An improved Henry gas solubility optimization algorithm for task scheduling in cloud computing

Mohamed Abd Elaziz [1,2]· Ibrahim Attiya [2]

Abstract

In cloud computing, task scheduling plays a major role and the efficient schedule of tasks can increase the cloud system efficiency. To successfully meet the dynamic requirements of end-users' applications, advanced scheduling techniques should be in place to ensure optimal mapping of tasks to cloud resources. In this paper, a modified Henry gas solubility optimization (HGSO) is presented which is based on the whale optimization algorithm (WOA) and a comprehensive opposition-based learning (COBL) for optimum task scheduling. The proposed method is named HGSWC. In the proposed HGSWC, WOA is utilized as a local search procedure in order to improve the quality of solutions, whereas COBL is employed to improve the worst solutions by computing their opposite solutions and then selecting the best among them. HGSWC is validated on a set of thirty-six optimization benchmark functions, and it is contrasted with conventional HGSO and WOA. The proposed HGSWC has been proved to perform better than the comparison algorithms. Moreover, the performance of HGSWC has also been tested on a set of synthetic and real workloads including fifteen different task scheduling problems. The results obtained through simulation experiments demonstrate that HGSWC finds near optimal solutions with no computational overhead as well as outperforms six well-known metaheuristic algorithms.

Keywords Task scheduling · Cloud computing · Henry gas solubility optimization (HGSO) · Whale optimization algorithm (WOA) · Comprehensive opposition-based learning (COBL)· Metaheuristics

Mohamed Abd Elaziz
abd_el_aziz_m@yahoo.com
Ibrahim Attiya
ibrahimateya@zu.edu.eg
1 School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
2 Department of Mathematics, Faculty of Science, Zagazig University, Zagazig, Egypt submitted to the cloud environment. These tasks should be arranged for

execution in such a manner that enables every task to be completed on time, while achieving maximum utilization of cloud resources. This decision-making process is known as "cloud task scheduling", and it is one of the major challenges that strongly affects the performance of the whole cloud environment. Thus task scheduling in cloud systems has received remarkable attention from the research community (Bittencourt et al. 2012; Rodriguez and Buyya 2014; Li et al. 2020). Accordingly, miscellaneous techniques have been developed and used in addressing this issue, for example, Minimum Completion Time, First Come First Serve, Minimum Execution Time, and Min-min and Max-min technique. Although these algorithms have been adopted on many cloud computing systems, some research works have verified that these strategies are inappropriate for large-scale scheduling problems. On the other hand, metaheuristic algorithms have proven to be effective and robust in tackling many real-world problems (Houssein et al. 2020; Tharwat et al. 2018). Some of the most popular metaheuristic algorithms that are considered for task scheduling include genetic algorithm (GA) (Zhao et al. 2009), ant colony optimization (ACO) (Jia et al. 2019), particle swarm optimization (PSO) (Pandey et al. 2010), cuckoo search (CS) (Burnwal and Deb 2013), symbiotic organisms search (SOS) (Abdullahi et al. 2019), whale optimization algorithm (WOA) (Gharehchopogh and Gholizadeh 2019), and Harris Hawks optimization (HHO) (Attiya et al. 2020).

In the same context, a new metaheuristic technique called Henry gas solubility optimization (HGSO) has been proposed as a global optimization (Hashim et al. 2019) which simulates the Henry gas solubility. HGSO has recently been found to be successful in solving various optimization problems such as feature selection (FS) (Neggaz et al. 2020), unit commitment (UC) problem (Saranya and Saravanan 2020), and motif discovery (MD) problem (Hashim et al. 2020). However, it suffers from several limitations including inability to exploit the search space and this can be observed from its behaviour during the updating process which in turn affects on the quality of the obtained solution. In addition, HGSO basically depends on generating new random solutions to update the worst solutions obtained. However, this can't be the optimal method to handle this issue, which means that HGSO still needs more improvements. This motivated us to provide an improved version of HGSO based on the WOA (Mirjalili and Lewis 2016) and comprehensive opposite-based learning (COBL) method (Seif and Ahmadi 2015) to overcome the mentioned challenges. Recall that, the two techniques (i.e., WOA, and COBL) have established their performance in several applications. For instance, the WOA has been applied to image segmentation (Abd El Aziz et al. 2017, 2018b; Ewees et al. 2018a), Parameter estimation of photovoltaic cells (Oliva et al. 2017; Abd Elaziz and Oliva 2018), feature selection (Mafarja and Mirjalili 2017), wind speed forecasting (Wang et al. 2017), and content-based image retrieval (Abd El Aziz et al. 2018a). Also, COBL has proved its applicability to solve optimization problems in various domains (Seif and Ahmadi 2015).

The proposed algorithm, called HGSWC, is similar to the traditional HGSO algorithm in its starting stage, which generates a set of solutions and split them into groups. Then it determines the best solution from each group and the best one overall the groups. Next, the HGSWC updates the solutions using either the operators of HGSO or WOA depending on the quality of the fitness value for each solution within its group. This process increases the exploitation ability across the search space. The next step is to determine the worst solutions then using the COBL to update them. This can be performed through computing the comprehensive opposition solutions for the current worst solutions then selecting the best solutions among them. Thereafter, the process of updating the solutions is performed until the stopping condition is reached. This paper's major contributions can be summarized as follows:

- Design a modified version of Henry gas solubility optimization (HGSO) algorithm for optimum task scheduling in a cloud computing environment.
- Employment of two techniques namely, WOA and COBL to improve the rate of convergence and quality of solutions obtained.
- Comparative and statistical analysis of HGSWC to ensure it's applicability for global optimization problems.
- Extensive performance evaluation of HGSWC with other popular metaheuristics for different workload instances by considering performance metrics such as makespan and performance improvement rate.

The organization of the remainder of this study is given as follows. Section 2 puts forward the related work. Section 3 formulates the Cloud task scheduling problem and presents the original WOA and HGSO algorithms, while Sect. 4 describes the details of the proposed HGSWC algorithm. Performance evaluation and detailed analysis of the results obtained are given in Sect. 5. In Sect. 6, some concluding remarks and suggestions for potential future works are drawn.

1 Related work

Cloud computing empowers on-demand service delivery with consistency while overcoming difficulties related with traditional computing paradigms such as limitations of computing resources, lack of scalability and elasticity, and inefficiency in responsiveness and performance. In cloud computing, consumers can utilize hundreds or even thousands of the available virtualized resources and it is impractical for everyone to manually allocate each task. Thus, task scheduling plays a crucial role in cloud computing to allocate the resources for each submitted task effectively and efficiently. This action can be considered as a mapping between computing tasks and available resources in the cloud environment, which depends on the premise of achieving certain optimization objectives. The authors in Arunarani et al. (2019) and Kumar et al. (2019) presented a review of scheduling methodologies and the objectives appropriate for cloud computing.

Indeed, a wide variety of task scheduling algorithms for cloud systems have been proposed over the past decade. Among them, heuristic and metaheuristic-based techniques have shown their ability to attain near optimal solutions within a reasonable time for scheduling in the cloud. For example, task scheduling based on genetic algorithm (GA) was proposed in Rekha and Dakshayini (2019). The proposed GA scheduler is aimed to find appropriate allocation decisions while reducing the task completion time. The authors evaluated the performance of the proposed GA scheduler in terms of throughput and makespan, and the results exposed its efficiency against greedy and simple allocation strategies. Zhou et al. (2019) proposed an enhanced genetic algorithm integrated with greedy strategy named MGGS, to optimize the task scheduling issue in cloud computing. They considered the average response time, total execution time, and QoS parameters as evaluation metrics for MGGS. The work described in Akbari et al. (2017) proposes a GA with new operators to tackle static task scheduling within heterogeneous distributed computing systems (HDCSs). To enhance the performance of GA, the authors performed some significant changes in the genetic functions of the standard GA and introduced a new operator called Inversion to promote diversity among the generated samples as well as reducing the amount of repeated schedule solutions in each generation. Similarly, Keshanchi et al. (2017) presented an improved GA for static task scheduling in HDCSs using the priority queues. The authors employed a powerful one-point crossover operator to preserve the precedence constraints among produced solutions. They also adopted an Elitism technique with unusual selections to evade premature convergence. The purposes were to enhance the makespan and optimize the algorithm's execution time.

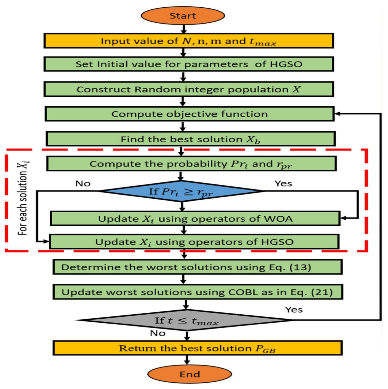
Some works related to PSO based task scheduling in cloud computing have been reported in Pandey et al. (2010) and Beegom and Rajasree (2019). Pandey et al. (2010) proposed an improved task scheduling method using PSO to reduce the total execution cost. The simulation results reveal the efficiency of the PSO by contrasting it with the 'Best Resource Selection' (BRS) algorithm. The disadvantage of this work is that it incurs higher transmission and storage costs. The authors in Guo et al. (2012) presented a task scheduling optimization approach based on an improved PSO algorithm to map tasks to resources in cloud computing in order to minimize the processing cost of user tasks. However, their optimization approach focuses only on optimizing the efficiency. Khalili and Babamir (2015) introduced a scheduling approach depends on a modified PSO to minimize the makespan of workload scheduling. In this method, different techniques were adopted to update the inertia weight of PSO. To overcome the difficulties with PSO, Mansouri et al. (2019) have proposed a hybrid task scheduling algorithm using PSO and fuzzy theory, which enhances the PSO performance, minimizes the makespan, reduces the total execution time, and maximizes the resources utilization. This FMPSO strategy was planned by means of four modified velocity updating methods and two operators to improve the PSO search capability and overcome some of its drawbacks. They additionally employed a fuzzy system for the purpose of fitness evaluations. Similarly, Alla et al. (2018) suggested two hybrid PSO strategies based on Dynamic dispatch Queues

(TSDQ) for cloud task scheduling. The first approach called TSDQ-FLPSO combines the Fuzzy logic with PSO. The second approach called TSDQSAPSO integrates Simulated Annealing with the PSO algorithm. The objectives of them were to optimize waiting time, makespan, resource utilization, and load balancing.

Moreover, other techniques have been developed for task scheduling as well. Ding et al. (2014) introduced a task scheduling approach based on the ACO paradigm. In their study, makespan was considered as an objective. Similarly, Ari et al. (2017) presented a modified version of ACO for cloud task scheduling. The objectives of them were to minimize makespan and energy consumption. Moreover, the symbiotic organisms search (SOS) has attracted much attention as an alternative approach to address the task scheduling issue. Abdullahi et al. (2016) developed a discrete version of SOS to find optimum scheduling of user requests which minimizes the makespan. The simulation result exposed the efficacy of their DSOS algorithm by comparing it with two scheduling strategies, SAPSO and PSO. To improve the performance of SOS algorithm for task scheduling, Abdullahi and Ngadi (2016) advanced a modified version of SOS based on SA procedure for scheduling of tasks in the cloud. This study presents an optimization policy to optimize the search capability of SOS, improve the convergence rate, and minimize the makespan. Moreover, Navimipour and Milani (2015) developed a task scheduling algorithm based on the cuckoo search algorithm (CSA) to allocate user tasks to cloud resources so as to minimize the total waiting time of the tasks. However, this work only tested on homogeneous cloud infrastructure. To avoid premature convergence associated with CSA, Akbari and Rashidi (2016) have explained a multi-objective scheduling approach based on cuckoo optimization algorithm (MOSCOA) for scheduling of tasks in heterogeneous systems. The core idea was to minimize the total execution time of the task while allowing for maximum parallelization. Here, speedup, efficiency, and scheduling length ratio (SLR) were employed to evaluate the performance of the scheduling methods.

2 updating stage

This stage starts by computing the ftness value (F) for each solution X as defned in Eq. (5). Then the best solution Xb is determined. After that, the probability of each solution will be computed depending on its ftness value as in Eq. (27).


Start
Input value of N, n, m and t_{max}
Set Initial value for parameters of HGSO
Construct Random integer population X
Compute objective function
Find the best solution X_b
For each solution X_i
Compute the probability Pr_i and r_{pr}
No
If $Pr_i \geq r_{pr}$
Yes
Update X_i using operators of WOA
Update X_i using operators of HGSO
Determine the worst solutions using Eq. (13)
Update worst solutions using COBL as in Eq. (21)
Yes
If $t \leq t_{max}$
No
Return the best solution P_{GB}
End

3 Experimental evaluation

This section presents a set of experiments to verify the performance of HGSWC. The frst set of experiments (described in Sect. 5.1) have been conducted over 36 benchmark functions to ascertain the ability of HGSWC to solve global optimization problems. In the other set of experiments (described in Sect. 5.2), a large variety of task scheduling problems including both real and synthetic workload instances are employed to test the capability of HGSWC to tackling the cloud task scheduling problem.

4 Global optimization

The performance of the HGSWC approach is assessed as a global optimization algorithm in this experimental series by comparing it with traditional HGSO and WOA algorithms. In addition, the floor function used in the step of generating initial solutions is removed in (29) rpr = LPr + rand × (UPr LPr) M. Abd Elaziz, I. Attiya 1 3 this experiment. A set of global optimization functions from CEC2005 benchmark (Suganthan et al. 2005) is used which contains thirty-six functions as defned in Table 1. These functions are divided into unimodal (i.e., F1–F10) and multimodal (i.e., F11–F36) groups. In the unimodal, there is a single extreme solution in the search domain. Meanwhile, the multimodal functions have more than one extreme solution, and this group consists of two types, (1) variant dimension (i.e., F11–F26), and (2) fxed dimension (i.e, F27–F35).

5 Results and discussion

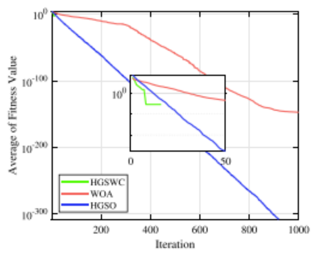
Tables 2 and 3 show the average, standard deviation, best and worst values of ftness results. According to the results given in Table 2, it can be noticed that the HGSWC method has an average of the ftness value better than that of the two other methods (i.e., HGSO and WOA). More specifcally, it has the smallest average on 14 functions out of the 36 functions, while in other eight functions the produced values were nearly similar to that of the others two methods. In addition, the STD value for HGSWC is better than the traditional HGSO, but it's nearly the same as that of WOA. However, the comparison algorithms obtain the same smallest STD value for 14 test functions, which in turn indicate the high performance and stability of HGSWC. Besides, according to the best values of the ftness value (as shown in Table 3), it can be seen that the WOA and HGSWC nearly have the same performance, where each of them has the best value for seven functions However, it can be noticed that the HGSWC has the smallest ftness value similar to other two algorithms at eighteen functions, while WOA has the smallest ftness value at only fourteen functions similar to HGSO. In terms of the worst ftness value it can be seen that the HGSWC approach performs better than the two

6 Statistical analysis

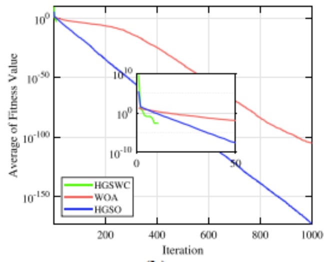
To further provide an evident on the performance of the proposed HGSWC as a global optimization method, the non-parametric Wilcoxon's rank sum test is applied. This test checks if there is a signifcant diference between the average of ftness value for the control group (HGSWC) against the other tested groups (WOA and HGSO). In this study, the test is performed at 5(P-val) is less than , then the hypothesis which assumes that the average of HGSWC is not equal to the average of WOA or HGSO, is accepted (i.e., H = 1); otherwise, the hypothesis will be rejected (i.e., H = 0). The statistical results of the Wilcoxon's test are given in Table 4. According to the results, we can notice that the hypothesis (H = 1) is accepted for most of the tested functions since the p value is less than 0.05. In contrast, there was no signifcant diference in performance between HGSWC and HGSO for functions F1, F8, F9, F12, F13, F14, F19, and F23. Also, for the functions F12, F19, F24, F27, F28, and F34–F36, there was no signifcant diference between HGSWC and WOA

7 Experimental series 2: task scheduling problem in cloud computing

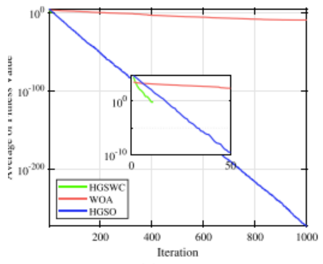
This section presents the experimental setup, results and the analysis of the results. In order to help understand and support the scenario used in our evaluation, in Sect. 5.2.1 we present the simulation environment and the experimental data which are employed in this study. In Sect. 5.2.2 we explain the metrics used in our experiments for evaluating the performance of the proposed HGSWC algorithm. Finally, in Sect. 5.2.3 we summarize the achieved results and provide some concluding remarks.



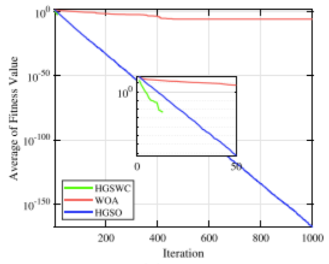
(a) F1



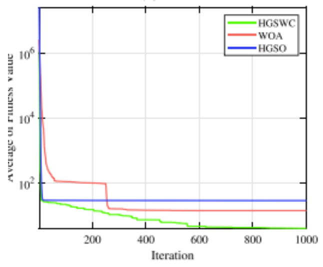
(b) F2



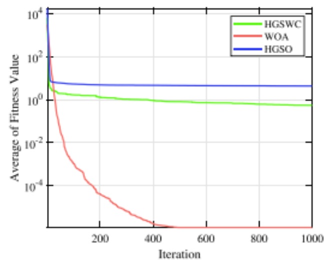
(c) F3



(d) F4



(e) F5



(f) F6