International Journal of Cognitive Informatics and Natural Intelligence
Volume 17 • Issue 1

An Improved Bat Algorithm With Time-Varying Wavelet Perturbations for Cloud Computing Resources Scheduling

Fahong Yu, Shanwei Polytechnic, China*
https://orcid.org/0000-0002-9342-6419
Meijia Chen, Shanwei Polytechnic, China
Bolin Yu, Shenzhen Institute of Information Technology, China

ABSTRACT

Resources scheduling is a major challenge in cloud computing because of its ability to provide many on-demand information technology services according to needs of customers. In order to acquire the best balance between speed of operation, average response time, and integrated system utilization in the resource allocation process in cloud computing, an improved bat algorithm with time-varying wavelet perturbations was proposed. The algorithm provided a perturbation strategy of time-varying Morlet wavelet with the waving property to prevent from local optimum greatly and improve the converging speed and accuracy through the guide of individual distribution to control diversity and time-varying coefficient of wavelets. The experiments showed the proposed could significantly upgrade the overall performance and the capability of resource scheduling in cloud service compared to similar algorithms.

KEYWORDS

Bat Algorithm, Cloud Computing, Morlet Wavelet, Resources Scheduling, Time-Varying Wavelet, Wavelet

INTRODUCTION

Cloud computing is the growing computational technique that depends upon virtualization equipment in response to the user's request via the Internet and dynamic distribution of resources. The services provided by cloud computing are becoming increasingly more diverse and the ability to perform varied and complex tasks involving extremely large amounts of task data, has become necessary. Researches, aimed at meeting the needs of cloud-computing systems considering wider cloud service resource types and collaborative resource scheduling has become the main topic in current studies, which contain cloud computing architecture, management mechanisms, reliability, security, and scalability of resources in cloud-computing environments, and which are capable of hosting more resource models, modeling methods for cloud service resources, resource allocation with collaborative strategy for cloud service, and formulating an optimal, dynamic scheduling method of cloud resources allocation.

Improving the utilization of computing resources, enabling a shorter time to complete tasks, decreasing response time, and improving service quality are the main objectives of resource scheduling

DOI: 10.4018/IJCINI.318651
*Corresponding Author

in cloud computing, which is primarily achieved by allocating tasks to different computing resources in a reasonable manner (Yuan, et al., 2021).

With the growth and applications of the cloud-computing environments, the scheduling problems have achieved gradually more research focuses in recent years. Resource scheduling is the main issue in cloud computing and considers several features, such as total execution cost, fault tolerance, resource utilization, execution time, and energy consumption. The significant measure in task scheduling is load balancing (Jia et al., 2019). It is critical to increase the speed of completing computing tasks and to store less data to take up as little space as possible for customers of pay-as-you-go cloud computing. At the same time, to solve more customers' computing and storage tasks, the cloud service providers improve the efficiency of computing resources by allocating computing resources scientifically and rationally to address specific computing tasks (Gabi et al., 2022). It is critical to reduce the task completion time of users and reasonably consume the resources of operators in order to avoid the waste of computing resources by adapting an effective resource scheduling strategy (Houssein et al., 2021). In selecting the scheduling algorithm, researchers need to concentrate on decreasing the task execution time and makespan value (Sanaj & Prathap, 2020). It is a very critical issue to find an efficient and appropriate cloud-computing resource scheduling strategy and its related algorithm in a cloud-computing environment. The virtual machine plan utilization requests the physical machine (PM) for a certain data point. In consideration of the alterations in the workload over time in the cloud-computing environment, the static algorithms cannot operate; thereby, it requires dynamic methods for balancing the workload among the virtual machines (Abdullahi et al., 2019). Due to the dynamic assets of the cloud-computing environment and heterogeneity, resource scheduling is considered as the nondeterministic polynomial (NP)-hard problem.

The resource-scheduling problem can be solved by applying metaheuristic algorithms, which have acquired growing concentration in recent years. Compared with traditional methods, metaheuristic algorithms are very effective for finding the optimal solution in polynomial time rather than exponential time (Singh & Singh, 2017). There are many metaheuristics, and their alterations were used as resource scheduling strategies in cloud computing. The most typical heuristic algorithms for resource scheduling problems are: the genetic algorithm (GA) (Zhou, et al., 2020), ant colony optimization (ACO) (Jia et al., 2019), particle swarm optimization (PSO) (Pradhan et al., 2021), fish swarm algorithms, and frog leap algorithms. An improved objective function considering the resource scheduling time cost and transportation cost in emergency scheduling of cloud computing resources, based on the genetic algorithm, is proposed, which can effectively improve the efficiency of emergency resource scheduling (Liu et al., 2021). Aimed to improve response time as well as minimize VM failure rate, a multi-objective genetic algorithm for load balancing in a mobile cloud-computing environment was improved, and it performed well by reducing execution time and task wait time at the server (Ramasubbareddy et al., 2021). To improve the service quality of cloud computing, a hybrid multi-objective bandwidth aware divisible (BAT) algorithm, based on the mean square error and the conjugate gradient method, was proposed, and obtained a slightly better cost than the multi-objective genetic algorithm (Zheng & Wang, 2021). These bionic intelligent algorithms can be used to optimize the relative resource scheduling problems to improve resource utilization effectively for the capability of searching and high parallelism. Many researchers developed different algorithms to solve the task-scheduling problem, but there was no perfect algorithm to find the optimal solution for resource scheduling. Although a few of these algorithms describe the enhancement in computing the global optimal solution of the resource scheduling problem, it is affected by premature convergence, so it is difficult to overcome the local minimum when facing a large solution space. These shortcomings lead to sub-optimal solutions that affect system performance and abuse quality of service assurance. It is necessary to construct a new flexible and effective algorithm to solve the global optimal solution of resource scheduling in cloud-computing environments.

The bat algorithm (BA) (Yuan et al., 2021) is a metaheuristic algorithm inspired by emulating a range of bat behaviors, such as searching, locating, and pre-dating on prey through bionic simulation.

To improve the excellent performance for solving resource scheduling in cloud computing, an improved bat algorithm with time-varying wavelet perturbations (IBATWP) was proposed through introducing wavelet perturbation to significantly reduce the convergence time, and using differential perturbation on individual bandwidth aware divisible scheduling (BATS) to prevent the algorithm from being trapped in local optimum, and to address the quality of service and resource scheduling efficiency in cloud computing. The algorithm provided a perturbation strategy of a time-varying Morlet wavelet with the waving property to prevent being trapped,, greatly improving the converging speed and accuracy through the guidance of individual distribution to control diversity and time-varying coefficient of the wavelet.

RELATED WORKS

Definition of the Issue

The issue of resource scheduling in cloud computing has become a hot topic in the research field of cloud computing, which refers to the process of scheduling resources between different resource consumers in a particular cloud environment on the basis of specific rules of resource utilization. Resource scheduling is an important aspect of the cloud-computing paradigm, where the goal is to execute the tasks to determine the best resource for executing each of the requested tasks. The virtual machines (VMs) can be regulated to execute applications by cloud hosts based on the quality of service (QoS) that is user-defined and executed resources, which are sharing in parallel (Srichandan & Kumar, 2018). The basic goal of tackling the resource scheduling problem in cloud infrastructure is the allocation of available cloud-system resources to submitted tasks by achieving one or more objectives, such as minimization of the makespan or time completion of the tasks from the queue, minimization of the total cost, and energy consumption by cloud resources, etc. (abd Elaziz et al., 2019).

In the cloud-computing environment, the objective function of resource scheduling is more focused on the goals of cloud providers and cloud users. For general cloud computing, the service capacity of different resource nodes usually varies and the scheduling relationship changes during execution in the cloud-computing environment. It is useful to implement resource scheduling on applications inside virtual machines by relying on VM-level scheduling techniques combined with a scheduling policy, which are often overly simplified to their scheduling algorithms to degrade overall task performance, as well as too many tasks being considered during scheduling. Optimal resource scheduling generally considers the following goals: service availability, computing cost, load balancing, makespan, and throughput. Therefore, the optimization problem can be transformed into mapping candidate solutions to fitness measurement functions, $R^n \to R$, and obtaining the optimal solution $z \in R^n$ from all feasible solutions, thereby satisfying the function minimization problem described as Equation (1):

$$f\left(x\right) < f\left(y\right), \forall \left(x, y\right) \in R^n \tag{1}$$

Resource scheduling in cloud computing is a valuable cloud resource to balance the load and ensure equal distribution of resources based on demand. The fundamental goal of resource scheduling is to serve the most tasks in the shortest time and to maximize resource utilization, as well as ensure the minimum target scheduling time. From the perspective of cloud computing VMs' task assignment, the resource scheduling model can be represented as a model with four variables set, i.e. $M = \left(U, V, F, A\right)$, where $U = \left(t_0, t_1, \cdots t_i\right)$ is a set of the *i-th* tasks that the user U needs to calculate $V = \left(v_0, v_1, \cdots v_j\right)$, which refers to a set of virtual machines. *F* is an objective function of resource scheduling in cloud computing, and *A* is the scheduling algorithm. The variable *k* representing the

scheduling cost, the variable *c* representing the performing time, the variable *F* representing the objective function for optimizing cloud resource scheduling can be calculated as shown in Equation (2):

$$\min F \begin{cases} s.t. \sum_{\substack{1\le i\le n \\ 1\le j\le m}} b_{ij} t_{ij} + b_{ij} c_{ij} + b_{ij} s_{ij} + b_{ij} k_{ij} \\ b_{ij} = \begin{cases} 0 \\ 1 \end{cases} \end{cases} \tag{2}$$

where the variable $i = \{1, 2, \ldots m\}$ refers to the quantity of tasks, the variable $j = \{1, 2, \ldots n\}$ refers to the quantity of virtual machines, the variable b_{ij} refers to an element of resource scheduling matrix used to describe the utilization of task the *i-th* on virtual machine *j*. The variable b_{ij} refers to a binary matrix b_{ij} in which the task *i* is to be executed in virtual machine *j* will be 1 and zero otherwise. The variables t_{ij}, c_{ij}, s_{ij}, k_{ij} represent running time, computing cost, resource utilization rate, and scheduling cost of task *i* on virtual machine *j*.

To illustrate the resource consumption in cloud computing and general processing capability, each resource node can be represented using three attributes. The first one is processing capabilities, which can be specified by CPU computing power, and second one is load capability, which is indicated by memory size of a node. The resource bandwidth of a node is the third attribute described by a function of the first and second attributes, i.e. if the size and CPU power of the memory is larger, there will be a larger bandwidth. Three vectors for the basic computing system were designed, which concluded the processing capability vector P_n for task T_t, load capability vector L_n for task T_t, and the resource bandwidth vector B_n for task T_t. The notation is specified in Table 1.

The need for sophisticated resource-scheduling algorithms rises with the constant growth in number of submitted tasks. Existing scheduling algorithms and techniques should be improved and new ones should be devised, accordingly. The scheduling of resources must be efficient in order to satisfy end-users' requirements with no negative impact on the service level agreement.

Table 1. The Resource Scheduling Model's Notation

Notation	Meaning
U	a set of users
V	a set of virtual machines
F	objective function of the cloud computing scheduling
A	scheduling algorithm
b_{ij}	i-th task is assigned to j-th VM
k_{ij}	scheduling cost of the task *i* on the virtual machine *j*
t_{ij}	running time of the task *i* on the virtual machine *j*
c_{ij}	computing cost of the task *i* on the virtual machine *j*
s_{ij}	resource utilization rate of task i on virtual node j.
P_n	processing capability vector
B_t	resource bandwidth vector
T_t	computing task
L_n	load capability vector

Scheduling Algorithms

Cloud-computing resource scheduling is a typical NP problem involving multiple coefficients subjected to multiple constraints. The problem of resource scheduling is more challenging for its performance on jobs and virtual machines compared with other distributed environments. Therefore, rational allocation of computing resources has become the key to solving the problem of resource scheduling in cloud-computing environments. Several researchers have proposed many intelligent optimization-based algorithms, which introduced population intelligence-optimization strategies into cloud computing by modeling the biological behavior of certain species to solve resource scheduling optimization problems. As a heuristic policy optimization method, the genetic algorithm has attracted many researchers and is widely applied in scheduling strategies in cloud computing (Pacini et al., 2016; Shukria et al., 2021).

To minimize the time of task completion and transmission, an improved genetic algorithm for a cloud resource-scheduling optimization algorithm was proposed (Liu, et al., 2017). To improve the efficiency and effectiveness of cloud computing specifically, a method that combines immune cloning algorithms with fast multi-objective optimization in cloud resource scheduling is proposed to improve availability and effectiveness (Abualigah et al., 2021; Bezdan et al., 2022; Liu & Wang, 2020). Nowadays, increasingly more intelligent optimization algorithms are also being applied to solve resource scheduling. Particle swarm optimization (PSO), a popular heuristic algorithm, is also used to improve resource scheduling strategies in cloud-computing environments, reduce time of task execution, and improve the quality of service (Ji, 2019; Li et al., 2017), because of enhancing the capability of global search. The main drawback of PSO tends to fall into local optimality as the single objective is the scheduling evaluation index rather than the impact of other coefficients from various aspects. These intelligence optimization algorithms have achieved significant improvements in cloud-computing resource scheduling, but there are still some drawbacks, such as the tendency to fall into local optimality.

As the number of resources in cloud-computing systems continues to increase, resource scheduling becomes increasingly more complex, and the shortcomings of traditional cloud-computing resource scheduling algorithms, such as low resource utilization and load imbalance between nodes, become increasingly prominent. Therefore, to optimize resource utilization is the primary goal.

Original Bat Algorithm

The bat algorithm (BA), which emulates a range of bat behaviors, such as searching, locating, and predating on prey through bionic simulation, optimizes the target to enhance local and global search by simulating the behavior of bats in search space, and compares search targets and flight movements iteratively updating to detect ultrasound and a localized range of behaviors (Garg et al., 2021; Yang, 2010).

For a *N*-dimensional search space, let bat's frequency band be distributed in the range $\left[f_{min}, f_{max}\right]$, with a corresponding wavelength between λ_{min} and λ_{max}, and bat's loudness/volume be *A* and pulse frequency be r, then the substitute strategy for the position x_i^t and speed v_i^t of bat i at time t can be calculated according to Equations (3), (4), and (5):

$$f_i = f_{min} + \beta(f_{max} - f_{min}) \tag{3}$$

$$v_i^t = v_i^{t-1} + \left(x_i^t - x^*\right) f_i \tag{4}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{5}$$

where the variable f_i refers to the frequency of the bat *i*, the variable $\beta \in [0,1]$ refers to a variable that obeys the uniform distribution. The variable x^* refers to an optimal position. The variable λ is defined as a constant and the function λf is monotonously increasing.

Every bat may get a new position in the current local search calculation as Equation (6) after the local search is being carried out:

$$x_{new} = x_{old} + \alpha \overline{A^t} \tag{6}$$

where the variable $\alpha \in [-1,1]$, $\overline{A^t}$ refers to loudness/volume.

The sound intensity, loudness/volume, and frequency of the ultrasonic pulses emitted by the bat are relatively high at the beginning of the hunt. It will be notable that the loudness/volume decreases while the pulse frequency increases when the prey is easy to catch. The updating formulas can be calculated as Equations (7) and (8):

$$A_i^{t+1} = \varphi A_i^t \tag{7}$$

$$r_i^{t+1} = r_i^0 \left[1 - \exp(-\kappa t)\right] \tag{8}$$

where the constant φ, $\kappa \in [0,1]$, in which $A_i^t \to 0$, $r_i^t \to r_i^0$ when $t \to \infty$.

As BA is essentially a globally optimal solution-centric algorithm, premature convergence often happens when it is used to solve multi-peaked function optimization problems. This usually leads to a unidirectional flow of information throughout the group and forces the individuals to converge quickly in a search region near the optimal solution, which leads to a low level of diversity and premature convergence.

IMPROVED BAT ALGORITHM WITH TIME-VARYING WAVELET PERTURBATIONS

The improved bat algorithm adopted a perturbation strategy of a time-varying Morlet wavelet with the waving property to prevent local optimum, and improve the converging speed and accuracy through the guidance of individual distribution to control diversity and the time-varying coefficient of the wavelet.

Time-Varying Wavelet Perturbations

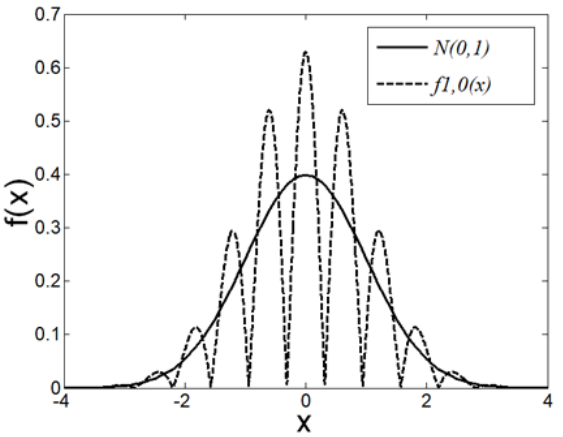
The Morlet wavelet model is a single-frequency sinusoidal exponential that functions under a Gaussian envelope with finite length, zero-mean, and alternating fluctuations around the zero-mean. A modified Morlet mother wavelet is proposed to establish the probability density function of wavelet perturbation in the paper. The synthetic wavelet is a multi-peaked function with the symmetric relation varying periodically along the axis, which can be calculated as as shown in Equation (9):

$$f_{a,0}(x) = C \frac{1}{\sqrt{a}} \left| e^{-x^2/2a^2} \cos\left(5\frac{x}{a}\right) \right| - 2.8a \le x \le 2.8a \tag{9}$$

where the constant *C* refers to the normalization constant, the variable refers to the scale coefficient of the wavelet. The comparing result of the probability density function with a = 1, $C = 0.627$, and the standard Gaussian distribution function *N (0,1)* are shown in Figure 1.

As can be seen in Figure 1, the entire distribution $f_{1,0}$ was wrapped under a Gaussian envelope appearing to be in a wave-like fluctuation. The probability was 99% distribution in [-2.8, 2.8]. This probability distribution exhibits a distinct periodic variation like that of the wave feature and the distribution *N (0, 1)* in its envelope compared to the Gaussian distribution, making it well suited for use as a perturbation coefficient to adjust the position of the cluster and maintain cluster diversity.

Figure 1. Image of f1,0 (x) and N (0,1)



IBATWP Description

In order to correct the shortcomings of the bat algorithm, a wavelet perturbation coefficient, based on the probability density distribution $f_{1,0}$ was applied to improve the bat algorithm.

The centroid of the bat population can be acquired, and the individual position with reference to the population centroid can be solved through perturbation of the Morlet wavelet in each iteration of the bat algorithm evolution. Therefore, the optimal solution set will be produced by replacing the original position with the current best optimal position.

The vector $x_i^k = \left(x_{i1}^k, x_{i2}^k, \ldots, x_{ij}^k\right)$ of the i-th bat in the k-th iteration represents the bat position set to be the vector of the center-of-group positions. The current center position of population used as the reference position for a single perturbation operation is displayed in Equation (10):

$$\overline{x^k} = \frac{1}{n}\sum_{i=1}^{n} x_i^k \tag{10}$$

A wavelet perturbation can be added to the positions of individual bats with a probability. The wavelet perturbation function is indicated as Equation (11):

$$\left(x_i^k\right) = x_i^k + \sigma\left|x_i^k - \overline{x^k}\right| \tag{11}$$

where the vector $\left(x_i^k\right)'$ indicates the position that obeys the randomly distributed $f_{1,0}$ under the perturbation range located at [0,0.8], it will get population diversity due to the distance $\left(x_i^k\right)'$ from the center $\overline{x^k}$ of mass of the population and the fact that the distribution was calculated based on the vector is periodic by perturbation operation.

Let the variable H_t refer to the entropy of bat group P_t, then the entropy H_t can be calculated as presented in Equation (12):

$$H_t = -\sum_{i=0}^{K-1} \frac{n_i}{N} \log \frac{n_i}{N} \tag{12}$$

$$q_t = 1 - H_t \tag{13}$$

The entropy H_t is a valid indicator to measure the diversity of the individual set. The population is more evenly distributed throughout the search space if the value of entropy H_t is close to 1, while the population diversity is lower if the value of entropy H_t is close to zero. The substitution rate is indicated as the variable q_t, which can be used to keep the diversity of the bat group simultaneously and improve the accuracy and speed of the bat algorithm.

Comparing the perturbations generated by uniform and Gaussian distributions, the wavelet perturbations have better Gaussian properties when the Gaussian envelope of the wavelet perturbation is set in the range of [-4, 4]. As the probability density of the wavelet perturbation is a periodic oscillating character, with more frequent and dramatic change uncertainty, the diversity of the bat group can be maintained by wavelet perturbation.

In view of the above design, the framework of the improved bat algorithm with time-varying wavelet perturbations for optimizing resource scheduling in cloud-computing environments is described in Table 2.

EXPERIMENTAL RESULTS AND ANALYSIS

The experiments were performed on standard benchmarks to solve a resource scheduling problem in cloud computing. In the first part, the authors presented simulation results for standard benchmarks of their proposed IBATWP algorithm. Real performance and improvements over the basic BA version can be evaluated only by conducting tests on a wider set of problems specifically designed for benchmark purposes. The second part of the experimental section provided simulation results for instances of cloud resource scheduling problems. IBATWP's performance metrics for challenging real world NP-hard problems were evaluated.

Tests With Standard Benchmark Functions

In order to verify the performance of the proposed algorithm, some experiments were implemented on four typical benchmark functions, and resources scheduling was simulated on Matlab r2020a simulation platform. There were two minimization problems including *f1* (Rastrigin function) and *f2* (Griewank function), and two maximization problems including *f3* (Schaffer function) and *f4* (NiH function). The settings of each function are listed in Table 3.

For IBATWP, the pulse frequency was initialized in the range from 0 to 10, the maximum pulse amplitude was set to 0.85, the maximum loudness/volume was set to 0.05, a loudness/volume decay

Table 2. Framework of Improved Bat Algorithm With Time-Varying Wavelet Perturbations

1)	Input: *MAXITER* (Maximum value of iteration), P_size (Population size), f_i (pulse frequency), r_0 (maximum pulse frequency), A (maximum pulse loudness), μ (loudness decay coefficient), γ (frequency increment coefficient) and ω (search accuracy);
2)	initializing initial population (P_size) based on uniform distribution
3)	while (Termination condition is not satisfied)
4)	Fitness evaluation according to Equation (1)
5)	for i = 1:n
6)	alter frequency, speeds, and positions according to Equations (2), (3) and (4)
7)	if (A generated random variable r1 > r_i) then
8)	Obtain the optimal set of solutions for the wavelet perturbation and a local optimal solution
9)	endif
10)	end for
11)	Generate a random solution set
12)	if (rand () > A_i) & ($f(x_i) < f(x^*)$) then
13)	update the position of all bats, the pulse frequency r and loudness amplitude A according to Equations (5), (6) and (7)
14)	endif
15)	Acquiring time-varying wavelet parameter according to Equation (8)
16)	calculate the center $\overline{x}$ of current population and the wavelet perturbation model according to Equations (9) and (10)
17)	Replace individuals according to Equations (11) and (12), and replace the optimal individuals based on the wavelet perturbation
18)	Acquiring the optimal solution set and the global optimal solution x_{best}^*
19)	end
20)	output x_{best}^*

Table 3. Settings for Test Functions

Function	Dimension	Domain	Population Size	Type	Optimum	Max. Eval
f_1	30	[-600, 600]	200	Min.	0	4×10^5
f_2	30	[-30, 30]	200	Min.	0	4×10^5
f_3	2	[-100, 100]	60	Max.	1	4×10^5
f_4	2	[-5.12, 5.12]	60	Max.	3600	4×10^5

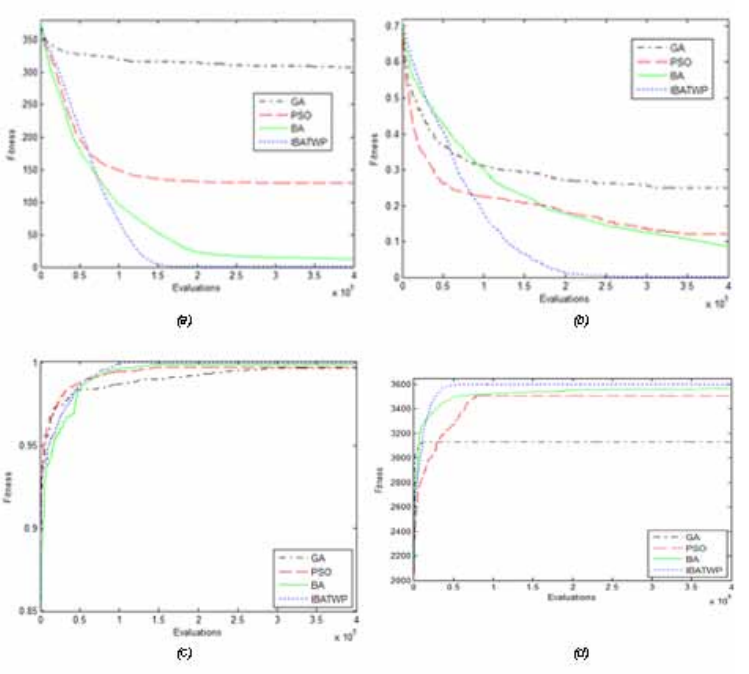
coefficient was set to 0.985, and the pulse amplitude coefficient was set to $K = 0.01$. The maximum pulse frequency r^0 was set to 0.75, the intensity of maximum pulse sound A was set to 0.25, the coefficient k of sound intensity attenuation was set to 0.975, the coefficient γ of pulse frequency increase was set to 0.01, the coefficient C of normalization constant of wavelet variation was set to -0.627, the scaling coefficient α was set to 1, and the range of wavelet variation coefficient was set to [-2.8, 2.8].

The experimental results, based on basic BA (Gandomi et a1., 2013; Potts et al., 1994) and PSO (Huang et a1., 2020), compared with IBATWP on the resource scheduling in cloud computing, were presented. All the algorithms ran 30 times independently on the test problems with setting pop to 200. The maximum number of evaluations was set to 4×10^5. The indicators used to weight the performance of the algorithms included mean and std, and the results are shown in Table 4. The convergence of GA, PSO, BA, and IBATWP, on above benchmark functions, were shown in Figure 2.

Table 4. Statistic Results on f1 ~ f4 for Four Algorithms

Function	GA		PSO		BA		IBATWP	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
f_1	307.6841	11.372068	126.0423	8.471938	18.3348	5.63531	3.2116e-2	7.3552e-4
f_2	0.24154	0.240343	0.98215	0.167724	0.087289	0.031586	2.5075e-2	4.3093e-2
f_3	0.97166	0.321034	0.98197	0.315614	0.98956	1.827e-1	0.9983	1.321e-3
f_4	3.1319e+3	43.2691	3.4831e+3	55.21177	3.5512e+3	25.50383	3.6e+3	0

Figure 2. Convergence Curves of f1 ~ f4 in Picture (a) ~ (d), respectively



The results of algorithms executed on test cases $f_1 \sim f_4$ for 30 times, respectively, were recorded in Table 4. The mean of IBATWP was superior to the other three algorithms, and the corresponding variance was also small, which showed that IBATWP has better convergence ability and stability than the other three algorithms. Viewing Figure 3, it can be seen that the proposed algorithm has the capability to decrease the potential to fall into a local optimum and the time required to reach the objective function. By comparing the convergence curves of algorithms GA, PSO, BA, and IBATWP on four test functions, the algorithm IBATWP converged slowly at an early stage, while it was capable of getting rid of the local trap effectively and converge had a high rate, which indicated that the performance and efficiency of the IBATWP was higher than that of GA, PSO, and BA. Therefore, it may be a very important algorithm for some engineering optimization problems.

Resource Scheduling in the Cloud Environment

In the paper, GA, PSO, BA, and the proposed algorithm were applied to solve resource-scheduling problems in cloud-computing environments at the optimization objective function as Equation (1). The individuals of the group were described as the resources in cloud computing scheduling. The CloudSim platform was applied as the test platform to simulate experiments, and the Datacenter Broker was applied as the container to execute the algorithms. In the resource scheduling, the virtual tasks were set to 500, the virtual nodes were set to 50, and the iterations value was set to 100. IBATWP was compared with the GA, PSO, and BA algorithms. The results are shown in Figures 3 and 4.

Figure 3. Comparison of Resources Scheduling Algorithms for Once Task Completion Time

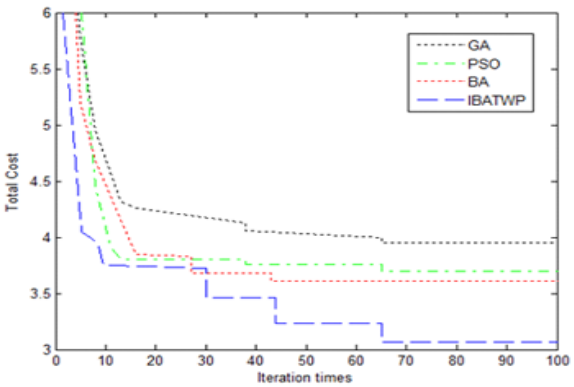
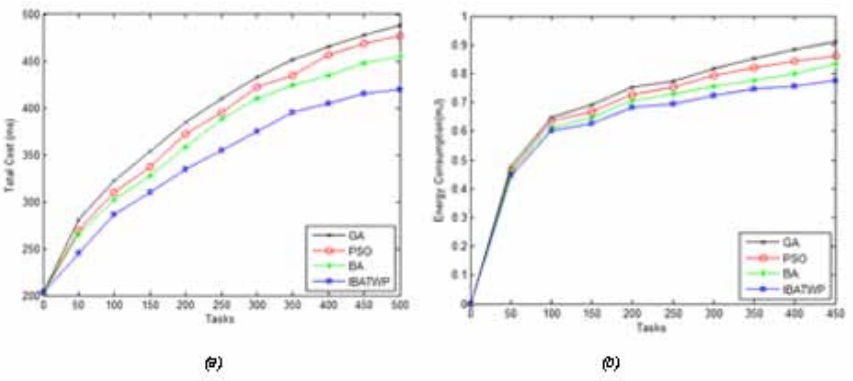


Figure 4. Comparison on Consumption

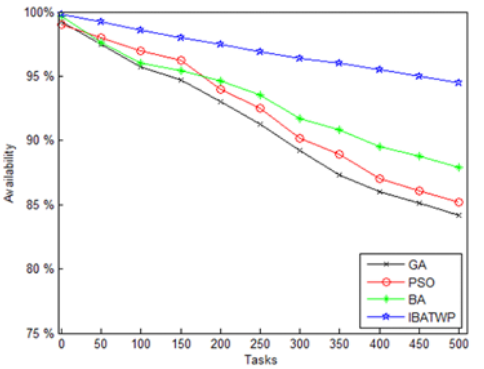


As can be seen from Figure 3, in terms of cost to complete the task, IBATWP shows a clear advantage as the number of iterations increases, while GA, BA, and POS indicated a slow response and poorer results in the later stages of evolution.

From Figure 4, it is clear that the total cost of the four algorithms executing for resource scheduling was not very different during the first stage, while the total cost of the proposed algorithm was lower than GA, BA, and POS, with the increasing of tasks. At the same time, the energy consumption of IBATWP was lower than GA, BA, and POS. IBATWP can save the total cost and obtain good results.

Validity represents the one-time success rate of resource scheduling. It can be seen in Figure 5 that the effective allocation of virtual machines for user tasks presents a decreasing trend as the number of tasks increases, but the success rate has little impact on the algorithm results. Comparing GA, BA, and POS, IBATWP outperforms the other three algorithms in selecting virtual machine resource allocations for user tasks, which indicates the proposed algorithm is effectiveness to solve the resources scheduling problem.

Figure 5. Comparison Availability of Three Resource Scheduling Algorithms



Comparison of Resource Scheduling Indicators

In order to analyze the effectiveness and suitability of the proposed algorithm, experiments were performed regarding resource allocation effectiveness by comparing IBATWP, GA, BA, and POS algorithms.

Executing Time

Executing time of the tasks in a cloud-computing environment contains the time of task waiting, task computing, and task transfer. In the experiment, the average execution time under IBATWP was much lower than that under GA, BA, and PSO, as shown in Figure 6, which indicated that the proposed algorithm was capable of facilitating the improvement of resource utilization.

Resource Utilization Analysis

In resource scheduling of cloud services, whether the resource utilization is reasonable has become an important measuring part for users' satisfaction, which can be indicated as ratio ($k=s/w$). In the ratio, the variable s represents the actual resource utilization and w represents the ideal resource utilization. When k is greater than 1, it means that the actual resource allocation is greater than the ideal situation. On the contrary, the actual resource allocation is less than the ideal situation. The result of average ratio of cloud-computing resource scheduling executed by GA, PSO, BA, and IBATWP is shown in Figure 7. It is clear that IBATWP can achieved good results compared to GA, PSO, and BA, in both time and resource allocation.

In the cloud computing system, the time required to complete tasks will be reduced greatly if more computing resources are provided. When the number of virtual machines increases from 20 to 50, the time required to complete tasks in IBATWP, GA, PSO, and BA algorithms are shown in Figure 8.

From Figure 8, it is clear that increasing computing resources can significantly reduce the time required for task execution in cloud-computing resource scheduling. With the increase of resources, the IBATWP algorithm has a more significant optimization effect than the PSO, GA, and BA algorithms.

Figure 6. Comparison of the Task Completion

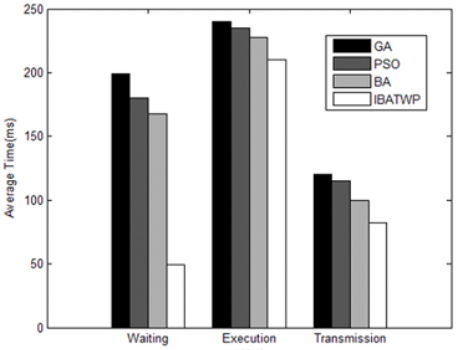


Figure 7. Average Ratio of Cloud Computing Resource Scheduling

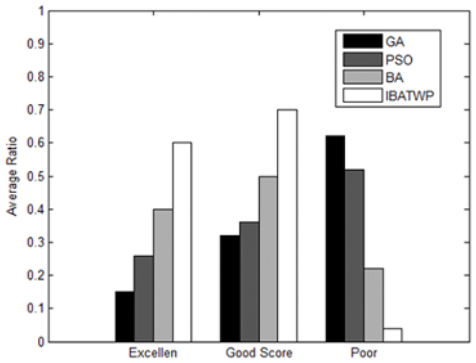
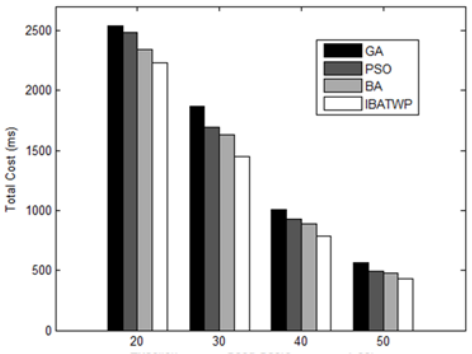


Figure 8. The Impact of the Number of Virtual Machines on Total Cost



The experimental results reveal that IBATWP showed up to 19.68% lower time cost than BA for resource scheduling in cloud computing. This is because the proposed approach use of a time-varying wavelet perturbations strategy to maintain a balance between the local and global search. The overall results reveal that the proposed approach outperforms time cost and energy consumption, which is more stable and scalable than the selected approaches. IBATWP in all experiment instances outperformed basic BA, PSO, and GA. It was clear that IBATWP proved to be robust and an efficient optimization technique for resolving the NP-hard problem of resource scheduling.

CONCLUSION

The BA resource-scheduling algorithm is considered a suitable approach for a load balanced scheduling of tasks due to its fast convergence and easy to implement nature. However, the BA suffers from a premature convergence issue. In this paper, attempting to achieve optimal balance between the executing speed, the average response time, and the overall system utilization during cloud computing's resource allocation, an improved bat algorithm with time-varying wavelet perturbations for resources scheduling optimization in cloud computing was proposed. The experimental results revealed that the IBATWP outperformed GA, PSO, and BA on resource scheduling in cloud computing services, and IBATWP showed up to 19.68% lower time cost than BA for resource scheduling in a cloud-computing environment. This is because the proposed approach used a time-varying wavelet perturbations strategy to maintain a balance between the local and global search. The overall results revealed that the proposed approach outperformed time cost and energy consumption, and was more stable and scalable than the selected approaches. The load balancing issues can be considered as future work in the cloud-computing resource management phase.

FUNDING AGENCY

This research was supported by the School Scientific Foundation Grant [No. SKQD2021B-035]; Innovative Industrial Design and Research Institute Project Grant [No. cxgy21-004]; Higher Education Collaborative Education Foundation of the Education Ministry Grants [No. 202101347006] and [No. 202101355031]; the National Natural Science Foundation of China Grant [No. U1801266]; Guangdong University Innovation Team Project Grant [No. 2020KCXTD045]; Guangdong Higher Vocational Education Teaching Reform Research and Practice Project Grant [No. GDJG2021385].

COMPETING INTEREST STATEMENT

We state that there is no known conflict of interest to disclose.

REFERENCES

abd Elaziz, M., Xiong, S., Jayasena, K. P. N., & Li, L. (2019). Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution. *Knowledge-Based Systems, 169*, 39-52.

Abdullahi, M., Ngadi, M. A., Dishing, S. I., & Ahmad, B. I. E. (2019). An efficient symbiotic organisms search algorithm with chaotic optimization strategy for multi-objective task scheduling problems in cloud computing environment. *Journal of Network and Computer Applications*, *133*, 60–74.

Abualigah, L., Diabat, A., & Elaziz, M. A. A. (2021). Intelligent workflow scheduling for Big Data applications in IoT cloud computing Diabat environments. *Cluster Computing*, *24*(4), 2957–2976.

Bezdan, T., Zivkovic, M., Bacanin, N., Strumberger, I., Tuba, E., & Tuba, M. (2022). Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm. *Journal of Intelligent & Fuzzy Systems*, *42*(1), 411–423.

Gabi, D., Dankolo, N. M., Muslim, A. A., Abraham, A., Joda, M. U., Zainal, A., & Zakaria, Z. (2022). Dynamic scheduling of heterogeneous resources across mobile edge-cloud continuum using fruit fly-based simulated annealing optimization scheme. *Neural Computing & Applications*, *34*(16), 14085–14105.

Gandomi, A. H., Yang, X. S., Alavi, A. H., & Talatahari, S. (2013). Bat algorithm for constrained optimization tasks. *Neural Computing & Applications*, *22*(6), 1239–1255.

Garg, M., Kaur, A., & Dhiman, G. (2021). *A Novel resource allocation and scheduling based on priority using metaheuristic for cloud computing environment*. Impacts and Challenges of Cloud Business Intelligence.

Houssein, E. H., Gad, A. G., Wazery, Y. M., & Suganthan, P. N. (2021). Task scheduling in cloud computing based on meta-heuristics: Review, taxonomy, open challenges, and future trends. *Swarm and Evolutionary Computation*, *62*(17), 100841.

Huang, X., Li, C., Chen, H., & An, D. (2020). Task scheduling in cloud computing using particle swarm optimization with time varying inertia weight strategies. *Cluster Computing*, *23*(7), 1137–1147.

Ji, K. (2019). Cloud computing resource scheduling optimization based on ant colony algorithm with dynamic trend prediction. *Bulletin of Science and Technology*, *16*(9), 216–4225.

Jia, Z., Yan, J., Leung, J. Y., Li, K., & Chen, H. (2019). Ant colony optimization algorithm for scheduling jobs with fuzzy processing time on parallel batch machines with different capacities. *Applied Soft Computing*, *75*, 548–561.

Li, J., Ma, T., Tang, M., Shen, W., & Jin, Y. T. (2017). Improved FIFO scheduling algorithm based on fuzzy clustering in cloud computing. *Information*, *8*(1), 25.

Liu, D., Yao, Z., & Chen, L. (2021). Emergency scheduling optimization simulation of cloud computing platform network public resources. *Complexity*, *2021*, 1–11.

Liu, S., & Wang, N. (2020). Collaborative optimization scheduling of cloud service resources based on improved genetic algorithm. *IEEE Access: Practical Innovations, Open Solutions*, *8*(2), 15878–15890.

Liu, X., Zhang, X., Li, W., & Zhang, X. (2017). Swarm optimization algorithms applied to multi-resource fair allocation in heterogeneous cloud computing systems. *Computing*, *99*, 1231–1255.

Pacini, E., Mateos, C., & Garino, C. G. (2016). Multi-objective swarm intelligence schedulers for online scientific clouds. *Computing*, *98*(5), 495–522.

Potts, J. C., Giddens, T. D., & Yadav, S. B. (1994). The development and evolution of an improved genetic algorithm based on migration an artificial selection. *IEEE Transactions on Systems, Man, and Cybernetics*, *24*(1), 73–86.

Pradhan, A., Bisoy, S. K., & Das, A. (2021). A survey on PSO based meta-heuristic scheduling mechanism in cloud computing environment. *Journal of King Saud University-Computer and Information Sciences.*

Prasanna Kumar, K. R., & Kousalya, K. (2020). Amelioration of task scheduling in cloud computing using crow search algorithm. *Neural Computing & Applications*, *32*(1), 5901–5907.

Ramasubbareddy, S., Swetha, E., Luhach, A. K., & Srinivas, T. A. S. (2021). A multi-objective genetic algorithm-based resource scheduling in mobile cloud computing. *International Journal of Cognitive Informatics and Natural Intelligence*, *15*(3), 58–73.

Sanaj, M. S., & Prathap, P. J. (2020). Nature inspired chaotic squirrel search algorithm (CSSA) for multi objective task scheduling in an IAAS cloud computing atmosphere. *Engineering Science and Technology, an International Journal, 23*(4), 891-902.

Shukri, S. E., Al-Sayyed, R., Hudaib, A., & Mirjalili, S. (2021). Enhanced multi-verse optimizer for task scheduling in cloud computing environments. *Expert Systems with Applications*, *168*, 114230.

Singh, N., & Singh, S. B. (2017). A modified mean gray wolf optimization approach for benchmark and biomedical problems. *Evolutionary Bioinformatics Online*, *13*.

Srichandan, S., & Kumar, T. A. (2018). Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm. *Future Computing and Informatics Journal*, *3*(2), 210–230.

Wei, X. (2020). Task scheduling optimization strategy using improved ant colony optimization algorithm in cloud computing. *Journal of Ambient Intelligence and Humanized Computing*, 1–12.

Yang, X. S. (2010). A new metaheuristic bat-Inspired algorithm. *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, 65-74.

Yuan, H., Bi, J., & Zhou, M. (2021). Spatial task scheduling for cost minimization in distributed green cloud data centers. *IEEE Transactions on Autoation. Science and Engineering*, *16*(2), 729–740.

Zheng, J., & Wang, Y. (2021). A hybrid multi-objective bat algorithm for solving cloud computing resource scheduling problems. *Sustainability*, *13*(14), 7933.

Zhou, Z., Li, F., Zhu, H., Xie, H., Abawajy, J. H., & Chowdhury, M. U. (2020). An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments. *Neural Computing & Applications*, *32*(11), 1531–1541.

Fahong Yu received his PhD in Computer Science and Technology from Wuhan University, China, in 2012. In 2012 he was hired as the assistant professor at Jiaxing University and is currently a vice-professor for Shanwei Polytechnic and a guest professor for Yangtze University. He writes and presents widely on issues of intelligent computing.