International Journal of Engineering

Journal Homepage: www.ije.ir

Energy-aware Task Scheduling in Cloud Compting Based on Discrete Pathfinder Algorithm

A. Zandvakili, N. Mansouri*, M. M. Javidi

Department of Computer Science, Shahid Bahonar University of kerman, Kerman, Iran

PAPER INFO

Paper history:
Received 10 April 2020
Received in revised form 29 July 2021
Accepted 30 July 2021

Keywords:
Resource Utilization
Energy Efficiency
Cloud Computing
Task Scheduling
Throughput
Makespan
Latency

ABSTRACT

Task scheduling is one of the fundamental issues that attract the attention of lots of researchers to enhance cloud performance and consumer satisfaction. Task scheduling is an NP (non-deterministic polynomial)-hard problem that is challenging due to the several conflicting objectives of users and service providers. Therefore, meta-heuristic algorithms are the more preferred option for solving scheduling problems in a reasonable time. Although many task scheduling algorithms are proposed, existing strategies mainly focus on minimizing makespan or energy consumption while ignoring other performance factors. In this paper, we propose a new task scheduling algorithm based on the Discrete Pathfinder Algorithm (DPFA) that is inspired by the collective movement of the animal group and mimics the guidance hierarchy of swarms to find hunt. The proposed scheduler considers five objectives (i.e., makespan, energy consumption, throughput, tardiness, and resource utilization) as cost functions. Finally, different algorithms such as Firefly Algorithm (FA), Particle Swarm Optimization (PSO), Grasshopper Optimization Algorithm (GOA), and Bat Algorithm (BA), are used for comparison. The experimental results indicate that the proposed scheduling algorithm with FA, BA, PSO, and GOA improved up to 9.16%, 38.44%, 3.59%, and 3.44%, respectively. Moreover, the results show dramatic improvements in terms of resource utilization, throughput, and energy consumption.

doi: *10.5829/ije.2021.34.09c.10*

1. INTRODUCTION

1. 1. Cloud Computing Cloud Computing refers to both the applications delivered as services over the internet and the hardware and system software in the datacenters that provide those services [1]. As shown in Figure 1, the benefits of cloud computing are: on-demand self-service, multi-tenancy, offers resilient computing, fast and effective virtualization, offers advanced online security, location, and device independence, always available, and scales automatically to adjust to allow pay-per-use. Cloud computing delivers different services as a utility to users through the internet. One consequence of this model is that large cloud data centers consume large amounts of energy and produce significant carbon footprints. Researchers have recently taken energy consumption seriously as a contribution to expand the green cloud space. In datacenters, many efficient technologies including dynamic voltage and frequency technology, resource hibernation, and memory optimization are utilized for reducing energy consumption. Achieving a reasonable trade-off among energy consumption, resource utilization, and quality of service (QoS) requirements is a challenging problem, especially with diverse tasks in a heterogeneous environment. A common objective of cloud providers is to develop resource provisioning and management solutions that minimize energy consumption [2]. To achieve this objective, a thorough understanding of energy consumption patterns in complex cloud systems is imperative. Resource utilization is the next challenge of cloud computing. Effective resource scheduling not only reduces resource consumption (increases the resource utilization ratio) but also executes incoming tasks in minimum time (minimizes the makespan).

*Corresponding Author Email: najme.mansouri@gmail.com, n.mansouri@uk.ac.ir (N. Mansouri)

Please cite this article as: A. Zandvakili, N. Mansouri, M. M. Javidi, Energy-aware Task Scheduling in Cloud Compting Based on Discrete Pathfinder Algorithm, International Journal of Engineering, Transactions C: Aspects, Vol. 34, No. 09, (2021) 2124-2136

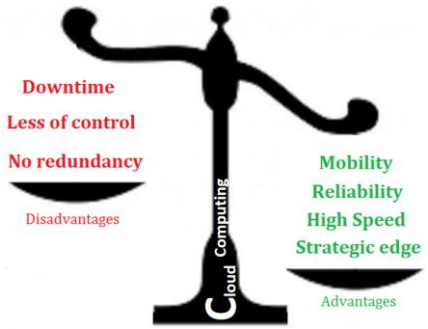

Figure 1. Advantages and disadvantages of cloud computing

At cloud datacenters, inefficient task scheduling may reduce revenue as a result of resource underutilization. In this context, to perform efficient scheduling of tasks on the cloud, the makespan needs to be reduced. In cloud computing, the tardiness parameter is another important objective function. Naturally, a low value for this parameter, makes the response time shorter and thus increases the satisfaction of cloud users. Throughput is the next challenge of cloud computing. Throughput refers to the performance of tasks by a server or device over a specific period. For transaction processing systems, it is normally measured as transactions-per-second. For systems processing bulk data, such as cloud servers, it is measured as a data rate (e.g., Megabytes per second) [3]. Adequate throughput is important to ensure all applications run with optimal efficiency. All these challenges, along with the timely receipt of services with the least delay by the server, depending on the optimal scheduling.

1. 2. Task Sheduling Task scheduling in a distributed heterogeneous computing environment can be identified as a non-linear, multi-objective, NP-hard optimization problem that strives to optimize cloud resource utilization and satisfy the QoS requirements. [4]. Task scheduling is a major challenge in cloud computing, which refers to the technique of mapping a set of tasks to a set of machines to fulfill users' demands. The task scheduling problem can be modeled in different modes. The difference can be based on the number of machines (single or multiple machines), the type of machines (the machines are the same or different), the dependence or independence between tasks. In this paper, machines are capable to perform all tasks, but each task is considered individually (each task does not include smaller parts). Undoubtedly, solving the task scheduling problem is very time-consuming and has a high computational load due to the nature of these types of problems with precise methods, so an acceptable solution can be obtained by choosing the proper meta-heuristic algorithms. There are various types of scheduling algorithms, some of them are static

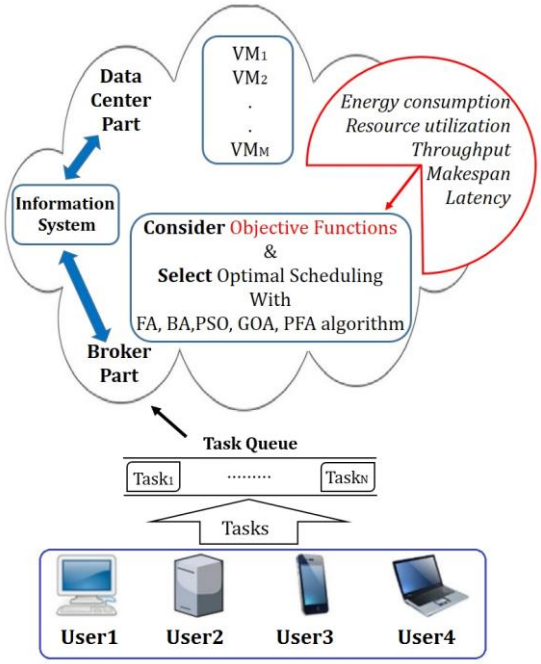

Figure 2. Architecture of task scheduling system

scheduling algorithms that are considered suitable for small or medium scale cloud computing, and dynamic scheduling algorithms that are considered suitable for large scale cloud computing environments.

Figure 2 shows the scheduling system architecture that users send their requests to the cloud and wait for the results. In the cloud environment, the scheduler must select the appropriate machine based on the indicators considered in the objective function and send the tasks to the machines.

Unworthy assignment of tasks to a cloud server can increase the waiting time, makespan, and energy consumption, with inefficient load distribution. An efficient algorithm needs to assign tasks to a suitable set of machines to achieve the desired objectives. Most of the existing task scheduling strategies ignore multiple-objective issues such as energy consumption, makespan, tardiness, throughput, and resource utilization simultaneously, and their main focus is to minimize the cost or completion time of the task scheduling without regarding the QoS metrics. Therefore, the objectives of this paper are to minimize resource utilization, makespan, and energy consumption while maximizing throughput of the cloud servers. We use a meta-heuristic algorithm, named PFA and proposed a new task scheduling algorithm based on Discrete PFA (DPFA). We select PFA for solving task scheduling problem since:

- It can mimic the collective movements of swarms by using the hierarchy between the head and other members of the swarm.

- It has two separate mathematical formulations for position updating of head and other members and tends to move locally in the final steps.
- Can explore the promising solutions, provide the abrupt changes in the initial iterations, and exploit the best one throughout iterations.

In recent years, many optimization algorithms have been used to solve multi-objective task scheduling problem, among them the FA [5], PSO [6], BA [7], and GOA [8] are popular.

Nicolas et al. [9] presented a Multi-Objective Discrete Firefly Algorithm (MO-DFFA) for solving the Flexible Job-shop Scheduling Problem (FJSP). The authors minimized three different objectives simultaneously, the sum of the completion times of the orders, the workload of the critical machine, and the total workload of all machines are their objective functions. They used Genetic Algorithm (GA) and the Greedy Randomized Adaptive Search Procedure (GRASP) for comparison.

The main goal of Kumar and Sharma [10] was to design and develop a task processing framework that has the decision-making capability to select the optimal resource at runtime to process the applications (diverse and complex nature) at virtual machines using modified PSO. The authors proposed algorithm that gives a non-dominance set of optimal solutions and improves various influential parameters (i.e., time, cost, throughput, task acceptance ratio). They used Cloudsim tool environment [11] and PSO, adaptive PSO, artificial bee colony (ABC), BA, and improved min–min load-balancing algorithm for comparison.

Shareh et al. [12] investigated the tasks scheduling problem in open shops using the BA based on ColReuse and substitution meta-heuristic functions. They performed simulations in the MATLAB and used GA for comparison.

In this paper, our main research contributions are as follows:

- Use PFA for scheduling in the cloud computing environment.
- Design a multi-objective scheduling strategy for finding optimal scheduling based on multiple conflicting objectives, namely energy consumption, makespan, tardiness, throughput, and resource utilization.
- Experimental analysis is performed to compare the proposed algorithm with FA, BA, PSO, and GOA.

The rest of the paper is organized as follows: Section 2 discusses the related works, which deal with existing task scheduling techniques in the cloud environment. Mathematical models and problem formulation are discussed in section 3. The technical solution is discussed in section 4. Section 5 deals with performance evaluation and section 6 consists of the conclusion and future work.

2. RELATED WORK

Scheduling is the art of analyzing the required QoS parameters to determine which activity should be performed. There are several conflicting parameters in the task scheduling problem. Using these parameters as objective functions in the optimization algorithms is difficult, time-consuming, and costly (Especially using more than three objective functions). Due to this reason, researchers usees meta-heuristic algorithms to achieve optimal scheduling. Meta-heuristic algorithms have revealed significant performance based on different scheduling approaches.

Agarwal and Srivastava [13] used PSO to minimize the execution time. The proposed PSO-based task scheduling mechanism keeps the overall response time minimum and uses the CloudSim simulator with the existing greedy and GA-based task scheduling mechanism.

Raju and Devarakonda [14] tried to reduce the makespan time in a cloud environment by introducing the new method Modified Greedy PSO with Clustered approach (MGPSOC). The MGPSOC algorithm makes use of clustering with bio-inspired techniques. The authors used PSO and Greedy-PSO algorithms for comparison.

Avinashi et al. [15] presented a novel hybrid method by combining Grey Wolf Optimization (GWO) and PSO. The proposed method optimizes the makespan, execution time, and response time. PSO improves the optimization performance of GWO in the proposed method. The proposed algorithm performance is evaluated with GA and GWO algorithms.

Tabrizchi et al. [16] proposed a novel self-adaptive hybrid Imperialist Competitive Algorithm (ICA)-PSO algorithm for dealing with associate multi-task scheduling problems. The authors combined ICA and PSO to improve the exploration. They used PSO and ICA algorithms for comparison.

Koneti et al. [17] presented a Cost-Effective Firefly-based Algorithm (CEFA) to solve workflow scheduling problems. The proposed CEFA uses a novel method for problem encoding, population initialization, and fitness evaluation intending to provide cost-effective and optimized workflow execution within the limited time. The performance of the proposed algorithm is compared with the PSO algorithm, Robustness-Cost-Time (RCT), Robustness-Time-Cost (RTC), and Regressive Whale Optimization (RWO), in terms of response time and makespan.

The number of iterations is very important in the meta-heuristic algorithm. In the existing approach, the number of iterations is very large which increases the total execution cost and time. Kaur and Mann [18], proposed a Hybrid Cost-Effective Genetic and Firefly Algorithm (GAFFA) for workflow scheduling in cloud

computing which will optimize the number of iterations. They selected execution time, execution cost, and termination delay parameters to compare their method with GA. The proposed method optimizes the makespan, execution time, and response time.

Rajagopalan et al. [19] propounded a novel meta-heuristic algorithm of hybrid FA-GA combination for task scheduling. The proposed algorithm combines the benefits of a mathematical optimization algorithm like FA with an evolutionary algorithm like GA to form a powerful meta-heuristic search algorithm. The performance of the proposed algorithm is compared with traditional First In First Out (FIFO) and Genetic algorithms in terms of execution time.

Adil et al. [20] introduced a job scheduling mechanism based on the Discrete Firefly Algorithm (DFA) to map the grid jobs to available resources. Traditional scheduling mechanisms such as Tabu Search (TS) and hill-climbing used single-path solutions. The proposed scheduling mechanism uses population-based candidate solutions rather than single path solutions, which helps to avoid trapping at the local optimum. The authors used simulation and real workload trace to evaluate their mechanism. The performance of the proposed algorithm is compared with GA and TS in terms of makespan. Bezdan et al. [21] proposed a hybridized BA-ABC for multi-objective task scheduling problems. In the proposed method, the exploration phase of BA is enhanced by the onlooker bee search from the ABC algorithm. The performance of the proposed algorithm was compared with Chaotic Symbiotic Organisms Search (CMSOS) in terms of makespan.

Zhou et al. [22] have used GA to improve the completion time, the quality of work, and the average response time to optimize task scheduling. They used greedy algorithms and improved GA. The novel algorithm named MGGS can find an optimal solution using a fewer number of iterations. The proposed algorithm is compared with GA, Min-Min, and First Come First Service (FCFS).

Taghizadehalvandi and Kamisli [23] discussed the issue of employee shift work schedule. In this case, the goal is to minimize the total amount of employee's work and provide preferences to employees. Under this multi-objective structure, a multi-objective decision model has been created, taking into account the needs of employees. The authors developed weights/priorities of the objective functions. This study is conducted for a company operating in the service part. The number of personnel required for each department and the number of departments are the limitations of these companies. But the proposed model can be used in different types of services and work shifts.

Table 1 shows the summary of the related task scheduling algorithms in the cloud environment. From the review of the existing strategies, we can see that most of those approaches focus on minimizing the execution time or makespan of the task scheduling without regarding the QoS metrics such as throughput and energy consumption of the cloud servers. It causes load imbalance and user dissatisfaction. To increase the efficiency, the proposed algorithm finds the suitable virtual machine for each task using multiple scheduling objectives (i.e., makespan, energy consumption, throughput, tardiness, and resource utilization).

TABLE 1. Some references in task scheduling

Ref.	Year	Algorithm(s)	Compared Methods	Objective Function(s)	Weaknesses
[13]	2019	PSO	GA Greedy	Execution time	Low scalability Not considering cost and deadline
[14]	2021	MGPSOC	PSO Greedy-PSO	Makespan	Single objective
[15]	2021	GWO-PSO	GA GWO	Makespan Execution time Response time	High time complexity High overhead in scheduling
[16]	2021	ICA-PSO	PSO ICA	Makespan	Not considering energy consumption and tardiness
[17]	2021	CEFA	PSO RCT RTC RWO	Makespan Response time	Low scalability Not considering resource utilization and throughput
[18]	2021	Hybrid Cost-Effective Genetic and Firefly Algorithm (GAFFA)	GA	Finish time Execution cost Delay	High time complexity Not comparison with other meta-heuristic algorithms

[19]	2020	Hybrid Firefly-Genetic	GA FIFO	Execution time	High time complexity Not optimized QOS metrixes
[20]	2014	DFA	GA TS	Makespan	Low availability Falling in the local optimum
[21]	2021	BA-ABC	Chaotic symbiotic organisms search (CMSOS)	Makespan Cost	Load imbalance Not considering tardiness
[22]	2020	Improved GA	GA Min-Min FCFS	Makespan Response time	Load imbalance Not comparison with other meta-heuristic algorithms

3. MATHEMATICAL MODELS AND PROBLEM FORMULATION

3. 1. Problem Definition

Definition 1: (Initial preparation time for each task). Which is marked with the symbol S_0.

Definition 2: (Preparation time between tasks). Which is marked with the symbol S. In fact, S is an $N \times N$ matrix.

Definition 3: (The ratio of computational requirements to processing rate of machine ($et_{i,j}$)) [24].

$$et_{i,j} = st_i \,/\, pr_j \tag{1}$$

In Equation (1), st_i is the computational requirements of the $i-th$ task, and pr_j is the processing rate of the $j-th$ machine.

Definition 4: (The completion time of tasks in each machine (CTM)). We indicate the execution time on each machine with the symbol $ST_{i,j}$. So, we have:

$$CTM_j = \sum_{i=1}^{k} \left(et_{i,j} + ST_{i,j} \right) \tag{2}$$

In Equation (2), k *is* the number of tasks performed on the $j-th$ machine.

Definition 5: (The completion time of each task (CTT_i)). Since each task can only be performed on one machine and does not leave the machine until it is completed. CTT_i, is the time interval between $i-th$ task and $(i-1)-th$ task, with the addition of the completion time of the previous tasks.

3. 2. Objective Functions

In this section, five objective functions are introduced.

3. 2. 1. Makespan The makespan of a task scheduling depends on the execution time of each task on the selected machine instance vm_j [25].

$$makespan = \max_{1 \prec j \prec m} \left\{ CTM_j \right\} \tag{3}$$

In Equation (3), CTM_j is completion time of tasks in the $j-th$ machine.

3. 2. 2. Energy Consumption The energy consumption (EC) is the sum of energy consumed on each selected machine. It is characterized by Equation (4) [24].

$$EC_u = \varepsilon_u * \sum_{\forall i \in u} et_{i,u} \tag{4}$$

In Equation (4), EC_u is the energy consumption in the $u-th$ machine, ε_u: is the static energy consumption per time unit of the $u-th$ machine [24].

$$EC = \sum_{\forall u} EC_u \tag{5}$$

In Equation (5), EC is the total energy consumption.

3. 2. 3. Tardiness Lateness is an amount of delay in executing certain operations which is characterized by Equation (6) [26].

$$lateness_i = CTT_i - d_i \tag{6}$$

In Equation (6), CTT_i is the completion time of the $i-th$ task and d_i is the due date of the $i-th$ task. Tardiness is a measure of a delay in executing certain operations (T_i). It is characterized by Equation (7) [27].

$$T_i = \max\left(lateness_i, 0 \right) \tag{7}$$

3. 2. 4. Resource Utilization The resource utilization can be defined as the ratio between the currently used resources to the maximum resource capacity of a vm. The higher resource utilization indicates that a vm with a higher load and low utilization indicates the minimum load of a vm. Resource utilization RU_j is calculated as follows [25]:

$$RU_j = \frac{load_j}{load_j^{\max}} / 100 \tag{8}$$

In Equation (8), $load_j^{\max}$ represents the maximum permissible load of the vm_j.

$$load = std_{1 \prec j \prec m}\left\{CTM_j\right\} \quad (9)$$

In Equation (9), std is the standard deviation of completion time on the $j-th$ machine.

3. 2. 5. Throughput Throughput (TP) is the number of data outputs produced at the end task per time unit. Throughput is calculated by finding the longest or slowest task. Throughput (TP) is calculated as follows [24]:

$$TP = 1/\max\left(et_{i,j}\right) \quad (10)$$

The throughput is directly related to the longest task. The throughput of a system is low if it has at least one long task.

Throughput refers to the number of tasks completed successfully per total processing time [28]. The longest task is a bottleneck in task scheduling. When there is a big task, the finishing time of tasks increased. Therefore, when processing time is limited, fewer tasks are completed. As a result, according to TABLE 2, the throughput is reduced. In this example, it is assumed that there is only one machine. Therefore, the processing power of the machine is the same in both systems.

The general objective function is in the form of Equation (11).

$$\min F(x) = \left(w_1.Z1 + w_2.Z2 + w_3.Z3 + w_4.Z4\right)/\left(w_5.Z5\right) \quad (11)$$

$$Z1: \min\ f_1(x) = \max\left\{CTM_j\right\} \quad (12)$$

$$Z2: \min\ f_2(x) = \max\left\{EC\right\} \quad (13)$$

$$Z3: \min\ f_3(x) = \max\left\{T_i\right\} \quad (14)$$

$$Z4: \min\ f_4(x) = \max\left\{RU_j\right\} \quad (15)$$

$$Z5: \max\ f_5(x) = \max\left\{TP\right\} \quad (16)$$

The used parameters are described in Table 3.

According to Table 4, Some information about tasks and machines is provided in the form of input parameters. During the scheduling, a series of information is calculated, which are considered as calculated parameters. Finally, the optimal scheduling is the output of the system.

TABLE 2. Calculation of throughput for two assumed systems

	Size of task 1	Size of task 2	Size of task 3	Size of task 4	Throughput
System 1	12	14	11	15	0.067
System 2	11	14	19	7	0.053

TABLE 3. Symbols and definitions

N	Number of tasks
M	Number of machines
st_i	The size of the $i-th$ task
$lateness_i$	The lateness of the $i-th$ task
d_i	The due date of the $i-th$ task
T_i	The Tardiness of the $i-th$ task
pr_j	Processing rate of the *j-th* machine
ptt_i	The processing time of the $i-th$ task
CTT_i	The completion time of the $i-th$ task
S_0	Initial preparation time for each task
CTM_j	The completion time on the $j-th$ machine
S_{ab}	Preparation time between tasks ($N \times N$ matrix)
std_j	The standard deviation of completion time on the $j-th$ machine
$et_{i,j}$	The size of the $i-th$ task / the processing power of the $j-th$ machine

TABLE 4. System input, output, and calculated parameters

Inputs	Calculated	Output
N	$lateness_i$	Optimal scheduling
M	T_i	
st_i	CTM_j	
pr_j	TP	
ptt_i	std_j	
CTT_i	EC	
S_0	RU_j	
d_i		
S_{ab}		
$et_{i,j}$		

4. TECHNICAL SOLUTION

4. 1. Pathfinder Algorithm-Continuous PFA is a new swarm-based meta-heuristic algorithm that solves optimization problems with different structure [29]. This

method is inspired by collective movement of animal group and mimics the leadership hierarchy of swarms to find best food area or prey. PFA is able to converge globally optimum and avoids the local optima effectively. In this algorithm, two separated mathematical formulations are used for the position updating of the leader and other members. The mathematical formulation is used for position updating of other members that is characterized by Equation (17) [29].

$$x_i^{k+1} = x_i^k + R_1.\left(x_j^k - x_i^k\right) + R_2.\left(x_p^k - x_i^k\right) + \varepsilon \quad (17)$$

In Equation (17), k is the current iteration, x_i is the position vector of $i-th$ member, x_j is the position vector of the $j-th$ member, R_1 and R_2 are the random vectors.

$$R_1 = \alpha r_1, R_2 = \beta r_2 \quad (18)$$

In Equation (18), α is the coefficient for interaction which defines the magnitude of movement of any member together with its neighbor and β is the coefficient of attraction which sets the random distance for keeping the herd roughly with leader. In this study, α and β are randomly selected in the range of [1, 2].

Also, r_1 and r_2 provide a random movement and are uniformly generated in the range of [0,1]. ε is for vibration that is generated in each iteration using Equation (19).

$$\varepsilon = \left(1 - \frac{k}{k_{\max}}\right).u_1.D_{ij}, D_{ij} = \left\|x_i - x_j\right\| \quad (19)$$

In Equation (19), u_1 is random vectors range in $[-1,1]$, D_{ij} is the distance between two members and $k_{\max}$ is the maximum number of iterations. To look for prey, the mathematical formulation is used for position updating of the pathfinder characterized by Equation (20).

$$x_p^{k+1} = x_p^k + 2r_3.\left(x_p^k - x_p^{k-1}\right) + A \quad (20)$$

In Equation (20), r_3 is a random vector that is uniformly generated in the range of [0,1], A is generated in each iteration using Equation (21).

$$A = u_2.e^{\frac{-2k}{k_{\max}}} \quad (21)$$

where, u_2 is random vectors range in $[-1,1]$.

4. 2. Discretization Technique The task scheduling problem is considered as a discrete optimization problem and PFA is proposed to deal with a continuous optimization problem. Therefore, in this paper, this issue can handle by initially generating solutions using Equation (22) [30].

TABLE 5. Pseudo code of PFA [29].

```
Load PFA parameter
Initialize the population
Calculate the fitness of initial population
Find the pathfinder
While k < k_max
    α and β = random number in [1,2]
    Update the position of pathfinder using Equation (20) and and check the bound
    If new pathfinder is better than old
        Update pathfinder
    end
    for i=2 to maximum number of populations
        Update the position of members using Equation (17) and and check the bound
    end
    calculate new fitness of members
    find the best fitness
    If best fitness< fitness of pathfinder
        Pathfinder=best members
        Fitness= best fitness
    end
    for i=2 to maximum number of populations
        If new fitness of member(i)<fitness of member (i)
            Update members
        end
    end
    generate new A and ε
end
```

$$X_{ij} = floor\left(Lb_{ij} + \alpha.\left(Ub_{ij} - Lb_{ij}\right)\right), \alpha \in [0,1], j = 1,2,...,n \quad (22)$$

In Equation (22), the lower boundary Lb is set to 1, while the upper boundary Ub is set to M. Figure 3 shows the main loop of the proposed algorithms. Discretization in the main loop does not add time complexity.

4. 2. Time Complexity Similar to population-based algorithms such as PSO and GOA, the proposed algorithm has an iteration loop in which some operations are performed according to the dimensions of the problem and the objective functions. Consequently, the computational complexity of the proposed method is $O\left(t\left(n * PS + \left(F\right) * PS\right)\right)$, where t is the number of iterations, n is the number of dimensions, PS is the population size of swarm, and F is the cost of objective [29]. This amount of time consumed to run the algorithm is reasonable and common.

4. 3. Implementation and Parameter Setting First, we determine the number of tasks, the number of machines, the size and time of execution of tasks, as well as the processing power of machines. Then, to solve the task scheduling problem using meta-heuristic algorithms, we need to create an initial population. To create the initial population, we must determine the number of tasks

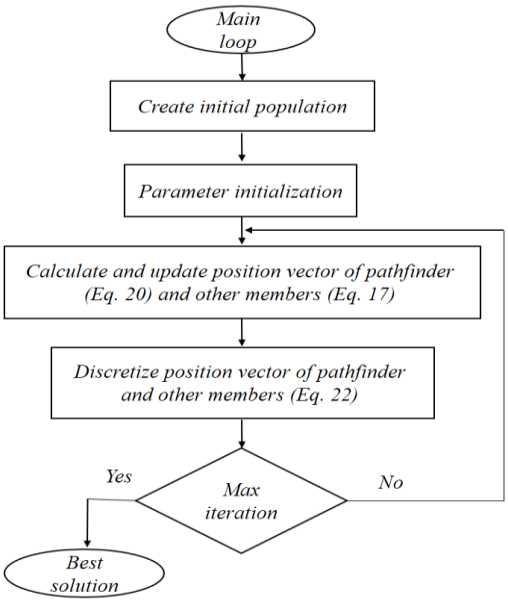

Figure 3. Flowchart of the proposed algorithm

and the number of machines. The x vector is a sample of the population shown in Equation (23):

$$x = \{4,3,1,3,2\} \quad (23)$$

Correspond to this representation, the first task will be accepted on the fourth machine. The second and fourth tasks are accepted on the third machine, while the third task will be accepted on the first machine, the fifth task will be accepted on the second machine. In this paper, we use FA, BA, PSO, and GOA algorithms for comparison. The adjustable parameters of these methods are introduced in Tables 6 to 10. MATLAB software has been used for simulation. First, we create several models, the number of machines is 20 and the number of tasks is 40, 60, and 120. Each of the algorithms is executed 10 times, the number of particles is 50, the max number of iteration is 100. The average results and the worst/best result are given in Tables 11 to 13.

5. PERFORMANCE EVALUATION

This section is dedicated to simulate algorithms and present results. In all tables, bold numbers mean the best results of the algorithms. W, B, and M character indicates worst, best, and average case, respectively.

TABLE 6. Adjustable parameters of BA

Loudness	0.9
Pulse rate	0.1
Frequency minimum	0
Frequency maximum	5

TABLE 7. Adjustable parameters of PSO

$P1$	2.05
$P2$	2.05
P	$P1+P2$
C	$2/\left(P-2+\sqrt{P^2-4P}\right)$
D	0.3
Inertia weight	C
Inertia Weight Damping Ratio	0.1
Social learning factor $c1$	$C\times P1$
Personal learning factor $c2$	$C\times P2$
Velocity of the particles	Range of vm_s

TABLE 8. Adjustable parameters of FA

Light Absorption Coefficient	2
Attraction Coefficient Base Value	2
Mutation Coefficient	0.2

TABLE 9. Adjustable parameters of GOA

$cMin$	0.00001
$cMax$	1

TABLE 10. Adjustable parameters of DPFA

Alpha	Random number in [0,1]
Beta	Random number in [0,1]

In Table 11, scheduling for 40 tasks is considered. The results shows that DPFA for *Z1*, *Z2*, and *Z5* objective functions with an average of 56.64, 706.13, and 0.018, respectively, is the best among the other. Moreover, DPFA has the best result for the total objective function (i.e., Z that is obtained by Equation (11)). The performance gap between DPFA and FA, BA, PSO, and GOA is 3.66%, 77.50%, 6.40%, and 3.58%, respectively.

Table 12 shows the results of the algorithms for scheduling 60 tasks. DPFA for *Z3*, *Z4*, and *Z5* objective functions with an average of 717.00, 5.09, and 0.010, respectively, is the best among the other algorithms. With these interpretations, DPFA has better results than others. GOA is better in terms of run-time, but because DPFA achieves the optimal answer in fewer iterations, it can be concluded that it is better.

Table 13 shows the results of the algorithms for scheduling 120 tasks. The DPFA for *Z1* objective

TABLE 11. Comparison of algorithms for scheduling 40 tasks on 20 machines based on five objective functions

		FA	BA	PSO	GOA	DPFA
	W	**66.19**	190.47	74.30	91.65	84.33
Z1	**B**	47.99	49.32	47.55	47.15	**43.42**
	M	57.21	91.52	65.49	68.41	**56.64**
	W	836.91	832.70	851.75	808.20	**804.90**
Z2	**B**	689.99	620.64	**605.65**	190.86	617.62
	M	749.70	715.29	723.71	519.57	**706.13**
	W	445.90	376.60	377.50	466.85	**302.15**
Z3	**B**	279.15	208.00	240.00	200.90	**150.10**
	M	329.82	279.37	317.13	**219.11**	248.97
	W	6.75	6.68	6.96	**6.01**	6.49
Z4	**B**	5.82	**3.95**	5.83	4.45	5.50
	M	6.33	5.65	6.39	**5.48**	6.11
	W	**0.018**	0.005	0.013	0.010	0.011
Z5	**B**	0.025	0.020	0.023	0.021	**0.028**
	M	0.014	0.012	0.015	0.015	**0.018**
Z		59404.9	101721.6	60977.5	59359.0	**57309.0**
Time		3.8	3.5	3.2	**3.0**	3.5

TABLE 12. Comparison of algorithms for scheduling 60 tasks on 20 machines based on five objective functions

		FA	BA	PSO	GOA	DPFA
	W	508.95	191.05	**111.99**	134.92	116.63
Z 1	**B**	68.80	95.28	71.65	70.85	**68.60**
	M	157.26	135.57	**89.48**	104.69	98.95
	W	1661.71	1759.19	**1551.18**	1835.59	1703.15
Z 2	**B**	1325.53	1343.39	**1220.76**	1063.87	1363.11
	M	1450.70	1500.17	**1373.54**	1577.08	1453.11
	W	1147.30	944.60	1024.75	1059.10	**955.70**
Z 3	**B**	707.95	613.10	657.65	**489.10**	587.55
	M	855.13	734.79	828.76	771.56	**717.00**
	W	6.44	6.12	6.53	**6.12**	6.08
Z 4	**B**	**2.81**	4.29	5.59	4.56	5.18
	M	5.33	5.20	5.98	5.24	**5.09**
	W	0.001	0.005	0.008	0.007	**0.008**
Z 5	**B**	0.010	0.010	0.013	0.014	**0.014**
	M	0.009	0.007	0.010	0.010	**0.010**
Z		415426.2	323743.9	216106.9	207979.9	**205710.3**
Time		4.8	4.1	4.8	**3.9**	5.8

function with an average of 422.17, for *Z2* objective function with an average of 4903.01, for *Z3* objective function with an average of 4140.54, for *Z5* objective function with an average of 0.002 has great efficiency. As a result, the DPFA has a 0.51%, 31.10%, 0.27%, and 5.29% improvement in results compared to the FA, BA, PSO, and GOA, respectively. The high run-time of DPFA is compensated by reaching the solution in a low number of iteration.

TABLE 13: Comparison of algorithms for scheduling 120 tasks on 20 machines based on five objective functions

		FA	BA	PSO	GOA	DPFA
	W	709.05	774.90	**513.25**	547.60	565.95
Z 1	**B**	**264**.62	377.10	368.65	340.45	**313.48**
	M	416.21	572.77	443.56	424.52	**422.17**
	W	6650.81	6249.32	**6410.62**	6410.63	6166.46
Z 2	**B**	4684.75	4842.42	4813.98	4517.74	**4353.34**
	M	5660.80	5671.36	5683.41	4938.35	**4903.01**
	W	4946.20	4973.20	4502.05	4917.95	**4295.50**
Z 3	**B**	3994.00	4048.85	3996.00	**3480.00**	4018.40
	M	4398.53	4329.38	4248.35	4241.53	**4140.54**
	W	4.78	4.26	4.24	4.76	**4.03**
Z 4	**B**	3.18	**3.08**	3.74	3.48	3.57
	M	4.10	**3.63**	3.94	3.91	4.11
	W	0.001	0.001	0.001	0.001	**0.001**
Z 5	**B**	0.003	0.002	0.002	0.003	**0.003**
	M	0.002	0.001	0.002	0.002	**0.002**
Z		4634241.3	6044950.7	4623503.0	4854967.3	**4610850.0**
Time		**2.4**	5.3	6.6	6.1	7.9

TABLE 14. Some weakness of optimization algorithms

Algorithm	weakness
FA	-Easy to fall into local optimum [31] -The solution accuracy is lower -Sticking to local minima [32]
BA	-The low precision of optimization [33] -Easy to fall into local optimum -A poor iterative ability -Its global searchability is weak -It is difficult to process high-dimensional data
PSO	-Easy to fall into local trapping [34] -Premature convergence
GOA	-Easy to fall into local optimum [35] -Slow convergence speed

In Table 14, some weaknesses of the algorithms used in this paper are listed.

Figure 4 compares the algorithms in terms of makespan. The average makespan value is gradually increased with increasing the size of tasks. In a small number of tasks, the algorithms performed almost similarly. As the number of tasks is increased, FA, GOA, and DPFA can complete tasks in less time. Among the algorithms, GOA and DPFA have the best performance. DPFA, in 40, 60, and 120 tasks has a suitable result with minimal fluctuations. Having two separate equations to update the position of the herd members makes the DPFA algorithm works properly. Compared with other optimization algorithms, BA has a poor iterative ability.

Figure 5 compares the algorithms in terms of energy consumption. The average energy consumption value is gradually increased with increasing the number of tasks. The best performance for 40 tasks is related to GOA and the best performance for 60 tasks is related to PSO. When we have 40 or 60 tasks, the other algorithms performed almost similarly. As the number of tasks increased,

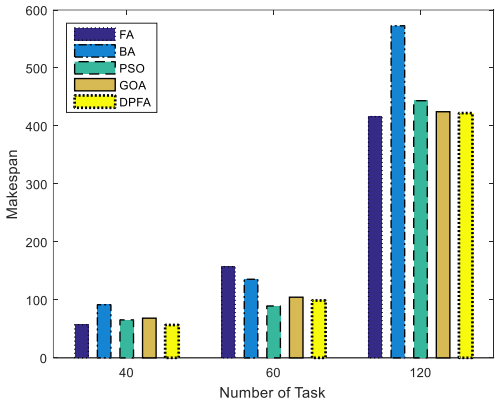

Figure 4. Algorithms' comparison based on makespan

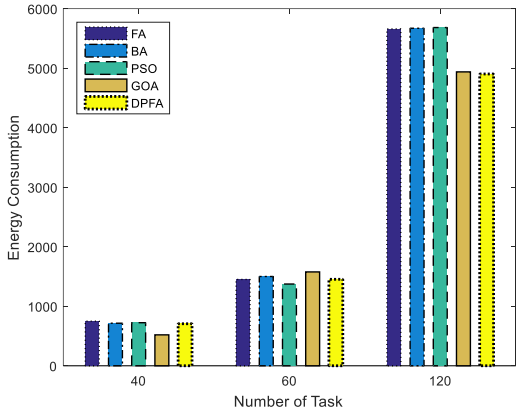

Figure 5. Algorithms' comparison based on energy consumption

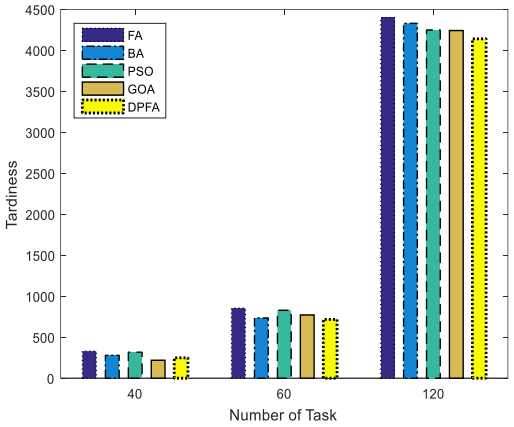

Figure 6. Algorithms' comparison based on tardiness

DPFA can complete tasks with less energy consumption and show the best performance. The high convergence speed of the DPFA causes it to converge in fewer iterations and reduces energy consumption.

Tardiness is the other parameter that is added to the objective function in this paper. Naturally, a small amount of this parameter makes the response time shorter. Among the algorithms, DPFA with the lowest latency is able to have better results than the rest of the algorithms (Figure 6). One of the important factors of DPFA is its speed and simplicity, which has appeared in this result.

Figure 7 compares the algorithms in terms of resource utilization. The lower value of this parameter is better and low utilization indicates the minimum load of a VM. The best performance for 40 tasks is related to GOA, the best performance for 60 tasks is related to DPFA, and the best performance for 120 tasks is related to BA. But on average and due to Table 13, he DPFA is better than the other algorithms.

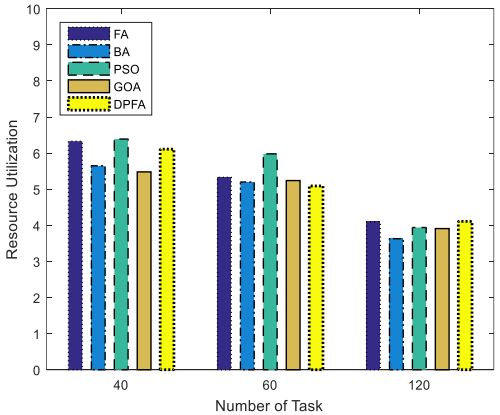

Figure 7. Algorithms' comparison based on resource utilization

Figure 8 compares the algorithms in terms of throughput. The higher value of this parameter is better. The best performance for 40, 60, and 120 tasks is related to DPFA. The DPFA has a high exploration capability in a low number of iterations, so it assigns suitable machines to the tasks and throughput increases. The worst performance for 40, 60, and 120 tasks is related to BA, because it has some shortcomings, such as its global searchability is weak and it is difficult to process high-dimensional data.

The number of iterations is an effective parameter for all algorithms. Figure 9 shows the proper performance of the FA algorithm. In 50 iterations, this algorithm has a completion time of 585426.2. It reduces this time to 233617.0 in 450 iterations that means a 60% improvement. In this respect, it has the largest reduction compared to other algorithms. DPFA does not change with an increasing number of iterations. If the number of iterations is increased, finding of new promising solutions will be difficult because of fluctuation rate A

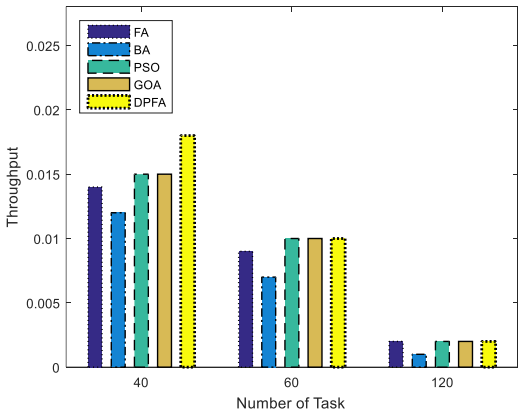

Figure 8. Algorithms' comparison based on throughput

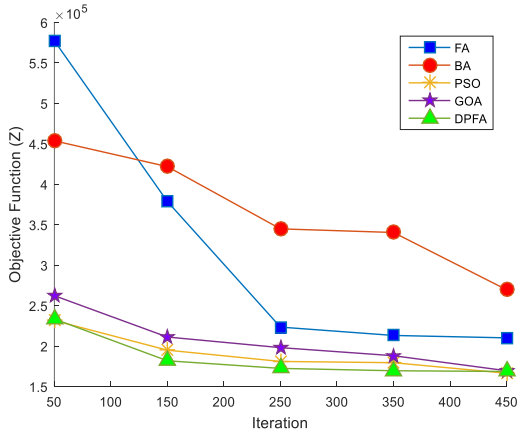

Figure 9. Algorithms' comparison based on iteration and objective function (Z) (60 tasks and 20 machines- each algorithm is run twice and averaged)

and vibration vector ε converging to 0. In fact, the convergence speed of this algorithm has appeared with very good results in a small number of iterations, and increasing the number of iterations does not change the result.

6. CONCLUSIONS

The task scheduling problem is one of the most critical issues in cloud computing because cloud performance depends mainly on it. The main aim of a novel scheduling technique is to evaluate the optimum set of resources available to execute an incoming task such that a scheduling algorithm (scheduler) can then be applied to optimize such diverse QoS parameters as cost, makespan, scalability, reliability, resource utilization, energy consumption, etc.

In this paper, we modeled the objective function based on five parameters (i.e., makespan, energy consumption, tardiness, resource utilization, and throughput). We used a meta-heuristic algorithm, named PFA and proposed a new task scheduling algorithm based on Discrete PFA (DPFA). DPFA has some features (i.e., has two separate mathematical formulations for position updating of head and other members- can explore promising solutions- provide the abrupt changes in the initial iterations- exploit the best one throughout iterations). Due to these reasons, when the number of tasks increases, it has the best performance for makespan, energy consumption, tardiness, and throughput objective functions. On average, among the algorithms, DPFA has the desirable performance and can optimize the total objective function. We suggest using chaos theory to create diversity in the population, adjusting parameters of PFA with the help of tools (i.e., fuzzy system). It can also be implemented to other problems in different areas.

7. REFERENCES

1. Helo. P, Hao. Y, Toshev. R, and Boldosova. V, "Cloud manufacturing ecosystem analysis and design", ***Robotics and Computer Integrated Manufacturing***, (2021), doi:10.1016/j.rcim.2020.102050.
2. Mohammad Hasani zade. B, Mansouri. N, and Javidi. MM, "Multi-objective scheduling technique based on hybrid hitchcock bird algorithm and fuzzy signature in cloud computing", ***Engineering Applications of Artificial Intelligence***, Vol. 104, (2021), doi: 10.1016/j.engappai.2021.104372.
3. Mansouri. N, "An effective weighted data replication strategy for data Grid", ***Australian Journal of Basic and Applied Sciences***, Vol. 6, No. 10, (2012), 336-346.
4. Wang. B, Wang. Ch , Huang. W, Song. Y, and Qin. X, "Security-aware task scheduling with deadline constraints on heterogeneous hybrid clouds", ***Journal of Parallel and Distributed Computing***, Vol. 153, (2021), 15-28, doi: 10.1016/j.jpdc.2021.03.003.
5. Yang. XS, "Firefly algorithms for multimodal optimization", ***Stochastic Algorithms: Foundations and Applications***, (2009), 169-178, doi: 10.1007/978-3-642-04944-6_14.

6. Kennedy. J, and Eberhart. RC, "Particle swarm optimization", ***IEEE International Conference on Neural Networks (Perth, Australia)***, Piscataway, Vol. 5, (1995), 1942-1948, doi: 10.1007/s11721-007-0002-0.
7. Yang. XS, "A new metaheuristic bat-inspired algorithm", ***Studies in Computational Intelligence***, Vol. 284, (2010), 65-74. https://doi.org/10.1007/978-3-642-12538-6_6.
8. Saremi. S, Mirjalili. S, and Lewis. A, "Grasshopper optimisation algorithm: theory and application", ***Advances in Engineering Software***, Vol. 105, (2017), 30-47, doi: 10.1016/j.advengsoft.2017.01.004.
1. 9. Nicolas. A, Rafael. R, David. F, and Raul. P, "A discrete firefly algorithm for solving the flexible job-shop scheduling problem in a make-to-order manufacturing system", ***Central European Journal of Operations Research,*** (2020), doi: 10.1007/s10100-020-00701-w.
9. Kumar. M, and Sharma. SC, "PSO-based novel resource scheduling technique to improve QoS parameters in cloud computing", ***Neural Computing and Applications,*** Vol. 32, (2020), 12103-12126, doi: 10.1007%2Fs00521-019-04266-x.
10. Mansouri. N, Ghafari. R, and Mohammad Hasani Zade B, "Cloud computing simulators: A comprehensive review", ***Simulation Modelling Practice and Theory***, Vol. 104, (2020), doi: 10.1016/j.simpat.2020.102144.
11. Shareh. MB, Bargh. SH, Hosseinabadi. AAR, and Slowik. A, "An improved bat optimization algorithm to solve the tasks scheduling problem in open shop", ***Neural Computing and Applications***, Vol. 33, (2021), 1559-1573, doi: 10.1007/s00521-020-05055-7.
12. Agarwal. M, and Srivastava. GMS, "A PSO algorithm-based task scheduling in cloud computing**",** ***International Journal of Applied Metaheuristic Computing (IJAMC)***, Vol. 10, (2019), doi: 10.4018/IJAMC.2019100101.
13. Raju. YHP, and Devarakonda. N, "Greedy-based PSO with clustering technique for cloud task scheduling", Proceedings of International Conference on Computational Intelligence and Data Engineering, (2021), 133-141, doi: 10.1007/978-981-15-8767-2_12.
14. Kumar. SMS, Krishnamoorthy. P, Soubraylu. S, Venugopal. And JK, Marimuthu. K, "An efficient task scheduling using GWO-PSO algorithm in a cloud computing environment", Proceedings of International Conference on Intelligent Computing, Information and Control Systems, (2021), 751-761, doi: 10.1007/978-981-15-8443-5_64.
15. Tabrizchi. H, Rafsanjani. MK, Balas VE., "Multi-task scheduling algorithm based on self-adaptive hybrid ICA–PSO algorithm in cloud environment", ***Soft Computing Applications,*** (2021), 422-431, doi: 10.1007/978-3-030-52190-5_30.
16. Chakravarthi. KK, Shyamala. L, Vaidehi. V, "Cost-effective workflow scheduling approach on cloud under deadline constraint using firefly algorithm", ***Applied Intelligence***, Vol. 51, (2021), 1629-1644, doi: 10.1007/s10489-020-01875-1.
17. Kaur. I, and Mann. PS, "A hybrid cost-effective genetic and firefly algorithm for workflow scheduling in cloud", International Conference on Innovative Computing and Communications, (2021), doi: 10.1007/978-981-15-5148-2_4.
18. Rajagopalan A, Modale. DR, and Senthilkumar. R, "Optimal scheduling of tasks in cloud computing using hybrid firefly-genetic algorithm", ***Advances in Decision Sciences, Image Processing, Security and Computer Vision***, (2020), 678-687, doi: 10.1007/978-3-030-24318-0_77.
19. Yousif. A, Nor. SM, Abdullah. AH, and Bashir. MB, "A discrete firefly algorithm for scheduling jobs on computational grid", ***Studies in Computational Intelligence***, (2014), 271-290, doi: 10.1007/978-3-319-02141-6_13.
20. Bezdan. T , Zivkovic. M , Tuba. E , Strumberger. I , Bacanin. N, and Tuba. M, "Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm", ***Advances in Intelligent Systems and Computing***, (2021), 718-725, doi: 10.1007/978-3-030-51156-2_83.
21. Zhou Zh, Li. F, Zhu. H, and Xie. H, Abawajy. JH, Chowdhury M U., "An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments", ***Neural Computing and Applications***, Vol. 32, (2020), 1531-1541, doi: 10.1007/s00521-019-04119-7.
22. Taghizadehalvandi. M, and Kamisli. OZ, "Multi-objective Solution Approaches for Employee Shift Scheduling Problems in Service Sectors", ***International Journal of Engineering, Trnsactions C: Aspects,*** Vol. 32, (2019), 1312-1319, doi: 10.5829/ije.2019.32.09c.12.
23. Gu. Y, and Budati C, "Energy-aware workflow scheduling and optimization in clouds using bat algorithm", ***Future Generation Computer Systems***, Vol. 113, (2020), 106-112, doi: 10.1016/j.future.2020.06.031.
24. Adhikari. M, Amgoth. T, and Srirama. SN, "Multi-objective scheduling strategy for scientific workflows in cloud environment a firefly-based approach," ***Applied Soft Computing Journal***, Vol. 93, (2020), doi: 10.1016/j.asoc.2020.106411.
25. Su S, and Yu. H, "Minimizing tardiness in data aggregation scheduling with due date consideration for single-hop wireless sensor networks", ***Wireless Networks***, Vol. 21, (2015), 1259-1273, doi: 10.1007/s11276-014-0853-4.
26. Akyol. DE, and Bayhan. GM, "Multi-machine earliness and tardiness scheduling problem: an interconnected neural network approach", ***The International Journal of Advanced Manufacturing Technology*** , Vol. 37, (2008), 576-588, doi: 10.1007/s00170-007-0993-0.
27. Kumar. M, and Sharma. SC, "PSO-COGENT: Cost and energy efficient scheduling in cloud environment with deadline constraint", ***Sustainable Computing: Informatics and Systems***, Vol. 19, (2018), 147-164, doi: 10.1016/j.suscom.2018.06.002.
28. Yapici. H, and Cetinkaya. N, "A new meta-heuristic optimizer: pathfinder algorithm", ***Applied Soft Computing Journal***, Vol. 78, (2019), 545-568, doi: 10.1016/j.asoc.2019.03.012.
29. Elaziz. MA, and Attiya. I, "An improved henry gas solubility optimization algorithm for task scheduling in cloud computing", ***Artificial Intelligence Review***, (2020), doi: 10.1007/s10462-020-09933-3.
30. Panda. MR, Panda. S, Priyadarshini. R, and Das. P, "Mobile robot path-planning using oppositional-based improved firefly algorithm under cluttered environment", ***Advances in Intelligent Computing and Communication***, (2020), doi: 10.1007/978-981-15-2774-6_18.
31. Yu. Y, Ren. X, Du. F, and Shi. J, "Application of improved PSO algorithm in hydraulic pressing system identification", ***Journal of Iron and Steel Research, International***, Vol. 19, (2012), 29-35, doi: 10.1016/S1006-706X(13)60005-9.
32. Li. X, Zhou. L, Deng. X, Wang. B, Qiu. C, Lu. M, and Li. C, "A resource allocation scheme based on improved bat algorithm for D2D communication system", International Conference in Communications, Signal Processing, and Systems CSPS 2018: Communications, Signal Processing, and Systems, Vol. 515, (2019), 852-861, doi: 10.1007/978-981-13-6264-4_100.
33. Song. B, Wang. Z, and Zou. L, "An improved PSO algorithm for smooth path planning of mobile robots using continuous high-degree bezier curve", ***Applied Soft Computing Journal***, Vol. 100, (2021), doi: 10.1016/j.asoc.2020.106960.
34. Luo. J, Chen. H, Zhang. Q, Xu. Y, Huang. H, and Zhao. X, "An improved grasshopper optimization algorithm with application to financial stress prediction", ***Applied Mathematical Modelling***, Vol. 64, (2018), 654-668, doi: 10.1016/j.apm.2018.07.044.

Persian Abstract

چکیده

زمان‌بندی کار یکی از موضوعات اساسی می‌باشد که توجه بسیاری از محققان را به منظور افزایش عملکرد ابر و رضایت مصرف‌کننده جلب کرده است. زمان‌بندی کار یک مسئله‌ی NP سخت است که به دلیل وجود چندین هدف متناقض، برای کاربران و ارائه دهندگان خدمات، چالش برانگیز است. بنابراین، الگوریتم‌های فراابتکاری برای حل مسئله‌ی زمان‌بندی کار در زمان قابل قبول، گزینه‌ی مناسب‌تری هستند. اگرچه الگوریتم‌های زمان‌بندی کار بسیاری ارائه شده‌اند، اما رویکردهای موجود عمدتا روی به حداقل رساندن زمان اتمام کارها یا میزان مصرف انرژی، متمرکز شده و سایر پارامترهای مهم را نادیده گرفته‌اند. در این مقاله، ما الگوریتم جدیدی را برای زمان‌بندی کار مبتنی بر الگوریتم بهینه‌سازی مسیریاب گسسته (DPFA)، پیشنهاد می‌دهیم که از حرکت جمعی گروه حیوانات الهام گرفته و از سلسله مراتب دسته‌ها برای یافتن شکار تقلید می‌کند. زمان‌بند پیشنهادی پنج هدف (به عنوان مثال، زمان اتمام کارها، مصرف انرژی، توان عملیاتی، تأخیر و استفاده از منابع) را به عنوان توابع هدف در نظر گرفته است. در نهایت، الگوریتم‌های مختلفی مانند الگوریتم کرم شب‌تاب (FA)، الگوریتم بهینه‌سازی ازدحام ذرات(PSO)، الگوریتم بهینه‌سازی خفاش(BA) و الگوریتم بهینه‌سازی ملخ(GOA)، برای مقایسه استفاده شده است. نتایج تجربی نشان می‌دهد که الگوریتم زمان‌بندی پیشنهادی برای تابع هدف زمان اتمام کارها، تا ۹/۱۶ درصد، ۳۸/۴۴ درصد، ۳/۵۹ درصد و ۳/۴۴ درصد به ترتیب در مقایسه با الگوریتم کرم شب‌تاب، الگوریتم بهینه‌سازی ازدحام ذرات، الگوریتم بهینه‌سازی خفاش و الگوریتم بهینه‌سازی ملخ بهبود داشته است. علاوه بر این، نتایج، بهبود چشمگیر از نظر استفاده از منابع، توان عملیاتی، تاخیر و مصرف انرژی را نشان می دهد.