IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 15, 2021, accepted May 3, 2021, date of publication May 6, 2021, date of current version May 25, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3077901

A Metaheuristic Framework for Dynamic Virtual Machine Allocation With Optimized Task Scheduling in Cloud Data Centers

DEAFALLAH ALSADIE, (Member, IEEE)
Department of Information Systems, Umm Al-Qura University, Makkah 24381, Saudi Arabia
e-mail: dbsadie@uqu.edu.sa

ABSTRACT Optimal allocation of virtual machines in a cloud computing environment for user-submitted tasks is a challenging task. Finding an optimal task scheduling solution is considered as NP-hard problem specifically for large task sizes in the cloud environment. The best solution involves scheduling the tasks to virtual machines data centre while minimizing the essential, influential and cost effective parameters such as energy usage, makespan and cost. In this direction, this work presents a metaheuristic framework called MDVMA for dynamic virtual machine allocation with optimized task scheduling in a cloud computing environment. The MDVMA focuses on developing a multi-objective scheduling method using non dominated sorting genetic algorithm (NSGA)-II algorithm-based metaheuristic algorithm for optimizing task scheduling with the aim of minimizing energy usage, makespan and cost simultaneously to provide trade-off to the cloud service providers as per their requirements. To evaluate the performance of the MDVMA approach, we compared the performances of two different scenarios of benchmark real-world workload data sets using the existing approaches, namely, Artificial Bee Colony (ABC) algorithm, Whale Optimization Algorithm (WOA) and Particle Swarm Optimization (PSO) algorithm. Simulation results demonstrate that optimizing task scheduling leads to better overall results in terms of minimizing energy usage, makespan and cost of the cloud data center. Finally, the paper concludes metaheuristic algorithms as a promising method for task scheduling in a cloud computing environment.

INDEX TERMS Cloud computing, energy consumption, task scheduling, meta-heuristics algorithm, optimization.

I. INTRODUCTION

In recent years, cloud computing has become an integral part of Information Technology based organizations and individual users. It provides on-demand computing resources for executing multi-tier applications in the form of virtual machines [1] equipped with different amount of computing resources. It provides the computing resources to the users to execute their application on suitable virtual machines at the agreed quality of service on the basis of the pay-as-you-go model. It has several advantages over traditional IT services, including on-demand network services, remote and reliable storage, rapid elasticity, enhanced security, multi-tenancy, and measured services [2].

The associate editor coordinating the review of this manuscript and approving it for publication was Asad Waqar Malik.

Cloud computing service providers offer software and hardware resources through cloud data centres. In order to meet the increasing demands of cloud computing resources, data centres are equipped with more powerful servers and other related hardware resources [3]. It has been observed that the significant portion of energy consumption is caused by servers in the cloud data centre as depicted in Fig. 1. Increase in energy consumption of cloud data servers has a direct impact on cost, performance, return on investment and emission of carbon dioxide. Therefore, the research community in the field has focused on minimizing the energy consumption of cloud data centres using learning innovative ideas.

An efficient task scheduling method is considered as one of the most critical methods to address the challenge of minimizing energy usage in cloud data centres [4]. A task scheduling method attempts to map user-submitted tasks to specific virtual machines with the aim of optimizing the objectives of the

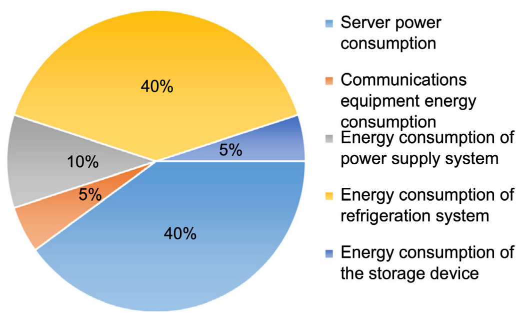

FIGURE 1. Energy usage distribution in the cloud data centre [4].

cloud data centre such as maintaining throughput, minimizing energy consumption while ensuring service level agreement and required quality of service [5]. However, considering a large number of tasks and virtual machines in the cloud data centre, solving the task scheduling problem is a challenge in polynomial time. Thus, it is considered as non-deterministic polynomial-time hard (NP hard) problem in the cloud computing context [6].

Several methods have been developed to solve NP-hard problems like task scheduling including heuristic methods and metaheuristic methods. Heuristic methods are used to design a specific kind of scheduling problems, whereas meta heuristic methods are employed for finding near optimal solutions. Meta heuristic methods can be used in three ways to solve task scheduling problems as depicted in Fig. 2, single objective (SO) scheduling, multi-objective as single objective (MOSO) and multi-objective as multi-objective (MOMO) scheduling methods [7], [8]. SO methods attempt to optimize only one objective or a single objective from a set of objectives at a time. MOSO methods attempt to optimize multiple objective functions by combining them into a single objective function. It requires in-depth domain knowledge that enables an accurate combination of the objective functions [9]. MOMO methods treat multiple objectives as separate objectives and attempt to optimize simultaneously. Such methods produce a set of non inferior solution in the form of the Pareto front and offer trade off to the cloud service provides. SO and MOSO have issues with solving multi-objective problems when concave shaped Pareto front is involved. Therefore, many MOMO based optimizing methods like PAES [10], SPEA2 [11], PESA-II [12], non dominated sorting genetic algorithm (NSGA)-II [13], MOEA/D [14] has been proposed in recent years.

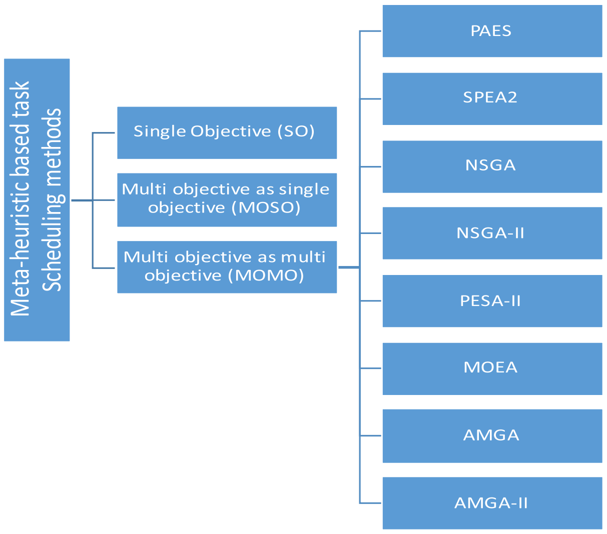
Meta-heuristic based task Scheduling methods
Single Objective (SO)
Multi objective as single objective (MOSO)
Multi objective as multi objective (MOMO)
PAES
SPEA2
NSGA
NSGA-II
PESA-II
MOEA
AMGA
AMGA-II

FIGURE 2. Meta-heuristic based task scheduling methods.

In this work, we mainly focused on formulating task scheduling problem as pure multi objective problem (MOMO) as described in Section I and II. In literature, prior work mainly focused on meta heuristic methods for optimizing single objective or multiple objective as single objectives in task scheduling context of cloud computing environment. Most prior work attempted to obtain single task scheduling solution that lacks in trade-offs for the cloud service provider. Here, we formulated task scheduling problem as multi objective problem where each objective is considered as a separate objective and multiple objectives are optimized simultaneously using MOGA. In this work, we aim to obtain a set of near optimal solutions that assure minimization of energy usage, makespan and cost for executing given set of cloud tasks on given virtual machines in the cloud data center simultaneously. The obtained solutions exhibit trade-offs for cloud service providers that was ignored in prior work targeting to obtain single task scheduling solution. We considered the most commonly used parameters used by evaluating task scheduling approaches as per findings of the survey paper by [3]. This combination of evaluating factors is mostly ignored in prior work in the field. Moreover, we highlighted the need of task scheduling, advantages of using meta heuristic algorithms and explained working of the proposed framework in the manuscript. The proposed framework is validated using synthetic and realistic benchmark data set. Results are compared with state of art methods. We demonstrated percentage improvement in results by using the proposed framework in comparison to other methods. Results are discussed to highlight the potential benefits of the proposed framework in this work.

The primary contribution of this work is a meta heuristic framework called MDVMA for dynamic virtual machine allocation with optimized task scheduling in a cloud computing environment. This work focuses on developing a multi-objective scheduling method based metaheuristic framework for optimizing task scheduling with the aim of minimizing energy usage, makespan and cost simultaneously. This paper mainly contributes in the following ways.

- Formulated mathematical models for computing energy usage, makespan and cost of the task scheduling problem in the cloud data centre.

- Formulated task scheduling problem as a multi-objective optimization problem having multiple conflicting objectives like energy usage and makespan under user defined constraints like the deadline of execution time.
- Performed multi-objective optimization for task scheduling problem using NSGA-II [13] to minimize energy usage, makespan and cost simultaneously.
- Simulated the proposed MDVMA framework and other approaches in CloudSim environment.
- Demonstrated the better performance of the proposed MDVMA approach in comparison to three other approaches, namely, Artificial Bee Colony (ABC) algorithm, Whale optimization algorithm (WOA) and Particle Swarm Optimization PSO) algorithm using two data sets, namely Heterogeneous Computing Scheduling Problem (HCSP) as benchmark dataset and synthetic dataset in terms of energy usage, makespan and cost.

benchmark dataset and syn- thetic

The remainder of this paper is structured as follows. Section II describes the state of the art in context to multi-objective metaheuristic methods for task scheduling in a cloud computing environment. Section III provides motivation for using multi-objective genetic algorithm and presents a flow chart of NSGA-II algorithm. Section IV describes the proposed framework and explain its working. Section V describes the details of the experiment and discusses the results. Finally, the paper is concluded at the end of VI.

II. RELATED WORK

Cloud computing environment has enabled the execution of a variety of applications that are more cost-effective for individual users as well as for organizations. However, the energy usage of a cloud data center has drastically increased with the increased demands of cloud based services [15]. Therefore, many research efforts have been made to minimize energy consumption along with other factors like makespan, ROI, cost etc.

Initially, many researchers focused on heuristic algorithms for solving task scheduling problem in the cloud computing environment. The heuristic algorithms mainly focus on finding an optimal or near optimal solution [16]. These algorithms explore search space for finding the best solution by using features of the problem. These algorithms are problem dependent [17]. Many heuristic algorithms have been developed and applied in the cloud environment for scheduling independent tasks. The most important developments in heuristic algorithms include minimum completion time algorithm, minimum execution time algorithm, First Come First serve basis algorithm, Min-Max algorithm, Min-Min algorithm and Suffrage algorithm, Greedy Scheduling, Shortest Task First, Sequence Scheduling, Balance Scheduling (BS), Opportunistic Load Balancing, Min-Min Opportunistic Load Balancing [18]–[20].

These algorithms have been widely used in solving the task scheduling problem, mainly the static task scheduling problem [21]. However, these algorithms suffer from a significant limitation of being trapped in local minima as task scheduling problem is a multi-modal problem [22].

To avoid trapping of heuristic algorithms in local minima issue, meta heuristic algorithms have been developed. Meta heuristic algorithms have several benefits over heuristic algorithms, including flexibility, simplicity and ergodicity [17]. These algorithms can be easily deployed and found to be less complex. These algorithms have the flexibility to solve many optimization problems. These algorithms also possess the capability of ergodicity for finding multi-modal search space with appropriate diversity by avoiding local minima simultaneously. The recent developments in metaheuristic algorithms indicate that these algorithms are more efficient and effective in task scheduling over classic task scheduling methods [23]–[26].

Keeping advantages of meta heuristic algorithms into consideration, several researchers have focused on task scheduling using meta heuristic methods, specifically multi-objective scheduling methods. Fig. 3 presents the general workflow of multi-objective scheduling methods. These methods require the user task along with constraints (if any), applies the optimization process involving crossover and mutation operators, passes through different generations to obtain Pareto front of solutions. The pareto front represents a set of non-inferior solutions as a result of optimizing multiple objectives simultaneously. The administrator of the cloud data centre can manually select or automatically apply some the desired solution as per requirements of the cloud data centre to allocate tasks to the virtual machines as per the selected solution.

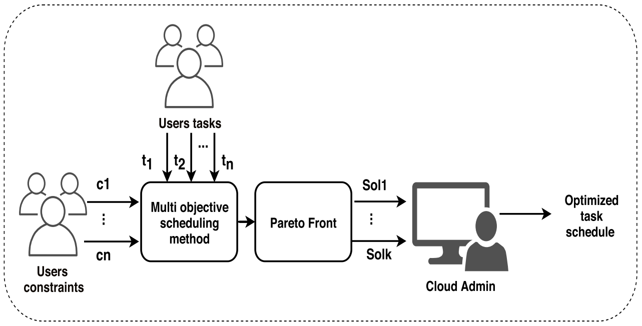

FIGURE 3. General workflow of multi objective scheduling methods.

Jena *et al.* [27] suggested an Artificial Bee Colony (ABC) based approach for optimizing energy usage, computing resource utilization, cost and processing time in the cloud computing environment. The authors proposed to assign a timestamp touser-submitted jobs on their arrival and maintain a queue to serve the jobs on First Come First Serve basis. The jobs are allocated to suitable data centres by using the multi-objective ABC algorithm. Here, the ABC algorithm maintains an external archive to save non-dominated vectors during the scheduling process. Initially, the external archive uses the Random numbers, followed by multi-objective ABC algorithm generated values. They simulated the proposed approach using cloudSim software. Zheng and Cai [28] suggested that in order to dispatch energy-aware load that minimizes the electricity consumption and cost for online cloud service providers, this model helps to save a considerable amount of electricity by considering its volatility using energy-efficient methods in cloud data centres. Luo and Zhou [29] proposed an optimization algorithm for minimizing the energy consumption by workflow scheduling as per service level agreement. The proposed method demonstrated a reduction of a considerable amount of energy while fulfilling the performance constraints in terms of cost and time.

Guoning *et al.* [30] proposed a task scheduling method based upon genetic simulated annealing algorithm. The main focus of their proposal was to optimize the execution time of the tasks in cloud data centres. Priyanto and Adiwijaya [31] attempted to find an optimal task schedule with minimum fluctuations using ant colony optimization algorithm. An Improvement over ant colony optimization was suggested by Preve [32]. The improved algorithm demonstrated the minimization of makespan and balance the workload of the cloud system in the grid environment. Banerjee *et al.* [33] also improved the basic ant colony optimization algorithm for allocating and scheduling services with the name of minimizing throughput to serve all incoming request as per service level agreement and resource allocators. Feller *et al.* [34] proposed to solve the problem of workload placement as multi-dimensional bin packing problem in the cloud computing environment. The proposed solution dynamically places cover cloth as per current load.

Pandey *et al.* [35] suggested a particle Swarm Optimization based forest method for task scheduling of data centered applications. The proposed approach considered computational overhead and data transfer costs simultaneously. Tawfeek *et al.* [36]Suggested the use of ant colony optimization algorithm for minimizing the execution time of tasks in the cloud computing environment. The authors demonstrated that ACO based approach as outperformed the conventional methods like Round Robin, and First Come First serve method. The authors of [37] proposed an approach using particle Swarm Optimization algorithm for task scheduling to optimize task execution time and utilization of computing resources in the cloud data centre. Some authors suggested integrating PSO and simulated annealing algorithms for task scheduling in the Cloud Computing environment [37]–[39]. It has been observed that the integrated approach enables the reduction in execution time of tasks and enhance the utilization of computing resources in the cloud data centre. A stochastic hill climbing algorithm based approach is developed in [40] solving for task scheduling problem in a cloud computing environment. The proposed approach was simulated in a cloud analyst. The authors demonstrated the superiority of the proposed approach over First Come First serve basis and Round Robin algorithms.

The authors of [41] performed the comparative analysis of metaheuristic algorithms, namely Artificial Bee Colony algorithm, Ant Colony Optimization algorithm and Particle Swarm Optimization algorithm for task scheduling problem. It has been concluded that meta heuristic algorithms outperform the conventional algorithms such as First Come First serve algorithm, largest ask first algorithm, and random algorithm for minimizing the execution time of tasks in the cloud data centre. They proved that artificial Bee colony algorithm outperforms the remaining metaheuristic algorithm in their experiment. The authors of [42] suggested a symbiotic organism search algorithm for task scheduling problem. They demonstrated that their approach outperforms the conventional Particle Swarm Optimization algorithm, particularly in large search space.

Most of the above mentioned work focused on minimizing execution time or energy consumption of the cloud data centre using meta heuristic algorithms. Only a few methods concentrated on minimizing energy consumption, makespan and cost simultaneously. Since makespan and energy consumption are inversely proportional, it is desired to obtain a set of non inferior solutions by optimizing multiple conflicting objectives.

In this direction, Suresh *et al.* [43] proposed a PSO approach based on the stimulated social behaviour of bird flocking to solve multi-objective optimization problem. They called each solution as a particle and adjusted its position in the search page as per its flying experience and experience of neighbouring particles. Omkar *et al.* [44], [45] proposed an integrated approach based on Quantum behaved Particle Swarm Optimization and vector evaluated particle Swarm Optimisation algorithms for solving multi-objective Optimization problem as single objective problem. They found appropriate weights to the objective functions to compute fitness function value as a single objective function value. But the proposed solution lacks in guaranteeing the global convergence. The global convergence issue was solved using quantum behaved particle swarm optimisation algorithm that uses state of the particle with wave function instead of velocity and position of the particles. However, determining the optimized values of the weight factor for each objective function is practically tricky for real life problems.

Chen *et al.* [46] suggested integrating the latest whale optimization algorithm with a multi-objective optimization model for obtaining an optimized task schedule in the cloud computing environment. The authors mainly focused on improving the performance of the cloud systems with available resources. Simulation-based experimental results demonstrate the improved performance in terms of convergence speed and accuracy for finding an optimal task scheduling solution in comparison to traditional methods. The authors provided improved system resource utilization for small and large scale task configuration in the cloud data centre.

On similar lines, multi-objective Wale Optimization algorithm is developed for task scheduling in a cloud computing environment in [47]. The authors attempted to schedule the tasks on the basis of fitness parameters, relying on three conditions, namely, quality of service, resource utilization and energy. They demonstrated that by considering these three parameters, task execution time and cost of the virtual machines could be minimized by using multi-objective Whale Optimization algorithm. The efficiency of their proposed algorithm depends upon fitness parameters. Sreenu *et al.* [48] proposed an approach called W-scheduler for solving task scheduling problem using an integrated approach of multi-objective model and whale optimization algorithm. W-scheduler involves the computation of fitness value by computing the cost function of memory and the central processing unit. It also adds makespan and budget cost function into fitness value. By using a whale optimization algorithm, W-scheduler attempts to find an optimal schedule for allocating tasks to a given number of virtual machines with an aim of minimizing costs and makespan. They evaluated the performance of W-scheduler in comparison to the traditional methods, namely, PBACO, SPSO-SA, and SLPSO-SA in terms of cost and makespan. Sanaj *et al.* [49] proposed a framework for map reduce and an integrated approach for efficient task scheduling based on genetic algorithm and whale optimization algorithm. They proposed to extract task related features in the beginning. The extracted features are further reduced using MRQFLDA algorithm. The pool of the tasks are divided into a number of sub tasks using the map reduce framework. Sub tasks are assigned to virtual machines as per schedule produced by genetic algorithm and whale optimization algorithm. These algorithms are simulated using the well known cloudSim environment. Experimental results demonstrate better performance of the proposed algorithms over traditional algorithms.

In this work, we focus to use a multi-objective optimization algorithm, namely non dominated sorting genetic algorithm (NSGA)-II. NSGA-II has been successfully implemented in a variety of optimization problems to find non-inferior solutions in the form of Pareto front having multiple conflicting objectives.

III. MULTI OBJECTIVE OPTIMIZATION

This section highlights the motivation behind using MOGA and working of NSGA-II [13] as multi-objective optimization algorithm.

A. MOTIVATION OF MOGA

Multi-objective genetic algorithms have been successfully implemented in different applications related to industry, science and engineering [50]. These algorithms have been widely used for solving multi-objective optimization problems in different areas. Most of the researchers use genetic algorithms in two different modes. Some researchers converted a multi-objective problem into a single-objective problem and applied the genetic algorithm to find an optimal solution [51]. In this mode, the genetic algorithm is capable of obtaining a single solution that optimizes the single objective. Generally, researches incorporate domain knowledge for converting multi-objective problem into a single objective problem.

The second mode uses genetic algorithms to solve multi-objective problems by considering different objectives separately. This mode produces a set of non-dominated solutions. The obtained set of solutions provides trade off between objectives for the users in the form of a Pareto front [52]–[56]. The user can choose a solution from the set of non-dominated solutions as per requirements. In this case, there is no requirement of any specific knowledge to obtain the set of non-dominated solutions. Multi-objective genetic algorithms have been proposed in the recent past. It has been concluded that NSGA-II is a fast and efficient algorithm in the category of multi-objective genetic algorithms.

B. WORKING OF NSGA-II

Deb *et al.* (2000) [13] improved the NSGA proposed by Srinivas & Deb (1994) [56] as NSGA-II. Figure 4 presents the working model of NSGA-II. NSGA-II is also a generation-based method wherein the working is based on the dominance concept. It uses the concept of crowding distance for maintaining the diversity of individual solutions. The parent chromosomes are selected using the tournament approach. This algorithm involves sorting of the population

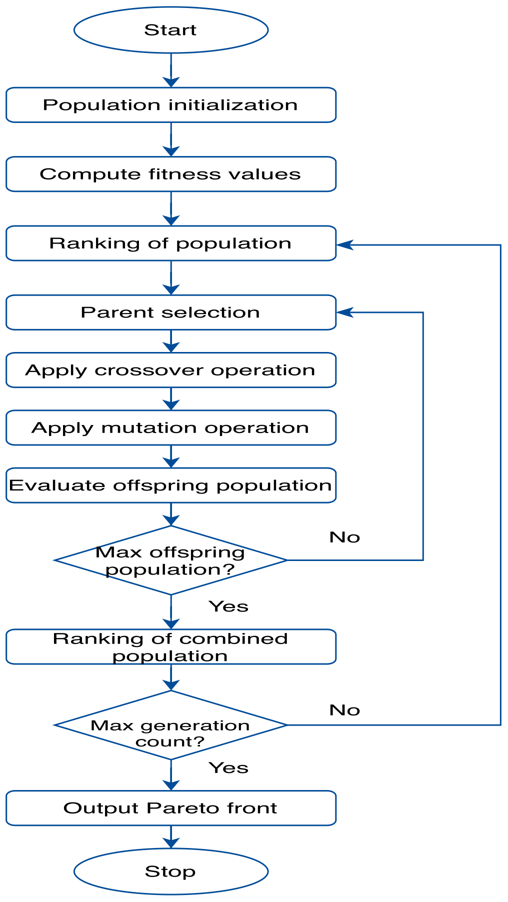

FIGURE 4. Flowchart of NSGA-II.

as per non-dominant levels by comparing solutions for determining if other solutions dominate the solution. Solutions are arranged in the form of Pareto front at different levels. Each generation of the algorithm creates a new population from the top level of Pareto front. If more solutions are required for forming a new population, then solutions are taken from the next front by considering their ranks. The rank of the solution is determined by the number of solutions it dominates and the number of solutions that dominates it [50]. The density of the solutions is determined by using the average distance between two points known as crowding distance. This algorithm uses crowd comparison operation by considering the non-dominance rank of the solutions. It does not use any external memory. Due to these advantages, NSGA-II is considered as an efficient genetic algorithm in comparison to two other variants of MOGA.

IV. THE PROPOSED FRAMEWORK

The proposed framework consists of four main modules as depicted in Fig. 5. It involves the optimization of energy consumption, makespan and cost of cloud services consisting of allocating $T = \{T_1, T_2, T_3, \ldots, T_m\}$ tasks over $V = \{V_1, V_2, V_3, \ldots, V_n\}$ virtual machines in cloud computing environment. In this work, we assume that virtual machine V_i works in the voltage range of $\{V\_Min_i, V\_Max_i\}$. Each Task T_i may consists of number of sub tasks as $\{st_1, st_2, st_3, \ldots, st_k\}$.

The detail of main modules is described below:

A. ENERGY USAGE COMPUTATION MODULE

This module employs dynamic voltage frequency scaling (DVFS) system for computing energy usage of m virtual machines in the cloud. DVFS system provides the energy usage by each resource running at different voltage frequencies. The energy usage of a virtual machine for running a sub task is computed using Eq. (1).

$$EU = \alpha \times V^2 \times W_{freq} \tag{1}$$

where V is the voltage level for operating the sub task T_i. α is a constant factor, that is a multiple of flip frequency and load capacitance. The value of α varies in the range of 0 to 1. W_{freq} represents the working frequency. The total energy usage for completing a task is computed using Eq. (2).

$$EU_{Task_i} = \sum_{i=1}^{k} \alpha \times V_i^2 \times W_{freq_i} \times CT_i \tag{2}$$

where, CT_i gives completion time for Task T_i on a given virtual machine.

B. TASK ESTIMATION MODULE

The proposed framework involves developing a task scheduling model based on evaluation factors employed in the multi-objective scheduling approaches for optimizing multiple objectives. As per survey conducted by [3], cost, energy consumption, task completion time, task waiting time, flow

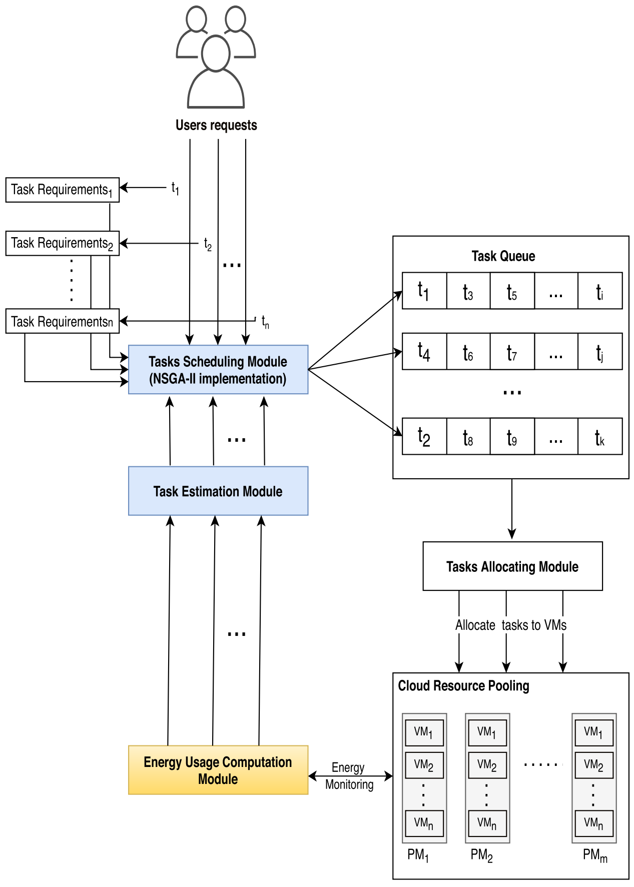

FIGURE 5. The proposed meta-heuristic framework.

time, failure rate, profit, carbon emission, makespan and reliability are the most commonly used evaluation factors in multi objective scheduling approaches.

In this work, we focused on the three most common metrics, energy usage, cost and makespan as evaluation factors in the proposed framework using task estimation module. Task estimation module computes energy usage for each task using Eq. (2). Makespan refers to the time for the complete execution of the last task. It is used as the most common evaluation factor for multi-objective scheduling approaches. It helps to reduce the execution time and meet the deadline of the task [15]. The lower value of makespan metric indicates the optimal schedule of virtual machines. It is computed as the maximum completion time of all tasks by a virtual machine among all possible schedules. In this work, we express makespan as the completion time of final task by a virtual machine to maintain simplicity using Eq. (3).

$$MS = Min_{Schedule_j}(Max_{Task_i}) \tag{3}$$

Cost refers to the cost of computing resources required for executing a task. We use Eq. (4) for computing cost [29].

$$Cost = \sum_{i=1}^{n}(ET_i \times RC_i) + \sum_{i=1}^{n}(CT_i \times TT_i) \tag{4}$$

where, ET_j represents the execution time of the task T_j. RCj gives the computing resource cost per unit time. CT_i and TT_i denote completion time and task transfer time of task T_i, respectively.

C. TASK SCHEDULING MODULE

This module enables the optimal scheduling of $T = T_1, T_2, T_3, \ldots, T_m$ tasks over $V = V_1, V_2, V_3, \ldots, V_n$ virtual machines in cloud data center by implementing NSGA-II. The task scheduling develops a correspondence between T and V to optimize the objective. In this work, we focus on optimizing energy usage, makespan and cost represented by Eqs.(3), and (4), respectively.

This module aims at scheduling tasks to virtual machines for achieving minimum energy usage, makespan and cost while meeting other constraints like availability of the resources and capacity of the physical machines. The module takes the input of estimation of evaluation factors, namely, energy usage, makespan and the cost along with user task requests. Further, the module applies a metaheuristic approach for optimizing the evaluation factors by finding a suitable schedule of tasks on the given virtual machines. Mathematically, the task scheduling problem is described as an optimization problem as per (5).

$$Best_{Schedule} = minimize(EU, MS, Cost) \tag{5}$$

The main objective of the optimization process involves identification of the best solution from the available solutions. Therefore, it requires a set of evaluation criteria called objective functions to compare all feasible solutions in the search space. Cloud Computing environment may consist of multiple objective functions that need to be optimized simultaneously like minimization of energy usage, minimization of makespan and minimization of cost.

This module formulates a task scheduling problem as a multi-objective optimization problem that optimizes the energy usage, makespan and cost simultaneously. In order to find an optimal solution for multi-objective task scheduling problem, this module applies a constrained non-dominated sorting genetic algorithm (NSGA-II) [13]. Mathematically, task scheduling problem is described below.

Objective functions:

- Minimize (energy usage) = $\sum EU_{ji}$, Where, EU_{ij} gives the energy usage for executing task T_i on virtual machine V_j.
- Minimize (makespan) = $Min_{Schedule_j}^{(Max_{Task_i})}$
- Minimize (cost) = $\sum_{i=1}^{n}(ET_i \times RC_i) + \sum_{i=1}^{n}(CT_i \times TT_i)$

Design constraints:

- Each task T to be assigned to only one virtual machine.
- Completion time of each task must fulfill its deadline.
- Resource requirements of all tasks on a given virtual machine must not exceed the maximum capacity of the virtual machine.
- There is no dependence of tasks and virtual machines.

Since, fitness function is a function for measuring quality of solutions, we proposed to use three objective functions for minimizing energy usage, makespan and cost directly and simultaneously as fitness functions for evaluating quality of the solution in this work.

1) CHROMOSOME

A chromosome represents a task schedule as a feasible solution in this module, which gives a sequence of task assigned to virtual machines as depicted in Table 1. Table 1 presents that $Task_1$, $Task_4$, $Task_5$, and $Task_10$ are assigned to VM_2, $Task_2$, $Task_6$ and $Task_8$ are assigned to VM_3, $Task_3$ is scheduled for VM_1 and $Task_7$ is scheduled for VM_4.

TABLE 1. **Chromosome representation.**

$Task_1$	$Task_2$	$Task_3$	$Task_4$	$Task_5$	$Task_6$	$Task_7$	$Task_8$	$Task_9$	$Task_{10}$
VM_2	VM_3	VM_1	VM_2	VM_2	VM_3	VM_4	VM_3	VM_2	VM_2

2) POPULATION INITIALIZATION

The module initializes the population using randomly as expected solutions in search space. It computes the required evaluation metrics for the randomly selected solutions corresponding to each chromosome in the population.

3) PARENT SELECTION

In order to further explore the search space for finding an optimal solution, this module applies the tournament selection strategy for selection of individual solutions. The selected individual solutions are used to obtain new solutions using crossover and mutation operators.

4) CROSSOVER AND MUTATION OPERATORS

This module uses a single point crossover operator for the selected chromosomes to explore the search space further. The mutation operator is applied by flipping random bits of the selected chromosome for exploiting the solution and generating a new population for the next iteration.

5) DIVERSITY PRESERVATION

This module maintains a good spread of solutions in the obtained set of solutions by using crowding function as the default function of NSGA-II [13]. NSGA-II algorithm sorts a population of assigned size according to the level of non-domination; each solution must be compared with every other solution in the population to find if it is dominated. Solutions of the first non-dominated front are stored in the first Pareto front, solutions of the second front on the second Pareto front and so on. Solutions on the first Pareto front constitute the new population if they are less than the initial population size: solutions from the next front are taken according to their ranks.

In NSGA-II [13], for each solution, one has to determine how many solutions dominate it and the set of solutions to which it dominates. The NSGA-II estimates the density of solutions surrounding a particular solution in the population by computing the average distance of two points on either side of this point along each of the objectives of the problem. This value is the so-called crowding distance. During selection, NSGA-II uses a crowded-comparison operator which takes into consideration both the non-domination rank of an individual in the population and its crowding distance (i.e., non-dominated solutions are preferred over dominated solutions, but between two solutions with the same non-domination rank, the one that resides in the less crowded region is preferred).

This module uses NSGA-II algorithm with above mentioned characteristics to find a set of non-inferior solutions as an optimal set of task schedules in the form of Pareto front. The set of non-inferior solutions provides a classification trade-off for cloud service providers.

D. TASK ALLOCATING MODULE

Task allocating module assigns the selected non-inferior schedule for optimized values of evolution factors to virtual machines. The task of the selected non-inferior task schedule can be selected by domain experts manually, or it can be selected automatically based on some criteria as per the requirement of cloud service providers.

V. EXPERIMENTS AND RESULTS

This section describes the experimental setup, including the hardware and software platform for conducting a comprehensive set of experiments in this work. It also describes evaluation data sets, synthetic data set and benchmark HCSP data set. Experiment results are presented and discussed in terms of evaluation factors, namely, energy usage, makespan and cost in the following subsections.

A. EXPERIMENTAL SETUP

It is a very challenging task to evaluate scheduling methods and other resource allocation strategies in a real Cloud Computing environment. As the real cloud computing environment is based upon the pay-as-you-go model, it is monetarily infeasible to conduct a repeatable experiment in a real cloud computing environment with a varying degree of uniformity.

In order to address this issue, the researchers in the field prefer to evaluate their scheduling methods using a simulated environment that is based upon benchmark data sets. In this work, we used the most commonly used cloud simulation software called CloudSim (version 5.0) [57].

CloudSim simulation software is the most commonly used simulation software for evaluating optimization techniques. It simulates cloud system elements including data centres, tasks and virtual machines. It also offers support for task scheduling strategies and different kinds of energy usage models for simulating different kinds of workloads. In this work, we simulate a cloud model like Infrastructure as a Service (IaaS) based on a single data centre. The suggested data centre offers eight physical hosts of two different categories. Each machine consists of four virtual machines with three different structures.

Table 2 presents the configuration of the simulated cloud data centre. Table 3 presents settings for virtual machines as per classification proposed by Amazon EC2. Table 4 presents settings for physical hosts. We performed experiments on a machine equipped with Intel (R) Core (TM) i5-4210 CPU @ 1.70 GHz, 8GBs RAM, 1TB HDD under Windows 10 64-bit operating system.

TABLE 2. Configuration of cloud simulated data centre.

Parameter	Value
Main memory	2 GB
HDD	1 TB
Bandwidh	10gbps
Scheduler	Time sharing
Virtual machine manager	Xen

TABLE 3. Setting of the simulated virtual machines.

VM type	Type 1 (small)	Type 2 (medium)	Type 3 (large)
MIPS	500	1000	1500
Pe	4	7	20
Capacity	2000	7000	30000

TABLE 4. Setting of the simulated physical hosts.

Physical host	H-I	H-II
Processor	Intel Core 2 Extreme X6800	Intel Core i7 Extreme 3960X
PE	2	6
MIPS	27079	177730

B. BENCHMARK DATASET

In order to demonstrate the applicability of the proposed task scheduling method and comprehensive comparison with the existing approaches, we reused benchmark data sets, namely, synthetic data set and HCSP data set developed by Braun *et al.* [58].

In this work, we consider a benchmark data set having different types of virtual machine instances and tasks for evaluating the proposed scheduling framework. Identified benchmark data sets in Worlds data pertaining to the execution time of task ok on different virtual machines. We followed a uniform distributed strategy for generating different instances of the data set. We represented the attributes of the data instances in terms of consistency, heterogeneity of task and heterogeneity of resource.

In this data set, the attribute of the data instance is represented as u_x_yyzz [58]. Here, u represents uniform distribution. x represents consistency type. Three Types of consistencies have been considered in this work as fully consistent, partially consistent and inconsistent. yy represents the heterogeneity of the task. zz represents the heterogeneity

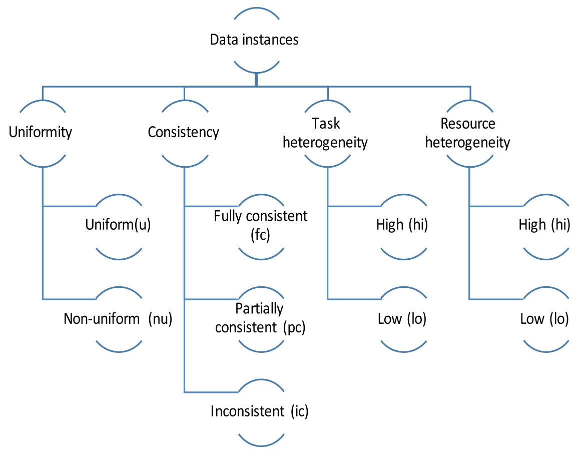

FIGURE 6. Attributes of data instances.

of the resource. Heterogeneity of task and resource can take values of high and low. Fig. 6 summarizes the values of the attributes of data instances. In this work, we focused on uniformly generated data instances.

In this set of experiments, we followed a methodology for evaluating the proposed task scheduling framework consisting of 1) selecting a benchmark instance 2) executing the task using proposed top scheduling method; 3) computing evaluation factors, namely, energy usage, makespan and cost.

The performance of the proposed fast scheduling method is compared with state-of-the-art in the field to demonstrate the superiority of the proposed method.

1) HETEROGENEOUS COMPUTING SCHEDULING PROBLEM (HCSP) DATASET

The benchmark data set HCSP data set is developed on the basis of execution time for a computing matrix n number of virtual machines and m number of tasks. HCSP data set considers all heterogeneity and consistency for both tasks and machines The HCSP model is based on the expected time to compute (ETC) matrix having m number of tasks and n number of VMs. This dataset considers three key properties of ETC model (machine heterogeneity, task heterogeneity and consistency). Here, uniformity denotes the instances that are generated by uniform distribution. In this paper, all task instances of HCSP dataset are assumed to be generated from by uniform distribution.

Consistency indicates the characteristics of realistic scenarios and can take three values as fully partially consistent and inconsistent. In a fully consistent model, whenever a given machine m_j executes any task t_i faster than other machine m_k, then machine m_j executes all tasks faster than machine m_k. This situation corresponds to an ideal case of high machine heterogeneity and very low task heterogeneity (or even identical tasks complexity), where the execution time of each task is mainly determined for the computational power of each machine, since tasks are supposed to perform at the same rate in all machines (and without requiring large amount of remote data or communications). Although such structured scenario seems to be unrealistic for general purpose applications, it captures the reality of many SPMD (single program, multiple data) applications executing with local input data. An inconsistent model lacks of any structure among the tasks computational requirements and machines computational power, and this absence of relationships means that a given machine m_j may be faster than machine m_k when executing some tasks and slower for others. This category comprises the most general scenario for a distributed infrastructure composed of highly heterogeneous resources which receive any kind of tasks, from easy-to-solve programs to very complex parallel models. In addition, a third category of partially consistent models is included, to define those inconsistent systems that include a consistent subsystem. In this last category, even though there is not a predefined structure on the whole sets of tasks and machines, some of them behave like a consistent system.

Resource or Machine heterogeneity evaluates the variation of execution times for a given task across the resources in the computing system. An environment comprised of similar computing resources (i.e. parallel systems and almost-homogeneous clusters of workstations) will be represented by low machine heterogeneity values, while machine high heterogeneity values will represent generic HC systems, composed by many computational resources of different types and power (e.g. workstations, supercomputers, distributed environments and grid systems).

Task or cloudlets heterogeneity represents the degree of variation among the tasks execution times for a given machine. High task heterogeneity values models those scenarios in which tasks of different types are submitted to execute in the computing system, from simple programs which demands very little computational effort, to large and complex tasks which require high CPU times to perform. On the other side, when the complexity and so the computation requirements of the tasks are quite similar, those tasks can be performed in rather similar execution times for a given machine, a situation modeled by low task heterogeneity values.

The structure of the data set instances is denoted as *u_xx_yyzz*, where u denotes the instances that are generated by uniform distribution, *xx* denotes the type of consistency [i.e., fully consistent (fc), inconsistent (ic) or partially consistent (pc)], yy represents the task heterogeneity [i.e., high (hi) or low (lo)] and zz represents the resource heterogeneity [i.e., high (hi) or low (lo)]. Each data set consists of 12 different instances (i.e., *u_fc_hihi*, *u_fc_hilo*, *u_fc_lohi*, *u_fc_lolo*, *u_ic_hihi*, *u_i_hilo*, *u_ic_lohi*, *u_ic_lolo*, *u_pc_hihi*, *u_pc_hilo*, *u_pc_lohi* and *u_pc_lolo*).

The HCSP dataset used in this research is comprised of four types of instances (i.e., c-hilo, c-lohi, i-hilo, i-lohi) having a different level of task and resource heterogeneity is given below:

TABLE 5. Workload characteristics of HCSP data set.

HCSP Data sets	Uniformity (u / nu)	Consistency (fc / pc / in)	Task Heterogeneity (hi / lo)	Resource Heterogeneity (hi / lo)	Data instance
(512 x 16), (1024 x 32), (2048 x 64), (128x 4096), (256 x 8192)	Uniform (u)	Fully Consistent (fc)	High (hi)	High (hi)	u_fc_hihi
			High (hi)	Low (lo)	u_fc_hilo
			Low (lo)	High (hi)	u_fc_lohi
			Low (lo)	Low (lo)	u_fc_lolo
		Partially Consistent (pc)	High (hi)	High (hi)	u_pc_hihi
			High (hi)	Low (lo)	u_pc_hilo
			Low (lo)	High (hi)	u_pc_lohi
			Low (lo)	Low (lo)	u_pc_lolo
		Inconsistent (ic)	High (hi)	High (hi)	u_ic_hihi
			High (hi)	Low (lo)	u_ic_hilo
			Low (lo)	High (hi)	u_ic_lohi
			Low (lo)	Low (lo)	u_ic_lolo
	Non uniform (nu)	Not applicable			

- hilo: High heterogeneity of Cloudlet with low heterogeneity of VMs.
- lohi: Low heterogeneity of Cloudlets and high heterogeneity of VMs.

This data set contains with different structures like (512×16), (1024×32), (2048×64), (128×4096), and (256×8192). Here, the first number represents the number of tasks, and the second number represents the number of virtual machines. For example, 512×16 indicates that 512 tasks need to be scheduled on 16 virtual machines. In the HCSP data set, there are a different number of instances corresponding to each data set. For example, there are four instances of 512×16 data set, eight instances of 1024×32 data set and eight instances of 2048×64 data set. The workload of the HCSP based data set is presented in Table 5. The selected data set instances of HCSP data set have been widely used for evaluating task scheduling problems [59]–[61].

2) SYNTHETIC DATA SET

In this set of experiments, we used synthetic data set for evaluating the proposed approach in comparison to the other existing approaches. The synthetic data set is generated using a pseudo random generation number. In this data set, we generated 512, 1024, 2048, 128, and 256 heterogeneous tasks randomly. We followed general structure as per HCSP data set like (512×16), (1024×32), (2048×64), (128×4096), and (256×8192). Here, 512, 1024, 2048, 128, and 256 represent the number of tasks to be mapped to 16, 32, 64, 4096, 8192 number of virtual machines respectively.

The generated tasks are represented as notation *u_xx_yyzz*, described in Section V-B. We considered three levels of cloud consistency as fully consistent, partially consistent, and inconsistent. We also considered two level of heterogeneity as high and low for tasks and cloud as described in Section V-B1 for HCSP benchmark data set.

C. RESULTS AND DISCUSSION

This section explains the experimental evaluation of the proposed task scheduling method using benchmark data sets in terms of energy usage, makespan and cost. The workload of distribution is computed and compared with each virtual machine in the simulated environment of cloudSim software. We evaluated the performance of the proposed approach (MDVMA) in comparison to three well known meta heuristic algorithms, namely, Artificial Bee Colony (ABC) algorithm, Whale Optimisation Algorithm (WOA) and Particle Swarm Optmisation (PSO) algorithm. We conducted 10 independent executions of our experiments for each dataset and presented average values the obtained results in the following sections. Experimental results are described as benchmark data set wise below.

1) EXPERIMENTAL RESULTS BASED ON SYNTHETIC DATA SET

This section describes performance results of MDVMA in comparison to the identified meta heuristic algorithms, ABC, WOA and PSO in terms of energy usage, makespan and cost based on synthetic data set as described in Section V-B2. The evaluation parameters are computed using Equations 2, 3 and 4 as mentioned in Section IV-A and IV-B.

Table 6 presents the energy usage of cloud data center for the given workload using synthetic dataset with different number of virtual machines. It can be seen from the results reported in Table 6 that energy usage using MDVMA approach corresponding to various setting of virtual machines is comparatively lower than other simulated algorithms in this work. The total energy usage in cloud data center using MDVMA approach is reduced by 55.80%, 34.92% and 19.23% over ABC algorithm, WOA algorithm and PSO algorithm respectively.

Figure 7 depicts the Comparative analysis of the performance of MDVMA approach with other simulated meta heuristic algorithms in terms of energy usage of the cloud data center. The reporting results indicate that energy usage

TABLE 6. Comparative summary results in terms of energy usage using the synthetic dataset.

Approach / No.of tasks	Energy usage (kwh)				Reduction in energy using MDVMA over		
	ABC	WOA	PSO	MDVMA	ABC	WOA	PSO
5	4.01	3.5	3.01	2.51	37.41 %	28.29 %	16.61 %
10	6.91	5.25	4.61	3.72	46.16 %	29.14 %	19.31 %
15	7.81	7.01	6.02	4.83	38.16 %	31.10 %	19.77 %
20	9.01	7.72	7.01	6.02	33.19 %	22.02 %	14.12 %
25	10.01	9.1	8.41	7.14	28.67 %	21.54 %	15.10 %
30	12	10.5	9.01	7.71	35.75 %	26.57 %	14.43 %
Total Energy usage (kwh)	49.75	43.08	38.07	31.93	**35.82%**	**25.88%**	**16.13 %**

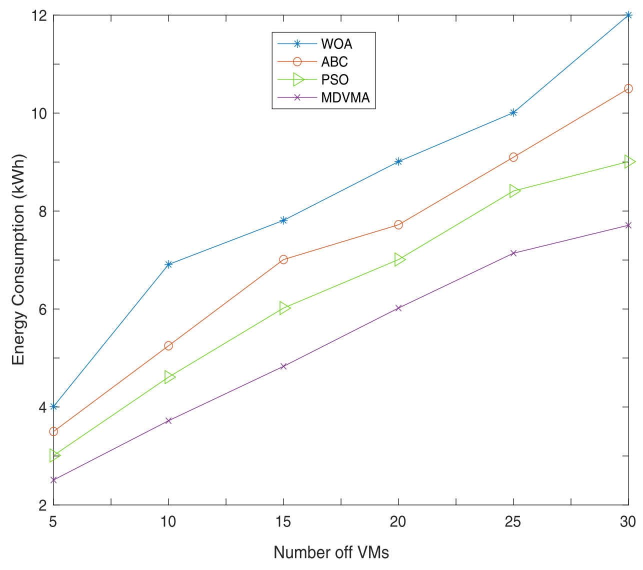

FIGURE 7. Comparative energy usage analysis of MDVMA approach with other simulated algorithms using synthetic data set.

of MDVMA approach at each level of the number of virtual machines is comparatively lower that of the state of art methods for task scheduling. MDVMA approach demonstrates a better solution for task scheduling problem using meta heuristic algorithm with a lower value of energy consumption at the cloud data center.

Table 7 presents the makespan metric values in seconds for the given workload using synthetic dataset with different number of task ranging from 100 to 600 recorded at an interval of 50. It can be seen from the results reported in Table 7 that makespan using MDVMA approach corresponding to various setting of tasks is comparatively lower than other simulated algorithms in this work. The total makespan in cloud data center using MDVMA approach is reduced by 12.22%, 7.78% and 5.56% over ABC algorithm, WOA algorithm and PSO algorithm respectively.

Figure 8 depicts the comparative analysis of the performance of MDVMA approach with other simulated meta heuristic algorithms in terms of makespan with varying load of number of tasks. The reporting results indicates that makespan of MDVMA approach at each level of number of tasks is comparatively lower that of the state of art methods

TABLE 7. Comparative summary results in terms of makespan using the synthetic dataset.

Approach / No.of tasks	makespan				Reduction in makespan using MDVMA over		
	ABC	WOA	PSO	MDVMA	ABC	WOA	PSO
100	900	700	600	450	50.00 %	35.71 %	25.00 %
150	1250	1050	950	800	36.00 %	23.81 %	15.79 %
200	1600	1400	1300	1150	28.13 %	17.86 %	11.54 %
250	1950	1750	1650	1500	23.08 %	14.29 %	9.09 %
300	2300	2100	2000	1850	19.57 %	11.90 %	7.50 %
350	2650	2450	2350	2200	16.98 %	10.20 %	6.38 %
400	3000	2800	2700	2550	15.00 %	8.93 %	5.56 %
450	3350	3150	3050	2900	13.43 %	7.94 %	4.92 %
500	3700	3500	3400	3150	14.86 %	10.00 %	7.35 %
550	4050	3850	3750	3500	13.58 %	9.09 %	6.67 %
600	4550	4350	4250	4000	12.09 %	8.05 %	5.88 %
650	5050	4850	4750	4500	10.89 %	7.22 %	5.26 %
Total Makespan	34350	31950	28550	7338	**16.89%**	**10.64%**	**7.15%**

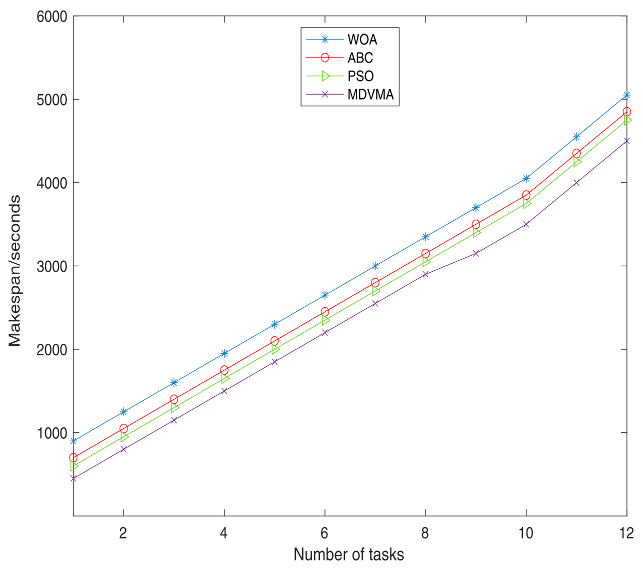

FIGURE 8. Comparative makespan analysis of MDVMA approach, ABC algorithm, WOA algorithm and PSO algorithm using synthetic data set.

for task scheduling. MDVMA approach demonstrates a better solution for task scheduling problem using NSGA-II algorithm with lower value of makespan at cloud data center at all levels of number of tasks in cloud data center.

We have also evaluated MDVMA algorithm in terms of cost and compared its performance with that of ABC algorithm, WOA algorithm and PSO algorithm using the synthetic data set. Table 8 presents the cost of the data center using MDVMA algorithm, ABC algorithm, WOA algorithm and PSO algorithm using synthetic data set for a given workload. The workload used in the synthetic dataset contains a different number of task ranging from 100 to 600 recorded at an interval of 50. It can be seen from the results reported in Table 3 that total cost using MDVMA approach corresponding to the various setting of tasks is comparatively lower than other simulated algorithms in this work. The total cost in the cloud data center using MDVMA approach is reduced

TABLE 8. Comparative summary of results in terms of cost using the synthetic dataset.

Approach / No.of tasks	Cost				Reduction in cost using MDVMA over		
	ABC	WOA	PSO	MDVMA	ABC	WOA	PSO
100	225	175	150	112	50.22 %	36.00 %	25.33 %
150	313	263	238	200	36.10 %	23.95 %	15.97 %
200	401	351	326	288	28.18 %	17.95 %	11.66 %
250	489	439	414	376	23.11 %	14.35 %	9.18 %
300	577	527	502	464	19.58 %	11.95 %	7.57 %
350	665	615	590	552	16.99 %	10.24 %	6.44 %
400	753	703	678	640	15.01 %	8.96 %	5.60 %
450	841	791	766	728	13.44 %	7.96 %	4.96 %
500	929	879	854	816	12.16 %	7.17 %	4.45 %
550	1017	967	942	904	11.11 %	6.51 %	4.03 %
600	1167	1117	1092	1054	9.68 %	5.64 %	3.48 %
650	1317	1267	1242	1204	8.58 %	4.97 %	3.06 %
Total cost	8694	8094	7794	7338	**15.60%**	**9.34%**	**5.85%**

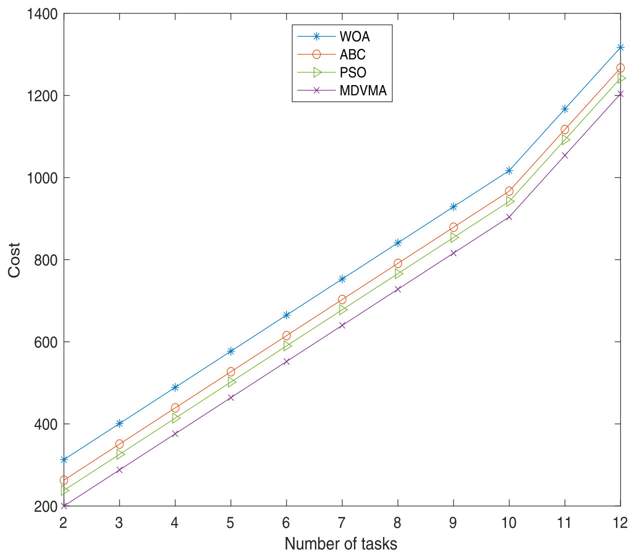

FIGURE 9. Comparative cost value analysis of MDVMA approach, ABC algorithm, WOA algorithm and PSO algorithm using synthetic data set.

by 15.60 %, 9.34 % and 5.85 % over ABC algorithm, WOA algorithm and PSO algorithm respectively.

Figure 9 presents the comparative analysis of the performance of MDVMA approach with other simulated meta heuristic algorithms in terms of total cost with varying load of number of tasks using synthetic data set. The reporting results indicates that cost of MDVMA approach at each level of number of tasks is comparatively lower that of the state of art methods for task scheduling. MDVMA approach demonstrates a better solution for task scheduling problem using NSGA-II algorithm with lower value of cost at cloud data center at all levels of number of tasks in cloud data center.

2) EXPERIMENTAL RESULTS BASED ON HCSP DATA SET

This section describes performance results of MDVMA in comparison to the identified meta heuristic algorithms, ABC, WOA and PSO based on benchmark HCSP data set as described in Section V-B1.

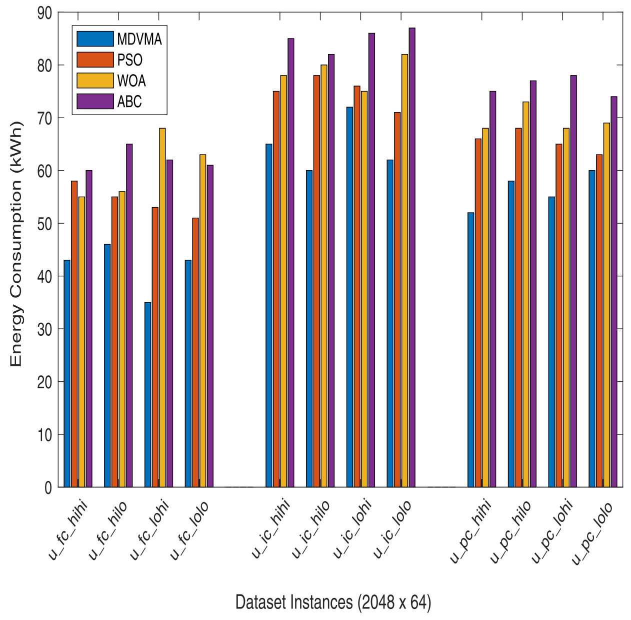

FIGURE 10. Performance analysis of MDVMA algorithm, ABC algorithm, PSO algorithm and WOA algorithm using HCSP dataset in terms of energy usage (kwh).

We presently used different data instances of HCSP data set. For the sake of simplicity and clarity, we presented and compared the results of one data instance denoted as 2048×64 consisting of 2048 tasks to be allocated to 64 virtual machines in terms of three evaluation parameters, namely, energy usage, makespan and cost. These evaluation parameters are computed using Equations 2, 3 and 4 as mentioned in Section IV-A and IV-B. Figure 10 presents comparative analysis of the performance of MDVMA approach, ABC algorithm, PSO algorithm and WOA algorithm in terms of energy usage based on a selected data instance (2048×64) of HCSP data set. It can be observed from the reported results that MDVMA approach out performs the other simulated algorithms by reducing the energy usage by an considerable amount of energy usage at all level of consistencies (fully consistent / partially consistent / inconsistent) and heterogeneity of tasks and computing resources (high / low) over ABC algorithm, PSO algorithm and WOA algorithm as presented in Table 9. The obtained results indicate that MDVMA approach has reduced more energy usage in case of fully consistent settings (u_fc_*) than corresponding inconsistent (u_ic_*) and partially consistent settings (u_pc_*).

Makespan represents overall execution time. It is considered as one of the most important evaluation factor in measuring cloud services. Lower value of Makespan show better performance and vice versa. Figure 11 presents comparative analysis of the performance of MDVMA approach, ABC algorithm, PSO algorithm and WOA algorithm in terms of makespan based on a selected data instance (2048×64) of

TABLE 9. Energy usage and reduction (% age) using MDVMA approach over ABC algorithm, PSO algorithm and WOA algorithm using HCSP dataset.

Approach / Data instance	Energy usage (kwh)				Reduction in cost using MDVMA over		
	PSO	WOA	ABC	MDVMA	PSO	WOA	ABC
u_fc_hihi	58	55	60	43	25.86 %	21.82 %	28.33 %
u_fc_hilo	55	56	65	46	16.36 %	17.86 %	29.23 %
u_fc_lohi	53	68	62	35	33.96 %	48.53 %	43.55 %
u_fc_lolo	51	63	61	43	15.69 %	31.75 %	29.51 %
u_ic_hihi	75	78	85	65	13.33 %	16.67 %	23.53 %
u_ic_hilo	78	80	82	60	23.08 %	25.00 %	26.83 %
u_ic_lohi	76	75	86	72	5.26 %	4.00 %	16.28 %
u_ic_lolo	71	82	87	62	12.68 %	24.39 %	28.74 %
u_pc_hihi	66	68	75	52	21.21 %	23.53 %	30.67 %
u_pc_hilo	68	73	77	58	14.71 %	20.55 %	24.68 %
u_pc_lohi	65	68	78	55	15.38 %	19.12 %	29.49 %
u_pc_lolo	63	69	74	60	4.76 %	13.04 %	18.92 %

TABLE 10. Makespan and reduction (%age) using MDVMA approach over ABC algorithm, PSO algorithm and WOA algorithm using HCSP dataset.

Approach / Data instance	Makespan (seconds)				Reduction in makespan using MDVMA over		
	PSO	WOA	ABC	MDVMA	PSO	WOA	ABC
u_fc_hihi	85	100	85	86	15.00 %	0.00 %	1.16 %
u_fc_hilo	72	74	73	78	2.70 %	1.37 %	7.69 %
u_fc_lohi	42	44	45	47	4.55 %	6.67 %	10.64 %
u_fc_lolo	33	36	33	35	8.33 %	0.00 %	5.71 %
u_ic_hihi	80	81	83	80	1.23 %	3.61 %	0.00 %
u_ic_hilo	69	70	72	73	1.43 %	4.17 %	5.48 %
u_ic_lohi	40	40	41	41	0.00 %	2.44 %	2.44 %
u_ic_lolo	30	31	33	30	3.23 %	9.09 %	0.00 %
u_pc_hihi	81	81.5	82	83	0.61 %	1.22 %	2.41 %
u_pc_hilo	71	71.5	73	74	0.70 %	2.74 %	4.05 %
u_pc_lohi	41	41.5	42	41.5	1.20 %	2.38 %	1.20 %
u_pc_lolo	30.5	31.5	32	32.5	3.17 %	4.69 %	6.15 %

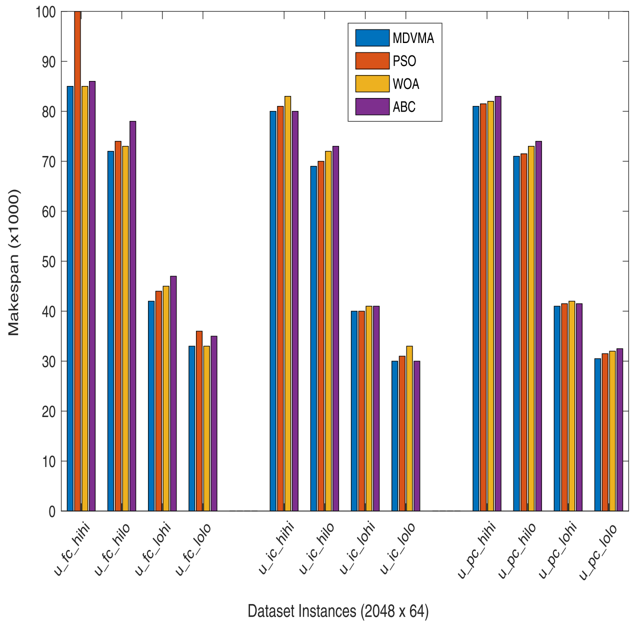

FIGURE 11. Performance analysis of MDVMA algorithm, ABC algorithm, PSO algorithm and WOA algorithm using HCSP dataset in terms of makespan (seconds).

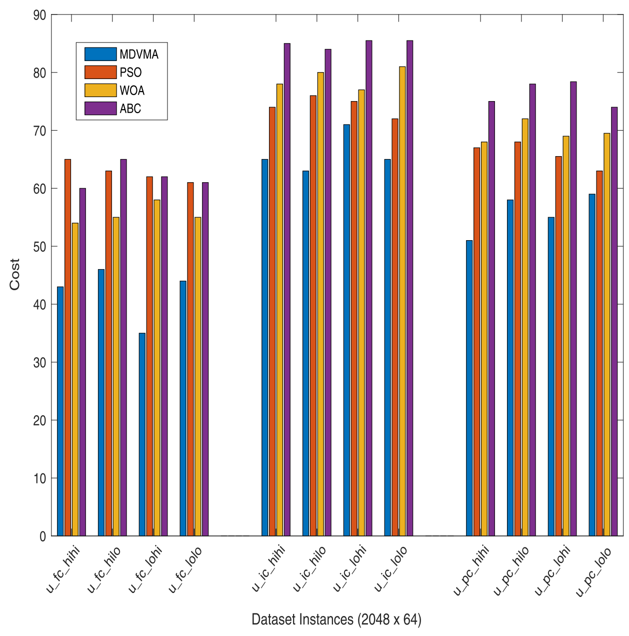

FIGURE 12. Performance analysis of MDVMA algorithm, ABC algorithm, PSO algorithm and WOA algorithm using HCSP dataset in terms of cost.

HCSP data set. It can be observed from the Figure 11 that MDVMA approach out performs the other simulated algorithms by reducing the makespan considerably at all level of consistencies (fully consistent / partially consistent / inconsistent) and heterogeneity of tasks and computing resources (high / low) over ABC algorithm, PSO algorithm and WOA algorithm as presented in Table 10.

It can be noticed from Table 10 that MDVMA approach can reduce makespan up to 15% (in case of PSO for u_fc_hihi data instance. However, in this setting MDVMA approach and WOA algorithm behave similarly. Similar to the energy usage results, MDVMA approach has resulted more reduction in makespan for fully consistent settings (*u_fc_*∗) than corresponding inconsistent (*u_ic_*∗) and partially consistent settings(*u_pc_*∗).

Figure 12 presents comparative analysis of the performance of MDVMA approach, ABC algorithm, PSO algorithm and WOA algorithm in terms of cost based on a selected data instance (2048×64) of HCSP data set. It can be observed from the Figure 6 that MDVMA approach out performs the other simulated algorithms by reducing the makespan considerably at all level of consistencies (fully consistent / partially consistent / inconsistent) and heterogeneity of tasks and computing resources (high / low) over ABC algorithm, PSO algorithm and WOA algorithm as presented in Table 11. Consequently, it can be observed from the above-reported results that MDVMA approach outperforms ABC algorithm, PSO algorithm and WOA algorithm in terms of energy usage, makespan and cost based on synthetic dataset as well as a selected data instance (2048 × 64) of HCSP data set.

TABLE 11. Cost and reduction (%age) using MDVMA approach over ABC algorithm, PSO algorithm and WOA algorithm using HCSP dataset.

Approach / Data instance	Cost				Reduction in cost using MDVMA over		
	PSO	WOA	ABC	MDVMA	PSO	WOA	ABC
u_fc_hihi	43	65	54	60	33.85 %	20.37 %	28.33 %
u_fc_hilo	46	63	55	65	26.98 %	16.36 %	29.23 %
u_fc_lohi	35	62	58	62	43.55 %	39.66 %	43.55 %
u_fc_lolo	44	61	55	61	27.87 %	20.00 %	27.87 %
u_ic_hihi	65	74	78	85	12.16 %	16.67 %	23.53 %
u_ic_hilo	63	76	80	84	17.11 %	21.25 %	25.00 %
u_ic_lohi	71	75	77	85.5	5.33 %	7.79 %	16.96 %
u_ic_lolo	65	72	81	85.5	9.72 %	19.75 %	23.98 %
u_pc_hihi	51	67	68	75	23.88 %	25.00 %	32.00 %
u_pc_hilo	58	68	72	78	14.71 %	19.44 %	25.64 %
u_pc_lohi	55	65.5	69	78.4	16.03 %	20.29 %	29.85 %
u_pc_lolo	59	63	69.5	74	6.35 %	15.11 %	20.27 %

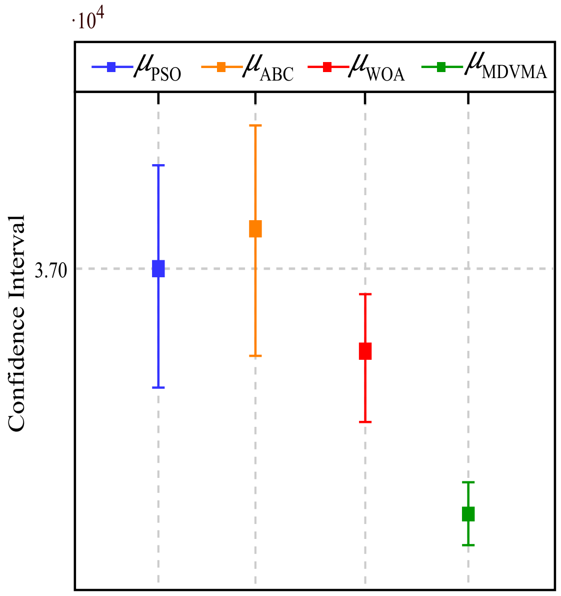

FIGURE 13. Comparative analysis of confidence interval for PSO, ABC, WOA and MDVMA methods using synthetic dataset.

The reduction in makespan (execution time) leads to more cloud customer satisfaction. Whereas, reduction in energy use and cost leads to increase revenue of cloud service providers by optimized task scheduling using MDVMA approach.

To prove the validity of MDVMA approach, we computed and plotted confidence interval for synthetic and benchmark datasets for cost in Figures 13 and 14 respectively. Confidence Intervals are used to quantify the uncertainty by providing a lower limit and upper limit to represent a range of values of the actual population parameter with a specified level of confidence. We assumed 95% of the confidence interval in our experiments. The comparative analysis of confidence intervals for PSO, ABC, WOA and MDVMA methods indicate that MDVMA outperforms existing methods.

Therefore, it can be concluded that MDVMA approach is a practical and useful solution for obtaining optimized task

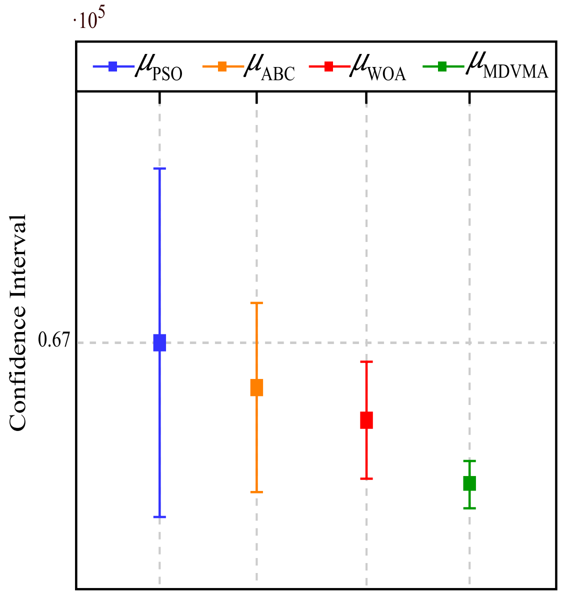

FIGURE 14. Comparative analysis of confidence interval for PSO, ABC, WOA and MDVMA methods using benchmark dataset.

scheduling with reduced energy usage, makespan and cost in comparison to other state of art optimization methods used for cloud task scheduling.

VI. CONCLUSION

This paper presents a metaheuristic framework called MDVMA for dynamic virtual machine allocation with optimized task scheduling in a cloud computing environment using NSGA-II algorithm. The MDVMA framework simultaneously optimizes multiple conflicting objectives like energy usage, makespan and cost and provides trade off to the Cloud Service Provider as per their requirements by offering a set of non inferior solutions to the task scheduling problem. Experimental results demonstrate the superior performance of MDVMA approach over the existing approaches, namely, Artificial Bee Colony (ABC) algorithm, Whale Optimization Algorithm (WOA) and Particle Swarm Optimization (PSO) algorithm using two data sets, namely HCSP as benchmark dataset and synthetic dataset in terms energy usage, makespan and cost. The proposed MDVMA framework outperforms the existing approaches by reducing energy usage by 35.82 %, 25.88 % and 16.13 %; makespan by 16.89 %, 10.64 % and 7.15 %; and cost by 15.60 %, 9.34 % and 5.85 % over ABC algorithm, WOA algorithm and PSO algorithm, respectively using synthetic data set. A considerable reduction in energy usage, makespan and cost has also been achieved for benchmark data instance 2048×64 of HCSP data set. In this paper, we mainly focused on dynamic task scheduling problem. The proposed framework is designed for deployment in cloud computing environment. We simulated cloud data center environment in CloudSim Simulator. The proposed framework is validated using realistic benchmark dataset of cloud tasks of different requirements like uniformity, consistency, task heterogeneity and resource heterogeneity. The obtained results demonstrated that proposed framework is practical for deployment in cloud data centers. Our future work will focus on developing optimal workflow scheduling strategies for virtual machine selection and placement in the cloud computing environment.

REFERENCES

[1] M. Masdari and M. Zangakani, "Green cloud computing using proactive virtual machine placement: Challenges and issues," *J. Grid Comput.*, vol. 24, pp. 1–33, Aug. 2019.

[2] J. Yang and Z. Chen, "Cloud computing research and security issues," in *Proc. Int. Conf. Comput. Intell. Softw. Eng.*, Dec. 2010, pp. 1–3.

[3] M. Hosseinzadeh, M. Y. Ghafour, H. K. Hama, B. Vo, and A. Khoshnevis, "Multi-objective task and workflow scheduling approaches in cloud computing: A comprehensive review," *J. Grid Comput.*, vol. 17, pp. 1–30, Sep. 2020.

[4] H. Rong, H. Zhang, S. Xiao, C. Li, and C. Hu, "Optimizing energy consumption for data centers," *Renew. Sustain. Energy Rev.*, vol. 58, pp. 674–691, May 2016.

[5] M. Agarwal and G. M. S. Srivastava, "A genetic algorithm inspired task scheduling in cloud computing," in *Proc. Int. Conf. Comput., Commun. Autom. (ICCCA)*, Apr. 2016, pp. 364–367.

[6] M. Masdari, F. Salehi, M. Jalali, and M. Bidaki, "A survey of pso-based scheduling algorithms in cloud computing," *J. Netw. Syst. Manage.*, vol. 25, no. 1, pp. 122–158, 2017.

[7] F. Ebadifard and S. M. Babamir, "Optimizing multi objective based workflow scheduling in cloud computing using black hole algorithm," in *Proc. 3th Int. Conf. Web Res. (ICWR)*, Apr. 2017, pp. 102–108.

[8] I. Pietri, Y. Chronis, and Y. Ioannidis, "Multi-objective optimization of scheduling dataflows on heterogeneous cloud resources," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 361–368.

[9] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Gener. Comput. Syst.*, vol. 28, no. 5, pp. 755–768, May 2012.

[10] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the Pareto archived evolution strategy," *Evol. Comput.*, vol. 8, no. 2, pp. 149–172, Jun. 2000.

[11] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," TIK, Yiming, Toutiao, Tech. Rep. 103, 2001.

[12] D. W. Corne, N. R. Jerram, J. D. Knowles, and M. J. Oates, "Pesa-II: Region-based selection in evolutionary multiobjective optimization," in *Proc. 3rd Annu. Conf. Genetic Evol. Comput.*, 2001, pp. 283–290.

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[14] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[15] A. S. Sofia and P. G. Kumar, "Multi-objective task scheduling to minimize energy consumption and makespan of cloud computing using NSGA-II," *J. Netw. Syst. Manage.*, vol. 26, no. 2, pp. 463–485, Apr. 2018.

[16] D. Hazra, A. Roy, S. Midya, and K. Majumder, "Distributed task scheduling in cloud platform: A survey," in *Smart Computing and Informatics*. Singapore: Springer, 2018, pp. 183–191.

[17] M. A. Elaziz, S. Xiong, K. P. N. Jayasena, and L. Li, "Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution," *Knowl.-Based Syst.*, vol. 169, pp. 39–52, Apr. 2019.

[18] K. V. Price, "Differential evolution: A fast and simple numerical optimizer," in *Proc. North Amer. Fuzzy Inf. Process.*, 1996, pp. 524–527.

[19] U. K. Sikdar, A. Ekbal, S. Saha, O. Uryupina, and M. Poesio, "Differential evolution-based feature selection technique for anaphora resolution," *Soft Comput.*, vol. 19, no. 8, pp. 2149–2161, Aug. 2015.

[20] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: Challenges and opportunities," in *Proc. Int. Conf. High Perform. Comput. Simul.*, Jun. 2009, pp. 1–11.

[21] S. K. Panda and P. K. Jana, "SLA-based task scheduling algorithms for heterogeneous multi-cloud environment," *J. Supercomput.*, vol. 73, no. 6, pp. 2730–2762, Jun. 2017.

[22] E. K. Tabak, B. B. Cambazoglu, and C. Aykanat, "Improving the performance of independenttask assignment heuristics minmin, maxmin and sufferage," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 5, pp. 1244–1256, May 2014.

[23] C.-W. Tsai and J. J. P. C. Rodrigues, "Metaheuristic scheduling for cloud: A survey," *IEEE Syst. J.*, vol. 8, no. 1, pp. 279–291, Mar. 2014.

[24] M. Kalra and S. Singh, "A review of Metaheuristic scheduling techniques in cloud computing," *Egyptian Informat. J.*, vol. 16, no. 3, pp. 275–295, Nov. 2015.

[25] F. Ramezani, J. Lu, J. Taheri, and F. K. Hussain, "Evolutionary algorithm-based multi-objective task scheduling optimization model in cloud environments," *World Wide Web*, vol. 18, no. 6, pp. 1737–1757, Nov. 2015.

[26] F. Ramezani, J. Lu, J. Taheri, and A. Y. Zomaya, "A multi-objective load balancing system for cloud environments," *Comput. J.*, vol. 7, pp. 1316–1337, Jan. 2017.

[27] R. K. Jena, "Task scheduling in cloud environment: A multi-objective ABC framework," *J. Inf. Optim. Sci.*, vol. 38, no. 1, pp. 1–19, Jan. 2017.

[28] X. Zheng and Y. Cai, "Energy-aware load dispatching in geographically located Internet data centers," *Sustain. Comput., Informat. Syst.*, vol. 1, no. 4, pp. 275–285, Dec. 2011.

[29] Y. Luo and S. Zhou, "Power consumption optimization strategy of cloud workflow scheduling based on sla," *WSEAS Trans. Syst.*, vol. 13, pp. 368–377, Oct. 2014.

[30] G.-N. Gan, T.-L. Huang, and S. Gao, "Genetic simulated annealing algorithm for task scheduling based on cloud computing environment," in *Proc. Int. Conf. Intell. Comput. Integr. Syst.*, Oct. 2010, pp. 60–63.

[31] A. A. Priyato and W. M. Adiwijaya, "Implementation of ant colony optimization algorithm on the project resource scheduling problem," Fac. Inform., Inst. Technol., Telkom, Bandung, 2008.

[32] N. Preve, "Balanced job scheduling based on ant algorithm for grid network," in *Grid Cloud Computing: Concepts, Methodologies, Tools Application*. Hershey, PA, USA: IGI Global, 2012, pp. 1114–1131.

[33] S. Banerjee, I. Mukherjee, and P. Mahanti, "Cloud computing initiative using modified ant colony framework," *World Academy Sci., Eng. Technol.*, vol. 56, pp. 221–224, Oct. 2009.

[34] E. Feller, L. Rilling, and C. Morin, "Energy-aware ant colony based workload placement in clouds," in *Proc. IEEE/ACM 12th Int. Conf. Grid Comput.*, Sep. 2011, pp. 26–33.

[35] S. Pandey, L. Wu, S. M. Guru, and R. Buyya, "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments," in *Proc. 24th IEEE Int. Conf. Adv. Inf. Netw. Appl.*, Oct. 2010, pp. 400–407.

[36] M. A. Tawfeek, A. El-Sisi, A. E. Keshk, and F. A. Torkey, "Cloud task scheduling based on ant colony optimization," in *Proc. 8th Int. Conf. Comput. Eng. Syst.*, 2013, pp. 64–69.

[37] S. Zhan and H. Huo, "Improved PSO-based task scheduling algorithm in cloud computing," *J. Inf. Comput. Sci.*, vol. 9, no. 13, pp. 3821–3829, 2012.

[38] Z. Liu and X. Wang, "A pso-based algorithm for load balancing in virtual machines of cloud computing environment," in *Proc. Int. Conf. Swarm Intell.* Berlin, Germany: Springer, 2012, pp. 142–147.

[39] H. S. Al-Olimat, M. Alam, R. Green, and J. K. Lee, "Cloudlet scheduling with particle swarm optimization," in *Proc. 5th Int. Conf. Commun. Syst. Netw. Technol.*, Apr. 2015, pp. 991–995.

[40] B. Mondal, K. Dasgupta, and P. Dutta, "Load balancing in cloud computing using stochastic hill Climbing–A soft computing approach," *Procedia Technol.*, vol. 4, pp. 783–789, 2012.

[41] G. F. Elhady and M. A. Tawfeek, "A comparative study into swarm intelligence algorithms for dynamic tasks scheduling in cloud computing," in *Proc. IEEE 7th Int. Conf. Intell. Comput. Inf. Syst. (ICICIS)*, Dec. 2015, pp. 362–369.

[42] M. Abdullahi, M. A. Ngadi, and S. M. Abdulhamid, "Symbiotic organism search optimization based task scheduling in cloud computing environment," *Future Gener. Comput. Syst.*, vol. 56, pp. 640–650, Mar. 2016.

[43] S. Suresh, P. B. Sujit, and A. K. Rao, "Particle swarm optimization approach for multi-objective composite box-beam design," *Compos. Struct.*, vol. 81, no. 4, pp. 598–605, Dec. 2007.

[44] S. N. Omkar, R. Khandelwal, T. V. S. Ananth, G. N. Naik, and S. Gopalakrishnan, "Quantum behaved particle swarm optimization (QPSO) for multi-objective design optimization of composite structures," *Expert Syst. Appl.*, vol. 36, no. 8, pp. 11312–11322, Oct. 2009.

[45] S. N. Omkar, D. Mudigere, G. N. Naik, and S. Gopalakrishnan, "Vector evaluated particle swarm optimization (VEPSO) for multi-objective design optimization of composite structures," *Comput. Struct.*, vol. 86, nos. 1–2, pp. 1–14, Jan. 2008.

[46] X. Chen, L. Cheng, C. Liu, Q. Liu, J. Liu, Y. Mao, and J. Murphy, "A WOA-based optimization approach for task scheduling in cloud computing systems," *IEEE Syst. J.*, vol. 14, no. 3, pp. 3117–3128, Sep. 2020.

[47] G. N. Reddy and S. P. Kumar, "Multi objective task scheduling algorithm for cloud computing using whale optimization technique," in *Proc. Int. Conf. Next Gener. Comput. Technol.* Singapore: Springer, 2017, pp. 286–297.

[48] K. Sreenu and M. Sreelatha, "W-scheduler: Whale optimization for task scheduling in cloud computing," *Cluster Comput.*, vol. 7, pp. 1–12, Jun. 2019.

[49] M. S. Sanaj and P. M. Joe Prathap, "An efficient approach to the map-reduce framework and genetic algorithm based whale optimization algorithm for task scheduling in cloud computing environment," *Mater. Today, Process.*, vol. 37, pp. 3199–3208, Mar. 2021.

[50] C. C. Coello, "Evolutionary multi-objective optimization: A historical view of the field," *IEEE Comput. Intell. Mag.*, vol. 1, no. 1, pp. 28–36, 2006.

[51] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, vol. 16. Hoboken, NJ, USA: Wiley, 2001.

[52] P. Ngatchou, A. Zarei, and A. El-Sharkawi, "Pareto multi objective optimization," in *Proc. 13th Int. Conf. Intell. Syst. Appl. Power Syst.*, May 2005, pp. 84–91.

[53] J. Liu, X.-G. Luo, X.-M. Zhang, F. Zhang, and B.-N. Li, "Job scheduling model for cloud computing based on multi-objective genetic algorithm," *Int. J. Comput. Sci.*, vol. 10, no. 1, p. 134, 2013.

[54] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing," *J. Comput. Syst. Sci.*, vol. 79, no. 8, pp. 1230–1242, 2013.

[55] H. Jiang, J. Yi, S. Chen, and X. Zhu, "A multi-objective algorithm for task scheduling and resource allocation in cloud-based disassembly," *J. Manuf. Syst.*, vol. 41, pp. 239–255, Oct. 2016.

[56] N. Srinivas and K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," *Evol. Comput.*, vol. 2, no. 3, pp. 221–248, Sep. 1994.

[57] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw., Pract. Exper.*, vol. 41, no. 1, pp. 23–50, Jan. 2011.

[58] A. Hussain and M. Aleem, "GoCJ: Google cloud jobs dataset for distributed and cloud computing infrastructures," *Data*, vol. 3, no. 4, p. 38, Sep. 2018.

[59] S. K. Panda and P. K. Jana, "A multi-objective task scheduling algorithm for heterogeneous multi-cloud environment," in *Proc. Int. Conf. Electron. Design, Comput. Netw. Automated Verification (EDCAV)*, Jan. 2015, pp. 82–87.

[60] V. Gajera, Shubham, R. Gupta, and P. K. Jana, "An effective multi-objective task scheduling algorithm using min-max normalization in cloud computing," in *Proc. 2nd Int. Conf. Appl. Theor. Comput. Commun. Technol. (iCATccT)*, 2016, pp. 812–816.

[61] S. K. Panda and P. K. Jana, "Efficient task scheduling algorithms for heterogeneous multi-cloud environment," *J. Supercomput.*, vol. 71, no. 4, pp. 1505–1533, Apr. 2015.

DEAFALLAH ALSADIE (Member, IEEE) received the B.Sc. degree in computer science from Taibah University, in 2007, the M.A.Sc. degree in computer science from Latrobe University, Australia, in 2011, and the Ph.D. degree in computer science from RMIT University, Melbourne, VIC, Australia, in 2019. He is currently an Assistant Professor with the Department of Information Systems, College of Computers and Information Systems, Umm Al-Qura University, Saudi Arabia. His research interests include scheduling and resource allocation for parallel and distributed computing systems, data centers, edge computing, and the Internet of Things.