end for
            goto step 7
        else return shortest ant tour
        end if
    end for
end for

The complexity of the ACO algorithm is $O(|E||V|n)$, where n is the population size, $|E|$ is the number of edges $|V|$ is the number of vertices. ACO may work effectively for some of the applications and for others it may be slow convergence speed. Stagnation problem may also result when all the jobs are assigned to the same resource. So there is a need of developing an efficient metaheuristic scheduling technique that overcomes the limitations of the ACO algorithm.

5.3. Particle Swarm Optimization (PSO) algorithm

PSO algorithm optimizes a problem by iterative process of improving a candidate solution with regard to a given measure of quality. It imitates the behaviour of a bird flocking to find the food. PSO describes the particles and minimize them on the basis of the fitness function of each particle. Particle consist of the tasks and the mapped resources. Fitness function determines the effectiveness of the schedule [69]. Each particle is similar to chromosomes in genetic algorithm and have a fitness value, which will be assessed by a fitness capacity that need to be enhanced in each iteration [70,71].

Every particle has its best position termed as pbest where the fitness value is optimal as compared to other particles of whole population locally. The same particle has its best position termed as gbest where the fitness value is optimal as compared to other

particles of whole population globally [72,73]. In each cycle, every particle moves to another position and the new position is guided by the particle's velocity [74].

5.3.1. Pseudocode of PSO algorithm

The pseudocode of PSO Algorithm is shown in Algorithm 2.
private(t, i) = best found position of tth particle by i iteration
global(i) = best found solution by all particles on ith iteration
let x_t^i be the position vector of tth particle on ith instance

$$argMin_{Y=x_t^0,x_t^1,x_t^2,\ldots,x_t^i}[f(Y)] \tag{9}$$

$$l_i = argMin_i[f(private(k,i))] \tag{10}$$

where, k = 0 to m

$$global(i) = private(li,i) \tag{11}$$

where, f is the fitness function
$V_t(i)$ = denotes the velocity of the tth particle at ith iteration, it updates using the equation as follows:

$$V_t(i+1) = \omega V_t(i) + \alpha_1 r_1(private(t_i) - x_t^i) + \alpha_2 r_2(global(i) - x_t^i) \tag{12}$$

$$x_t^{i+1} = x_t^i + V_t(i+1) \tag{13}$$

where, r_1 and $r_2 \simeq [0,1]$
$\alpha_1 \& \alpha_2$ are small positive constant called acceleration coefficient
The complexity of PSO algorithm is $O(max_{iter} * n^2)$, where, n is population size and max_{iter} is maximum number of iterations.

Algorithm 2 Particle Swarm Optimization (PSO)

Input: Position vector x randomly initialized
Output: Best found solution

Initialize each position vector x_0^t to the random value in the solution space
$private(t,0) \leftarrow x_0^t$
Compute $global(0)$ using Eq. (11)
randomly select a valid initial velocity V_t for all particle
$i \leftarrow 0$
while $(i < max_{iter})$ **do**
    Select valid r_1 and r_2
    Compute each particle t velocity $V_t(i)$ by Eq. (12)
    Update each particle t position x_i^t by Eq. (13)
    Compute $private(t,i)$ for each $particle(t)$
    Check and update for each $particle(t)$
    if $(f(x_i^t) < f(global(i)))$ **then**
        $global(i) = x_i^t$
    end if
    $i \leftarrow i + 1$
end while
return $global(i)$ as the best found solution.

PSO technique may work efficiently for some of the applications and for others it may achieve optimal convergence rate. Moreover during convergence, the constant values of the parameters might cause the needless fluctuation of particles. So it is required to develop an effective metaheuristic technique that overcomes the limitations of the PSO algorithm.

5.4. Genetic Algorithm (GA)

Genetic algorithm is based on initial set of random solutions known as populations. Chromosomes are the individuals in the population. Solutions of one population is taken forward to reproduce the new population. This is done in the expectation that the new population generated are superior than the older one. The suitable resources are used to reproduce the new populations [75]. GA makes a populace of arrangements and applies control operators like mutation and crossover to find the best among them. At each step, there is a random selection of individuals from the current population [76,77].

Genetic algorithms are flexible and provides better optimize solutions whenever working with the large data sets. Easy implementation, simple architecture and heuristic properties are the advantages of genetic algorithms. Various researchers are working on the different parameters related to genetic algorithms and are trying to optimize it further for specific set of applications [78].

5.4.1. Pseudocode of GA algorithm

The pseudocode of GA Algorithm is shown in Algorithm 3.

The function $crossover(x, y)$ takes two solutions x, y and return offspring of $x\&y$

The function $mutate(x)$ takes one solution x and return the x

f is the fitness function

The complexity of the GA algorithm is $O(max_{iter} * n * S)$, where S is the computation for mutation and crossover function, n is the maximum population, max_{iter} is the maximum number of iterations.

Algorithm 3 Genetic Algorithm (GA)

Input: Randomly initiated N valid solutions
Output: Solution S

Let $i = 1$, where i is the iteration index.
Randomly initiate N solutions $S = \{x_1^1, x_1^2, x_1^3, ..., x_1^N\}$.
Compute fitness value $f(x_i^t), \forall t \in [1, N]$.
Select set $C \subset S \times S$ for the offspring generation based on fitness value.
if $(i < max_{iter})$ **then**
    For all $c \subset C$, apply crossover function on c & add the offspring back in S.
    Apply mutate function on each element of S.
    Compute fitness value of each solution in S & remove infeasible solution.
    Based on the fitness value, select N solutions from S and remove $(|S| - N)$ solution from S.
    $i \leftarrow i + 1$.
end if
Return S

Genetic scheduling technique may work efficiently for some of the applications and for others it may have long execution time. Moreover, GA has slow convergence rate and it may converge towards the local optima. Hence it is required to develop an efficient metaheuristic algorithm that overcomes the limitations of GA algorithm.

5.5. Artificial Bee Colony (ABC) algorithm

ABC algorithm is an technique which relies on behaviour of honey bee. It utilizes control parameters such as maximum number of cycles and colony size. ABC is a streamlining algorithm having a population based search methodology in which sustenance positions are changed by simulating honey bees with iteration [79]. In a multidimensional search space, bee flies around to search the food. ABC framework joins neighbourhood-seek techniques, completed by utilized and spectator honey bees, with worldwide hunt strategies, overseen by spectators and scouts, endeavouring to adjust investigation process [80].

5.5.1. Pseudocode of ABC algorithm

The pseudocode of ABC Algorithm is shown in Algorithm 4. $Neighbour(x_k, x_j) = x_j + \phi_j(x_j - x_k)$ where, ϕ_j is a random number associated with jth Bee.

The complexity of the ABC algorithm is $O(max_{iter} * n * d)$, where d is the problem dimension, n is the population size, max_{iter} is the maximum number of iterations.

Artificial bee colony scheduling technique may work efficiently for some of the applications and for others it may require random initializations. Moreover, ABC follows probabilistic approach in local search. Hence, it is required to design an effective metaheuristic algorithm that overcomes the limitations of the ABC algorithm.

5.6. Crow Search Algorithm (CSA)

The CSA algorithm is another population based metaheuristic technique. Crows are widely disseminated variety of feathered creatures which are currently viewed as world's most canny creatures. Crows demonstrate exceptional cases of ability and perform consistently on ability tests. These birds can remember faces, can utilize apparatuses, convey in advanced ways, cover up and recover food over different seasons. Motivated by nature, CSA works in light of this thought that crows store their abundance food in particular positions of the environment and recover food from hidden location when it is required. Crows tails other crows to acquire better sustenance sources. On the off chance that a crow notice another is tailing it, the crow misleads that crow by setting off to another position [56].

As of optimization, crows are denoted as searchers. Environment is denoted as search space. Each position of the search space corresponds to a feasible solution. The nature of food source is objective function. Finally the best food source represents the global solution of the problem.

Algorithm 4 Artificial Bee Colony Algorithm (ABC)

Input: $S \leftarrow number\ of\ onlooker\ bees$, $l \leftarrow lower\ bound$, $u \leftarrow upper\ bound$
Output: V with max fit(V)

Generate FS = $\begin{bmatrix} x_{00} & x_{01} & . & . & x_{0(n-1)} \\ x_{10} & x_{11} & . & . & x_{1(n-1)} \\ . & . & . & . & . \\ . & . & . & . & . \\ x_{m0} & x_{m1} & . & . & x_{m(n-1)} \end{bmatrix}$
where $x_{ij} = R(l_j, u_j)$
Select some employed bees
for (each employed bee j) **do**
  $k \leftarrow Random(m, 0)$
  $V_j \leftarrow max(fit(\overrightarrow{V_j}), fit(Neighbour(\overrightarrow{x_k}, \overrightarrow{x_j})))$
  Compute $d \leftarrow \sum_{i=0}^{E} fit(\overrightarrow{V_i})$
end for
for (each onlooker bee k) **do**
  Set selection probability of each food location, $l = \frac{fit(\overrightarrow{x_m})}{d}$
  Select a location according to probability for bee k
end for
Select some scout bees: randomly select a location in search space

5.6.1. Pseudocode of Crow Search Algorithm (CSA)

The pseudocode of Crow Search Algorithm is shown in Algorithm 5. The solution vector of crow k on ith iteration is given in Eq. (14)

$$l_k^i = \begin{cases} m_k^i + r_k * s_k^i (m_t^i - l_t^i) & r_j \geq P_t^i \\ random\ location & otherwise \end{cases} \tag{14}$$

$$m_k^{i+1} = \begin{cases} l_k^{i+1} & \Phi(l_k^{i+1}) is\ better\ than\ \Phi(m_k^i) \\ m_k^i & otherwise \end{cases} \tag{15}$$

where, $\Phi \rightarrow$ objective function and
$l_k^i \rightarrow$ solution vector of crow k on ith iteration,
$P_k^i \rightarrow$ awareness probability of kth crow at ith instance,
$r_k \rightarrow$ random number for crow k,
t $\rightarrow$ targeted crow,
$iter_{max} \rightarrow$ maximum number of iterations.
$m_k^i \rightarrow$ best known hiding location of kth crow at ith instance,
$s_k^i \rightarrow$ stamina factor of kth crow at ith instance,
The complexity of the CSA algorithm is $O(n * d * max_{iter})$, where d is the problem dimension and n is the population size, max_{iter} is the maximum number of iterations.

Algorithm 5 Crow Search Algorithm.

Input: Vector r, Awareness probability matrix P, max_{iter}
Output: Optimum location in search space

Randomly place each crow k in a valid location in search space represented by l_k^1.
Initialize the memory location $m_k^1 = l_k^1$.
$i \leftarrow 1$
while ($i < max_{iter}$) **do**
  for each crow k **do**
    Select the target crow t.
    Update the k crow location, l_k^i using Eq. (14).
    Check the validity of new location.
    Update the memory location of crow k using Eq. (15).
  end for
  $i \leftarrow i + 1$
end while

In the event of these situations, CSA reproduce the intelligent behaviour of crows, in discovering the solution to scheduling optimization issue. Moreover, CSA follows probabilistic approach in local search. Hence, it is required to develop an metaheuristic technique in which diversity of the algorithm can be more effectively controlled in order to solve complex optimization problem.

5.7. Penguin Search Optimization Scheduling Algorithm (PeSOA)

PeSOA algorithm depends on hunting behaviour of penguins. This penguin technique is interesting as they can work together with their efforts and synchronize their plunges to optimize the cumulative energy during the duration of aggregate hunting. Every penguin contribute to a solution and are distributed in groups. Each group looks for food in specific holes with different height levels. In this procedure, penguins arrange in groups and begin to pursuit in a particular hole and level according to food likelihood.

In each iteration, situation of the penguin with new solution is arrived, which is adjusted and therefore provide three solutions. Which are best local solution, the last solution, and the new solution. After multiple iterations the solution of every penguin in each group is re-evaluated and best solution is conveyed to other penguins in a group [57].

5.7.1. Pseudocode of Penguin Swarm Optimization Algorithm (PeSOA)

The pseudocode of Penguin Swarm Optimization Algorithm is shown in Algorithm 6. The next optimal location can be identified by using Eq. (16).

$$X_{i+1}^{t} = X_{i}^{t} + r_{i}^{t} | X_{best} - X_{previousbest} | \tag{16}$$

where, r_i^t is a random number for distribution

X_{best} is the globally found best solution

$X_{previousbest}$ previous best solution.

The complexity of the PeSOA algorithm is $O(max_{iter} * n * cf)$, where cf is the computation required for evaluating fitness function, n is the population size, max_{iter} is the maximum number of iterations.

Algorithm 6 Penguin Swarm Optimization Algorithm (PeSOA)

Input: max_{iter}, Total Penguins T
Output: Best Hunter Penguin

```
Randomly place the penguin in the solution space, represented by X_i^t ∈ S.
i ← 1
while (i < max_iter) do
    for (each X_i^t ∈ S) do
        Check the oxygen reserve of t^th penguin & take a random step accordingly.
        Update the t^th penguin location, X_i^t using Eq. (16) while satisfying the reserve oxygen criteria.
        Update the fitness function of X_i^t solution (fish eaten by t^th penguin)
    end for
    Select the best group by computing the fitness function of entire group.
    Update the best found solution X_best
    i ← i + 1
end while
```

Penguin scheduling technique may work effectively for some of the applications and for others it may detect all local and global minimum depending on the size of the group of penguins is big or small. Moreover, reproduction and migration rules used in PeSOA can be modified to make it more efficient. PeSOA scheduling algorithm can be combined with other scheduling algorithms to make it more effective metaheuristic scheduling algorithm.

The key observations extracted from the study of the literature must be addressed and the existing multi-objective metaheuristic optimization scheduling algorithm should be analysed quantitatively so as to determine their level of efficiency and effectiveness in solving scheduling optimization problem.

6. Comparisons of scheduling simulators with metaheuristics

In real cloud environment, it is expensive and difficult to do experimentation on real world data. Moreover if experiment does not performed as planned, it may led to loss of important data. Thus a simulator which can mimic the environment of real cloud is required [81]. So that various techniques and mechanisms proposed for the benefits of the stakeholders can be tested and validated [82]. Once approved, these techniques and mechanisms can be implemented in real cloud environment without any fear of losing data [59] [83]. The evaluation of various QoS constraints as per the service level agreement between cloud providers and cloud consumers, can be done with the help of cloud simulators. There are number of cloud simulators available in the market that differs in terms of their layers of implementation, applicability, constraints modelling, portability and so on [84] [85]. The comparative analysis of various cloud simulators on the basis of different parameters is shown in Table 5.

It is concluded from the comparative analysis performed in Table 5 that as the CloudSim simulator supports cost modelling, energy modelling, federation modelling, communication modelling and its available openly, thus it can be used for performing experimentation in almost all application areas of cloud computing.

Table 5
Comparison of Cloud Simulators on the basis of various parameters.

Cloud Simulators	1	2	3	4	5	6	7	8	9	10
ThermoSim [86]	2020	✓	✓	✓	×	×	×	✓	✓	×
BigDataSDNSim [87]	2019	✓	×	✓	×	×	✓	✓	✓	✓
SCORE [88]	2018	×	✓	✓	×	×	✓	×	✓	×
GAME-SCORE [89]	2018	×	✓	✓	×	×	✓	×	✓	×
DynamicCloudSim [90]	2014	✓	×	✓	×	✓	✓	✓	✓	×
CloudShed [91]	2013	✓	×	✓	✓	×	✓	✓	✓	✓
FTCloudSim [92]	2013	✓	×	✓	×	✓	✓	✓	✓	×
WorkflowSim [93]	2012	✓	×	✓	×	✓	✓	✓	✓	×
iCanCloud [94]	2011	✓	×	×	✓	×	✓	✓	✓	×
GDCSim [95]	2011	×	✓	✓	×	×	✓	×	×	✓
NetworkCloudSim [96]	2011	✓	×	✓	×	✓	✓	✓	✓	×
GreenCloud [97]	2010	×	✓	✓	✓	×	✓	✓	×	×
CloudAnalyst [98]	2010	✓	×	✓	✓	✓	✓	✓	✓	×
MDCSim [99]	2009	×	×	✓	×	×	×	✓	✓	×
CloudSim [100]	2009	✓	×	✓	×	✓	✓	✓	✓	×

1—Publication Year, 2—Cost Model. 3—SLA Support, 4—Energy Model, 5—GUI Support, 6—Federation Model, 7—Open Source Availability, 8—Communication Model, 9—Platform Portability, 10—Congestion Control.

7. Performance evaluation

This section represents the results and discuss the quantitative analysis of various existing metaheuristic scheduling algorithms in cloud environment and an elaborated comparison is provided on the basis of scheduling parameters like makespan and resource utilization cost. The state-of-the-art metaheuristic scheduling algorithms in cloud environment employed for comparison are PSO, PeSOA, CSA, ACO, GA and ABC. Even though there are multiple improved versions of ABC, ACO, PeSOA, ABC, CSA and GA are available in the literature. But in this paper, their standard versions are used for performing experiments. In this section, the comparative analysis of various existing metaheuristic scheduling algorithms is provided and the analysis has been performed using two scenarios.

7.1. Experimental setup

Cloudsim simulator has been used for quantitative analysis of various existing metaheuristic techniques [98,100] [101]. We also used CloudSim for simulating a cloud computing environment under two different scenarios. The scenarios use 70 physical machines and 100 VMs and the analysis is carried out based on the two different scenarios. The MIPS of VM varied from 5000 to 15 000 MIPS, processor from 1 to 100 and memory from 1 to 100.

7.2. Implementation details

The parameters for the algorithms under investigation are tuned by frequent experimentation since finding optimal value of these parameters is a NP-Hard problem itself. For ACO algorithm, the parameters α, β, τ and Q directly affect the computation results. This paper utilizes the optimal configuration of parameters i.e. $\alpha = \beta = 1$, $\tau = 0.5$ and $Q = 100$. For GA experimentation, the population size of 20 have been selected with bad individual 0.9 and crossover rate bandwidth being between 0.2 to 0.4. For PeSOA algorithm, diversification and intensification can be controlled by the parameters such as number of generations, population size, initial oxygen reserve etc. Hill-Climbing algorithm have been used to compute the near optimal value of these parameters for PeSOA experimentation. For CSA algorithm, the values of parameters are set to be as follows: Total no of crows = 100, stamina factor = 1.5 and r = [0,1] with awareness probability being 0.1 for all. The population size for ABC algorithm experimental setup was set to 100 and max iteration was 50. In PSO algorithm, both accelerate constants are set to be 2.05, whereas maximum and minimum inertia values are 0.9 and 0.4 respectively.

In cloud services, the major part of quality assurance can be improved through the latency efficiency. Distributed tracing systems like Google Traces can be used to collect the data generated by various applications. In our experimentation, the historical data of latency records provide by Google traces have been utilized. Instead of using random data, this approach had been followed for achieving realistic effectiveness of metaheuristic techniques.

7.3. Simulation results

The experiments on various state-of-the-art metaheuristic scheduling techniques have been carried out in cloud environment. This section portrays the results of the experimentation performed. The objective of the simulation is to compute the makespan of various metaheuristic scheduling algorithms under test. The algorithm that will possess minimum makespan is considered to be the optimal metaheuristic scheduling algorithm. The performance of various metaheuristic scheduling algorithms is calculated using following different test scenarios.

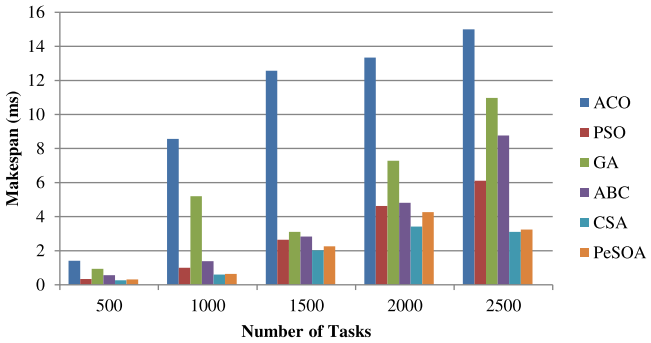

Fig. 5. Graphical representation of makespan of different metaheuristic scheduling techniques on varied set of tasks.

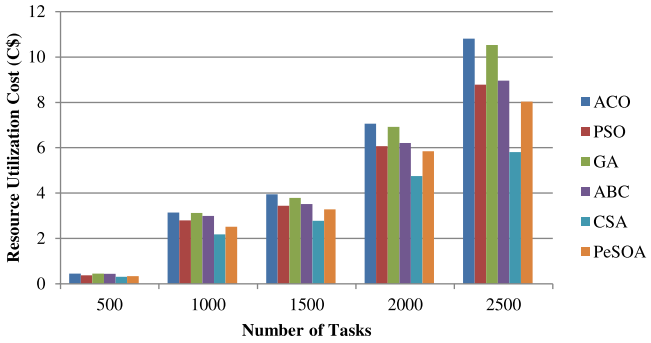

Fig. 6. Graphical representation of resource utilization cost of different metaheuristic scheduling techniques on varied set of tasks.

7.3.1. Scenario 1: Experimentation with task set size varied from 500 to 2500 tasks with fixed number of iterations i.e. 40

During experimentation, the task set size has been varied from 500 to 2500 tasks, the values of makespan for each metaheuristic algorithm under test has been calculated and represented graphically in Fig. 5. The makespan is measured in terms of milliseconds (ms) and resource utilization cost is measured in terms of Cloud Dollars (C$) [100]. The graphical representation clearly depicts the variations of makespan with the increasing number of tasks.

When the techniques under investigation were implemented for the minimum value of task set size and fixed number of iterations i.e. 40, the value of makespan for CSA, PeSOA, PSO, ABC, GA and ACO was 0.26789559, 0.3107328675, 0.3396585325, 0.5642057725, 0.937479035 and 1.407698615, respectively. From the graphical representation, it is evident that the CSA algorithm has emerged as the most optimal metaheuristic algorithm in contrast to other considered state-of-the-art algorithms.

Whereas, for the maximum task set size, the value of makespan for the ACO, PSO, GA, ABC, CSA and PeSOA algorithms was 15, 6.11661085, 10.9737904, 8.7644275, 3.10493175 and 3.2421324, respectively. It is observed that the CSA algorithm has outrun all the other algorithms and has achieved minimum makespan, hence it has turned out to be the most optimal metaheuristic algorithm. Thus, in case of makespan determined on varied task set size, the resultant optimal pattern for the algorithms under investigation is $CSA > PeSOA > PSO > ABC > GA > ACO$.

Furthermore, the aim of the simulation is to compute the resource utilization cost of various metaheuristic algorithms under test. The algorithm running with the minimum resource utilization cost is considered to be the most optimal metaheuristic scheduling algorithm. The graphical representation shown in Fig. 6 clearly reflects the variation of resource utilization cost with the increasing number of tasks.

When the algorithms under investigation were implemented for the minimum value of task set size and fixed number of iterations i.e. 40, the value of resource utilization cost for CSA, PeSOA, PSO, ABC, GA and ACO was 0.31011076, 0.334315355, 0.374337445, 0.437624605, 0.4494047515 and 0.4501328475, respectively. From the graphical representation, it is evident that the CSA algorithm has emerged as the most optimal metaheuristic algorithm in contrast to other considered state-of-the-art algorithms.

Whereas, when the task set size value considered for experimentation was taken as maximum, the value of resource utilization cost for the ACO, PSO, GA, ABC, CSA and PeSOA algorithms was 10.81321875, 8.7769473, 10.5269793, 8.9610782, 5.8065959 and 8.03324605, respectively. It is noted from the resource utilization cost values, that the CSA algorithm has outrun all the other algorithms and has achieved minimum resource utilization cost, hence it has turned out to be the most optimal metaheuristic algorithm. So, in case of resource utilization cost calculated on varied task set size, the resultant optimal pattern for the algorithms under test is $CSA > PeSOA > PSO > ABC > GA > ACO$.

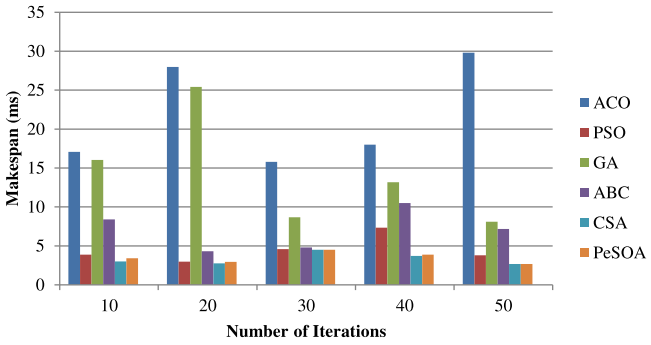

Fig. 7. Graphical representation of comparative analysis of different metaheuristic scheduling algorithms based on makespan using setup 1.

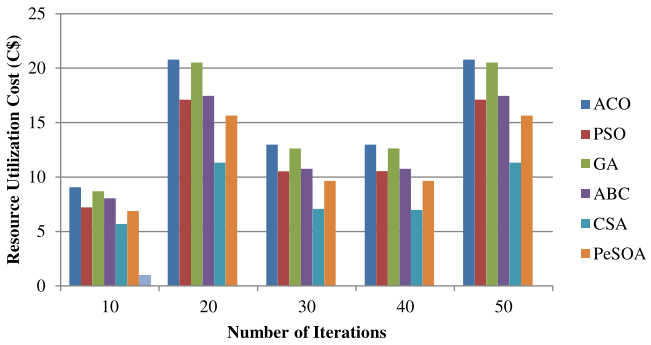

Fig. 8. Graphical representation of comparative analysis of different metaheuristic scheduling algorithms based on resource utilization cost using setup 1.

7.3.2. Scenario 2: Experimentation with fixed task set size of 3000 and 5000 tasks with the number of iterations varied from 10 to 50

To determine most optimal scheduling algorithm among various existing metaheuristic scheduling algorithms, makespan and resource utilization cost will be calculated. The experimentation has been performed using two different simulation setups.

- **Setup 1 with task set size of 3000:** The scheduling algorithm which exhibits the minimum makespan, is considered to be most optimal algorithm. The graphical representation of comparative analysis of different metaheuristic scheduling algorithms based on makespan for setup 1 with task set size of 3000 tasks has been portrayed in Fig. 7. When the number of iterations were varied from 10 to 50, it was observed that CSA algorithm is the most optimal in all the cases as compared to other considered metaheuristic algorithms.
  Further, the metaheuristic algorithm also exhibiting the minimum resource utilization cost will be considered as the most optimal one. The pictorial representation of the analysis of different metaheuristic algorithms based on resource utilization cost for setup 1 with task set size of 3000 has been shown in Fig. 8. The CSA algorithm exhibits the minimum resource utilization cost for all the varied number of iterations (10 to 50), which is the most optimal, in contrast to other metaheuristic algorithms examined.

- **Setup 2 with task set size of 5000:** For setup 2 with task set size of 5000, the graphical representation of comparative analysis of different metaheuristic algorithms based on makespan has been displayed in Fig. 9. When the number of iterations were varied from 10 to 50, it was observed that CSA algorithm is again the most optimal in all the cases as compared to other existing metaheuristic algorithms tested in this section.
  Further, the pictorial representation of comparative analysis of different metaheuristic algorithms based on resource utilization cost for the task set size of 5000 has been displayed in Fig. 10. The CSA algorithm exhibits the minimum resource utilization cost for all the varied number of iterations (10 to 50), which is the most optimal, in contrast to other metaheuristic algorithms examined.

7.3.3. Comparative discussion

It has been observed from the graphical representations that CSA metaheuristic algorithm reduces both makespan and resource utilization cost to minimum on the varied number of iterations, as compared to existing methods like ACO, PSO, GA, ABC and PeSOA. So, the CSA metaheuristic algorithm turns out to be most optimal among various other metaheuristic algorithms considered for

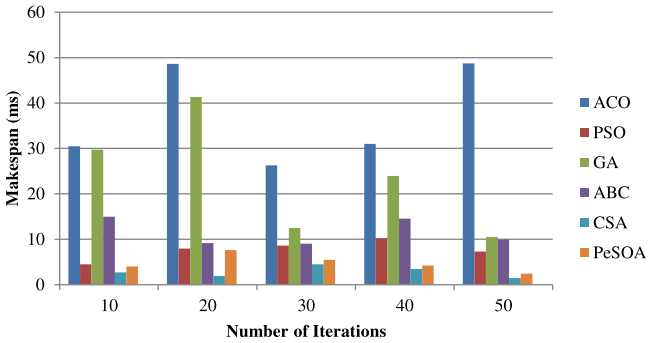

Fig. 9. Graphical representation of comparative analysis of different metaheuristic scheduling algorithms based on makespan using setup 2.

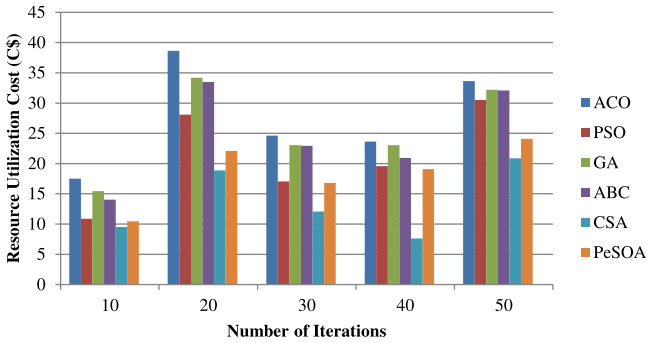

Fig. 10. Graphical representation of comparative analysis of different metaheuristic scheduling algorithms based on resource utilization cost using setup 2.

comparison. Further, the makespan and resource utilization cost of the PeSOA algorithm is the second to minimum when compared with existing methods like ACO, PSO, GA, ABC and CSA on varied number of iterations (10 to 50). So PeSOA algorithm is proved to be the second optimal algorithm, whereas, ACO algorithm is proved to be the least optimal metaheuristic algorithm among PSO, GA, ABC, CSA and PeSOA metaheuristic algorithms. Further, it has been observed during experimentation that the resource cost of the metaheuristic algorithms under investigation itself is an important evaluation factor. So, the resource cost of each of the metaheuristic algorithm under investigation has been calculated and the comparative analysis has been represented graphically in Fig. 11. Resource cost is defined as the multiplication of price of a particular resource and its execution time [102]. The price of a resource is a constant amount represented in Cloud Dollars (C$) whereas execution time is the amount of time required to execute application successfully and is represented in seconds. During experimentation, it has been observed that the execution cost is increasing as the number of tasks submitted by the users increases for PeSOA, GA, ACO, ABC, PSO and CSA metaheuristic algorithms. The average value of resource cost in CSA is 4.59%, 14.08%, 25.43%, 38.64% and 42.54% less than PeSOA, PSO, ABC, GA and ACO algorithms respectively. Therefore, it is evident from the experiment that CSA consumes less cost as compared to other considered metaheuristic algorithms.

8. Conclusions and future directions

In this paper, the goal of systematic review is to showcase insights of different multi-criteria metaheuristic scheduling techniques in cloud environment. The six state-of-the-art metaheuristic techniques: CSA, ABC, GA, PeSOA, ACO, and PSO have been analysed quantitatively depending on scheduling parameters such as RUC and makespan. Before performing analysis, different cloud simulators have been compared based on various parameters to find out which is the best cloud simulator to carry on the experimentation. As a result, the CloudSim simulator has turned out as the best choice for performing experimentation in cloud environment. Thus, the comparative analysis of different metaheuristic techniques has been done using CloudSim simulator. During experimentation, the values of both makespan and resource utilization cost were obtained on varied task set size. The resultant optimal pattern for the algorithms under investigation was $CSA > PeSOA > PSO > ABC > GA > ACO$. The experimental results demonstrate that CSA provides effective outcomes as compared to other considered metaheuristic techniques. Thus, it is the most optimal metaheuristic scheduling algorithm among the examined techniques and PeSOA has figured out to be the second optimal metaheuristic scheduling algorithm. Finally, various promising future research directions have been proposed.

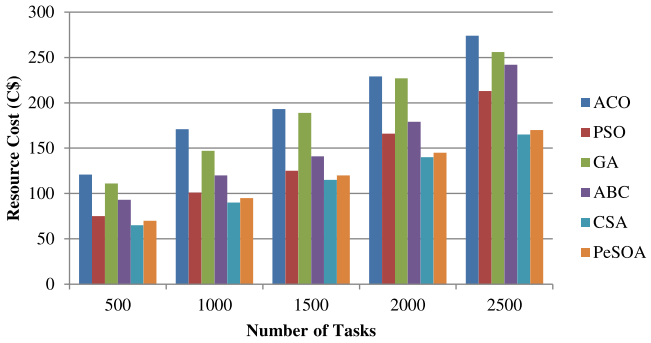

Fig. 11. Graphical representation of comparative analysis of different metaheuristic scheduling algorithms based on resource cost.

8.1. Future research directions

In cloud environment, both the stakeholders i.e. cloud providers and consumers, can use the conclusive results obtained in this paper, in selecting the most optimal metaheuristic technique for executing their submitted applications on to the allocated resources. We have identified various promising future research directions based on the analysis, which are discussed below:

- **Hybridization:** The objective of hybridization concept is to harness the advantages of each algorithm used in making hybrid algorithm [103,104]. Based on the analysis done in this paper, in future, it is proposed to integrate the CSA with PeSOA and develop a new algorithm named as CPO (Crow Penguin Optimizer) for finding the new solution for multi-criteria scheduling optimization problem [105]. The performance of CPO algorithm can be evaluated and compared with the existing standard versions of metaheuristic techniques like ACO, PSO, GA, ABC, traditional CSA and PeSOA in terms of QoS, resource utilization cost, makespan and load based on most recent data extracted from google traces.
- **Industry 4.0:** Task scheduling algorithms using cloud computing architecture are useful in Industry 4.0-based applications as well. Industry 4.0 standards includes cognitive computing, artificial intelligence, cloud computing [106]. Cyber–physical systems and IoT offer easy go and efficient solutions to handle industrial practices in manufacturing, supply chain and other sectors [106]. The tasks scheduling algorithms can improve the applicability of Industrial Internet of Things (IIoT) practices in these sectors, and it would be interesting to measure the statistics and explore the possibilities in different specialized sectors [107].
- **Security:** Blockchain is an emerging technology and in future it can be used for ensuring the security of the resources in the cloud environment. However, as cloud service providers are handling the responsibility of managing the cloud resources, they can use Blockchain technology for tracing the usage and provisioning of cloud resources as per the demand of consumers [108].
- **Software-Defined Network (SDN):** In future, the virtualization concept in cloud computing can be applied in more secure and effective way by enabling the virtualization in the physical layer of the cloud computing architecture. The SDN paradigm when combined with cloud computing paradigm will minimize the consumption of power and improves the usage of the network while executing the user's applications [109].
- **Artificial Intelligence and Machine Learning:** Effective artificial intelligence and machine learning techniques can be employed to predict the future workload on available resources [2]. Users can do predictive analysis using artificial intelligence and machine learning techniques to forecast the optimal value of control parameters of metaheuristic techniques deployed in their optimization problem [110].
- **IoT Applications:** Tasks distribution over reduced number of computing resources is useful to various IoT applications including healthcare, agriculture, traffic management. In healthcare, tasks (like patient's historical data processing, data sharing, secure data storage and retrieval, doctor's specialization, and success rate, staff rating) can be processed in parallel using advanced task scheduling algorithms. Similarly, soil fertility measurement and environment conditions detection using sensors, data collection, processing, statistics generation and visualization are important IoT and cloud computing-based application in agriculture domain. In traffic management, automated image-based vehicle detection, on-road rule violation detection and other traffic engineering tasks requires a lot of heavy computational jobs that can be scheduled easily using task scheduling algorithms over cloud infrastructure. Thus, these applications and their performance analysis can be measured using discussed algorithms in future [111].
- **Mobile Cloud Computing:** The mobile edge computing paradigm is the recent trend in the field of computing. The latency should be less, as far as the gaming applications submitted by the users is concerned. The mobile edge computing enables users to run their applications on the provisioned resources efficiently with less communication delays [112].
- **Quantum Computing:** Quantum computing can help and transform the processes and techniques used in the field of machine learning for performing predictive analysis on finding the optimal value of control parameters which will further let the metaheuristic algorithm to converge to a global optimal solution in large search space [113,114].

- **Serverless Edge Computing:** The concept of Serverless Edge Computing can be utilized to improve the scalability and reduce the computing cost while processing the incoming requests from various IoT or edge devices [115]. Further, the popular technologies such as Quantum Computing and Blockchain can be used along with Serverless Edge Computing to improve computational speed and security [113,115].
- **Experimental Environments:** IoT, Edge and Fog computing came into existence with the advancements in the cloud computing paradigm. In future, more advanced simulators such as iFogSim [116], FogNetSim++ [117] and iThermoFog [118] are required to test and validate the scheduling techniques developed in these computing paradigms [59,81].
- **Trust and Privacy:** There are various research challenges are emerging as the computing paradigm shifts is going from personal computing to cloud computing. [2,119]. To facilitate this paradigm shift, there is a need for the development of new trust and privacy models developed for cloud computing to offer a safe environment to all the stakeholders. Therefore, trust and privacy constraints must be consider for developing of new resource scheduling algorithms to maximize resource utilization dynamically.
- **5G/6G:** There is a need to adopt fifth/sixth (5G/6G) generation mobile telecommunication services to provide the ability for fast transmission of data with minimum latency and energy consumption [120]. In future, the real time mission-critical and sophisticated applications can utilize 5G/6G technologies to provide the robust communication during resource sharing [121].

CRediT authorship contribution statement

Harvinder Singh: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Visualization, Validation, Writing - original draft, Writing - review & editing. **Sanjay Tyagi:** Writing - original draft, Conceptualization, Data curation, Methodology, Software, Validation, Supervision. **Pardeep Kumar:** Writing - original draft, Conceptualization, Data curation, Methodology, Software, Validation, Supervision. **Sukhpal Singh Gill:** Supervision, Visualization, Methodology, Writing - review & editing. **Rajkumar Buyya:** Supervision, Visualization, Methodology, Writing - review & editing.

Acknowledgements

We would like to thank the Editor-in-Chief (Prof. Helen Karatza), area editor and anonymous reviewers for their valuable comments and suggestions to help and improve our research paper.

References

[1] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, Ivona Brandic, Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility, Future Gener. Comput. Syst. 25 (6) (2009) 599–616.
[2] Sukhpal Singh Gill, Shreshth Tuli, Minxian Xu, Inderpreet Singh, Karan Vijay Singh, Dominic Lindsay, Shikhar Tuli, Daria Smirnova, Manmeet Singh, Udit Jain, et al., Transformative effects of IoT, blockchain and artificial intelligence on cloud computing: Evolution, vision, trends and open challenges, Internet Things 8 (2019) 100118.
[3] Atul Vikas Lakra, Dharmendra Kumar Yadav, Multi-objective tasks scheduling algorithm for cloud computing throughput optimization, Procedia Comput. Sci. 48 (C) (2015) 107–113, http://dx.doi.org/10.1016/j.procs.2015.04.158, http://dx.doi.org/10.1016/j.procs.2015.04.158.
[4] Amit Kumar Bhardwaj, Yuvraj Gajpal, Chirag Surti, Sukhpal Singh Gill, HEART: Unrelated parallel machines problem with precedence constraints for task scheduling in cloud computing using heuristic and meta-heuristic algorithms, Softw. - Pract. Exp. (2020).
[5] Michele Carillo, Gennaro Cordasco, Flavio Serrapica, Vittorio Scarano, Carmine Spagnuolo, Przemysław Szufel, Distributed simulation optimization and parameter exploration framework for the cloud, Simul. Model. Pract. Theory 83 (2018) 108–123, http://dx.doi.org/10.1016/j.simpat.2017.12.005.
[6] Mala Kalra, Sarbjeet Singh, A review of metaheuristic scheduling techniques in cloud computing, Egypt. Inform. J. 16 (3) (2015) 275–295, http://dx.doi.org/10.1016/j.eij.2015.07.001.
[7] M.X. Xu, R. Buyya, Brownout approach for adaptive management of resources and applications in cloud computing systems: A taxonomy and future directions, ACM Comput. Surv. 52 (1) (2019) 27, http://dx.doi.org/10.1145/3234151.
[8] S. Singh, I. Chana, A survey on resource scheduling in cloud computing: Issues and challenges, J. Grid Comput. 14 (2) (2016) 217–264, http://dx.doi.org/10.1007/s10723-015-9359-2.
[9] S.S. Gill, R. Buyya, A taxonomy and future directions for sustainable cloud computing: 360 degree view, ACM Comput. Surv. 51 (5) (2019) 33, http://dx.doi.org/10.1145/3241038.
[10] A. Hameed, A. Khoshkbarforoushha, R. Ranjan, P.P. Jayaraman, J. Kolodziej, P. Balaji, S. Zeadally, Q.M. Malluhi, N. Tziritas, A. Vishnu, S.U. Khan, A. Zomaya, A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems, Computing 98 (7) (2016) 751–774, http://dx.doi.org/10.1007/s00607-014-0407-8.
[11] A. Beloglazov, R. Buyya, Y.C. Lee, A. Zomaya, A taxonomy and survey of energy-efficient data centers and cloud computing systems, in: M.V. Zelkowitz (Ed.), Advances in Computers, Vol. 82, in: Advances in Computers, vol. 82, Elsevier Academic Press Inc, San Diego, 2011, pp. 47–111, http://dx.doi.org/10.1016/b978-0-12-385512-1.00003-7.
[12] Ilhem Boussaid, Julien Lepagnot, Patrick Siarry, A survey on optimization metaheuristics, Inform. Sci. 237 (2013) 82–117, http://dx.doi.org/10.1016/j.ins.2013.02.041.
[13] C.W. Tsai, Jjpc Rodrigues, Metaheuristic scheduling for cloud: A survey, IEEE Syst. J. 8 (1) (2014) 279–291, http://dx.doi.org/10.1109/jsyst.2013.2256731.
[14] Rajkumar Buyya, Satish Narayana Srirama, Giuliano Casale, Rodrigo Calheiros, Yogesh Simmhan, Blesson Varghese, Erol Gelenbe, Bahman Javadi, Luis Miguel Vaquero, Marco A.S. Netto, Adel Nadjaran Toosi, Maria Alejandra Rodriguez, Ignacio M. Llorente, Sabrina De Capitani Di Vimercati, Pierangela Samarati, Dejan Milojicic, Carlos Varela, Rami Bahsoon, Marcos Dias De Assuncao, Omer Rana, Wanlei Zhou, Hai Jin, Wolfgang Gentzsch, Albert Y. Zomaya, Haiying Shen, A manifesto for future generation cloud computing: Research directions for the next decade, ACM Comput. Surv. 51 (5) (2019) http://dx.doi.org/10.1145/3241737.
[15] M. Kumar, S.C. Sharma, A. Goel, S.P. Singh, A comprehensive survey for scheduling techniques in cloud computing, J. Netw. Comput. Appl. 143 (2019) 1–33, http://dx.doi.org/10.1016/j.jnca.2019.06.006.
[16] T.O. Ting, Xin-She Yang, Shi Cheng, Kaizhu Huang, Hybrid metaheuristic algorithms: past, present, and future, in: Recent Advances in Swarm Intelligence and Evolutionary Computation, Springer, 2015, pp. 71–83.

[17] A.R. Arunarani, D. Manjula, V. Sugumaran, Task scheduling techniques in cloud computing: A literature survey, Future Gener. Comput. Syst. Int. J. Escience 91 (2019) 407–415, http://dx.doi.org/10.1016/j.future.2018.09.014.

[18] Harvinder Singh, Sanjay Tyagi, Pardeep Kumar, Scheduling in cloud computing environment using metaheuristic techniques: A survey, in: Advances in Intelligent Systems and Computing, Vol. 937, 2020, pp. 753–763, http://dx.doi.org/10.1007/978-981-13-7403-6_66.

[19] Vijindra, Sudhir Shenai, Survey on scheduling issues in cloud computing, Procedia Eng. 38 (2012) 2881–2888, http://dx.doi.org/10.1016/j.proeng.2012.06.337.

[20] M. Lavanya, B. Shanthi, S. Saravanan, Multi objective task scheduling algorithm based on SLA and processing time suitable for cloud environment, Comput. Commun. 151 (2020) 183–195, http://dx.doi.org/10.1016/j.comcom.2019.12.050.

[21] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, A view of cloud computing, Commun. ACM 53 (4) (2010) 50–58, http://dx.doi.org/10.1145/1721654.1721672.

[22] L.C. Canon, A.K.W. Chang, Y. Robert, F. Vivien, Scheduling independent stochastic tasks under deadline and budget constraints, Int. J. High Perform. Comput. Appl. 34 (2) (2020) 246–264, http://dx.doi.org/10.1177/1094342019852135.

[23] Rajkumar Buyya, Manzur Murshed, David Abramson, Srikumar Venugopal, Scheduling parameter sweep applications on global grids: A deadline and budget constrained cost-time optimization algorithm, Softw. Pract. Exper. 35 (5) (2005) 491–512, http://dx.doi.org/10.1002/spe.646, https://doi.org/10.1002/spe.646.

[24] Yongkui Liu, Xun Xu, Lin Zhang, Long Wang, Ray Y Zhong, Workload-based multi-task scheduling in cloud manufacturing, Robot. Comput.-Integr. Manuf. 45 (September 2016) (2017) 3–20, http://dx.doi.org/10.1016/j.rcim.2016.09.008.

[25] Harvinder Singh, Sanjey Tyagi, Pardeep Kumar, Energy-aware virtual machine selection and allocation strategies in cloud data centers, in: 2018 5th International Conference on Parallel, Distributed and Grid Computing, PDGC 2018, 2018, pp. 312–317, http://dx.doi.org/10.1109/PDGC.2018.8745764.

[26] Harvinder Singh, Sanjay Tyagi, Pardeep Kumar, Crow search based scheduling algorithm for load balancing in cloud environment, Int. J. Innov. Technol. Explor. Eng. 8 (9) (2019) 1058–1064, http://dx.doi.org/10.35940/ijitee.I7787.078919.

[27] Feng Ye, Zhijian Wang, Feng Xu, Yuanchao Zhou, Fachao Zhou, Shaosong Yang, A novel cloud load balancing mechanism in premise of ensuring QOS, Intell. Autom. Soft Comput. 19 (2) (2013) 151–163, http://dx.doi.org/10.1080/10798587.2013.786968.

[28] P. Kumar, R. Kumar, Issues and challenges of load balancing techniques in cloud computing: A survey, ACM Comput. Surv. 51 (6) (2019) 35, http://dx.doi.org/10.1145/3281010.

[29] S. Singh, I. Chana, QoS-aware autonomic resource management in cloud computing: A systematic review, ACM Comput. Surv. 48 (3) (2015) 46, http://dx.doi.org/10.1145/2843889.

[30] L.Y. Zuo, L. Shu, S.B. Dong, Y.F. Chen, L. Yan, A multi-objective hybrid cloud resource scheduling method based on deadline and cost constraints, IEEE Access 5 (2017) 22067–22080, http://dx.doi.org/10.1109/access.2016.2633288.

[31] Sukhpal Singh Gill, Rajkumar Buyya, Inderveer Chana, Maninder Singh, Ajith Abraham, BULLET: Particle swarm optimization based scheduling technique for provisioned cloud resources, J. Netw. Syst. Manage. 26 (2) (2018) 361–400, http://dx.doi.org/10.1007/s10922-017-9419-y.

[32] N. Netjinda, B. Sirinaovakul, T. Achalakul, Cost optimal scheduling in IaaS for dependent workload with particle swarm optimization, J. Supercomput. 68 (3) (2014) 1579–1603, http://dx.doi.org/10.1007/s11227-014-1126-9.

[33] Thamarai Selvi Somasundaram, Kannan Govindarajan, CLOUDRB: A framework for scheduling and managing high-performance computing (HPC) applications in science cloud, Future Gener. Comput. Syst. Int. J. Escience 34 (2014) 47–65, http://dx.doi.org/10.1016/j.future.2013.12.024.

[34] Sonia Yassa, Rachid Chelouah, Hubert Kadima, Bertrand Granado, Multi-objective approach for energy-aware workflow scheduling in cloud computing environments, Sci. World J. (2013) http://dx.doi.org/10.1155/2013/350934.

[35] P.W. Wang, Y.H. Lei, P.R. Agbedanu, Z.H. Zhang, Makespan-driven workflow scheduling in clouds using immune-based pso algorithm, IEEE Access 8 (2020) 29281–29290, http://dx.doi.org/10.1109/access.2020.2972963.

[36] M. Salimi, A. Majd, M. Loni, T. Seceleanu, C. Seceleanu, M. Sirjani, M. Daneshtalab, E. Troubitsyna, Multi-objective optimization of real-time task scheduling problem for distributed environments, in: Proceedings of the 6th Conference on the Engineering of Computer Based Systems, Assoc Computing Machinery, New York, 2020, http://dx.doi.org/10.1145/3352700.3352713.

[37] Sukhpal Singh Gill, Rajkumar Buyya, Resource provisioning based scheduling framework for execution of heterogeneous and clustered workloads in clouds: from fundamental to autonomic offering, J. Grid Comput. 17 (3) (2019) 385–417, http://dx.doi.org/10.1007/s10723-017-9424-0.

[38] Mohamed Abd Elaziz, Shengwu Xiong, K.P.N. Jayasena, Lin Li, Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution, Knowl.-Based Syst. 169 (2019) 39–52, http://dx.doi.org/10.1016/j.knosys.2019.01.023.

[39] Mainak Adhikari, Sudiirshan Nandy, Tarachand Amgoth, Meta heuristic-based task deployment mechanism for load balancing in IaaS cloud, J. Netw. Comput. Appl. 128 (2019) 64–77, http://dx.doi.org/10.1016/j.jnca.2018.12.010.

[40] N. Almezeini, A. Hafez, Task scheduling in cloud computing using lion optimization algorithm, Int. J. Adv. Comput. Sci. Appl. 8 (11) (2017) 77–83.

[41] R.K. Jena, Task scheduling in cloud environment: A multi-objective ABC framework, J. Inf. Optim. Sci. 38 (1) (2017) 1–19, http://dx.doi.org/10.1080/02522667.2016.1250460.

[42] S. Mirjalili, A. Lewis, The whale optimization algorithm, Adv. Eng. Softw. 95 (2016) 51–67, http://dx.doi.org/10.1016/j.advengsoft.2016.01.008.

[43] Elina Pacini, Cristian Mateos, Carlos Garcia Garino, Balancing throughput and response time in online scientific clouds via ant colony optimization (SP2013/2013/00006), Adv. Eng. Softw. 84 (2015) 31–47, http://dx.doi.org/10.1016/j.advengsoft.2015.01.005.

[44] Gang Zhao, Cost-aware scheduling algorithm based on PSO in cloud computing environment, Int. J. Grid Distrib. Comput. 7 (1) (2014) 33–42, http://dx.doi.org/10.14257/ijgdc.2014.7.1.04.

[45] F. Ramezani, J. Lu, F.K. Hussain, Task-based system load balancing in cloud computing using particle swarm optimization, Int. J. Parallel Program. 42 (5) (2014) 739–754, http://dx.doi.org/10.1007/s10766-013-0275-4.

[46] Dhinesh L.D. Babu, P. Venkata Krishna, Honey bee behavior inspired load balancing of tasks in cloud computing environments, Appl. Soft Comput. 13 (5) (2013) 2292–2303, http://dx.doi.org/10.1016/j.asoc.2013.01.025.

[47] K. Dasgupta, B. Mandal, P. Dutta, J.K. Mondal, S. Dam, A genetic algorithm (GA) based load balancing strategy for cloud computing, in: First International Conference on Computational Intelligence: Modeling Techniques and Applications (Cimta) 2013, Vol. 10, 2013, pp. 340–347, http://dx.doi.org/10.1016/j.protcy.2013.12.369.

[48] W. Venters, E.A. Whitley, A critical review of cloud computing: researching desires and realities, J. Inf. Technol. 27 (3) (2012) 179–197, http://dx.doi.org/10.1057/jit.2012.17.

[49] I. Chopra, M. Singh, SHAPE-an approach for self-healing and self-protection in complex distributed networks, J. Supercomput. 67 (2) (2014) 585–613, http://dx.doi.org/10.1007/s11227-013-1019-3.

[50] M. Sheikhalishahi, L. Grandinetti, R.M. Wallace, J.L. Vazquez-Poletti, Autonomic resource contention-aware scheduling, Softw.-Pract. Exp. 45 (2) (2015) 161–175, http://dx.doi.org/10.1002/spe.2223.

[51] B. Vasumathi, S. Moorthi, Implementation of hybrid ANN–PSO algorithm on FPGA for harmonic estimation, Eng. Appl. Artif. Intell. 25 (3) (2012) 476–483, http://dx.doi.org/10.1016/j.engappai.2011.12.005.

[52] L.Y. Zuo, L. Shu, S.B. Dong, C.S. Zhu, T. Hara, A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing, IEEE Access 3 (2015) 2687–2699, http://dx.doi.org/10.1109/access.2015.2508940.

[53] M. Clerc, J. Kennedy, The particle swarm - explosion, stability, and convergence in a multidimensional complex space, IEEE Trans. Evol. Comput. 6 (1) (2002) 58–73, http://dx.doi.org/10.1109/4235.985692.

[54] A. Konak, D.W. Coit, A.E. Smith, Multi-objective optimization using genetic algorithms: A tutorial, Reliab. Eng. Syst. Saf. 91 (9) (2006) 992–1007, http://dx.doi.org/10.1016/j.ress.2005.11.018.

[55] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Global Optim. 39 (3) (2007) 459–471, http://dx.doi.org/10.1007/s10898-007-9149-x.

[56] Alireza Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm, Comput. Struct. 169 (2016) 1–12, http://dx.doi.org/10.1016/j.compstruc.2016.03.001, arXiv:1708.01368.

[57] Youcef Gheraibia, Abdelouahab Moussaoui, Penguins search optimization algorithm (PeSOA), Lecture Notes in Comput. Sci. 7906 LNAI (2013) 222–231, http://dx.doi.org/10.1007/978-3-642-38577-3_23.

[58] Z.A. Mann, Allocation of virtual machines in cloud data centers-A survey of problem models and optimization algorithms, ACM Comput. Surv. 48 (1) (2015) 34, http://dx.doi.org/10.1145/2797211.

[59] Mohammad S. Aslanpour, Sukhpal Singh Gill, Adel N. Toosi, Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research, Internet Things (2020) 100273.

[60] Sukhpal Singh, Inderveer Chana, QRSF: QoS-aware resource scheduling framework in cloud computing, J. Supercomput. 71 (1) (2014) 241–292, http://dx.doi.org/10.1007/s11227-014-1295-6.

[61] Hend Gamal El Din Hassan Ali, Imane Aly Saroit, Amira Mohamed Kotb, Grouped tasks scheduling algorithm based on QoS in cloud computing network, Egypt. Inform. J. 18 (1) (2017) 11–19, http://dx.doi.org/10.1016/j.eij.2016.07.002.

[62] P. Chitra, P. Venkatesh, R. Rajaram, Comparison of evolutionary computation algorithms for solving bi-objective task scheduling problem on heterogeneous distributed computing systems, Sadhana - Acad. Proc. Eng. Sci. 36 (2) (2011) 167–180, http://dx.doi.org/10.1007/s12046-011-0014-8.

[63] Yu Xin, Zhi-Qiang Xie, Jing Yang, A load balance oriented cost efficient scheduling method for parallel tasks, J. Netw. Comput. Appl. 81 (2017) 37–46, http://dx.doi.org/10.1016/j.jnca.2016.12.032, http://linkinghub.elsevier.com/retrieve/pii/S1084804516303496.

[64] Teena Mathew, K. Chandra Sekaran, John Jose, Study and analysis of various task scheduling algorithms in the cloud computing environment, in: Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on, IEEE, 2014, pp. 658–664.

[65] S.R. Shishira, A. Kandasamy, K. Chandrasekaran, Survey on meta heuristic optimization techniques in cloud computing, in: 2016 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2016, 2016, pp. 1434–1440, http://dx.doi.org/10.1109/ICACCI.2016.7732249.

[66] V.K. Manupati, G. Rajyalakshmi, Felix T.S. Chan, J.J. Thakkar, A hybrid multi-objective evolutionary algorithm approach for handling sequence- and machine-dependent set-up times in unrelated parallel machine scheduling problem, Sadhana - Acad. Proc. Eng. Sci. 42 (3) (2017) 391–403, http://dx.doi.org/10.1007/s12046-017-0611-2.

[67] Yu Liu, Changjie Zhang, Bo Li, Jianwei Niu, DeMS: A hybrid scheme of task scheduling and load balancing in computing clusters, J. Netw. Comput. Appl. 83 (2017) 213–220, http://dx.doi.org/10.1016/j.jnca.2015.04.017.

[68] Jyoti Malhotra, Jagdish Bakal, Second order mutual information based grey wolf optimization for effective storage and de-duplication, Sadhana - Acad. Proc. Eng. Sci. 43 (11) (2018) 34–37, http://dx.doi.org/10.1007/s12046-018-0939-2, https://doi.org/10.1007/s12046-018-0939-2.

[69] T. Vairam, S. Sarathambekai, K. Umamaheswari, Multiprocessor task scheduling problem using hybrid discrete particle swarm optimization, Sadhana - Acad. Proc. Eng. Sci. 43 (12) (2018) http://dx.doi.org/10.1007/s12046-018-0984-x.

[70] Hua He, Guangquan Xu, Shanchen Pang, Zenghua Zhao, AMTS: Adaptive multi-objective task scheduling strategy in cloud computing, China Commun. 13 (4) (2016) 162–171, http://dx.doi.org/10.1109/CC.2016.7464133.

[71] S. Pandey, L. Wu, S.M. Guru, R. Buyya, A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments, in: 2010 24th IEEE International Conference on Advanced Information Networking and Applications, 2010, pp. 400–407, http://dx.doi.org/10.1109/AINA.2010.31.

[72] Zhongjin Li, Jidong Ge, Hongji Yang, Liguo Huang, Haiyang Hu, Hao Hu, Bin Luo, A security and cost aware scheduling algorithm for heterogeneous tasks of scientific workflow in clouds, Future Gener. Comput. Syst. 65 (2016) 140–152, http://dx.doi.org/10.1016/j.future.2015.12.014, http://dx.doi.org/10.1016/j.future.2015.12.014.

[73] Amandeep Verma, Sakshi Kaushal, A hybrid multi-objective particle swarm optimization for scientific workflow scheduling, Parallel Comput. 62 (2017) 1–19, http://dx.doi.org/10.1016/j.parco.2017.01.002.

[74] Xingquan Zuo, Guoxiang Zhang, Wei Tan, Self-adaptive learning pso-based deadline constrained task scheduling for hybrid IaaS cloud, IEEE Trans. Autom. Sci. Eng. 11 (2) (2014) 564–573, http://dx.doi.org/10.1109/TASE.2013.2272758.

[75] Imran Ali Chaudhry, Abdul Munem Khan, Minimizing makespan for a no-wait flowshop using genetic algorithm, Sadhana - Acad. Proc. Eng. Sci. 37 (6) (2012) 695–707, http://dx.doi.org/10.1007/s12046-012-0105-1.

[76] G. Subashini, M.C. Bhuvaneswari, Comparison of multi-objective evolutionary approaches for task scheduling in distributed computing systems, Sadhana-Acad. Proc. Eng. Sci. 37 (6) (2012) 675–694, http://dx.doi.org/10.1007/s12046-012-0102-4.

[77] Xiaodong Sheng, Qiang Li, Template-based genetic algorithm for QoS-aware task scheduling in cloud computing, in: 2016 International Conference on Advanced Cloud and Big Data, 2016, pp. 0–5, http://dx.doi.org/10.1109/CBD.2016.37.

[78] Dorothea Heiss-Czedik, An introduction to genetic algorithms, Artif. Life 3 (1) (2009) 63–65, http://dx.doi.org/10.1162/artl.1997.3.63, arXiv:arXiv:1011.1669v3.

[79] Bennedeto Andrea, Andrea Bennedeto, Nuevas alternativas para pensar el desarrollo de los territorios rurales. Posibilidades y riesgos 1, Cuadernos Desarrollo Rural 57 (57) (2006) 101–131, http://dx.doi.org/10.1109/SCIS, arXiv:1312.2709.

[80] L.D. Dhinesh Babu, P. Venkata Krishna, Honey bee behavior inspired load balancing of tasks in cloud computing environments, Appl. Soft Comput. 13 (5) (2013) 2292–2303, http://dx.doi.org/10.1016/j.asoc.2013.01.025.

[81] Wenhong Tian, Minxian Xu, Aiguo Chen, Guozhong Li, Xinyang Wang, Yu Chen, Open-source simulators for cloud computing: Comparative study and challenging issues, Simul. Model. Pract. Theory 58 (2015) 239–254.

[82] Gokcecicek Tasoglu, Gokalp Yildiz, Simulated annealing based simulation optimization method for solving integrated berth allocation and quay crane scheduling problems, Simul. Model. Pract. Theory 97 (2019) 101948.

[83] Markus Rabe, Maik Deininger, Angel A. Juan, Speeding up computational times in simheuristics combining genetic algorithms with discrete-event simulation, Simul. Model. Pract. Theory 103 (2020) 17, http://dx.doi.org/10.1016/j.simpat.2020.102089.

[84] N. Mansouri, R. Ghafari, B. Mohammad Hasani Zade, Cloud computing simulators: A comprehensive review, Simul. Model. Pract. Theory 104 (2020) 50, http://dx.doi.org/10.1016/j.simpat.2020.102144.

[85] Harvinder Singh, Sanjay Tyagi, Pardeep Kumar, Comparative analysis of various simulation tools used in a cloud environment for task-resource mapping, 2021, pp. 419–430, http://dx.doi.org/10.1007/978-981-15-7533-4_32.

[86] Sukhpal Singh Gill, Shreshth Tuli, Adel Nadjaran Toosi, Felix Cuadrado, Peter Garraghan, Rami Bahsoon, Hanan Lutfiyya, Rizos Sakellariou, Omer Rana, Schahram Dustdar, et al., ThermoSim: Deep learning based framework for modeling and simulation of thermal-aware resource management for cloud computing environments, J. Syst. Softw. (2020) 110596, http://dx.doi.org/10.1016/j.jss.2020.110596.

[87] Khaled Alwasel, Rodrigo N. Calheiros, Saurabh Garg, Rajkumar Buyya, Mukaddim Pathan, Dimitrios Georgakopoulos, Rajiv Ranjan, BigDataSDNSim: A simulator for analyzing big data applications in software-defined cloud data centers, Softw. - Pract. Exp. (2020) spe.2917, http://dx.doi.org/10.1002/spe.2917.

[88] Damian Fernandez-Cerero, Alejandro Fernandez-Montes, Agnieszka Jakobik, Joanna Kolodziej, Miguel Toro, SCORE: Simulator for cloud optimization of resources and energy consumption, Simul. Model. Pract. Theory 82 (2018) 160–173, http://dx.doi.org/10.1016/j.simpat.2018.01.004.

[89] Damian Fernández-Cerero, Agnieszka Jakóbik, Alejandro Fernández-Montes, Joanna Kołodziej, GAME-SCORE: Game-based energy-aware cloud scheduler and simulator for computational clouds, Simul. Model. Pract. Theory 93 (2019) 3–20, http://dx.doi.org/10.1016/j.simpat.2018.09.001.

[90] Marc Bux, Ulf Leser, Dynamiccloudsim: Simulating heterogeneity in computational clouds, Future Gener. Comput. Syst. 46 (2015) 85–99, http://dx.doi.org/10.1016/j.future.2014.09.007.

[91] W.H. Tian, Y. Zhao, M.X. Xu, Y.L. Zhong, X.S. Sun, A toolkit for modeling and simulation of real-time virtual machine allocation in a cloud data center, IEEE Trans. Autom. Sci. Eng. 12 (1) (2015) 153–161, http://dx.doi.org/10.1109/tase.2013.2266338.

[92] Andreas Kohne, Marc Spohr, Lars Nagel, Olaf Spinczyk, FederatedCloudSim: a SLA-aware federated cloud simulation framework, in: Proceedings of the 2nd International Workshop on CrossCloud Systems, 2014, pp. 1–5.

[93] Weiwei Chen, Ewa Deelman, Workflowsim: A toolkit for simulating scientific workflows in distributed environments, in: 2012 IEEE 8th International Conference on E-Science, IEEE, 2012, pp. 1–8.

[94] A. Nunez, J.L. Vazquez-Poletti, A.C. Caminero, G.G. Castane, J. Carretero, I.M. Llorente, iCanCloud: A flexible and scalable cloud infrastructure simulator, J. Grid Comput. 10 (1) (2012) 185–209, http://dx.doi.org/10.1007/s10723-012-9208-5.

[95] Sandeep KS Gupta, Rose Robin Gilbert, Ayan Banerjee, Zahra Abbasi, Tridib Mukherjee, Georgios Varsamopoulos, Gdcsim: A tool for analyzing green data center design and resource management techniques, in: 2011 International Green Computing Conference and Workshops, IEEE, 2011, pp. 1–8.

[96] Saurabh Kumar Garg, Rajkumar Buyya, Networkcloudsim: Modelling parallel applications in cloud simulations, in: 2011 Fourth IEEE International Conference on Utility and Cloud Computing, IEEE, 2011, pp. 105–113.

[97] Dzmitry Kliazovich, Pascal Bouvry, Samee Ullah Khan, GreenCloud: a packet-level simulator of energy-aware cloud computing data centers, J. Supercomput. 62 (3) (2012) 1263–1283.

[98] B. Wickremasinghe, R.N. Calheiros, R. Buyya, CloudAnalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications, in: 24th IEEE International Conference on Advanced Information Networking and Applications, AINA, 2010, pp. 446–452, http://dx.doi.org/10.1109/aina.2010.32.

[99] Seung-Hwan Lim, Bikash Sharma, Gunwoo Nam, Eun Kyoung Kim, Chita R Das, MDCSim: A multi-tier data center simulation, platform, in: 2009 IEEE International Conference on Cluster Computing and Workshops, IEEE, 2009, pp. 1–9.

[100] R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A. F. De Rose, R. Buyya, Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Softw.-Pract. Exp. 41 (1) (2011) 23–50, http://dx.doi.org/10.1002/spe.995.

[101] Tarun Goyal, Ajit Singh, Aakanksha Agrawa, Cloudsim: Simulator for cloud computing infrastructure and modeling, Procedia Eng. 38 (2012) 3566–3572, http://dx.doi.org/10.1016/j.proeng.2012.06.412.

[102] Sukhpal Singh Gill, Peter Garraghan, Vlado Stankovski, Giuliano Casale, Ruppa K Thulasiram, Soumya K Ghosh, Kotagiri Ramamohanarao, Rajkumar Buyya, Holistic resource management for sustainable and reliable cloud computing: An innovative solution to global challenge, J. Syst. Softw. 155 (2019) 104–129.

[103] Farzaneh Abazari, Morteza Analoui, Hassan Takabi, Song Fu, MOWS: multi-objective workflow scheduling in cloud computing based on heuristic algorithm, Simul. Model. Pract. Theory 93 (2019) 119–132, http://dx.doi.org/10.1016/j.simpat.2018.10.004.

[104] Hülya Güçdemir, Hasan Selim, Integrating simulation modelling and multi criteria decision making for customer focused scheduling in job shops, Simul. Model. Pract. Theory 88 (2018) 17–31, http://dx.doi.org/10.1016/j.simpat.2018.08.001.

[105] Harvinder Singh, Sanjay Tyagi, Pardeep Kumar, Crow–penguin optimizer for multiobjective task scheduling strategy in cloud computing, Int. J. Commun. Syst. 33 (14) (2020) e4467.

[106] Y.K. Teoh, S.S. Gill, A.K. Parlikad, IoT And fog computing based predictive maintenance model for effective asset management in industry 4.0 using machine learning, IEEE Internet Things J. (2021) 1, http://dx.doi.org/10.1109/JIOT.2021.3050441.

[107] M. Shamim Hossain, Ghulam Muhammad, Cloud-assisted industrial internet of things (iiot)–enabled framework for health monitoring, Comput. Netw. 101 (2016) 192–202.

[108] Minhaj Ahmad Khan, Khaled Salah, IoT security: Review, blockchain solutions, and open challenges, Future Gener. Comput. Syst. 82 (2018) 395–411.

[109] Diego Kreutz, Fernando MV Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, Steve Uhlig, Software-defined networking: A comprehensive survey, Proc. IEEE 103 (1) (2014) 14–76.

[110] Sabri Bicakci, Huseyin Gunes, Hybrid simulation system for testing artificial intelligence algorithms used in smart homes, Simul. Model. Pract. Theory 102 (2020) 101993.

[111] In Lee, Kyoochun Lee, The internet of things (IoT): Applications, investments, and challenges for enterprises, Bus. Horiz. 58 (4) (2015) 431–440.

[112] Talal H Noor, Sherali Zeadally, Abdullah Alfazi, Quan Z Sheng, Mobile cloud computing: Challenges and future research directions, J. Netw. Comput. Appl. 115 (2018) 70–85.

[113] Sukhpal Singh Gill, Adarsh Kumar, Harvinder Singh, Kamalpreet Kaur, Muhammad Usman, Rajkumar Buyya, Quantum computing: A taxonomy, systematic review and future directions, 2020, http://www.buyya.com/papers/QuantumComputing-Taxonomy.pdf.

[114] Mihai Udrescu, Lucian Prodan, Mircea Vlăduţiu, Simulated fault injection methodology for gate-level quantum circuit reliability assessment, Simul. Model. Pract. Theory 23 (2012) 60–70.

[115] Sukhpal Singh Gill, Quantum and blockchain based serverless edge computing: A vision, model, new trends and future directions, Internet Technol. Lett. (2021).

[116] Harshit Gupta, Amir Vahid Dastjerdi, Soumya K Ghosh, Rajkumar Buyya, 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) (2017) 1275–1296.

[117] Tariq Qayyum, Asad Waqar Malik, Muazzam A Khan Khattak, Osman Khalid, Samee U Khan, FogNetSim++: A toolkit for modeling and simulation of distributed fog environment, IEEE Access 6 (2018) 63570–63583.

[118] Shreshth Tuli, Sukhpal S. Gill, Giuliano Casale, Nicholas R. Jennings, iThermoFog: Iot-fog based automatic thermal profile creation for cloud data centers using artificial intelligence techniques, Internet Technol. Lett. 3 (5) (2020) e198, http://dx.doi.org/10.1002/itl2.198, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/itl2.198. e198 ITL-20-0074.R1.

[119] Andrzej Wilczyński, Joanna Kołodziej, Modelling and simulation of security-aware task scheduling in cloud computing based on blockchain technology, Simul. Model. Pract. Theory 99 (2020) 102038, http://dx.doi.org/10.1016/j.simpat.2019.102038.

[120] Klaus David, Hendrik Berndt, 6G vision and requirements: Is there any need for beyond 5G? IEEE Veh. Technol. Mag. 13 (3) (2018) 72–80.

[121] Rodrigo Izidoro Tinini, Matias Romário Pinheiro dos Santos, Gustavo Bittencourt Figueiredo, Daniel Macêdo Batista, 5GPy: A SimPy-based simulator for performance evaluations in 5G hybrid cloud-fog RAN architectures, Simul. Model. Pract. Theory 101 (2020) 102030.