Cloud task scheduling using enhanced sunflower optimization algorithm

[H]ojjat imami

***Abstract*—The objective of cloud task scheduling is to partition tasks on shared resources to minimize energy consumption and makespan. Recently, several meta-heuristics for task scheduling were proposed and achieved encouraging results. However, their performance is far from the ideal state and needs more improvement. This paper introduces an enhanced sunflower optimization (ESFO) algorithm for improving the performance of existing task scheduling. It finds optimal scheduling in a polynomial time. The experiments show that ESFO outperformed its counterparts. The amount of improvement in comparison with the best counterpart is 0.73and energy consumption. c 2021 The Korean Institute of Communications and Information Sciences (KICS). Publishing services by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)**

***Index Terms*—Keywords: Cloud computing; Task scheduling; Energy consumption; Enhanced sunflower optimization algorithm**

I. Introduction

Cloud computing as an important branch of distributed

computing allows users to access on-demand computing services including storage, servers, databases, and software over

the Internet [1]. As shown in Fig. 1, to use services, users submit their requests to the task manager module. The task scheduler gets the requests from the task manager, analyses

the available resources with the help of the resource information server, and partitions tasks among virtual machines

(VMs) by using task scheduling algorithms. An efficient task scheduling method permits the SP to maximize the utilization of resources. Task scheduling is an NP-hard problem [2]. Recently, several task scheduling algorithms have been proposed [3]. The existing algorithms can be classified into three groups [4]:

game theory-based method, dynamic method, and meta-heuristic. Table 1 lists some of the recent algorithms for

task scheduling in cloud computing. The game theory-based algorithms model the scheduling process as a game and define utility functions to assess the performance of nodes and balance the resource utilization rate of multiple nodes.

Dynamic scheduling algorithms focus on the real-time situation of available resources, tasks, and dynamic changes to schedule tasks [4]. Meta-heuristics are an efficient choice to solve NP-hard problems. They formulated task scheduling as an optimization problem. Meta-heuristics are not quarantined to find optimal solutions but provide near-optimal solutions

Identify applicable funding agency here. If none, delete this.

H. Emami

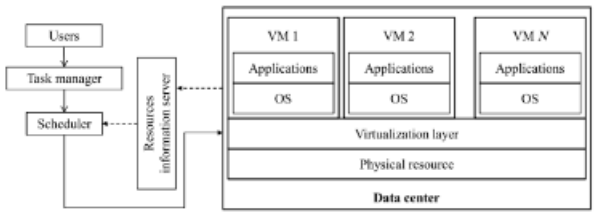

Fig. 1. An overview of task scheduling architecture in cloud computing [2].

with minimum complexity. The majority of meta-heuristic-based methods achieve encouraging results; however, they

suffer from trapping in local optima, and their performance is far from the ideal state. To improve the performance of meta-heuristic-based scheduling, we propose an ESFO algorithm to find an optimal scheduling plan with minimum search complexity. With a test on several task scheduling benchmarks, the proposed ESFO algorithm is proved to have significant improvement over the counterpart algorithms in terms of energy consumption and makespan measures. To summarize, our contributions are as follows: • Introducing the ESFO algorithm for solving the task scheduling problem in the cloud environment. In ESFO, we introduce a new pollination strategy to improve the exploitation and exploration capabilities. • Evaluating the proposed ESFO algorithm using different task scheduling benchmarks to identify potentials and weaknesses. The simulation results show the superiority of the ESFO algorithm compared with its counterparts in terms of energy consumption and makespan measures.

II. Problem statement

Let V = v1, v2, . . . , vN be a set of virtual machines

(VMs), and T = t1, t2, . . . , tM be a set of M tasks to be ex- ecuted. Each task can be processed by either of the machines.

The scheduling is defined as a function f : T $\rightarrow$ V, which maps tasks to VMs. The main objective is to find a schedule with minimum complexity for which the maximum completion time (makespan) and energy consumption are minimized [2]. The makespan is calculated as follows: P(i, j) is the processing time of task j on machine i, and x(i, j) is the boolean variable

Table 1
Task scheduling algorithms in literature.

Type	Algorithm	Reference
Meta-heuristic	Grey wolf optimization (GWO)	[2]
	Ant colony optimization (ACO)	[4]
	Particle swarm optimization (PSO)	[5]
	Glowworm swarm optimization (GSO)	[6]
	Genetic algorithm (GA)	[7]
	Artificial bee colony (ABC)	[8]
	Fireworks algorithm (FWA)	[9]
Dynamic	Dynamic two-stage scheduling	[10]
	Q-learning scheduling	[11]
Game Theory	Balancing task scheduling	[12]
	Biogeography-based algorithm	[13]

Fig. 2. table

$$f_1 = \max_{i=1}^{N} \left\{ \sum_{j=1}^{M} x(i,j) \times P(i,j) \right\}$$

Fig. 3. formula1

which identifies whether task j is processed by machine i or not. Since the VMs work in parallel, the makespan of a solution corresponds to the maximum execution time of VMs. The energy consumption corresponds to the CPU utilization, which is computed as follows: *Go to formula2* Ei j is the energy utilized by ith VM during the execution of jth task.

III. The ESFO algorithm for task scheduling

The sunflower optimization (SFO) algorithm [14] simulates the movement of sunflowers to absorb the solar radiation. The SFO is composed of two phases: pollination and movement. In the pollination phase, sunflowers cooperate to produce pollen gamete. In the movement phase, sunflowers take haphazard steps towards the best sunflower, which is considered as the sun. Proper exploration of the solution space is the most

important factor in finding the global optimum of an optimiza- tion problem. To realize this issue, we introduce an enhanced

version of SFO, namely the ESFO algorithm. The ESFO is equipped with a new pollination operator, which provides a proper balance between exploitation and exploration abilities. In the following, we describe how the ESFO algorithm is applied for task scheduling. **Create population :** The initial population composed of Z sunflowers. Pop = S1, S2, . . . , SZ (3)

$$f_2 = \frac{1}{N \times M} \sum_{i=1}^{N} \sum_{j=1}^{M} E_{ij}$$

Fig. 4. formula2

Each sunflower Si = si,1,si,2, . . . ,si,M in the population is composed of M numbers, each one denotes the identifier of a VM that tasks run on it. si j is defined as $s_{ij} = s_{min} + \theta_{ij}(s_{max} s_{min})(4)$

where i j is a uniform random number in the range [0, 1], smin and smax are respectively the lower and upper bounds of si j . For task scheduling, smin and smax respectively initialized as 1 and M. Let, there are six tasks t1, t2, t3, t4, t5, t6, and two VMs. A possible solution is Si = [1, 2, 2, 2, 1, 1] that maps tasks t1, t5, t6 into the first VM, and t2, t3, t4 into the second VM. Two constraints should be satisfied in the task assignment process. Every task should be contained in the scheduling process, and each task can appear only once. Since the task scheduling is a discrete problem, the indices of tasks and VMs are positive integers. To apply ESFO to task scheduling, the elements of sunflowers are rounded to be discrete numbers. **Power calculation:** The power of each sunflower Si is

calculated using Eq. (1) or Eq. (2). In the task scheduling prob- lem, the word "power" corresponds to the cost of a solution.

After power calculation, the sunflower with the minimum cost (S) is considered as the sun. **Pollination:** To model the pollination process, the standard SFO randomly selects p × Z sunflowers from population, and then updates each selected sunflower as

(5)

where p denotes the pollination rate. S t i and S t j

are the posi- tions of sunflowers i and j at iteration t. ri

is a random number in the range [0, 1]. In the standard SFO, p = 0.5 is used. In ESFO, we proposed the following equation to model the pollination phase S t+1 i = t × A + (1 t) × B

A = S t j + U(1, +1) × (S t i S t j)

B = S t i + × (S S t i)

(6)

where t is the switch probability at iteration t, which controls the ratio of local and global pollination. S is the current best sunflower found so far. U(1, +1) generates a uniform

random number in the range [1, 1], which adds some de- viation to increase the searching around the sunflower Si

,

is scaling factor that controls the amplitude of the search-direction (S

S t i). We used the value = 3., where The main motivation in introducing the new pollination

operator is to improve exploration and exploration simultane- ously. Term A in Eq. (6) improves the ESFO's exploitation

through simulating the local pollination process. Term B in Eq. (6) improves the exploration by modelling the global pollination mechanism. Movement: In this phase, (1 p)×Z sun- flowers is selected and updated as follows: Eq. (8) simulates the direction change of the sunflower Si towards the sun S . Finally, the most fittest sunflower is found

and replaced with the sun.

$$S_i^{t+1} = S_i^t + r.\frac{S^* - S_i^t}{\left\| S^* - S_i^t \right\|} \tag{8}$$

Fig. 5. formula

Algorithm 1: The ESFO for task scheduling

Input:
ESFO parameters;
M tasks, $T = \{t_1, t_2, ..., t_M\}$;
N virtual machines, $V = \{v_1, v_2, ..., v_N\}$;
Output:
An optimal solution for task scheduling;
Initialize population of sunflowers by Eqs. (3) and (4);
Evaluate the power of sunflowers by Eqs. (1) or (2);
$t = 0$;
while $(t < I)$ **do**
Compute the pollination of sunflowers by Eq. (6);
Compute the movement of sunflowers by Eq. (8);
Evaluate the power of sunflowers by Eqs. (1) or (2);
Update sun S^* by Eq. (9);
$t = t + 1$;
end
Return the sun S^* as the best solution;

Fig. 6. Caption

IV. Experiments and analysis

To simulate the cloud environment, we used the Cloudsim toolkit and implemented the core processes in MATLAB release 2017a. The proposed ESFO is compared with three algorithms: mean grey wolf optimization (MGWO) [2], hybrid glowworm swarm optimization (HGSO) [6], and standard SFO [14]. The population size (Z) and the maximum iteration

Table 2
Simulation parameters.

Parameter	Value
Number of data centre	5
Number of VMs (M)	50
VM policy	Time sha
Number of virtual CPUs	[1–5]
VM RAM size	[512–204
Virtual CPU capacity	[500–250
Number of tasks (N)	[100–500

Fig. 7. Caption

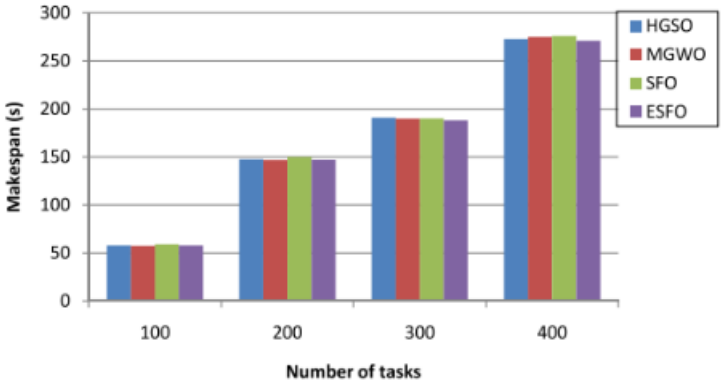

Fig. 8. Caption

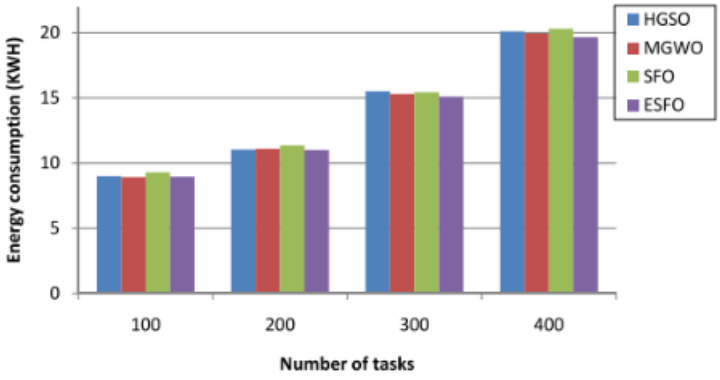

Fig. 3. Comparison of algorithms in terms of energy consumpti

Fig. 9. Caption