Improved chaotic binary grey wolf optimization algorithm for workfow scheduling in green cloud computing

Transportation Research Record
2020, Vol. XX(X) 1–9

DOI: 10.1177/ToBeAssigned
journals.sagepub.com/home/trr
SAGE

Ali Mohammadzadeh[1]· Mohammad Masdari[1].Farhad Soleimanian Gharehchopogh[1].Ahmad Jafarian[2]

Abstract

The workfow scheduling in the cloud computing environment is a well-known NP-complete problem, and metaheuristic algorithms are successfully adapted to solve this problem more efciently. Grey wolf optimization (GWO) is a recently proposed interesting metaheuristic algorithm to deal with continuous optimization problems. In this paper, we proposed IGWO, an improved version of the GWO algorithm which uses the hill-climbing method and chaos theory to achieve better results. The proposed algorithm can increase the convergence speed of the GWO and prevents falling into the local optimum. Afterward, a binary version of the proposed IGWO algorithm, using various S functions and V functions, is introduced to deal with the workfow scheduling problem in cloud computing data centers, aiming to minimize their executions' cost, makespan, and the power consumption. The proposed workfow scheduling scheme is simulated using the CloudSim simulator and the results show that our scheme can outperform other scheduling approaches in terms of metrics such as power consumption, cost, and makespan.

1 Introduction

Cloud computing provides an interesting technology that makes scientifc and industrial projects easier to implement. Infrastructure as a service (IaaS) is a type of cloud computing that provides online resources for virtualized computing. In addition to software as a service (SaaS) and platform as a service (PaaS), IaaS is one of the three main categories of cloud computing services [1]. Cloud computing can be used to deploy highly complex applications for scientifc workfow. Workfows break down complex, data-intensive applications into smaller tasks and perform them in serial or parallel depending on the application's nature. Workfow models are commonly used in felds such as science, business, and engineering. In the planning of the scientifc workfow, we need to take the following questions: (1) how to allocate tasks to VMs; (2) in what order the VMs will perform tasks taking into account the data dependency between tasks. Usually, these scientifc workfows require a great deal of data of diferent sizes and simulations of long-term computers. We need high computing power and the availability of large infrastructure that ofers diferent levels of QoS for the grid and more recently cloud computing environments. There are various strategies to solving the problem of scheduling; scheduling schemes can usually be defned as follows: metaheuristic-based scheduling, heuristic workfow scheduling, hybrid metaheuristic, and heuristic scheduling, and Task and workfow scheduling [2]. Metaheuristic scheduling schemes can use algorithms such as simulated annealing (SA), ant colony optimization (ACO), and particle swarm optimization (PSO) to generate optimum schedules. For metaheuristic scheduling workfow schemes, the goals sought are often found in the metaheuristic algorithm's ftness function. For this reason, once multiple objections are implemented to scheduling, diferent coefcients are allocated for each goal that can change the efect of each goal on the overall scheduling of workfow. Also, Metaheuristic algorithms were commonly used to solve various optimization issues, such as the selection of features [3]. The metaheuristic optimization algorithms have become so popular over the past two decades. Some of these algorithms are even known among scholars from other sciences. They are usually inspired by physical phenomena and animals' behaviors. Additionally, learning, using, and combining these algorithms are done easily. The only point in using these algorithms is how to give input and get output from the system. These algorithms act better than conventional optimization methods in the comparison with local optimizations given their stochastic nature. On the contrary, all meta-heuristic methods have

Corresponding author:
Alistair Smith, alistair.smith@sunrise-setting.co.uk

some weaknesses [4–7]. A common feature among the meta-heuristic approaches is dividing the search process into two stages: exploration and exploitation [8]. The exploration step is the broad random search in the search space to reach more desirable segments. The exploitation step is the local and more precise search in the desirable explored segments of the search space. Finding the right balance between these two stages is one of the challenging issues of metaheuristic methods. Several approaches have been proposed concerning the metaheuristic algorithms in the past few decades, which we intend to review in some cases. Particle swarm optimization (PSO) algorithm [9, 10] is inspired by the collective behavior of the birds where each particle tries to fnd the best response in the search space by changing its speed and direction. Gandomi [11] introduced the krill herd (KH) optimization algorithm according to krill feeding behavior. In this algorithm, particle motion is specifed according to three factors: food search behavior, random scattering, and the motion created by other particles [11]. Bat algorithm is inspired by the echolocation behavior of bats that was proposed by Yang in 2012. In this algorithm, the particles randomly navigate the search space at a diferent speed, position, frequency, and sound band [12]. Ant lion optimizer (ALO) algorithm [13] was frst proposed by mirjalili according to ant lion feeding behavior for continuous problems. Ant lion performance uses specifc movements of this insect to trap the ants in a pit. Multi-verse optimizer (MVO) algorithm is a kind of the new powerful meta-heuristic algorithms inspired by nature, where it has been commonly applied in a lot of felds, and based on three cosmological concepts: black hole, a white hole, and wormhole. These three concepts are used for exploration, exploitation, and local search [14]. Moth-fame optimizer algorithm [15] is a population-based algorithm that is proposed by Mirjalili. The moths in this algorithm move in a single, two, or multidimensional space. Cosine and logarithmic spiral function are used to implement moth-fame motion. In the shufed frog leaping algorithm [16], the frogs mimic each other and try to improve this behavior locally and turn into a model that others can imitate. This is a conscious imitation and not a mere imitation. The new optimization algorithm, inspired by the static and dynamic behavior of dragonfies' accumulation and its two main phases of exploration and exploitation, has been done by modeling the grasshopper optimization algorithm's social relationship in guiding, searching for food, and avoiding the enemies [17]. Several papers have been presented regarding grey wolves in recent years. In [18], a hybrid algorithm of PSO and GWO algorithms is presented. The particle swarm position is frst updated by the PSO algorithm and then by the GWO algorithm. In [19], Kumar et al. proposed three strategies for improving the GWO algorithm. The frst strategy is to use weighted baits. He then uses the laws of astrophysics to better guide the grey wolves to more desirable goals. According to this strategy, the wolves perform both exploration and exploitation processes. In the third strategy, the features and benefts of the two strategies are applied together. Various schemes have also been proposed to improve the GWO algorithm [20, 21]. The basic GWO algorithm does not perform well in the identifcation and exploration of global optimums that afect the convergence rate, despite good convergence. Hence, the purpose of the paper is to present the improved GWO algorithm that performs better in global optimizations. This algorithm is based on collective intelligence and is inspired by the collective behavior and hunting of grey wolves in nature. The innovation of the paper is using hill-climbing problem and random numbers based on chaos problem in the proposed algorithm. With the improvements done, we have seen good results with the basic method based on the chaos theory of the GWO algorithm. Many papers have used random numbers based on chaos problem and hill-climbing to improve the optimization algorithms [22–28]. The main contributions of this paper are as follows: (1) inspired by the GWO algorithm, an improved version of this algorithm was designed in this paper. The proposed algorithm is search improvement using hill-climbing problem and random numbers based on the theory of chaos. In this scheme, the update of the particle's position at each iteration is afected by the last few positions. (2) We used some standard mathematical optimization benchmark functions to compare the other algorithms. These benchmark functions were chosen as single exponential, multi-exponential, and fnite-dimensional, with varying levels of hardness. (3) Using the transfer functions in the proposed algorithm for the solution of scientifc workfow scheduling. (4) To evaluate the performance of the proposed algorithm practically, we implemented the extended proposed algorithm using the WorkfowSim tool, which is based on the CloudSim tool.The results of the proposed algorithm were compared with some other algorithms. The rest of this paper is organized as follows. The second section explores the context and the related work in recent literature. This section divided into two parts: heuristic, and meta-heuristic algorithms. The third section briefy explains the basic GWO algorithm. The proposed algorithm is provided in the fourth part, and in this part, we describe the innovation in the proposed algorithm, such as the improvement in search using hill-climbing problem and random numbers based on chaos theory. In the ffth part, the optimization functions and simulation results will be discussed. Section 5 explains the simulation component in detail, focusing on the optimization functions which divided into three groups: unimodal, multimodal, and constrained multimodal benchmark functions. Part six is a workfow scheduling in green cloud computing. In this section, we highlight the scheduling problem and describe a binary version of the proposed algorithm to use in discrete problems. In this section, we integrate the S-shaped and

V-shaped transfer functions into the algorithm to convert the continuous proposed algorithm into the binary version, and simulation of workfow in the cloud-sim simulator are presented in this section, respectively. Section seven and eight exhibits the discussing and concluding remarks.

2 Literature review

Workfow scheduling is a NP-complete problem. Various heuristic and meta-heuristic algorithms have been proposed that solve workfow scheduling problems. The rest of this section is classifed as heuristic and meta-heuristic algorithms.

2.1 Heuristic algorithms

Many heuristic algorithms such as MIN–MIN [29], and MAX–MIN [30] are proposed in the scheduling literature. List scheduling is one of the most common methods of scheduling workfow [31], in which a priority is given to each of the workfow tasks. Among the list-based heuristic algorithms, the critical path on the processor (CPOP) [32], dynamic level scheduling (DLS) [33], heterogeneous earliest fnish time (HEFT) [34], dynamic heterogeneous earliest fnish time (DHEFT) [35] and dynamic critical path (DCP) can be cited [36]. All these algorithms are aimed to reduce workfow makespan [37]. Some papers have considered two main criteria such as makespan and cost for scheduling. In [38], a multi-objective list-scheduling algorithm is presented to fnd dominant responses using Pareto for heterogeneous environments.There are many bi-objective and multi-objective heuristic algorithms which solve workfow scheduling problem. The following are some examples of these algorithms. A bi-objective dynamic level scheduling algorithm performs the assignment of tasks in conditions where the runtime is acceptable against reliability [39]. The authors considered makespan and cost and used the concept of Pareto dominance to run large cloud applications to reduce fnancial costs regardless of budget and time constraints. Camelo et al. [40], have presented a multi-objective heuristic algorithm to solve the workfow scheduling problem in the grid environment and they have used branch and bound's deterministic algorithm to fnd real Pareto front solutions. Juan et al. [37], have enhanced a new multi-objective listbased scheduling algorithm to deal with numerous contradictory objectives such as makespan, cost, and suggested a genuinely multi-criteria optimization reaching the Pareto front using a variety of well-distributed trading solutions chosen from the crowding distance and analyzed that use the HV metric. Multi-objective HEFT [41] has been provided for scheduling workfows on Amazon EC2. Cost and makespan have been considered to create an optimal Pareto and the users have been allowed to select the best solution manually

2.2 Meta-heuristic algorithms

Matthews et al. [42], have proposed a new scheduling algorithm based on ant colony algorithm, aimed to minimize the workfow time and the makespan. Unfortunately, these have been ignored job priorities. In [43], a cost-based algorithm is presented for efcient mapping of tasks to available cloud resources. This scheduling algorithm considers the cost of resource use and efciency. The scheduler divides all tasks by priority into three categories: high, medium, and low priorities. Mozmar et al. [44], have proposed a bi-objective hybrid genetic scheduling algorithm for parallel applications, limited priority in heterogeneous distributed systems like cloud computing infrastructure. One of the advantages of this algorithm is the reduction of makespan and communication costs. One way to solve multi-objective problems is to convert them to weighted single-objective problems. In [45], Li et al. transform the multi-objective problem into a singleobjective problem with the heuristic algorithm. They have been focused on using cloud resources to provide large-scale graph computing tasks. This algorithm produces a priority task list and passes the highest priority task in a cloud environment to a cost-efective virtual machine. Dangra et al. [46], have proposed a scheduling method with the technique of transforming a multi-objective problem into single-objective to increase efciency and reliability. They have proposed a reliable dynamic level scheduling (RDLS) algorithm based on dynamic level scheduling (DLS) [47]. Unlike the previous method where only one defnite solution is proposed as a result of the algorithm, a set of dominant solutions provided to the user. Yu et al. [48], have used the multi-objective evolutionary algorithm (MOEA) to solve the workfow scheduling problem. This algorithm is used to reduce two contradictory criteria of cost and runtime. Besides these two criteria, budget constraints and deadlines are considered in the algorithm as well. Population-based algorithms SPEA2 and NSGA-II [49, 50] and local search algorithms such as MOEA and PAES [51] have been used to solve workfow scheduling problems with conficting objectives and various constraints. Another multi-objective algorithm with the R-NSGA-II approach [52] obtains Pareto optimal solutions according to three conficting objectives: runtime, total cost, and reliability. By examining the past studies, one can conclude that most multi-objective heuristic algorithms are suitable for the grid model and studies on the cloud environment are few. On the other hand, most studies have been conducted to reduce makespan and cost, ignoring the energy issue. In their proposed approach, Khalili et al. [53], have considered throughput besides these two criteria. In our proposed method, besides reducing the makespan and implementation cost, the consumed energy and throughput have been considered as criteria. This paper presents a multi-objective optimization solution to create optimal Pareto responses for the workfow in the green cloud

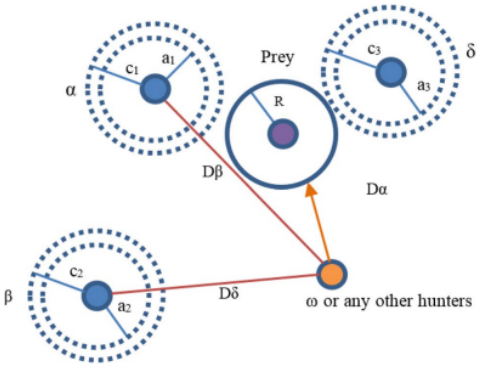

Figure 1. Particle position updating in GWO algorithm

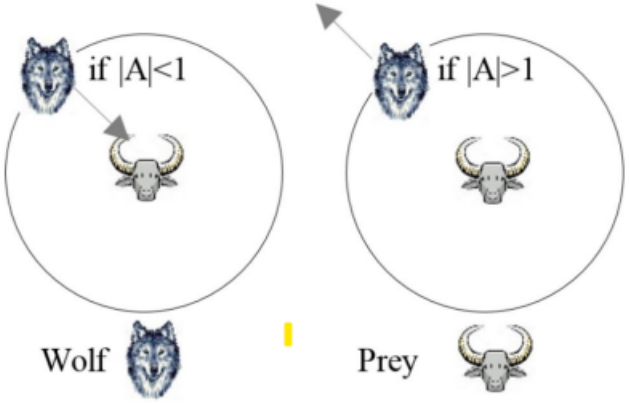

Figure 2. Exploration phase versus exploitation

environment.

In these equations, t shows the number of current iterations. A and C are coefcients vectors, Xp the hunting position vector, and x the position vector of a grey wolf. D is the wolf's distance from the prey or the current position distance from the optimal response. Equation 2 is the new position of the wolf after moving towards the target. Vectors A and C are calculated by Eqs. 3 and 4 [54]:
= 2a.r a (2)
C = 2.r (3)

3 Grey wolf optimization algorithm

This algorithm imitates the leadership hierarchy and grey wolves' hunting mechanism in nature. In this algorithm, four types of a grey wolf—alpha, beta, delta, and omega—are used to simulate the leadership hierarchy. Moreover, three main stages of hunting—Hunt for a target, encircling target, and attacking target—are simulated. According to Moro et al., the main stages of grey wolves' hunting are as follows [54]: tracking, chasing, and approaching prey. Chasing, seizing, and teasing the prey until it stops moving and attacking the prey. For mathematical modeling of the social hierarchies of wolves, we name the most appropriate solution as alpha, and among the best solutions, we name the second and third as beta and delta, respectively. The rest of the candidate solutions are considered as omega. The optimization process is directed by alpha, beta, and delta, and the fourth group follows these three groups. Equations 1 and 2 are used to model wolf siege behavior [54]:
D = —C.Xp(t) X(t)—
Vector decreases linearly from 2 to 0 during the iteration period in both the exploration and exploitation phases and r is a random vector from 0 to 1. Given the stochastic nature of r1 and r2 vectors, the wolves are allowed to reach any position between the points shown in Fig. 1. Thus, a grey wolf can change its position within the space that encompasses the

$$\overrightarrow{D_{\alpha|\beta|\delta}} = \left|\overrightarrow{C_{1|2|3}}.\vec{X}_{\alpha|\beta|\delta} - \vec{X}\right|$$

$$\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \overrightarrow{A_1}.\overrightarrow{D_\alpha}$$
$$\overrightarrow{X_2} = \overrightarrow{X_\beta} - \overrightarrow{A_2}.\overrightarrow{D_\beta}$$
$$\overrightarrow{X_3} = \overrightarrow{X_\delta} - \overrightarrow{A_3}.\overrightarrow{D_\delta}$$

prey at random using Eqs. 5 and 6. The same concept can be extended to n-dimensional search space. In this case, the grey wolves move around the cubes around the best solution. In Eq. 5, variable D shows the spatial distance of the alpha, beta, and delta wolves from the prey position or variable X. In Eq. 6, variables X1, X2, and X3 are the new positions of the alpha, beta, and delta wolves are after changing the location and approaching the prey. Equation 7 calculates the new location of the hunter based on the mean of the three newer locations of alpha, beta, and delta wolves. Grey wolves' hunting is usually guided by alpha. Beta and delta sometimes take part in hunting as well. We save three of the best solutions obtained and force the other search agents according to Eq. 7 to update their position according to the best search factors to model this behavior. Figure 1 shows how
 to update the search agent position in 2D space. According to Fig. 1, alpha, beta and delta estimate the hunting position, and other wolves randomly update their position around the hunting area. In Fig. 1, alpha, beta, delta, and omega wolves are shown as circles. D, D, and D show the hunter's distance from other wolves. Variables a and c are the radius of spatial variations of alpha, beta, and delta particles and can be calculated through Eqs. 3 and 4. Variable R is the radius of prey locations change. In the exploitation phase or prey attack, the grey wolves will attack if the prey stops. We reduce the value of a from 2 to 0 to model this. The value of A, depending on a, decreases as well. The decrease

Set the initial values of the population size n, para
coefficient vectors A and C, and the maximum nu
iterations Maxiter.
Set t = 0.
for (i = 1 : n) do
Generate an initial population of Xi(t) randoı
Evaluate the fitness function of each search agent (sol
end for Assign the values of the 1st , 2nd, 3rd best solution X, X
for (i = 1 : n) do
Update each search agent in the population as showı
Decrease the parameter a from 2 to 0.
Update the coefficients A, C as shown in Eq. (3) and (4), respectively.
Evaluate the fitness function of each search agent (vector) f(Xi). end for
Update the vectors X, X, and X.
Set t = t + 1.
until (t Maxiter). (Termination criteria are satisfied)
Produce the best solution X.

Table 1. Caption

in the value of A from 1 makes the wolves attack the prey. To avoid trapping at a local minimum of this algorithm, it provides a search or exploration phase for the prey. The wolves are separated from each other in search of prey and work together to attack it. To simulate this divergence, we use vector A with random values greater than 1 or smaller than 1. Figure 2 shows this problem. Another component afecting the exploration process is C value. The value of this random number vector is in the range [0, 2]. If the random value C is greater than 1, the prey position will afect the wolf and prey distance (variable D in Eq. 5). However, if this value is less than 1, the prey position will be less efective. This vector can be considered as the efect of obstacles that prevent approaching the prey in nature. The pseudo-code of this algorithm is given in Table 1. All variables like the number of algorithm iterations, random variables A and C, the population of wolves, and the parameter a are initialized. In each iteration, which is shown by variable t, the wolves' population is randomly generated and the ft function is run for each case. In each iteration among the population, the best wolves are identifed by alpha, beta, and delta based on their identifed ftness. Then, the new position of the hunter wolves is determined based on the average location values of the top three wolves and all parameters and spatial vectors are updated. The best position of the wolves (X) is recorded as the response in each iteration.

4 Proposed algorithm

In the GWO algorithm, the population moves towards the optimal responses—alpha, beta, and delta wolves move. In

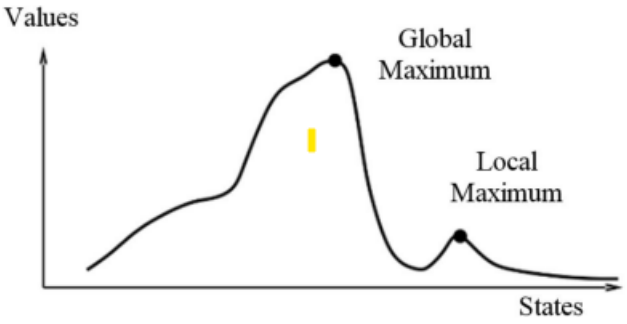

Figure 3. Hill–Climbing problem

each iteration, the optimal particles are identifed based on the ftness function and the best particle is called alpha. Each dimension of the new location is equal to the corresponding dimension mean of the superior particles, fully explained in Sect. 2 in Eqs. 1–7. The movement of the particles at each stage is done regardless of the degree of ftness; i.e., non-greedily. In the proposed algorithm, the number of steps of a particle can change without an increase in the degree of ftness. Every change in the ftness of the new location is compared to the best location it used to be. If there are no defnite steps to improve, the particle is returned to the last optimal response [55]. This is well shown in Fig. 3. As is seen in this fgure, after reaching the global optimum, it continues to explore several steps until it reaches a better response; however, it returns to the general optimum after not fnding appropriate responses. The basic GWO algorithm does not perform well in the exploration of global optimizations [21]; thus, we used 10 functions according to chaos theory such as circular, Gaussian and logistic instead of conventional stochastic functions to reduce this efect and increase the efciency of the proposed algorithm [56]. These functions are used to create numbers between [0, 1] as seen in Table 2. The initial value of all random numbers is considered 0.7 [57]. Overall, chaos, deterministic and quasi-random functions on dynamic and nonlinear systems are non-periodic, nonconvergent, and fnite. Mathematically, chaos functions are a random deterministic dynamic system. Chaos maps different from alternate mathematical functions can be used to use these functions in the optimization algorithm. Since the last decade, these functions have widely been focused on optimization because of their dynamic behavior that helps optimization algorithms in dynamic and more general discovery. Most importantly, chaos functions are used in real-world applications to make the algorithms applied. The results show that using chaos theory-based functions is efective to avoid being trapped in local optimum and to increase convergence speed. The implementation of some of these functions can be seen in Fig. 5. A random chaos function is used in each iteration of the GWO algorithm. Table 3 shows the pseudo-code for the proposed

algorithm. The fowchart of the algorithm is also shown in Fig. 4 for more clarity. Chaotic random numbers have a good efect on the convergence rate of the algorithm. Maps of the chaos functions generate random numbers within a permissible range. These numbers are initially predictable for a very short time and are random for a long time then.

5 Simulation and results

We used 23 standard mathematical optimization functions presented as CEC 2005 to compare the GWO algorithm and the proposed algorithm [56, 58, 59]. These benchmark functions have been selected as single exponential, multiexponential, and fnite-dimensional with varying hardness levels. The simulation and the resulting numerical results are performed in MATLAB 2017. The simulator uses a computer with a Core i7 processor with 2 GHz processing power and 4 GB of main memory. We run the proposed algorithm 10 times over the relevant functions and obtain the maximum, minimum, median, and mean of the iterations as are shown in Tables 5, 7, and 9. All the results shown in this paper are based on the IEEE CEC 2005 approved format. In these tables, the results are better distinguished by the thick pen. Each time the algorithm is fully run, 1000 searches are performed. We have a population size of 30 and each response is assumed to be a set of 30. The population and computational power of the compared algorithms are considered similar to have a correct and fair comparison.

5.1 Unimodal benchmark functions

Figure 6 is a three-dimensional drawing of these benchmark functions. Moreover, the cost functions along with the dimensions, ranges, and minimum inputs related to the single exponential benchmark functions are shown in Table 4. In Tables 4, 6, and 8, n shows the number of x members. We used an array of length 30 for each particle. These functions are suitable for measuring the exploitation process. Table 5 shows the statistical results (mean, median, minimum, and maximum) of the basic GWO algorithm, the proposed algorithm, and several new algorithms on unimodal exponential functions. Figure 7 shows the convergence graph for the best response to the algorithms. In

The Article Header Information

The heading for any file using TRR.cls is shown in Figure 9. *Transportation Research Record* is published using Times fonts and this is achieved by using the `times` option as

```
\documentclass[times]{TRR}
```

If for any reason you have a problem using Times you can easily resort to Computer Modern fonts by removing the `times` option.

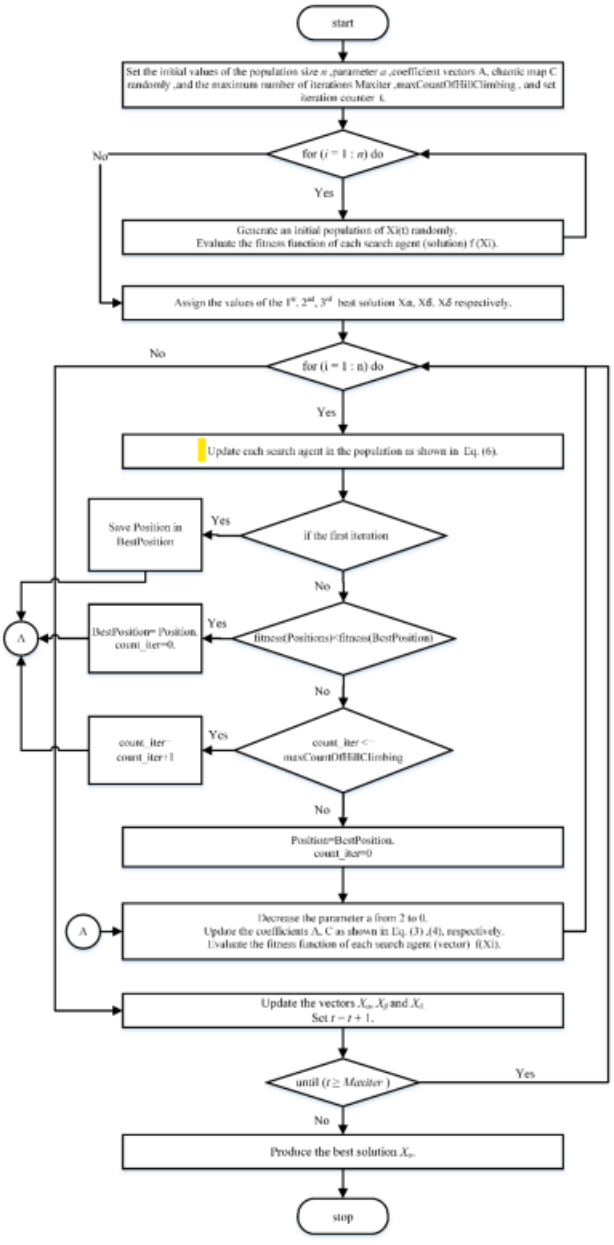

Figure 4. Flowchart of the proposed HCGWO algorithm

Remarks

(i) In `\runninghead` use ‘et al’ if there are three or more authors.

(ii) For multiple author papers please note the use of `\affilnum` to link names and affiliations. The corresponding author details need to be included using the `\corrauth` command.

(iii) For submitting a double-spaced manuscript, add `doublespace` as an option to the documentclass line.

(iv) The abstract should be capable of standing by itself, in the absence of the body of the article and of

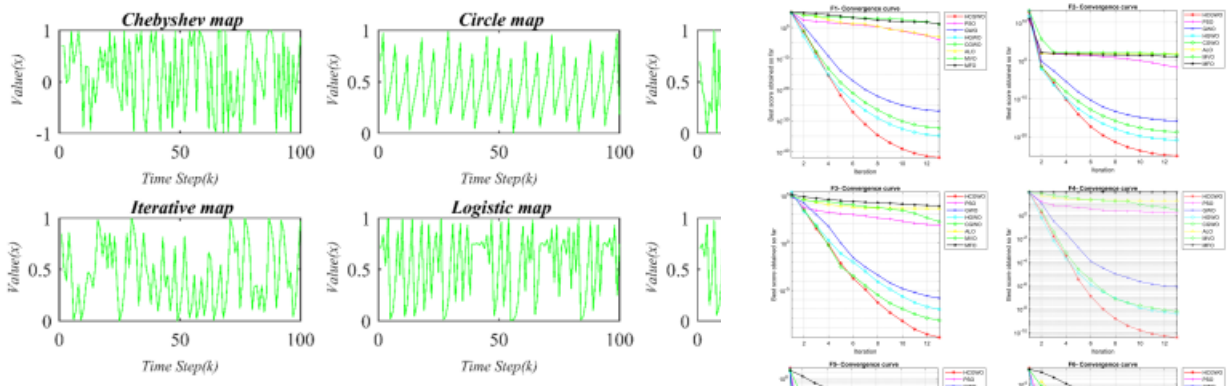

Figure 5. Flowchart of the proposed HCGWO algorithm

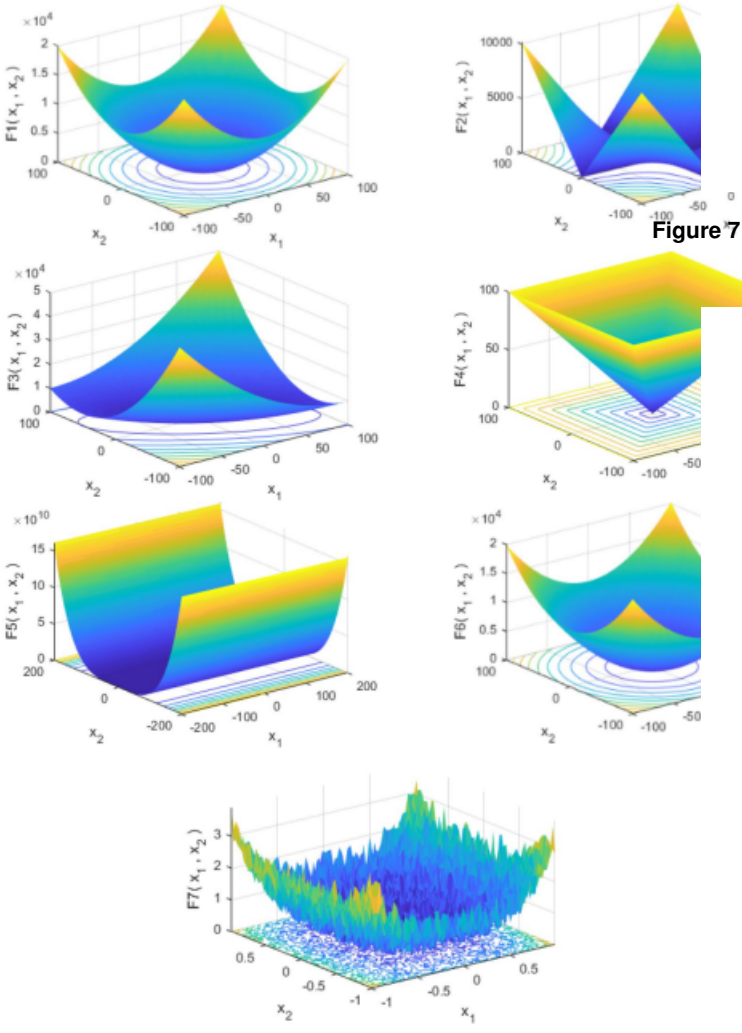

Figure 6

Figure 7

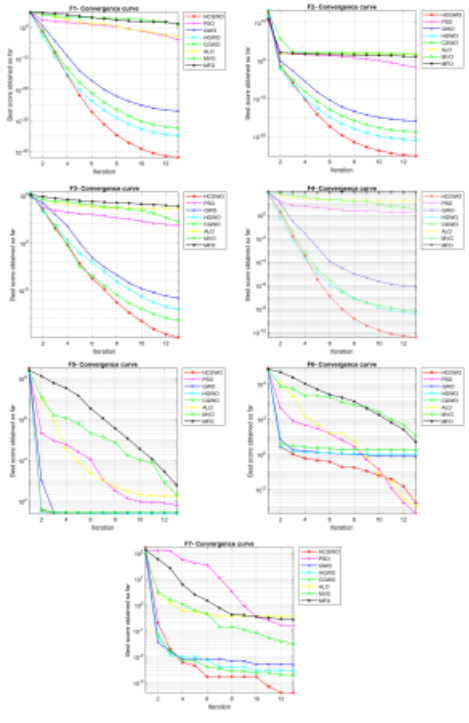

Figure 8

the bibliography. Therefore, it must not contain any reference citations.

(vi) If you are submitting to a *SAGE* journal that requires numbered sections (for example, IJRR), please add

```
\documentclass[<options>]{TRR}

\begin{document}

\runninghead{<Author surnames>}

\title{<Title case>}

\author{<An Author\affilnum{1},
Someone Else\affilnum{2} and
Perhaps Another\affilnum{1}>}

\affiliation{<\affilnum{1}First and third authors' affiliation\\
\affilnum{2}Second author affiliation>}

\corrauth{<Corresponding author's name and email address>}

\begin{abstract}
<Text>
\end{abstract}

\maketitle

\noindent <The first paragraph of text should not have a heading like 'Background'
           or 'Introduction'>
.
.
.
```

Figure 9. Example header text.

the command `\setcounter{secnumdepth}{3}` just above the `\begin{document}` line.

The Body of the Article

Mathematics

TRR.cls makes the full functionality of $\mathcal{AMS}$TEX available. We encourage the use of the `align`, `gather` and `multline` environments for displayed mathematics. amsthm is used for setting theorem-like and proof environments. The usual `\newtheorem` command needs to be used to set up the environments for your particular document.

Figures and Tables

TRR.cls includes the graphicx package for handling figures.

Figures are called in as follows:

```
\begin{figure}
\centering
\includegraphics{<figure name>}
\caption{<Figure caption>}
\end{figure}
```

```
\begin{table}
\small\sf\centering
\caption{<Table caption>}
\begin{tabular}{<table alignment>}
\toprule
<column headings>\\
\midrule
<table entries
(separated by & as usual)>\\
<table entries>\\
.
.
.\\
\bottomrule
\end{tabular}
\end{table}
```

Figure 10. Example table layout.

For further details on how to size figures, etc., with the graphicx package see, for example, (*1*) or (*3*).

The standard coding for a table is shown in Figure 10.

Cross-referencing

The use of the LaTeX cross-reference system for figures, tables, equations, etc., is encouraged (using `\ref{<name>}` and `\label{<name>}`).

End of Paper Special Sections

The endmatter order should be as follows (please note that only the Author Contributions, Declaration of Conflicting Interests, and Funding statements will appear in every article):

- Authors' Note
- Acknowledgments
- Author Contributions
- Declaration of Conflicting Interests
- Funding
- Data Accessibility Statement

The commands available are:

```
\begin{an}
To typeset an
  "Authors' Note" section.
\end{an}

\begin{acks}
To typeset an
  "Acknowledgements" section.
\end{acks}

\begin{ac}
<To typeset an
  "Author Contributions" section.>

The authors confirm contribution to
the paper as follows: study conception
and design: X. Author, Y. Author;
data collection: Y. Author;
analysis and interpretation of
results: X. Author, Y. Author. Z.
Author; draft manuscript preparation:
Y. Author. Z. Author.
All authors reviewed the results and
approved the final version
of the manuscript.
\end{ac}

\begin{dci}
To typeset a "Declaration of
  conflicting interests" section.
\end{dci}

\begin{funding}
To typeset a "Funding" section.
\end{funding}

\begin{das}
To typeset a "Data Accessibility
  Statement" section.
\end{das}
```

References

Please note that the file TRR.bst is included with the class file for those authors using BibTeX. The bst file works in a completely standard way:

```
\bibliographystyle{TRR}
\bibliography{<YourBibfile.bib>}
```

Copyright Statement

Please be aware that the use of this LaTeX 2$_\varepsilon$ class file is governed by the following conditions.

Copyright

Rules of Use

This class file is made available for use by authors who wish to prepare an article for publication in *Transportation Research Record*. The user may not exploit any part of the class file commercially.

This class file is provided on an *as is* basis, without warranties of any kind, either express or implied, including but not limited to warranties of title, or implied warranties of merchantablility or fitness for a particular purpose. There will be no duty on the author[s] of the software or SAGE Publications Ltd to correct any errors or defects in the software. Any statutory rights you may have remain unaffected by your acceptance of these rules of use.

Acknowledgements

This class file was developed by Sunrise Setting Ltd, Brixham, Devon, UK.
Website: http://www.sunrise-setting.co.uk

References

1. Kopka, H., and P. W. Daly. *A Guide to LaTeX*, 4th edn. Addison-Wesley, 2003.
2. Lamport, L. *LaTeX: a Document Preparation System*, 2nd edn. Addison-Wesley, 1994.
3. Mittelbach, F., and M. Goossens. *The LaTeX Companion*, 2nd edn. Addison-Wesley, 2004.