King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

A novel hybrid algorithm of GSA with Kepler algorithm for numerical optimization

Soroor Sarafrazi, Hossein Nezamabadi-pour [*], Saeid R. Seydnejad

Department of Electrical Engineering, Shahid Bahonar University of Kerman, P.O. Box 76169-133, Kerman, Iran

Received 26 March 2014; revised 20 June 2014; accepted 23 October 2014
Available online 18 June 2015

KEYWORDS

Numerical function optimization;
Hybrid algorithm;
Swarm intelligence;
Astrophysical concepts;
Gravitational Search Algorithm;
Kepler algorithm

Abstract It is now well recognized that pure algorithms can be promisingly improved by hybridization with other techniques. One of the relatively new metaheuristic algorithms is Gravitational Search Algorithm (GSA) which is based on the Newton laws. In this paper, to enhance the performance of GSA, a novel algorithm called "Kepler", inspired by the astrophysics, is introduced. The Kepler algorithm is based on the principle of the first Kepler law. The hybridization of GSA and Kepler algorithm is an efficient approach to provide much stronger specialization in intensification and/or diversification. The performance of GSA–Kepler is evaluated by applying it to 14 benchmark functions with 20–1000 dimensions and the optimal approximation of linear system as a practical optimization problem. The results obtained reveal that the proposed hybrid algorithm is robust enough to optimize the benchmark functions and practical optimization problems.

1. Introduction

Numerical optimization is an active and attractive field of research and in the past decade, more researchers are working on it. Many problems from very various domains are modeled as the optimization of a continuous function (Wang and Zhang, 2007; Zeng et al., 2009; Ong et al., 2009). Classical optimization algorithms cannot provide a suitable solution in so many complex fields due to the big problem size, dependency of these algorithms on initial solutions, etc. Therefore, solving these problems using classical techniques is impractical and this causes a growth interest in population-based Metaheuristic Algorithms (MA) and as a consequence, numerical problems have been tackled using MAs (Talbi, 2002). Greedy Heuristic (GH) (Papadimitriou and Steiglitz, 1982), simulated annealing (SA) (Kirkpatrick et al., 1983), Tabu Search (TS) (Glover, 1986), Iterated Local Search (ILS) (Lourenco et al., 2002), genetic algorithms (GA) (Holland, 1975), Evolution Strategies (ES) (Rechenberg, 1973), Genetic Programming (GP) (Koza, 1992), Ant Colony Optimization (ACO) (Dorigo et al., 1991), immune systems (Kephart, 1994), Gravitational Search Algorithm (Rashedi et al., 2009), Scatter Search (SS) (Talbi, 2002), etc. are listed as examples of MAs.

The abilities of exploration and exploitation are two main concerns for each metaheuristic algorithm which should be addressed when it is designed. In general, exploration (also known as diversification) indicates the capability of visiting

[*] Corresponding author. Tel./fax: +98 341 3235900.
E-mail address: nezam@uk.ac.ir (H. Nezamabadi-pour).

Peer review under responsibility of King Saud University.

http://dx.doi.org/10.1016/j.jksuci.2014.10.003

many different parts of the search space (unseen regions), whereas exploitation (also known as intensification) denotes the ability to attain more precise (high quality) solutions inside those explored parts. Each MA has several specific components for intensification and diversification (Blum and Roli, 2003).

Generally, a very difficult task in an MA is to provide an acceptable tradeoff between exploration and exploitation components (Talbi and Bachelet, 2006). Although most MAs attempt to accomplish this goal by their own way, it turns out that some of them tend to explore more and others are specialized in exploitation. Therefore, they are not able to find the optimal solution for all problems and some convergence problems may be encountered (e.g. in some cases their convergence speed is not fast enough) (Talbi, 2002; Jourdan et al., 2009). An alternative scheme to aid MAs to have both exploitation and exploration capability is hybridization of MAs with different specializing in intensification or diversification. This scheme combines MAs with the complementary behaviors, by which they compensate the weakness of each other (Lozano and G-Martinez, 2010).

Some reasons for hybridization can be noted as (Grosan and Abraham, 2007; Sinha and Goldberg, 2004): (1) to amend the performance of MAs, (2) to increase the quality of the solutions attained by MAs, and (3) to incorporate the MA as part of a larger system. Many authors have highlighted the need for hybridization of MAs with other optimization algorithms, MAs, machine learning techniques, etc. A concise summary of Hybrid Algorithms (HAs) is presented beneath:

The proposed HA in Caponio et al. (2009), called Super-Fit Memetic Differential Evolution (SFMDE), uses the Rosenbrock Algorithm (RA), the Nelder–Mead Algorithm (NMA) and a Particle Swarm Optimization (PSO), as the local searchers within the DE framework. The GA-based HA in Ullah et al. (2009) is proposed to handle constrained real-valued optimization problems. An agent chooses a Life Span Learning Processes (LSLP) as a self-adaptive local search operator. The knowledge experienced by the parents is used in the choosing process. Therefore, the adaptation level of both the algorithms is local-level adaptation (Ong et al., 2006). A framework of memetic optimization is introduced in Tenne and Armfield (2009) using variable global and local surrogate-models for optimizing costly functions. The models such as shape parameters and basis function vary during the optimization process. Radial Basis Functions (RBFs) are used to obtain an accurate and computationally efficient global model (Tenne and Armfield, 2009).

An adaptive hill climbing method is used in Wang et al. (2009) as a local search algorithm in the GA-based HA, which combines the features of steepest mutation-based hill climbing and greedy crossover-based hill climbing. In order to prevent premature convergence, two methods for maintaining diversity are also introduced in solving dynamic optimization problems. Reference Ishibuchi et al. (2009) clearly revealed the performance of biased neighborhood structures (i.e., biased to the problem size and the problem type) in multi-objective HAs while their implementation was not so sophisticated. A variable population-size genetic algorithm is hybridized by PSO algorithm in Shi et al. (2005).

References Liu et al. (2007), Shelokar et al. (2007) designed PSO-based HAs for multi-objective optimization and continuous optimization problems, respectively. Reference Shelokar et al. (2007) used ACO algorithm as a meta-heuristic local search in the HA. Dynamic Random Search Technique (RSET) as a local search is embedded into GA in Hamzacebi (2008). Two additional operations are combined with the original version of differential evolutions in Chiou and Wang (1998) to speed up the convergence rate without decreasing the exploration among the population.

The proposed algorithm in Wang and Zhang (2007) is the hybridization of DE with chaotic systems and pattern search method in order to extract the search space information and to speed up local exploiting, respectively. The DE algorithm in Wang et al. (2007) is hybridized with dynamic clustering technique to keep both diversification and intensification of population with higher precision. To achieve a fast convergence rate and high global convergence capability, an effective Hybrid Particle Swarm Cooperative Optimization (HPSCO) algorithm is proposed in Song et al. (2008) that combines simulated annealing and simplex algorithms.

HAs are successfully applied to various fields such as heterogeneous multiprocessor scheduling (Goh et al., 2009), multi-objective no-wait flow-shop scheduling (Qian et al., 2009), multi-objective 0/1 knapsack problems (Ozcan and Basaran, 2009), 3D-reconstruction of forensic objects (Santamaria et al., 2009), the optimal winner determination problem in combinatorial auctions (Boughaci et al., 2009), power-generator scheduling problem (Balci and Valenzuela, 2004), tumor classification (Shen et al., 2008), time-series forecasting (Behnamian and Fatemi, 2010), mobile ad hoc network (Wang et al., 2009). A hybrid algorithm of PSO and SA has been utilized to get the optimal solution for resource-constrained FMS scheduling problem (Wang and Liu, 2008) and so on.

The Gravitational Search Algorithm (GSA) is a stochastic swarm based metaheuristic algorithm which simulated the Newtonian laws of gravity and motion. As GSA is recently proposed, there are few hybrid algorithms based on it and for today's problems, GSA needs a better trade-off between exploration and exploitation. Therefore, this paper has introduced a new search algorithm called as Kepler algorithm inspired by the astrophysics and the second novelty of this paper is the hybridization of GSA with Kepler algorithm. This algorithm is a high-level Relay hybrid algorithm since GSA and Kepler algorithm are self-contained algorithms and are executed in sequence. Hybridization of GSA with the Kepler algorithm cooperates to speed up the search and increase the accuracy of the new hybrid algorithm because the Kepler algorithm helps GSA not to have a soon convergence and also makes it more specialized in I&D.

This paper is organized as follows: Section 2 offers a brief review of GSA. In Section 3, the proposed hybrid algorithm including the proposed algorithm as Kepler algorithm and its features are described. Section 4 presents 14 benchmark functions with an optimal approximation of linear system. Section 5 explains the experimentations on the functions presented in Section 4. Eventually, Section 6 draws the conclusion from the results.

2. Gravitational Search Algorithm

The Gravitational Search Algorithm (GSA) has been introduced by Rashedi et al. as a new swarm-based metaheuristic optimization tool (Rashedi et al., 2009). This algorithm provides an iterative search technique that simulates mass interactions, and moves through a multi-dimensional search space in the affection of gravitation. The performance of GSA and its variants like binary GSA (BGSA) (Harwit, 1998) in solving real-word application (Gong et al., 2008; Zhong et al., 2004) as well as a set of standard functions has been confirmed (Rashedi et al., 2009; Harwit, 1998).

To describe the GSA, let us to consider a system with N objects (agents) in which the position of the ith object is presented as follows:

$$X_i = (x_i^1, \ldots, x_i^d, \ldots, x_i^n), \quad i = 1, 2, \ldots, N \tag{1}$$

where x_i^d is the position of ith object in the dth dimension and n is dimension of the problem to be optimized. It is noted that the positions of objects correspond to the solutions of the problem. Based on Rashedi et al. (2009), the mass value of each object is estimated after calculating the fitness of the current population by Eq. (2):

$$M_i(t) = \frac{fit_i(t) - worst(t)}{\sum_{j=1}^{N}(fit_j(t) - worst(t))} \tag{2}$$

where $M_i(t)$ and $fit_i(t)$ denote the mass value and the fitness value of the object i at t, and, $worst(t)$ is defined as follows (for a minimization problem):

$$worst(t) = \max_{j \in \{1,\ldots,N\}} fit_j(t) \tag{3}$$

To calculate the acceleration of an agent, total forces from a set of heavier objects that apply to it must be measured based on law of gravity (Eq. (4)), which is followed by Eq. (5). Then, the next velocity of an object is computed using Eq. (6). Finally, its next position is calculated using Eq. (7).

$$F_i^d(t) = \sum_{j \in kbest, j \neq i} rand_j \; G(t) \frac{M_j(t) M_i(t)}{R_{ij}(t) + \varepsilon} \left(x_j^d(t) - x_i^d(t) \right) \tag{4}$$

$$a_i^d(t) = \frac{F_i^d(t)}{M_i(t)} = \sum_{j \in kbest, j \neq i} rand_j \; G(t) \frac{M_j(t)}{R_{ij}(t) + \varepsilon} \left(x_j^d(t) - x_i^d(t) \right) \tag{5}$$

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \tag{6}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{7}$$

where $rand_i$ and $rand_j$ are two uniformly distributed random numbers in the interval $[0, 1]$, ε is a small value, and $R_{ij}(t)$ is the Euclidian distance between two objects i and j defined as $R_{ij}(t) = \|X_i(t), X_j(t)\|_2$. $kbest$ is the set of first K objects with the best fitness value and biggest mass. $kbest$ is a function of time, which is initialized to K_0 at the beginning and is decreased with time. Here, K_0 is set to N (swarm size) and is decreased linearly to 1. G decreases exponentially from G_0 toward zero by time. The pseudo code of the GSA is given by Fig. 1.

```
1  Search space identification, t=0;
2  Randomized initialization $X_i(t)$ $for\ i = 1,2,\ldots,N$ ;
3  Fitness evaluation of agents;
4  Update $G(t)$, $worst(t)$ and $M_i(t)$ $for\ i = 1,2,\ldots,N$ ;
5  Calculation of the total force in different directions;
6  Calculation of acceleration and velocity;
7  Updating agents' position to yield $X_i(t+1)$ $for\ i = 1,2,\ldots,N$ , t=t+1;
8  Repeat steps 3 to 8 until the stopping criterion is reached.
```

Figure 1 Pseudo code of the GSA.

3. The proposed GSA based HA algorithm

In this Section the proposed HA, hybridization of GSA and new proposed algorithm, is presented. This new proposed algorithm is Kepler algorithm. Section 3.1 presents a brief summary about astrophysical concepts used in the proposed algorithm.

3.1. Astrophysical concepts

The first Kepler's law says (Leung and Wang, 2001): "The orbits along which planets move about the Sun are ellipses." And it means that the planets have different distances to the Sun, therefore, it motivated us to make an algorithm in which the best solution is the Sun and the planets are the other solutions and called it "Kepler". In Kepler algorithm, the planets (solutions or objects) have different situations to the Sun (the best solution) in different times and this causes to explore and exploit the space search more efficiently. Its reason is that in different times, the other planets get so much near to the Sun which means the algorithm exploits more accurately since it searches new places around the best solution. They sometimes get so far from the Sun, this leads to explore the whole search space more efficiently. In Fig. 2, it is shown more clearly why elliptical orbits cause both exploration and exploitation.

For implementation, we calculate the distance between each planet and the Sun and by Eq. (8) the new situations for the chosen planets are defined. According to the movement of the Sun in the space and in order to have more exploration and not get stuck in local optimum, the Sun (the best solution) has the opportunity to change its position by Eq. (9). The pseudo code of the Kepler is given by Fig. 3.

$$X_{i,new}(t+1) = X_{best}(t) + R_{i,best} \times U(-2, 2) \tag{8}$$

$$X_{best,new}(t+1) = X_{best}(t) \times U(-2, 2) \tag{9}$$

In this equation, $U(-2, 2)$ returns uniformly distributed pseudo random number in the interval $[-2, 2]$. In Eq. (8), a new distance from the best solution for each solution is produced by $R_{i,best} \times U(-2, 2)$. The algorithm exploits if $U(-2, 2)$ returns a number near 1, otherwise it explores. Following by the elitism, the next position of the chosen planets and the Sun will be determined by Eq. (10).

$$X_i(t+1) = \begin{cases} X_{i,new}(t+1) if\ fit(X_{i,new}) < fit(X_i) \\ X_i(t)\ otherwise \end{cases} \tag{10}$$

3.2. GSA–Kepler algorithm

In this hybrid algorithm, GSA and Kepler are applied one after another; each is using the output of the previous as its input, acting in a pipeline fashion which is shown in Fig. 4.

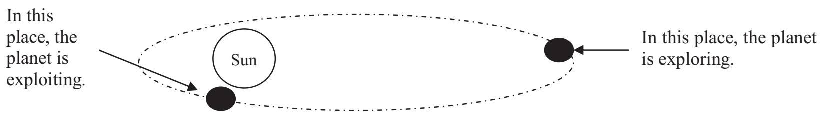
In this place, the planet is exploiting.
Sun
In this place, the planet is exploring.

Figure 2 Elliptical orbits of planets around the Sun.

```
1. Choosing K agents from $X_i(t)$ $for\ i = 1,2,...,N$;
2. Calculation of the Euclidean distance between two agents $i$ and the best agent;
3. Calculation the new positions of K agents and best agent $X_{j,new}(t+1)$ $for\ j = 1,2,...,K$ by Eqs.8 and 9.
4. Updating agents' position to yield $X_j(t+1)$ $for\ j = 1,2,...,K$ on elitism, t=t+1;
5. Repeat steps 1 to 5 until the stopping criterion is reached.
```

Figure 3 Pseudo code of the Kepler.

4. Benchmark functions and practical optimization problems in linear system

To validate the performance of the proposed algorithm, we applied it to 14 benchmark functions (Rashedi et al., 2009). These functions are given in Section 4.1 and optimal estimation of linear system is presented in Section 4.2.

4.1. Benchmark functions

In the experiments, a set of standard benchmark functions are taken from Rashedi et al. (2009). The functions are presented in Tables 1 and 2. In these tables n, and f_{opt} indicate the number of the dimensions of the function, and the optimum value of the function respectively and $S \subseteq R^n$. The functions of Table 1 (F_1–F_7) are unimodal. For an unimodal function, the convergence rate of the algorithm is more interesting than the final result of optimization. The functions of Table 2 (F_8–F_{14}) are multimodal having many local minima and the algorithm should be able in finding the optimum solution and it should not be trapped in local optima. More details regarding these functions are available in appendix of Rashedi et al. (2009).

In order to have a fair comparison of the proposed algorithm with the algorithm introduced in Gong et al. (2008), the same problems (benchmark functions) should be used and the values of the dimensions and other parameters in the problems should be the same as the values in Gong et al. (2008). Therefore, Dimension is considered to be 30 ($n = 30$) for all functions except for F_{14} which is set to 100. The optimum of all functions is 0 except for F_8 and F_{14} which are $-412.9829n$ and -78.33236, respectively.

```
1  Search space identification, t=0;
2  Randomized initialization $X_i(t)$ $for\ i = 1,2,...,N$;
3  Fitness evaluation of agents;
4  Update $G(t)$, $worst(t)$ and $M_i(t)$ $for\ i = 1,2,...,N$;
5  Calculation of the total force in different directions;
6  Calculation of acceleration and velocity;
7  Updating agents' position to yield $X_i(t+1)$ $for\ i = 1,2,...,N$, t=t+1;
8  Implementing the Kepler algorithm.
9  Repeat steps 3 to 8 until the stopping criterion is reached.
```

Figure 4 Pseudo code of the GSA–Kepler.

Table 1 Unimodal test functions.

Test function	S
$F_1(X) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^n$
$F_2(X) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-10, 10]^n$
$F_3(X) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	$[-100, 100]^n$
$F_4(X) = \max_i\{\lvert x_i \rvert, 1 \leqslant i \leqslant n\}$	$[-100, 100]^n$
$F_5(X) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$	$[-30, 30]^n$
$F_6(X) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	$[-100, 100]^n$
$F_7(X) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	$[-1.28, 1.28]^n$

4.2. Approximation of linear system

Optimal estimation of linear system is a prominent task in the simulation of dynamic complicated systems (Cheng and Hwang, 2001). Up to now, many methods have been proposed to solve the model approximation problem which can be categorized into two main groups: the performance-oriented and the nonperformance-oriented schemes. An approximation method, belonging to the performance-oriented group, approximates models by minimizing a certain error function.

4.2.1. Problem statement

Assume that we have a high-order rational/irrational transfer function $G(s)$, our desire is to approximate the model of the form in Eq. (12) such that $H_m(s)$ covers the desirable characteristic of the system $G(s)$.

$$H_m(s) = \frac{b_0 + b_1 s + \ldots + b_{m-1} s^{m-1}}{a_0 + a_1 s + \cdots + a_{m-1} s^{m-1} + s^m} e^{-\tau_d s} \tag{12}$$

To achieve the goal of find optimal approximate model $H_m(s)$, the frequency domain error function in Eq. (13) is minimized, where the frequency points, $\omega_i\ i = 0, 1, 2, \ldots, N$, and the integer N are taken a priori.

$$J = \sum_{i=0}^{N} \lvert G(j\omega_i) - H_m(j\omega_i) \rvert^2 \tag{13}$$

It is worth noticing that once the system $G(s)$ is asymptotically stable, the constraint, $H_m(0) = G(0)$, is considered to guarantee that the steady-state responses of the system $G(s)$ and the approximate model are the same for the unit-step input. The

Table 2 Multimodal test functions.

Test function	S
$F_8(X) = \sum_{i=1}^{n} - x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)$	$[-500, 500]^n$
$F_9(X) = \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i) + 10]$	$[-5.1\prime\prime, 5.12]^n$
$F_{10}(X) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	$[-32, \#2]^n$
$F_{11}(X) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^n$
$F_{12}(x) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i+1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m x_i > a \\ 0 - a < x_i < a \\ k(-x_i - a)^m x_i < -a \end{cases}$	$[-50, 50]^n$
$F_{13}(X) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	$[-50, 50]^n$
$F_{14}(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i^4 - 16x_i^2 + 5x_i)$	$[-5, 5]^n$

minimization of the error function given in Eq. (13) is an optimal parameter selection problem.

4.2.2. Optimal approximation of an unstable linear system

The information of this system is taken from Zhong et al. (2004), Cheng and Hwang (2001), Guo and Hwang (1996). Assume the fourth-order unstable and non-minimum-phase transfer function:

$$G(s) = \frac{60s^3 + 25850s^2 + 685000 - 2500000}{s^4 + 105s^3 + 10450s^2 + 45000s - 500000} \tag{14}$$

The desire is to approximate this transfer function, $G(s)$, by the second-order model

$$H_2(s) = \frac{c_{2,1}s + c_{2,0}}{s^2 + b_{2,1}s + b_{2,0}} \tag{15}$$

in such a way that the error function defined in Eq. (13) with $\omega_i = 10^{-2+0.2i}$, $i = 0, 1, \ldots, N = 60$, is minimized. The optimal parameters are searched by GSA–Kepler. In this process, the acceptable interval for each parameter is $(-\infty, +\infty)$.

5. Results and discussion

We use 9 out of 14 experiments to evaluate the performance of GSA–Kepler hybrid algorithm by comparing with Modified Artificial Immune System algorithm (MAISA) Gong et al., 2008 and also GSA and the other benchmark functions are just applied to compare GSA–Kepler and GSA. The existing results reported in Gong et al. (2008) can be used for comparing with the proposed algorithm. First, a brief explanation of MAISA is presented in the following.

MAISA: It is a metaheuristic algorithm for global optimization. In fact, it is a dynamic version of the population-based architecture of an artificial immune system. The MAISA simulates the dynamic behaviors of human immune reaction as a quaternion (R, I, G, Al), where R, I, G, and Al indicate the set of response rules which explain how antibodies interact with each other, the set of valid antibodies, antigen or exterior stimulus, and the dynamic algorithm describing how the response rules are exert to antibody population, respectively.

The three mentioned algorithms (competing algorithms) are applied to the benchmark functions, and the results for unimodal and multimodal functions are:

5.1. Unimodal high-dimensional functions

As aforementioned, functions F_1 to F_7 are unimodal functions. For a meaningful comparison, the results obtained are averaged over 30 independent runs and the Average Best-So-Far solutions (ABSF), and Standard Deviation of the best-so-far solutions (STDV) in the last iteration of 30 independent runs are reported for unimodal functions in Table 3. It is noticed that the number of Fitness Evaluations (FEs) for GSA and GSA–Kepler is fixed to 2500.

In GSA and GSA–Kepler, gravitational constant, G, is considered to be an exponential decreasing function of time (Eq. (11)), in which G_0 and β are set to 100 and 20, respectively and $t_{\max}$ is the total number of iterations (Rashedi et al., 2009):

$$G = G_0 \exp\left(-\frac{\beta \times t}{t_{\max}}\right) \tag{11}$$

This table demonstrates the robust power of GSA–Kepler for finding the optimum. The proposed algorithm increases the ability of GSA in both exploring and exploiting, and also has a high convergence rate; therefore these characteristics significantly cause good results. As this table shows not only the GSA–Kepler is more capable than MAISA to exploit but also it is much faster (it uses less function evaluation than MAISA); thus, the proposed algorithm is stronger than MAISA except in F_7.

5.2. Multimodal high-dimensional functions

The experiments have been carried out on the functions F_8–F_{14} for these functions, the number of local optima increases exponentially with the increase of dimension of the function. Similar to the Section 5.1 the results are averaged over 30 independent runs and the Average Best-So-Far solutions (ABSF), and Standard Deviation of the best-so-far solutions (STDV) in the last iteration of 30 runs are reported for Multimodal functions in Table 4. It is noticed that the number of fitness evaluations for GSA and GSA–Kepler is fixed to 2500.

This table again demonstrates the power of GSA–Kepler. In all cases, it is much faster than MAISA except in F_8. GSA–Kepler was not able to find very good result for F_8 since the population has converged and more exploration is needed to get better result for this case. However, its result is better than GSA and Kepler algorithm has improved GSA.

Table 3 Performance comparisons of GSA–Kepler, GSA and MAISA in benchmark functions in Table 1.

F	Mean number of functions evaluations	Mean function value (standard deviation)		
	MAISA	GSA–Kepler	GSA	MAISA
F_1	3956	$2.38 \times 10^{-38} \pm 9.87 \times 10^{-38}$	$4.34 \times 10^{3} \pm 1.49 \times 10^{3}$	$2.51 \times 10^{-17} \pm 4.00 \times 10^{-18}$
F_2	3206	$1.78 \times 10^{-19} \pm 6.41 \times 10^{-19}$	37.25 ± 16.21	$7.99 \times 10^{-14} \pm 9.35 \times 10^{-15}$
F_3	7146	$3.14 \times 10^{-30} \pm 1.71 \times 10^{-29}$	$8.25 \times 10^{3} \pm 3.56 \times 10^{3}$	$2.44 \times 10^{-11} \pm 1.44 \times 10^{-11}$
F_4	—	$1.51 \times 10^{-18} \pm 8.28 \times 10^{-18}$	27.24 ± 4.96	—
F_5	—	11.43 ± 13.55	$1.74 \times 10^{5} \pm 1.77 \times 10^{5}$	—
F_6	—	0.3629 ± 0.2082	$3.68 \times 10^{3} \pm 1.38 \times 10^{3}$	—
F_7	2708	0.0032 ± 0.0026	1.68 ± 1.09	$3.06 \times 10^{-11} \pm 1.36 \times 10^{-11}$

GSA–Kepler has the ability to explore more and it exactly leads to find the global optimums in F_9 and F_{11}. On the other hand, it is less powerful than MAISA to explore in F_8 and F_{14}; however, the GSA–Kepler is much faster than MAISA in $F_{14.}$

From Tables 3 and 4, it is concluded that GSA–Kepler gives the opportunities to agents to get close or become far from the best agents and these lead to exploit and explore, respectively and GSA–Kepler is able to maintain the population diversity whenever GSA is exploiting and the power of its exploration decreases. Therefore, the proposed algorithm, GSA–Kepler, in these sets of experiments could attain sufficient solutions with a smaller number of FEs.

5.2.1. Performance evaluations of GSA, MAISA and GSA–Kepler

For the functions of Table 2, with the increase of dimensions the number of local optima increases and the problem optimization will be more difficult. Therefore, in this set of experiment, the performance of GSA–Kepler on functions with 20–1000 dimensions is examined. The termination criterion of GSA–Kepler similar to (Gong et al., 2008) is one of the objectives, $|f_{opt} - f_{\min}| < \epsilon$, $|f_{\min}|$ or $|f_{opt}| < \epsilon$, where $f_{\min}$ represents the best so far solution. The parameter ϵ is set to 10^{-4} for all functions, which is the same as that of Zhong et al. (2004). The statistical results of GSA–Kepler, GSA and MAISA when optimizing the functions F_8, F_9, F_{10} and F_{11} with different dimensions are given in Table 5. The reported results of MAISA are obtained from Gong et al. (2008). Here, each result of GSA–Kepler and GSA is obtained from 50 independent runs.

The power of hybridization of Kepler and GSA is the primary deduction from Table 5. In this table, comparing with other algorithms, it is concluded that the performance of GSA can be improved by hybridization with Kepler, except in F_8. The second conclusion is the satisfying solutions obtained by GSA–Kepler at a lower computation cost than other algorithms and the third one is the high ability of this algorithm to solve large parameter optimization problems and as this table shows, this algorithm can solve large parameter optimization problems as well as small optimization problems.

5.3. Comparison of GSA, GSA–Kepler and GSA-Disruption

In order to have a better study of GSA–Kepler performance, GSA, GSA–Kepler and GSA-Disruption are compared in this section. GSA-Disruption is an improved GSA which has a new operator inspired by astrophysics and it is called "disruption" (Sarafrazi et al., 2011). The comparison of these algorithms with the same number of function evaluations (50,000) is shown in Fig. 5.

This figure shows that Kepler algorithm was robust enough to increase the performance of GSA in exploration and exploitation.

5.4. Comparative studies on approximation of linear system

The optimal value of error in Eq.13 for parameters a_i, b_i, $i = 1, 2, \ldots, m-1$ and τ_d of the approximate model (Eq. (12)) has been estimated by a gradient-based method (Aplevich, 1973), by a direct search optimization (Zhong et al., 2004; Cheng and Hwang, 2001; Guo and Hwang, 1996; Aplevich, 1973) or by metaheuristic algorithms (Gong et al., 2008). The methods of this group employ numerical optimization algorithms (Cheng and Hwang, 2001; Du et al., 2005).

To validate the performance of the proposed GSA–Kepler in attaining optimal solutions, some experiments are carried

Table 4 Performance comparisons of GSA–Kepler, GSA and MAISA in benchmark functions in Table 2.

F	Mean number of functions evaluations	Mean function value (standard deviation)		
	MAISA	GSA–Kepler	GSA	MAISA
F_8	1984	$-3.5123 \times 10^{3} \pm 587.47$	$-2.2128 \times 10^{3} \pm 550.95$	$-12569.49 \pm 1.04 \times 10^{-7}$
F_9	2528	0 ± 0	98.78 ± 23.48	$1.70 \times 10^{-12} \pm 1.67 \times 10^{-11}$
F_{10}	2774	$8.88 \times 10^{-16} \pm 0$	8.05 ± 1.68	$3.51 \times 10^{-16} \pm 2.19 \times 10^{-17}$
F_{11}	2612	0 ± 0	282.45 ± 39.89	$1.01 \times 10^{-15} \pm 3.85 \times 10^{-16}$
F_{12}	—	0.0422 ± 0.0313	$613.42 \pm 1.33 \times 10^{3}$	—
F_{13}	—	0.0489 ± 0.1042	$1.49 \times 10^{5} \pm 2.74 \times 10^{5}$	—
F_{14}	4794	-41.3864 ± 3.2184	-49.5801 ± 2.2996	$-78.3323 \pm 1.97 \times 10^{-9}$

Table 5 Performance comparison of GSA–Kepler, GSA, MAISA in benchmark functions in Table 2.

D	Mean function value (standard deviation)		
	GSA–Kepler	GSA	MAISA
F_8			
20	_	_	957
100	_	_	2787
200	_	_	3618
400	_	_	6285
1000	_	_	10,920
F_9			
20	496	_	1497
100	491	_	4914
200	518	_	8937
400	515	_	13,728
1000	534	_	17,585
F_{10}			
20	589	6978	1372
100	613	_	3036
200	588	_	4664
400	603	_	6126
1000	608	_	7192
F_{11}			
20	492	_	1018
100	499	_	3990
200	499	_	4982
400	508	_	5740
1000	514	_	6988

out for optimal approximate models of a system. This system, which is quoted from Guo and Hwang (1996), Parker and Anderson (1987), is unstable and non-minimum-phase. The parameters of GSA–Kepler are the same as those of Section 5.1. The number of evaluations of both GSA and GSA–Kepler is set to 18,000.

In search for the parameters $X^4 = [c_{2,1}, c_{2,0}, b_{2,1}, b_{2,0}]$, we have tested four fixed intervals: $[-500, 500]^4$, $[-1000, 1000]^4$, $[-2000, 2000]^4$ and $[-5000, 5000]^4$. The obtained approximate models for $G(s)$ and the corresponding error values by competing algorithms are given in Table 6. This table also presents the L^{∞}-norm of the approximation error function, $\varepsilon = |G(j\omega)\_H_2(j\omega)|$ for each model. The algorithm which gives smaller values of J and ε has more optimal results and it is known as a more powerful algorithm among the competing ones.

In the next step, the optimal parameters are searched by GSA–Kepler while the initial search space for the parameters $X^4 = [c_{2,1}, c_{2,0}, b_{2,1}, b_{2,0}]$ are set as $[-0.1, 0.1]^4$.The optimal approximate models for $G(s)$ and the corresponding error values achieved by GSA–Kepler are compared to competing algorithms in Table 7.

As can be seen from Table 6, GSA–Kepler has a good convergence in the four fixed search spaces but a too big search space affects the performance of the GSA in approximating the unstable linear system. The results of Table 7 show that the optimal error value (J) of the model obtained by GSA–Kepler is as good as those of GSA and MAISA but the L^{∞}-norm of the approximation error of the models obtained by GSA–Kepler is the best.

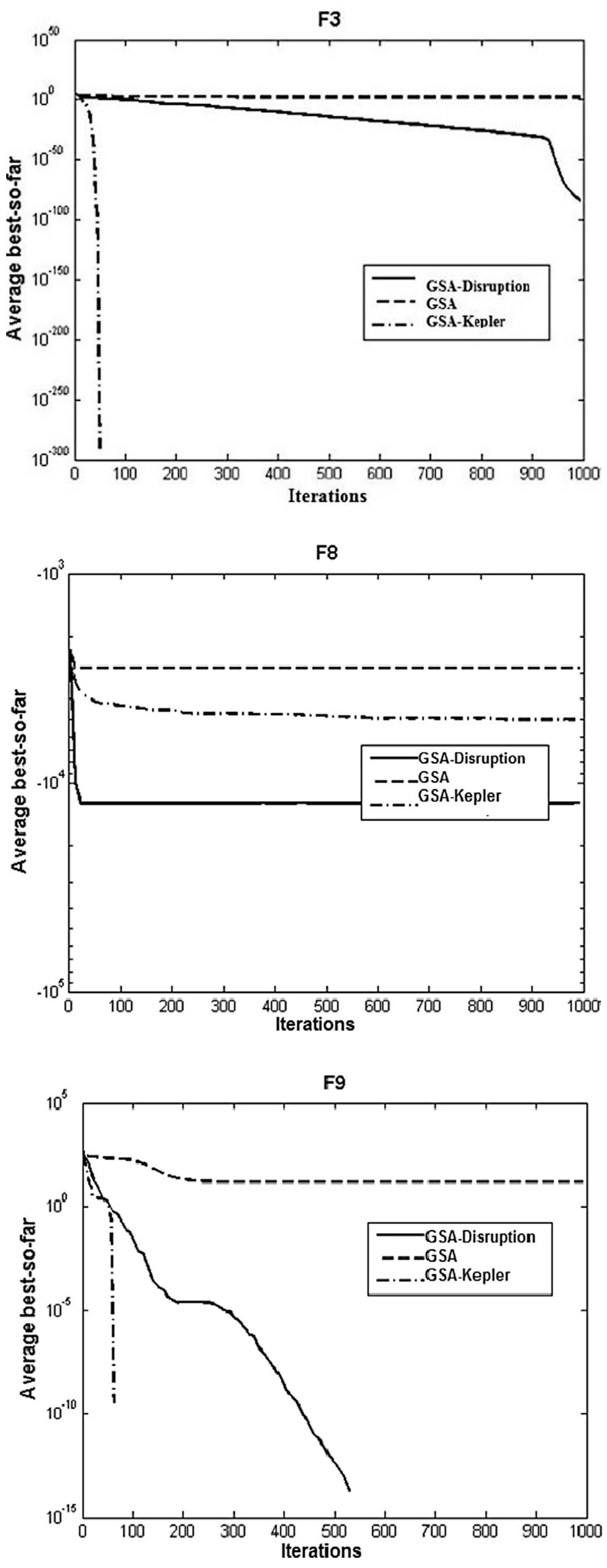

Figure 5 Comparison of GSA, GSA–Kepler and GSA-Disruption for functions 3, 8 and 9.

Table 6 Comparisons of the approximate models of the unstable linear system obtained by GSA and GSA–Kepler with fixed search spaces.

Search space	Method	Approximate model	J	ε
$[-500, 500]^4$	GSA–Kepler	$H_2(s) = \frac{129.438454370s - 453.295383346}{s^2 + 13.078275324s - 92.263688320}$	8.795	1.378
	GSA	$H_2(s) = \frac{125.708605004s - 391.791310133}{s^2 + 13.285565830s - 79.981542679}$	8.840	1.419
$[-1000, 1000]^4$	GSA–Kepler	$H_2(s) = \frac{128.911415273s - 443.794051176}{s^2 + 13.115496129s - 90.373143350}$	8.796	1.384
	GSA	$H_2(s) = \frac{129.135318268s - 447.925936973}{s^2 + 13.098233831s - 91.194024300}$	8.795	1.382
$[-2000, 2000]^4$	GSA–Kepler	$H_2(s) = \frac{128.691611201s - 439.913092050}{s^2 + 13.130374280s - 89.599959550}$	8.796	1.386
	GSA	$H_2(s) = \frac{134.509054370s - 549.740724214}{s^2 + 12.710681134s - 111.447767325}$	8.884	1.325
$[-5000, 5000]^4$	GSA–Kepler	$H_2(s) = \frac{128.036059193s - 416.566312673}{s^2 + 13.329604555s - 84.973975862}$	8.811	1.392
	GSA	$H_2(s) = \frac{321.649803301s + 77.190237460}{s^2 + 85.748991891s + 15.572507128}$	62.107	3.228

Table 7 Comparisons of the approximate models of the unstable linear system obtained by GSA–Kepler, GSA and MAISA.

Method	Approximate model	J	ε
MAISA Gong et al., 2008	$H_2(s) = \frac{129.310457004s - 450.941328300}{s^2 + 13.088391129s - 91.794882948}$	8.795	1.380
GSA	$H_2(s) = \frac{129.135318268s - 447.925936973}{s^2 + 13.098233831s - 91.194024300}$	8.795	1.382
GSA–Kepler	$H_2(s) = \frac{129.438454370s - 453.295383346}{s^2 + 13.078275324s - 92.263688320}$	8.795	1.378

6. Conclusion

In this paper, we introduced a novel algorithm inspired by the astrophysical concepts and called it Kepler because it is based on the first Kepler's law. By the hybridization of the Kepler algorithm with GSA, the performance of GSA and the quality of its solutions have been improved since the Kepler algorithm gives opportunities to solutions to get close to the best solution or far from it. Being close or far provides exploitation or exploration, respectively. Therefore, the hybridization of these algorithms (GSA and Kepler) makes a useful algorithm for numerical optimization.

References

Aplevich, J.D., 1973. Gradient methods for optimal linear system reduction. Int. J. Control 18 (4), 767–772.

Balci, H.H., Valenzuela, J.F., 2004. Scheduling electrical power generators using particle swarm optimization combined with the Langrangian relaxation method. Int. J. Appl. Math. Comput. Sci. 3 (14), 411–421.

Behnamian, J., Fatemi Ghomi, S.M.T., 2010. Development of a PSO–SA hybrid meta-heuristic for a new comprehensive regression model to time-series forecasting. Expert Syst. Appl. 37 (2), 974–984.

Blum, C., Roli, A., 2003. Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput. Surv. 35 (3), 268–308.

Boughaci, D., Benhamou, B., Drias, H., 2009. A memetic algorithm for the optimal winner determination problem. Soft Comput. 13, 905–917.

Caponio, A., Neri, F., Tirronen, V., 2009. Super-fit control adaption in memetic differential evolution frameworks. Soft. Comput. 13, 811–831.

Cheng, S.L., Hwang, C., 2001. Optimal approximation of linear systems by a differential evolution algorithm. IEEE Trans. Syst. Man Cybern. A 31 (6), 698–707.

Chiou, J.P., Wang, F.S., 1998. A hybrid method of differential evolution with application to optimal control problems of a bioprocess system. In: The IEEE International Conference on Evolutionary Computation Proceedings, pp. 627–632.

Dorigo, M., Maniezzo, V., Colorni, A., 1991. Positive feedback as a search strategy, Technical Report No. 91016. Dipartimento di Elettronica e Informatica, Politecnico di Milano, Italie.

Du, H.F., Gong, M.G., Jiao, L.C., Liu, R.C., 2005. A novel artificial immune system algorithm for high-dimensional function numerical optimization. Prog. Nat. Sci. 15 (5), 463–471.

Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. Comput. Oper. Res. 13, 533–549.

Goh, C.K., Teoh, E.J., Tan, K.C., 2009. A hybrid evolutionary approach for heterogeneous multiprocessor scheduling. Soft Comput. 13, 833–846.

Gong, M., Jio, L., Zhang, X., 2008. A population-based artificial immune system for numerical optimization. Neurocomputing 72, 149–161.

Grosan, C., Abraham, A., 2007. Hybrid evolutionary algorithms: methodologies, architectures, and reviews. In: Grosan, C., Abraham, Ishibuchi, H. (Eds.), Hybrid Evolutionary Algorithms. Springer, Berlin, Heidelberg, pp. 1–17.

Guo, T.Y., Hwang, C., 1996. Optimal reduced-order models for unstable and nonminimum-phase systems. IEEE Trans. Circuits Syst. I 43 (9), 800–805.

Hamzacebi, C., 2008. Improving genetic algorithms' performance by local search for continuous function optimization. Appl. Math. Comput. 196, 309–317.

Harwit, M., 1998. The Astrophysical Concepts, third ed., New York, p. 65.

Holland, J., 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press.

Ishibuchi, H., Hitotsuyanagi, Y., Tsukamoto, N., Nojima, Y., 2009. Use of biased neighborhood structures in multi-objective memetic algorithms. Soft. Comput. 13, 795–810.

Jourdan, L., Basseur, M., Talbi, E.G., 2009. Hybridizing exact methods and metaheuristics: a taxonomy. Eur. J. Oper. Res. 199, 620–629.

Kephart, J.O., 1994. A biologically inspired immune system for computers. In: Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems. MIT Press, Cambridge, MA, USA, pp. 130–139.

Kirkpatrick, S., Gellato, C.D., Vecchi, M.P., 1983. Optimization by simulated annealing. Science 220, 671–680.

Koza, J.R., 1992. Genetic Programming: On the Programming of Computers by Natural Selection. MIT Press, Cambridge, MA.

Leung, Y.W., Wang, Y.P., 2001. An orthogonal genetic algorithm with quantization for global numerical optimization. IEEE Trans. Evol. Comput. 5 (1), 41–53.

Liu, D., Tan, K.C., Goh, C.K., Ho, W.K., 2007. A multi-objective memetic algorithm based on particle swarm optimization. IEEE Trans. Syst. Man Cybern. Part B 37 (1), 42–50.

Lourenco, H.R., Martin, O., Stutzle, T., 2002. Handbook of Metaheuristics, Iterated Local Search. Kluwer Academic Publishers, Norwell, MA.

Lozano, M., G-Martinez, C., 2010. Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: overview and progress report. Comput. Oper. Res. 37, 481–497.

Ong, Y., Lim, M., Zhu, N., Wong, K., 2006. Classification of adaptive memetic algorithms: a comparative study. IEEE Trans. Syst. Man Cybern. Part B 36 (1), 141–152.

Ong, Y.S., Lim, M.H., Neri, F., Ishibichi, H., 2009. Special issue on emerging trends in soft computing: memetic algorithms. Soft Comput. 13, 739–740.

Ozcan, E., Basaran, C., 2009. A case study of memetic algorithms for constraint optimization. Soft Comput. 13, 871–882.

Papadimitriou, C.H., Steiglitz, K., 1982. Combinatorial Optimization: Algorithms and Complexity. Prentice-Hall.

Parker, P.J., Anderson, B.D.O., 1987. Unstable rational function approximation. Int. J. Control 46 (5), 1783–1801.

Qian, B.K., Wang, L., Huang, D.X., Wang, X., 2009. Multi-objective no-wait flow-shop scheduling with a memetic algorithm based on differential evolution. Soft Comput. 13, 847–869.

Rashedi, E., Nezamabadi-pour, H., Saryazdi, S., 2009. GSA: a gravitational search algorithm. Inf. Sci. 179 (13), 2232–2248.

Rechenberg, I., 1973. Evolutions Strategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution. Frommann-Holzboog, Stuttgart.

Santamaria, J., Cordon, O., Damas, S., Garcia-Torres, J.M., Quirin, A., 2009. Performance evaluation of memetic approaches in 3D reconstruction of forensic objects. Soft Comput. 13, 883–904.

Sarafrazi, S., Nezamabadi-pour, H., Saryazdi, S., 2011. Disruption: a new operator in gravitational search algorithm. Sci. Iran. 18 (3), 539–548.

Shelokar, P.S., Siarry, P., Jayaraman, V.K., Kulkarni, B.D., 2007. Particle swarm and ant colony algorithms hybridized for improved continuous optimization. Appl. Math. Comput. 188, 129–142.

Shen, Q., Shi, Wei-Min, Kong, W., 2008. Hybrid particle swarm optimization and tabu search approach for selecting genes for tumor classification using gene expression data. Comput. Biol. Chem. 32 (2008), 53–60.

Shi, X.H., Liang, Y.C., Lee, H.P., Lub, C., Wang, L.M., 2005. An improved GA and a novel PSO–GA-based hybrid algorithm. Inf. Process. Lett. 93, 255–261.

Sinha, A., Goldberg, D.E., 2004. A survey of hybrid genetic and evolutionary algorithms, ILLIGAL Technical Report 2003004.

Song, Sh., Kong, L., Gan, Y., Su, R., 2008. Hybrid particle swarm cooperative optimization algorithm and its application to MBC in alumina production. Prog. Nat. Sci. 18, 1423–1428.

Talbi, E.G., 2002. A taxonomy of hybrid meta-heuristics. J. Heurist. 8, 541–564.

Talbi, E.-G., Bachelet, T., 2006. COSEARCH: a parallel cooperative metaheuristic. J. Math. Model. Algorithms 5, 5–22.

Tenne, Y., Armfield, S.W., 2009. A framework for memetic optimization using variable global and local surrogate models. Soft. Comput. 13, 781–793.

Ullah, A.S.S.M.B., Sarker, R., Cornforth, D., Lokan, Ch., 2009. AMA: a new approach for solving constrained real-valued optimization problems. Soft Comput. 13 (2009), 741–762.

Wang, D., Liu, L., 2008. Hybrid particle swarm optimization for solving resources-constrained FMS. Prog. Nat. Sci. 18, 1179–1183.

Wang, Y.J., Zhang, J.S., 2007. Global optimization by an improved differential evolutionary algorithm. Appl. Math. Comput. 188, 669–680.

Wang, Y.J., Zhang, J.Sh., 2007. Global optimization by an improved differential evolutionary algorithm. Appl. Math. Comput. 188, 669–680.

Wang, Y.J., Zhang, J.Sh., Zhang, G.Y., 2007. A dynamic clustering based differential evolution algorithm for global optimization. Eur. J. Oper. Res. 183, 56–73.

Wang, H., Wang, D., Yang, Sh., 2009. A memetic algorithm with hill climbing strategy for dynamic optimization problems. Soft. Comput. 13, 763–780.

Wang, J., Osagie, E., Thulasiraman, P., Thulasiram, R.K., 2009. HOPNET: a hybrid ant colony optimization routing algorithm for mobile ad hoc network. Ad Hoc Netw. 7, 690–705.

Zeng, X.P., Li, Y.M., Jian, Q., 2009. A dynamic chain-like agent genetic algorithm for global numerical optimization and feature selection. Neurocomputing 72, 1214–1228.

Zhong, W.C., Liu, J., Xue, M.Z., Jiao, L.C., 2004. A multiagent genetic algorithm for global numerical optimization. IEEE Trans. Syst. Man Cybern. B 34 (2), 1128–1141.