ARTICLE IN PRESS

Ain Shams Engineering Journal (2016) **xxx**, xxx–xxx

Ain Shams University

Ain Shams Engineering Journal

www.elsevier.com/locate/asej
www.sciencedirect.com

ENGINEERING PHYSICS AND MATHEMATICS

A hybrid particle swarm optimization and genetic algorithm with population partitioning for large scale optimization problems

Ahmed F. Ali [a,b], **Mohamed A. Tawhid** [a,c,*]

[a] *Department of Mathematics and Statistics, Faculty of Science, Thompson Rivers University, Kamloops, Canada*
[b] *Department of Computer Science, Faculty of Computers & Informatics, Suez Canal University, Ismailia, Egypt*
[c] *Department of Mathematics and Computer Science, Faculty of Science, Alexandria University, Moharam Bey 21511, Alexandria, Egypt*

Received 20 February 2016; revised 16 July 2016; accepted 28 July 2016

KEYWORDS

Particle swarm optimization;
Genetic algorithm;
Molecular energy function;
Large scale optimization;
Global optimization

Abstract In this paper, a new hybrid particle swarm optimization and genetic algorithm is proposed to minimize a simplified model of the energy function of the molecule. The proposed algorithm is called Hybrid Particle Swarm Optimization and Genetic Algorithm (HPSOGA). The HPSOGA is based on three mechanisms. The first mechanism is applying the particle swarm optimization to balance between the exploration and the exploitation process in the proposed algorithm. The second mechanism is the dimensionality reduction process and the population partitioning process by dividing the population into sub-populations and applying the arithmetical crossover operator in each sub-population in order to increase the diversity of the search in the algorithm. The last mechanism is applied in order to avoid the premature convergence and avoid trapping in local minima by using the genetic mutation operator in the whole population. Before applying the proposed HPSOGA to minimize the potential energy function of the molecule size, we test it on 13 unconstrained large scale global optimization problems with size up to 1000 dimensions in order to investigate the general performance of the proposed algorithm for solving large scale global optimization problems then we test the proposed algorithm with different molecule sizes with up to 200 dimensions. The proposed algorithm is compared against the standard particle swarm optimization to solve large scale global optimization problems and 9 benchmark algorithms, in order to verify the efficiency of the proposed algorithm for solving molecules potential energy function. The numerical experiment results show that the proposed algorithm is a promising and

* Corresponding author at: Department of Mathematics and Statistics, Faculty of Science, Thompson Rivers University, Kamloops, BC V2C 0C8, Canada.
E-mail addresses: ahmed_fouad@ci.suez.edu.eg (A.F. Ali), Mtawhid@tru.ca (M.A. Tawhid).

Peer review under responsibility of Ain Shams University.

ELSEVIER **Production and hosting by Elsevier**

http://dx.doi.org/10.1016/j.asej.2016.07.008

efficient algorithm and can obtain the global minimum or near global minimum of the molecular energy function faster than the other comparative algorithms.

1. Introduction

The potential energy of a molecule is derived from molecular mechanics, which describes molecular interactions based on the principles of Newtonian physics. An empirically derived set of potential energy contributions is used for approximating these molecular interactions. The minimization of the potential energy function is a difficult problem to solve since the number of the local minima increases exponentially with the molecular size [1]. The minimization of the potential energy function problem can be formulated as a global optimization problem. Finding the steady state (ground) of the molecules in the protein can help to predict the 3D structure of the protein, which helps to know the function of the protein.

Several optimization algorithms have been suggested to solve this problem, for example, the random method [1–4], branch and bound method [5], simulated annealing [6], genetic algorithm [7–9] and variable neighborhood search [10,11]. A stochastic swarm intelligence algorithm, known as Particle Swarm Optimization (PSO) [12], and PSO and the Fletcher–Reeves algorithm [13], have been applied to solve the energy minimization problem. PSO is simple, easy to implement, and requires only a small number of user-defined parameters, but it also suffers from premature convergence.

In this paper, new hybrid particle swarm optimization algorithm and genetic algorithm is proposed in order to minimize the molecular potential energy function. The proposed algorithm is called Hybrid Particle Swarm Optimization and Genetic Algorithm (HPSOGA). The proposed HPSOGA algorithm is based on three mechanisms. In the first mechanism, the particle swarm optimization algorithm is applied with its powerful performance with the exploration and the exploitation processes. The second mechanism is based on the dimensionality reduction and the population partitioning processes by dividing the population into sub-population and applying the arithmetical crossover operator on each sub-population. The partitioning idea can improve the diversity search of the proposed algorithm. The last mechanism is to avoid the premature convergence by applying the genetic algorithm mutation operator in the whole population. The combination between these three mechanisms accelerates the search and helps the algorithm to reach to the optimal or near optimal solution in reasonable time.

In order to investigate the general performance of the proposed algorithm, it has been tested on a scalable simplified molecular potential energy function with well-known properties established in [5].

This paper is organized as follows: Section 2 presents the definitions of the molecular energy function and the unconstrained optimization problem. Section 3 overviews the standard particle swarm optimization and genetic algorithms. Section 4 describes in detail the proposed algorithm. Section 5 demonstrates the numerical experimental results. Section 6 summarizes the contribution of this paper along with some future research directions.

2. Description of the problems

2.1. Minimizing the molecular potential energy function

The minimization of the potential energy function problem considered here is taken from [7]. The molecular model considered here consists of a chain of m atoms centered at $x_1, \ldots, x_m$, in a 3-dimensional space. For every pair of consecutive atoms x_i and x_{i+1}, let $r_{i,i+1}$ be the bond length which is the Euclidean distance between them as seen in Fig. 1(a). For every three consecutive atoms $x_i,\ x_{i+1},\ x_{i+2}$, let $\theta_{i,i+2}$ be the bond angle corresponding to the relative position of the third atom with respect to the line containing the previous two as seen in Fig. 1(b). Likewise, for every four consecutive atoms $x_i,\ x_{i+1},\ x_{i+2}, x_{i+3}$, let $\omega_{i,i+3}$ be the torsion angle, between the normal through the planes determined by the atoms $x_i,\ x_{i+1},\ x_{i+2}$ and $x_{i+1},\ x_{i+2},\ x_{i+3}$ as seen in Fig. 1(c).

The force field potentials correspond to bond lengths, bond angles, and torsion angles are defined respectively [11] as

$$\begin{aligned}
E_1 &= \sum_{(i,j)\in M_1} c^1_{ij}\left(r_{ij} - r^0_{ij}\right)^2, \\
E_2 &= \sum_{(i,j)\in M_2} c^2_{ij}\left(\theta_{ij} - \theta^0_{ij}\right)^2, \qquad (1) \\
E_3 &= \sum_{(i,j)\in M_3} c^3_{ij}\left(1 + \cos\left(3\omega_{ij} - \omega^0_{ij}\right)\right),
\end{aligned}$$

where c^1_{ij} is the bond stretching force constant, c^2_{ij} is the angle bending force constant, and c^3_{ij} is the torsion force constant. The constants r^0_{ij} and θ^0_{ij} represent the preferred bond length and bond angle, respectively. The constant ω^0_{ij} is the phase angle that defines the position of the minima. The set of pairs of atoms separated by k covalent bond is denoted by M_k for $k = 1, 2, 3$.

Also, there is a potential E_4 which characterizes the 2-body interaction between every pair of atoms separated by more than two covalent bonds along the chain. We use the following function to represent E_4:

$$E_4 = \sum_{(i,j)\in M_3} \left(\frac{(-1)^i}{r_{ij}}\right), \qquad (2)$$

where r_{ij} is the Euclidean distance between atoms x_i and x_j.

The general problem is the minimization of the total molecular potential energy function, $E_1 + E_2 + E_3 + E_4$, leading to the optimal spatial positions of the atoms. To reduce the number of parameters involved in the potentials above, we simplify the problem by considering a chain of carbon atoms.

In most molecular conformational predictions, all covalent bond lengths and covalent bond angles are assumed to be fixed at their equilibrium values r^0_{ij} and θ^0_{ij}, respectively. Thus, the molecular potential energy function reduces to $E_3 + E_4$ and

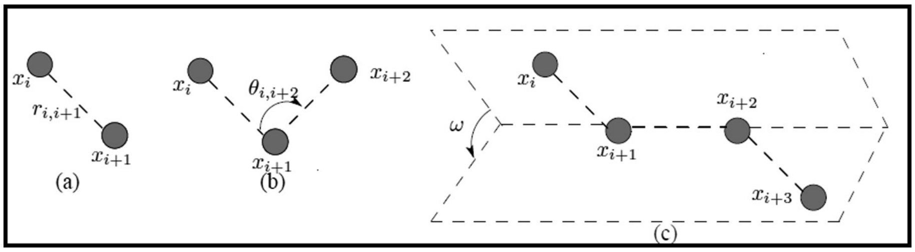

Figure 1 (a) Euclidean distance, (b) bond angle, (c) torsion (dihedral) angle.

the first three atoms in the chain can be fixed. The first atom, x_1, is fixed at the origin, $(0, 0, 0)$; the second atom, x_2, is positioned at $(-r_{12}, 0, 0)$; and the third atom, x_3, is fixed at $(r_{23}\cos(\theta_{13}) - r_{12}, r_{23}\sin(\theta_{13}), 0)$.

Using the parameters previously defined and Eqs. (1) and (2), we obtain

$$E = \sum_{(i,j)\in M_3} (1 + \cos(3\omega_{ij})) + \sum_{(i,j)\in M_3} \left(\frac{(-1)^i}{r_{ij}}\right). \tag{3}$$

Although the molecular potential energy function (3) does not actually model the real system, it allows one to understand the qualitative origin of the large number of local minimizers- the main computational difficulty of the problem, and is likely to be realistic in this respect.

Note that E_3 in Eq. (1) represents a function of torsion angles, and E_4 in Eq. (2) represents a function of Euclidean distance. To represent Eq. (3) as a function angles only, we can use the result established in [14] and obtain

$$r_{il}^2 = r_{ij}^2 + r_{jl}^2 - r_{ij}\left(\frac{r_{jl}^2 + r_{jk}^2 - r_{kl}^2}{r_{jk}}\right)\cos(\theta_{ik}) - r_{ij}\left(\frac{\sqrt{4r_{jl}^2 r_{jk}^2 - \left(r_{jl}^2 + r_{jk}^2 - r_{kl}^2\right)^2}}{r_{jk}}\right)\sin(\theta_{ik})\cos(\omega_{il}),$$

for every four consecutive atoms x_i, x_j, x_k, x_l. Using the parameters previously defined, we have

$$r_{ij} = \sqrt{10.60099896 - 4.141720682(\cos(\omega_{ij}))} \quad \text{for all } (i,j) \in M_3. \tag{4}$$

From Eqs. (3) and (4), the expression for the potential energy as a function of the torsion angles takes the form

$$E = \sum_{(i,j)\in M_3} \left(1 + \cos(3\omega_{ij}) + \frac{(-1)^i}{\sqrt{10.60099896 - 4.141720682(\cos(\omega_{ij}))}}\right), \tag{5}$$

where $i = 1, \ldots, m-3$ and m is the number of atoms in the given system. as shown in Fig. 1(c).

The problem is then to find $\omega_{14}, \omega_{25}, \ldots, \omega_{(m-3)m}$ where $\omega_{ij} \in [0, 5]$, which corresponds to the global minimum of the function E, represented by Eq. (5). E is a nonconvex function involving numerous local minimizers even for small molecules.

Finally, the function $f(x)$ can defined as

$$f(x) = \sum_{i=1}^{n} \left(1 + \cos(3x_i) + \frac{(-1)^i}{\sqrt{10.60099896 - 4.141720682(\cos(x_i))}}\right) \tag{6}$$

and $0 \leqslant x_i \leqslant 5$, $i = 1, \ldots, n$.

Despite this simplification, the problem remains very difficult. A molecule with as few as 30 atoms has $2^{27} = 134,217,728$ local minimizers.

2.2. Unconstrained optimization problems

Mathematically, the optimization is the minimization or maximization of a function of one or more variables by using the following notations:

- $x = (x_1, x_2, \ldots, x_n)$ - a vector of *variables* or *function parameters*;
- f - the *objective function* that is to be minimized or maximized; a function of x;
- $l = (l_1, l_2, \ldots, l_n)$ and $u = (u_1, u_2, \ldots, u_n)$ - the *lower and upper bounds* of the definition domain for x.

The optimization problem (minimization) can be defined as:

$$\min_{l \leqslant x \leqslant u} f(x) \tag{7}$$

3. The basic PSO and GA algorithms

3.1. Particle swarm optimization algorithm

We will give an overview of the main concepts and structure of the particle swarm optimization algorithm as follows.

Main concepts. Particle swarm optimization (PSO) is a population based method that inspired from the behavior (information exchange) of the birds in a swarm [15]. In PSO the population is called a swarm and the individuals are called particles. In the search space, each particle moves with a velocity. The particle adapts this velocity due to the information exchange between it and other neighbors. At each iteration, the particle uses a memory in order to save its best position and the overall best particle positions. The best particle position is saved as a best local position, which was assigned to a neighborhood particles, while the overall best particle position is saved as a best global position, which was assigned to all particles in the swarm.

Particle movement and velocity. Each particle is represented by a D dimensional vectors,

$$x_i = (x_{i1}, x_{i2}, \dots, x_{iD}) \in S. \tag{8}$$

The velocity of the initial population is randomly generated and each particle has the following initial velocity:

$$v_i = (v_{i1}, v_{i2}, \dots, v_{iD}). \tag{9}$$

The best local and global positions are assigned, where the best local position encounter by each particle is defined as

$$p_i = (p_{i1}, p_{i2}, \dots, p_{iD}) \in S. \tag{10}$$

At each iteration, the particle adjusts its personal position according to the best local position (Pbest) and the overall (global) best position (gbest) among particles in its neighborhood as follows:

$$x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)}, \quad i = 1, \dots, P \tag{11}$$

$$v_i^{(t+1)} = v_i^{(t)} + c_1 r_{i1} \times \left(pbest_i^{(t)} - x_i^{(t)}\right) + c_2 r_{i2} \times \left(gbest - x_i^{(t)}\right). \tag{12}$$

where c_1, c_2 are two acceleration constants called cognitive and social parameters, r_1, r_2 are random vector $\in [0, 1]$.

We can summarize the main steps of the PSO algorithm as follows.

- **Step 1.** The algorithm starts with the initial values of swarm size P, acceleration constants c_1, c_2.
- **Step 2.** The initial position and velocity of each solution (particle) in the population (swarm) are randomly generated as shown in Eqs. (8) and (9).
- **Step 3.** Each solution in the population is evaluated by calculating its corresponding fitness value $f(x_i)$.
- **Step 4.** The best personal solution *Pbest* and the best global solution *gbest* are assigned.
- **Step 5.** The following steps are repeated until the termination criterion is satisfied.
  - **Step 5.1.** At each iteration t, the position of each particle x_i^t is justified as shown in Eq. (11), while the velocity of each particle v_i^t is justified as shown in Eq. (12).
  - **Step 5.2.** Each solution in the population is evaluated $f(x_i)$ and the new best personal solution *Pbest* and best global solution *gbest* are assigned.
  - **Step 5.3.** The operation is repeated until the termination criteria are satisfied.
- **Step 6.** Produce the best found solution so far.

Algorithm 1. Particle swarm optimization algorithm.

```
1: Set the initial value of the swarm size $P$, acceleration constants $c_1$, $c_2$.
2: Set $t := 0$.
3: Generate $x_i^{(t)}, v_i^{(t)} \in [L, U]$ randomly, $i = 1, \dots, P$. {$P$ is the population (swarm) size}.
4: Evaluate the fitness function $f(x_i^{(t)})$.
5: Set $gbest^{(t)}$. {$gbest$ is the best global solution in the swarm}.
6: Set $pbest_i^{(t)}$. {$pbest_i^{(t)}$ is the best local solution in the swarm}.
7: repeat
8:   $v_i^{(t+1)} = v_i^{(t)} + c_1 r_{i1} \times \left(pbest_i^{(t)} - x_i^{(t)}\right) + c_2 r_{i2} \times \left(gbest - x_i^{(t)}\right)$. {$r_1, r_2$ are random vectors $\in [0,1]$}.
9:   $x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)}$, $i = 1, \dots, P$. {Update particles positions}.
10:  Evaluate the fitness function $f\left(x_i^{(t+1)}\right)$, $i = 1, \dots, P$.
11:  if $f\left(x_i^{(t+1)}\right) \leqslant f\left(pbest_i^{(t)}\right)$ then
12:    $pbest_i^{(t+1)} = x_i^{(t+1)}$.
13:  else
14:    $pbest_i^{(t+1)} = pbest_i^{(t)}$.
15:  end if
16:  if $x_i^{(t+1)} \leqslant f(gbest^{(t)})$ then
17:    $gbest^{(t+1)} = x_i^{(t+1)}$.
18:  else
19:    $gbest^{(t+1)} = gbest^{(t)}$.
20:  end if
21:  Set $t = t + 1$. {Iteration counter increasing}.
22: until Termination criteria are satisfied.
23: Produce the best particle.
```

3.2. Genetic algorithm

Genetic algorithms (GAs) have been developed by J. Holland to understand the adaptive processes of natural systems [16]. Then, they have been applied to optimization and machine learning in the 1980s [17,18]. GA usually applies a crossover operator by mating the parents (individuals) and a mutation operator that randomly modifies the individual contents to promote diversity to generate a new offspring. GAs use a probabilistic selection that is originally the proportional selection. The replacement (survival selection) is generational, that is, the parents are replaced systematically by the offsprings. The crossover operator is based on the n-point or uniform crossover while the mutation is a bit flipping. The general structure of GA is shown in Algorithm 2.

Algorithm 2. The structure of genetic algorithm.

```
1: Set the generation counter $t := 0$.
2: Generate an initial population $P^0$ randomly.
3: Evaluate the fitness function of all individuals in $P^0$.
4: repeat
5:   Set $t = t + 1$. {Generation counter increasing}.
6:   Select an intermediate population $P^t$ from $P^{t-1}$. {Selection operator}.
7:   Associate a random number $r$ from $(0, 1)$ with each row in $P^t$.
8:   if $r < p_c$ then
9:     Apply crossover operator to all selected pairs of $P^t$. {Crossover operator}.
10:    Update $P^t$.
11:  end if
12:  Associate a random number $r_1$ from $(0, 1)$ with each gene in each individual in $P^t$.
13:  if $r_1 < p_m$ then
14:    Mutate the gene by generating a new random value for
```

```
the selected gene with its domain. {Mutation operator}.
15:       Update $P^t$.
16:    end if
17:    Evaluate the fitness function of all individuals in $P^t$.
18: until Termination criteria are satisfied.
```

Procedure 1 (*Crossover* (p^1, p^2)).

1. Randomly choose $\lambda \in (0,1)$.
2. Two offspring $c^1 = (c^1_1, \ldots, c^1_D)$ and $c^2 = (c^2_1, \ldots, c^2_D)$ are generated from parents $p^1 = (p^1_1, \ldots, p^1_D)$ and $p^2 = (p^2_1, \ldots, p^2_D)$, where

$$c^1_i = \lambda p^1_i + (1-\lambda)p^2_i,$$
$$c^2_i = \lambda p^2_i + (1-\lambda)p^1_i,$$

$i = 1, \ldots, D$.

3. Return.

4. The proposed HPSOGA algorithm

The main structure of the proposed HPSOGA algorithm is presented in Algorithm 3.

Algorithm 3. Hybrid particle swarm optimization and genetic algorithm.

```
1: Set the initial values of the population size $P$, acceleration
   constant $c_1$ and $c_2$, crossover probability $P_c$, mutation probability
   $P_m$, partition number $part_{no}$, number of variables in each
   partition $v$, number of solutions in each partition $\eta$ and the
   maximum number of iterations $Max_{itr}$.
2: Set $t := 0$. {Counter initialization}.
3: for $(i = 1 : i \leqslant P)$ do
4:    Generate an initial population $X_i(\vec{t})$ randomly.
5:    Evaluate the fitness function of each search agent (solution)
      $f(\vec{X}_i)$.
6: end for
7: repeat
8:    Apply the standard particle swarm optimization (PSO)
      algorithm as shown in Algorithm 1 on the whole population $X(\vec{t})$.
9:    Apply the selection operator of the GA on the whole
      population $X(\vec{t})$.
10:   Partition the population $X(\vec{t})$ into $part_{no}$ sub-partitions,
      where each sub-partition $X'(\vec{t})$ size is $v \times \eta$.
11:   for $(i = 1 : i \leqslant part_{no})$ do
12:      Apply the arithmetical crossover as shown in Procedure 1
         on each sub-partition $X'(\vec{t})$.
13:   end for
14:   Apply the GA mutation operator on the whole population
      $X(\vec{t})$.
15:   Update the solutions in the population $X(\vec{t})$.
16:   Set $t = t + 1$. {Iteration counter is increasing}.
17: until $(t > Max_{itr})$. {Termination criteria are satisfied}.
18: Produce the best solution.
```

The main steps of the proposed algorithm are summarized as follows.

- **Step 1.** The proposed HPSOGA algorithm starts by setting its parameter values such as the population size P, acceleration constant c_1 and c_2, crossover probability P_c, mutation probability P_m, partition number $part_{no}$, the number of variables in partition v, the number of solutions in partition η and the maximum number of iterations Max_{itr}. **(Line 1)**
- **Step 2.** The iteration counter t is initialized and the initial population is randomly generated and each solution in the population is evaluated. **(Lines 2–6)**
- **Step 3.** The following steps are repeated until termination criteria are satisfied.
  - **Step 3.1.** The new solutions $\vec{X}^t$ are generated by applying the standard particle swarm optimization algorithm (PSO) on the whole population. **(Line 8)**
  - **Step 3.2.** Select an intermediate population from the current one by applying GA selection operator. **(Line 9)**
  - **Step 3.3.** In order to increase the diversity of the search and overcome the dimensionality problem, the current population is partitioned into $part_{no}$ sub-population, where each sub-population $X'(\vec{t})$ size is $v \times \eta$, where v is the number of variables in each partition and η is the number of solutions in each partition. **(Line 10)** Fig. 2 describes the applied population partitioning strategy.
  - **Step 3.4.** The arithmetical crossover operator is applied on each sub-population. **(Lines 11–13)**
  - **Step 3.5.** The genetic mutation operator is applied in the whole population in order to avoid the premature convergence. **(Line 14)**
- **Step 7.** The solutions in the population are evaluated by calculating its fitness function. The iteration counter t is increasing and the overall processes are repeated until termination criteria are satisfied. **(Lines 15–17)**
- **Step 8.** Finally, the best found solution is presented. **(Line 18)**

5. Numerical experiments

Before investigating the proposed algorithm on the molecular energy function, 13 benchmark unconstrained optimization problems with size up to 1000 dimensions are tested. The results of the proposed algorithm are compared against the standard particle swarm optimization for the unconstrained optimization problems and the 9 benchmark algorithms for the molecular potential energy function. HPSOGA is programmed by MATLAB, and the results of the comparative algorithms are taken from their original papers. In the following subsections, the parameter setting of the proposed algorithm with more details has been reported in Table 1.

5.1. Parameter setting

The parameters of the HPSOGA algorithm are reported with their assigned values in Table 1. These values are based on the common setting in the literature or determined through our preliminary numerical experiments.

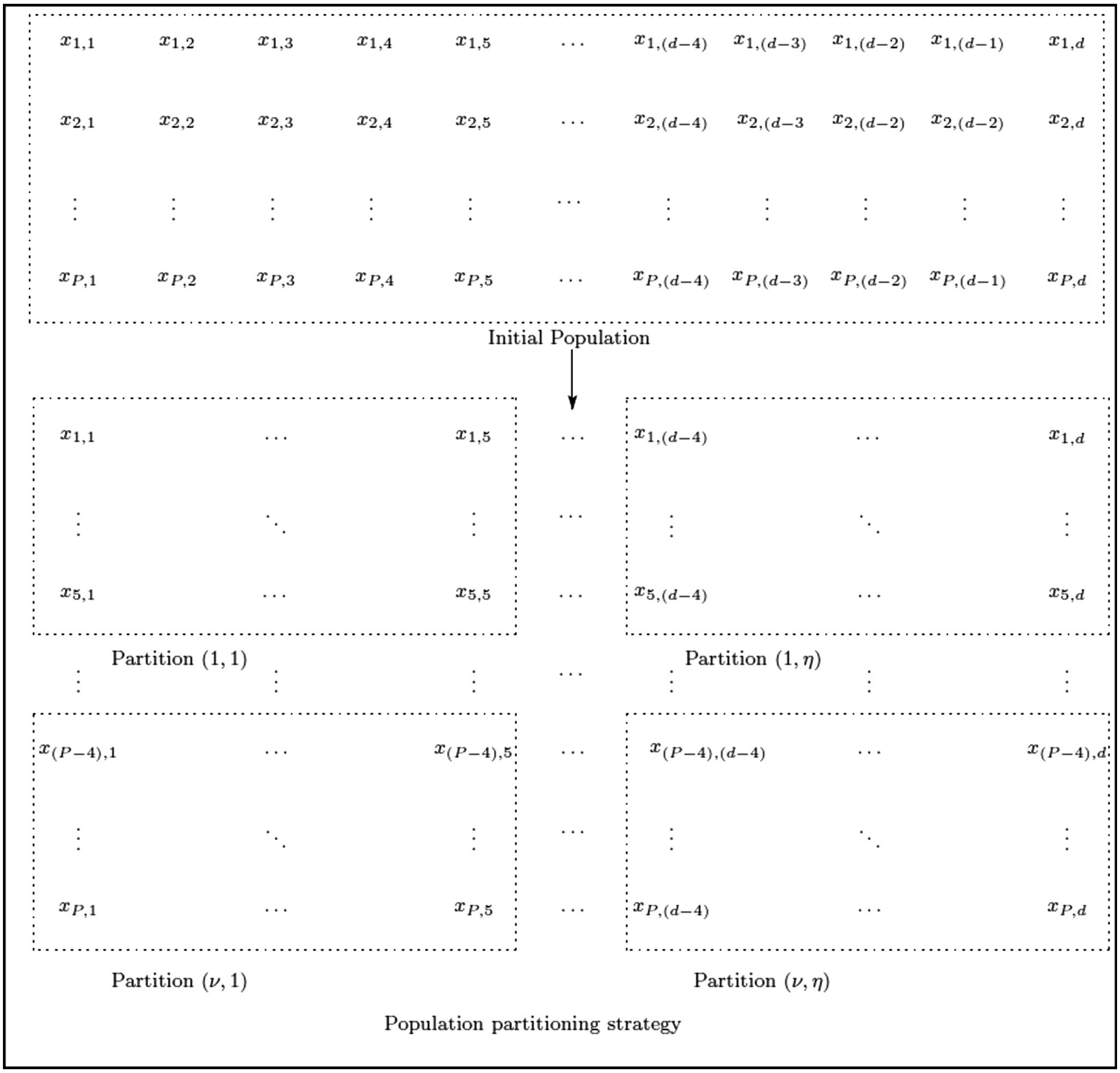

Figure 2 Population partitioning strategy.

Table 1 Parameter setting.

Parameters	Definitions	Values
P	Population size	25
c_1	Acceleration constant for cognition part	2
c_2	Acceleration constant for social part	2
P_c	Crossover rate	0.6
P_m	Mutation rate	0.01
ν	No of variables in each partition	5
η	No of solutions in each partition	5

Table 2 Unimodal test functions.

Test function	S	f_{opt}
$f_1(X) = \sum_{i=1}^{d} x_i^2$	$[-100, 100]^d$	0
$f_2(X) = \sum_{i=1}^{d} \lvert x_i \rvert + \prod_{i=1}^{d} \lvert x_i \rvert$	$[-10, 10]^d$	0
$f_3(X) = \sum_{i=1}^{d} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]^d$	0
$f_4(X) = \max_i \lvert x_i \rvert, \; 1 \leqslant i \leqslant d$	$[-100, 100]^d$	0
$f_5(X) = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30, 30]^d$	0
$f_6(X) = \sum_{i=1}^{d} ([x_i + 0.5])^2$	$[-100, 100]^d$	0
$f_7(X) = \sum_{i=1}^{d} i x_i^4 + \text{random}[0, 1)$	$[-1.28, 1.28]^d$	0

Table 3 Multimodal test functions.

Test function	S	f_{opt}
$f_8(X) = \sum_{i=1}^{d} -x_i \sin(\sqrt{\lvert x_i \rvert})$	$[-500, 500]^d$	$-418.9829 \times d$
$f_9(X) = \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i) + 10]$	$[-5.12, 5.12]^d$	0
$f_{10}(X) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + e$	$[-32, 32]^d$	0
$f_{11}(X) = \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{(i)}}\right) + 1$	$[-600, 600]^d$	0
$f_{12}(X) = \frac{\pi}{d} 10\sin^2(\pi y_1) + \sum_{i=1}^{m-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_d - 1)^2 + \sum_{i=1}^{m} u(x_i, 10, 100, 4) + y_i = 1 + \frac{x_i+1}{4}, \quad u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$[-50, 50]^d$	0
$f_{13}(X) = \{0.1\sin^2(3\pi x_1) + \sum_{i=1}^{d}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_d - 1)^2[1 + \sin^2(2\pi x_d)]\} + \sum_{i=1}^{d} u(x_i, 5, 100, 4)$	$[-50, 50]^d$	0

- **Population size** P. The experimental tests show that the best population size is $P = 25$, and increasing this number will increase the evaluation function values without any improvement in the obtained results.
- **Acceleration constant** c_1 **and** c_2. The parameters c_1 and c_2 are acceleration constants, and they are a weighting stochastic acceleration, which pull each particle toward personal best and global best positions. The values of c_1 and c_2 are set to 2.
- **Probability of crossover** P_c. Arithmetical crossover operator is applied for each partition in the population and It turns out that the best value of the probability of crossover is to set to 0.6.
- **Probability of mutation** P_m. In order to and avoid the premature convergence, a mutation is applied on the whole population with value 0.01.
- **Partitioning variables** v, η. It turns out that the best subpopulation size is to be $v \times \eta$, where v and η equal to 5.

5.2. Unconstrained test problems

Before testing the general performance of the proposed algorithm with different molecules sizes, 13 benchmark functions are tested and the results are reported in Table 2. In Table 2, there are 7 unimodel functions and 6 multimodel functions (see Table 3).

5.3. The efficiency of the proposed HPSOGA on large scale global optimization problems

In order to verify the efficiency of the partitioning process and the combining between the standard particle swarm optimization and genetic algorithm, the general performance of the proposed HPSOGA algorithm and the standard particle swarm optimization algorithm (PSO) are presented for functions f_3, f_4, f_9 and f_{10} by plotting the function values versus the number of iterations as shown in Figs. 3 and 4. In Figs. 3 and 4, the dotted line represents the standard particle swarm optimization, while the solid line represents the proposed HPSOGA algorithm. The data in Figs. 3 and 4 are plotted after d iterations, where d is the problem dimension. Figs. 3 and 4 show that the proposed algorithm is faster than the standard particle swarm optimization algorithm which verifies that the applied partitioning mechanism and the combination between the particle swarm optimization and the genetic algorithm can accelerate the convergence of the proposed algorithm.

5.4. The general performance of the proposed HPSOGA on large scale global optimization problems

The general performance of the proposed algorithm is presented in Figs. 5 and 6 by plotting the function values versus the iterations number for functions f_1, f_2, f_5 and f_6 with dimensions 30, 100, 400 and 1000. These functions are selected randomly.

5.5. The comparison between PSO and HPSOGA

The last investigation of the proposed algorithm HPSOGA is applied by testing on 13 benchmark functions with dimensions up to 1000 and comparing it against the standard particle

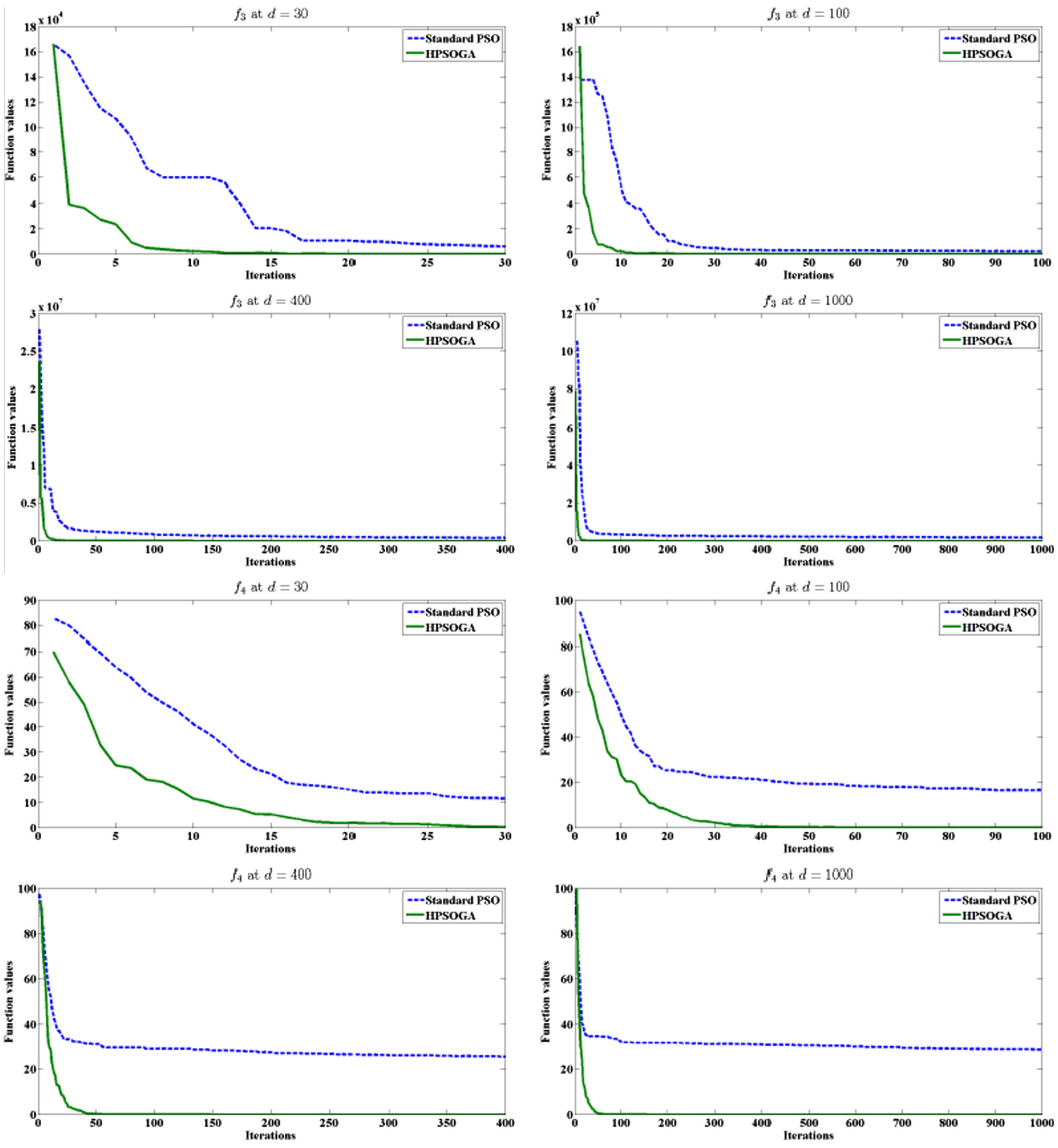

Figure 3 The efficiency of HPSOGA on large scale global optimization problems.

swarm optimization. The results of both algorithms (mean (Ave) and standard deviation (Std) of the evaluation function values) are reported over 30 runs and applied the same termination criterion, i.e., terminates the search when they reach to the optimal solution within an error of 10^{-4} before the 25,000, 50,000, 125,000 and 300,000 function evaluation values for dimensions 30, 100, 400 and 1000, respectively. The function evaluation is called cost function, which describes the maximum number of iterations and the execution time for each applied algorithm. The results in parentheses are the mean and the standard deviations of the function values and reported when the algorithm reaches the desired number of function evaluations without obtaining the desired optimal solutions. The reported results in Tables 4–7 show that the performance of the proposed HPSOGA is better than the standard particle swarm optimization algorithm and can obtain the optimal or near optimal solution in reasonable time.

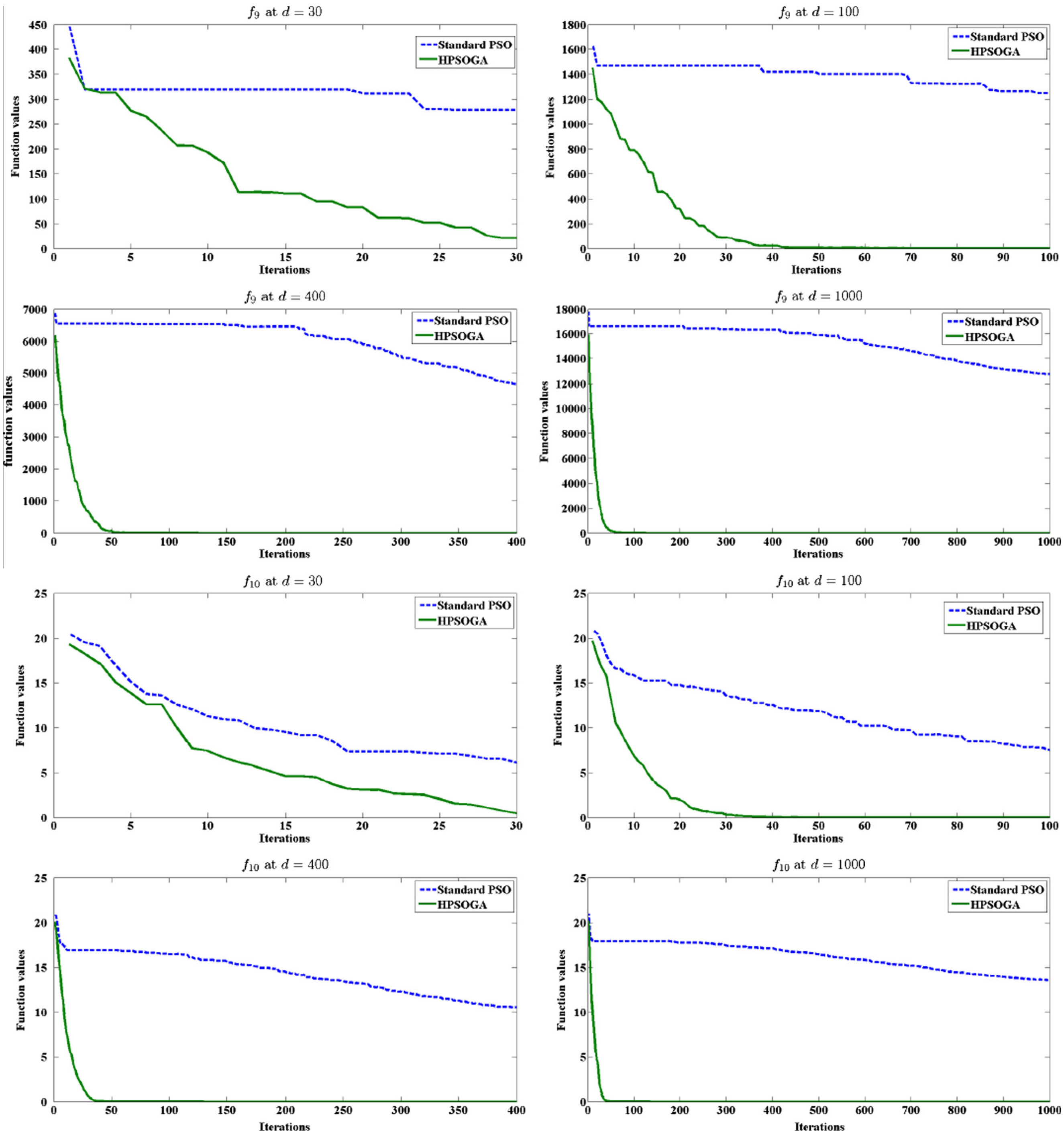

Figure 4 The efficiency of HPSOGA on large scale global optimization problems (cont.).

5.6. The efficiency of the proposed HPSOGA for minimizing the potential energy function

The general performance of the proposed algorithm is tested on a simplified model of the molecule with various dimensions from 20 to 200 by plotting the number of function values (mean error) versus the number of iterations (function evaluations) as shown in Fig. 7. The results in Fig. 7 show that the function values rapidly decrease while the number of iterations slightly increases. It can be concluded from Fig. 7 that the proposed HPSOGA can obtain the optimal or near optimal solutions within reasonable time.

5.7. HPSOGA and other algorithms

The HPSOGA algorithm is compared against two sets of benchmark methods. The first set of methods consists of four various real coded genetic algorithms (RCGAs),

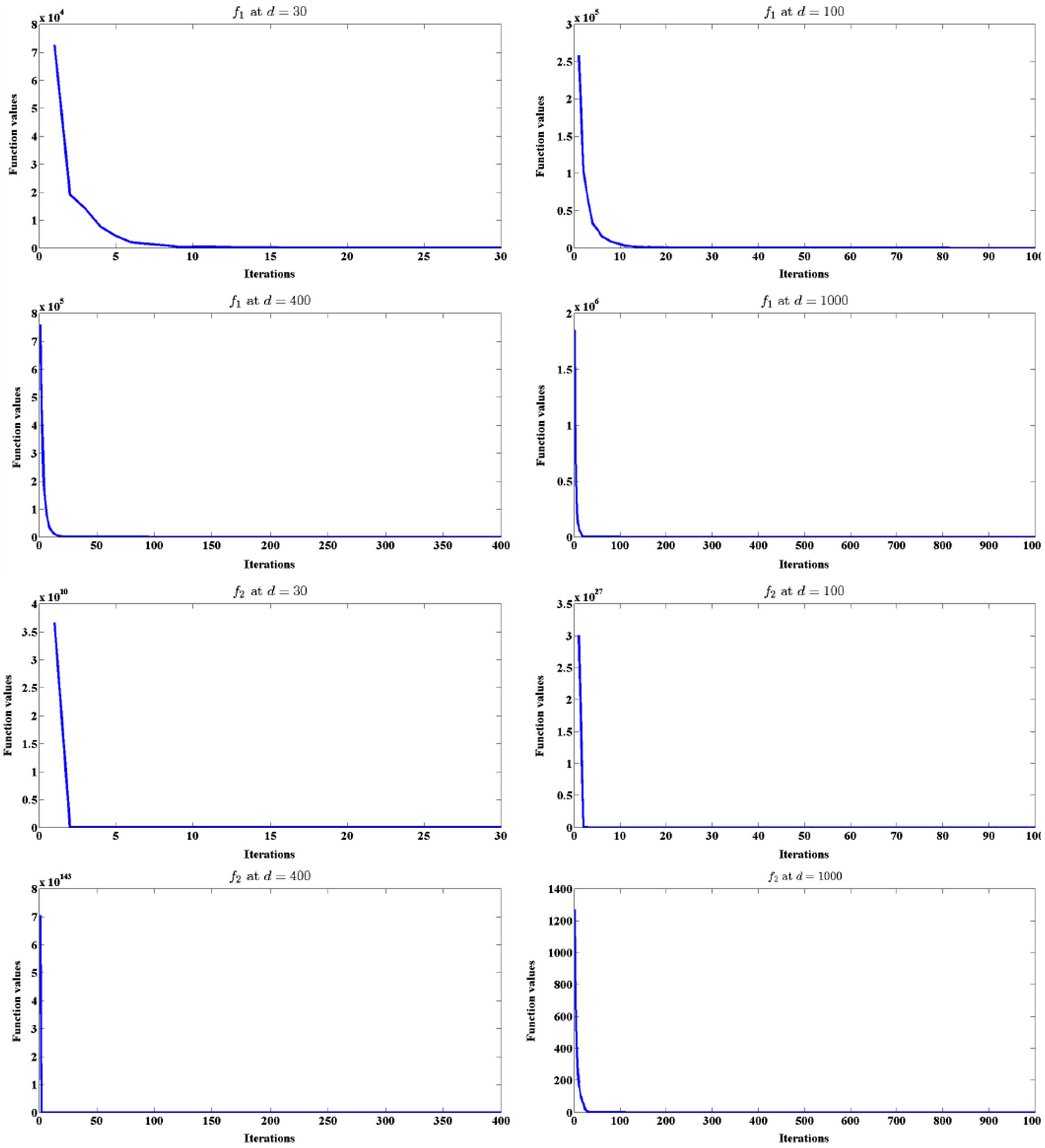

Figure 5 The general performance of HPSOGA on large scale global optimization problems.

WX-PM, WX-LLM, LX-LLM [8] and LX-PM [19]. These four methods are based on two real coded crossover operators, Weibull crossover WX and LX [20] and two mutation operators LLM and PM [19]. The second set of methods consists of 5 benchmark methods, variable neighborhood search based method (VNS), (VNS-123), (VNS-3) methods [11]. In [11], four variable neighborhood search methods, VNS-1, VNS-2, VNS-3, and VNS-123 were developed. They differ in the choice of random distribution used in the shaking step for minimization of a continuous function subject to box constraints. Here is the description of these four methods.

- **VNS-1**. In the first method, a random direction is uniformly distributed in a unit ℓ_∞ sphere. Random radius is chosen in such a way that the generated point is uniformly distributed in N_k, where N_k are the neighborhood structures, and $k = 1, \ldots, k_{max}$.

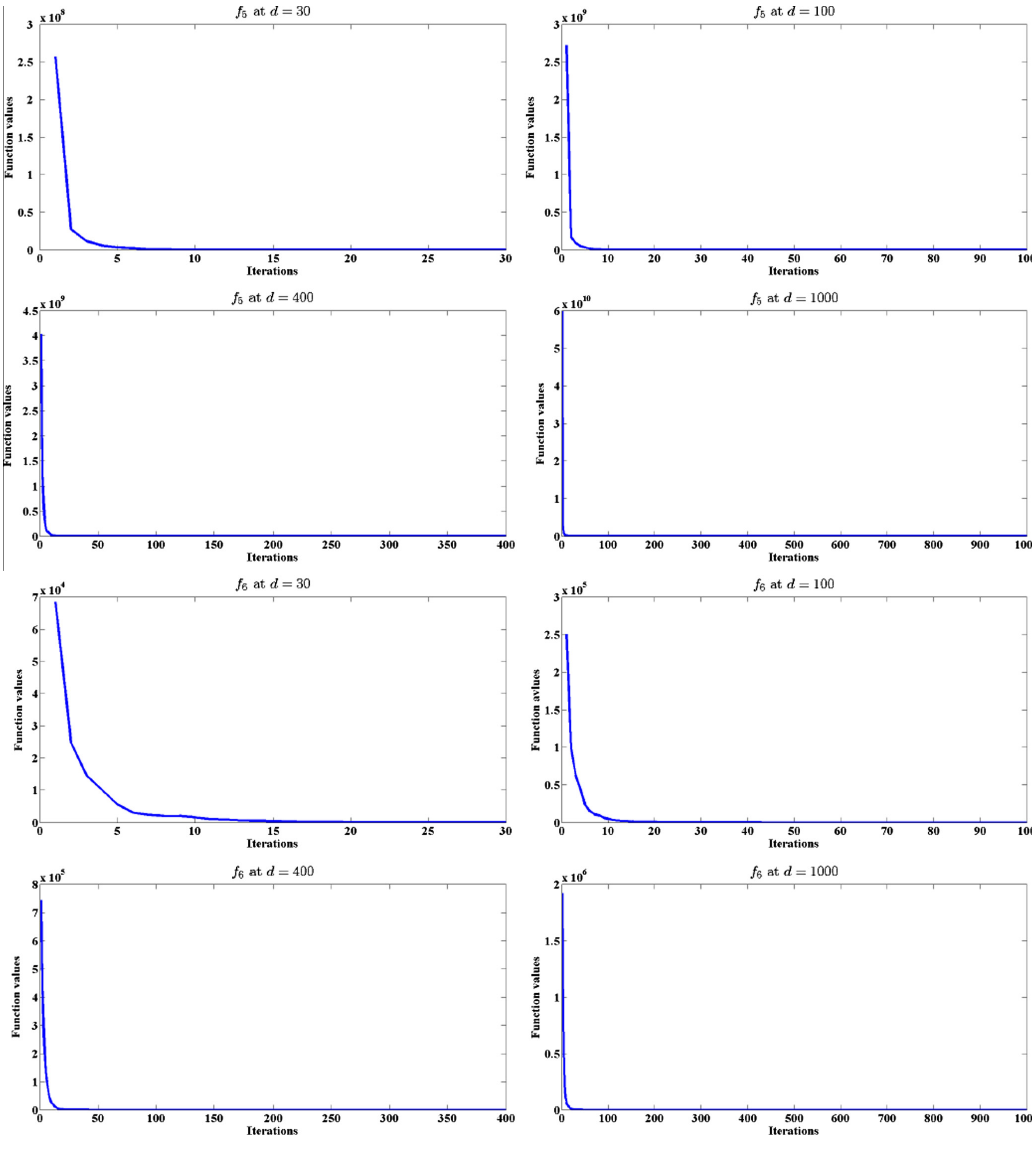

Figure 6 The general performance of HPSOGA on large scale global optimization problems (cont.).

- **VNS-2**. In the second method, a random direction is determined by random points uniformly distributed on a ℓ_1 sphere.
- **VNS-3**. In the third method, a random direction $x = (x_1, x_2, \ldots, x_n)$ is determined by a specially designed hypergeometric random point distribution on a unit ℓ_1 sphere as follows:
  1. x_1 is taken uniformly on $[-1, 1]$, x_k is taken uniformly from $[-A_k, A_k]$, where $A_k = 1 - |x_1| - \cdots - |x_{k-1}|$, $k = 2, \ldots, n-1$, and the last x_n takes A_n with random sign.
  2. coordinates of x are randomly permuted.
- **VNS-123**. In the fourth method, the combination of the three previously described methods is made to diversify the search.

(rHYB) method [7] denotes the staged hybrid Genetic algorithm (GA) with a reduced simplex and a fixed limit for simplex iterations and (qPSO) method [12] is a hybrid

Table 4 Comparison results (mean number (Ave) and standard deviation (Std) of function values) between PSO and HPSOGA at $d = 30$, FES = 25,000.

		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}
PSO	Ave	7215.37	8175.47	9165.19	10285.4	(29.45)	(0.0029)	16436.12	(−3958.36)	(3.534)	9336.16	5115.42	(0.01)	(2.14)
	Std	115.65	1115.24	1238.27	1205.48	(51.45)	(4.53)	1584.97	(1568.76)	(1.68)	246.18	123.15	(0.02)	(1.15)
HPSOGA	Ave	1119.15	1615.25	1585.15	2275.15	(45.14)	845.73	13135.75	8750.36	6690.74	7623.19	2462.18	8458.13	8148.19
	Std	15.22	24.57	65.84	75.86	(1.36)	115.49	512.78	512.34	1323.35	750.48	648.78	118.79	259.49

Table 5 Comparison results (mean number (Ave) and standard deviation (Std) of function values) between PSO and HPSOGA at $d = 100$, FES = 50,000.

		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}
PSO	Ave	10215.18	11435.29	49283.27	21320.13	(81.24)	(7.231)	(0.0012)	(−19128.69)	17335.15	11187.84	10589.14	(0.112)	(11.49)
	Std	1436.63	2212.81	6423.52	7142.18	(12.51)	(1.26)	(0.12)	(2135.14)	1343.15	2115.32	1514.25	(0.03)	(1.15)
HPSOGA	Ave	2115.35	2935.27	2985.46	2834.12	(78.16)	1887.19	17725.48	11215.19	7231.71	8915.23	4648.14	10645.24	10945.14
	Std	1231.42	432.18	462.49	745.81	(2.87)	248.73	2735.49	2134.26	1935.45	1589.25	1187.49	1848.48	1739.49

Table 6 Comparison results (mean number (Ave) and standard deviation (Std) of function values) between PSO and HPSOGA at $d = 400$, FES = 125,000.

		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}
PSO	Ave	18143.16	19224.36	(7313.2)	(398.31)	(1942.12)	(1234.22)	(1124)	(−34125.15)	29257.46	23167.34	22567.23	(12.47)	(456.254)
	Std	2512.15	3442.14	(1257.13)	(11.875)	(425.13)	(213.46)	(1.12)	(1225.58)	5334.36	5487.75	4238.22	(4.17)	(15.39)
HPSOGA	Ave	3131.15	5434.12	4178.19	3247.12	(333.43)	3734.19	21231.48	14335.23	10256.57	11.584.26	7564.36	13225.23	15227.56
	Std	125.12	864.14	815.48	258.48	(45.16)	941.15	1286.18	4224.16	2135.67	1347.32	1921.27	1976.16	2114.14

particle swarm optimization (PSO) in which quadratic approximation operator is hybridized with PSO.

The function E in Eq. (5) is minimized in the specified search space $[0, 5]^d$. The function E grows linearly with d as $E^*(d) = -0.0411183d$ [5] as shown in Table 8.

5.7.1. Comparison results between WX-PM, LX-PM, WX-LLM, LX-LLM and HPSOGA

In this subsection, the comparison results between our HPSOGA algorithm and other 4 variant genetic algorithms are presented. The five comparative algorithms are tested on

Table 7 Comparison results (mean number (Ave) and standard deviation (Std) of function values) between PSO and HPSOGA at $d = 1000$, FES = 300,000.

		f_1	f_2	f_3	f_4	f_5	f_6	f_7
PSO	Ave	42125.23	54113.22	(11371.183)	(125.04)	(939.14)	(919.23)	298215.85
	Std	1795.64	2954.75	(2373.15)	(17.42)	(361.05)	(513.39)	487.25
HPSOGA	Ave	6251.21	8434.18	9584.39	9845.12	(883.63)	8734.21	4611.19
	Std	925.39	1464.33	1215.18	1123.18	(158.96)	1128.85	5864.89

		f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}
PSO	Ave	(−42343.6)	56132.12	5312.17	47512.32	(0.096)	(81.94)
	Std	(1849.23)	4912.12	4158.32	512.22	(0.01)	(23.12)
HPSOGA	Ave	31115.15	21423.13	23334.19	21332.18	27112.23	32341.19
	Std	2425.17	3245.32	6712.21	3214.19	176512	4563.12

Table 8 The global minimum value E^* for chains of various sizes.

d	E^*
20	−0.822366
40	−1.644732
60	−2.467098
80	−3.289464
100	−4.111830
120	−4.934196
140	−5.756562
160	−6.578928
180	−7.401294
200	−8.22366

Table 9 Comparison results (mean number of function evaluations) between WX-PM, LX-PM, WX-LLM, LX-LLM and HPSOGA.

d	WX-PM	LX-PM	WX-LLM	LX-LLM	HPSOGA
20	15,574	23,257	28,969	14,586	**10,115**
40	59,999	71,336	89,478	39,366	**21,218**
60	175,865	280,131	225,008	105,892	**30,256**
80	302,011	326,287	372,836	237,621	**40,312**
100	369,376	379,998	443,786	320,146	**52,375**

Table 10 Comparison results (mean number of function evaluations) between VNS-123, VNS-3, GA, qPSO, rHYB and HPSOGA.

d	VNS-123	VNS-3	GA	qPSO	rHYB	HPSOGA
20	23,381	**9887**	36,626	–	35,836	10,115
40	57,681	25,723	133,581	–	129,611	**21,218**
60	142,882	39,315	263,266	–	249,963	**30,256**
80	180,999	74,328	413,948	–	387,787	**40,312**
100	254,899	79,263	588,827	–	554,026	**52,375**
120	375,970	99,778	–	–	–	**63,225**
140	460,519	117,391	–	–	–	**71,325**
160	652,916	167,972	–	–	–	**84,415**
180	663,722	173,513	–	–	–	**91,115**
200	792,537	213,718	–	–	–	**105,525**

Table 11 Wilcoxon test for comparison results in Table 10.

Compared methods		Solution evaluations			
Method 1	Method 2	R^-	R^+	ρ-value	Best method
HPSOGA	VNS-123	55	0	0.005062	HPSOGA
HPSOGA	VNS-3	54	1	0.006910	HPSOGA

different molecule sizes with dimension from 20 to 200. The results of the other comparative algorithms are taken from their original paper [8]. The mean number of the evaluation function values is reported over 30 runs in Table 4. The best results between the comparative algorithms are reported in **boldface** text. The results in Table 9 show that the proposed HPSOGA algorithm is successful to obtain the desired objective value of each function faster than the other algorithms in all cases.

5.7.2. Comparison results between VNS-123, VNS-3, GA, qPSO, rHYB and HPSOGA

Here is another comparison results between our HPSOGA algorithm and other 5 benchmark methods. The results are reported in Table 10. The results of the other comparative algorithms are taken from their original papers [7,11]. The mean number of the evaluation function values is reported over 30 runs as shown in Table 10. The best results between the comparative algorithms are reported in **boldface** text. The results in Table 10 show that the proposed HPSOGA algorithm succeeds and obtains the desired objective value of each molecular size faster than the other algorithms in most cases except when $d = 20$, the VNS-3 algorithm obtains the desired function value faster than the proposed algorithm.

5.8. Wilcoxon signed-ranks test

Wilcoxon's test is a nonparametric procedure employed in a hypothesis testing situation involving a design with two samples [21–23]. It is a pairwise test that aims to detect significant differences between the behavior of two algorithms. ρ is the probability of the null hypothesis being true. The result of

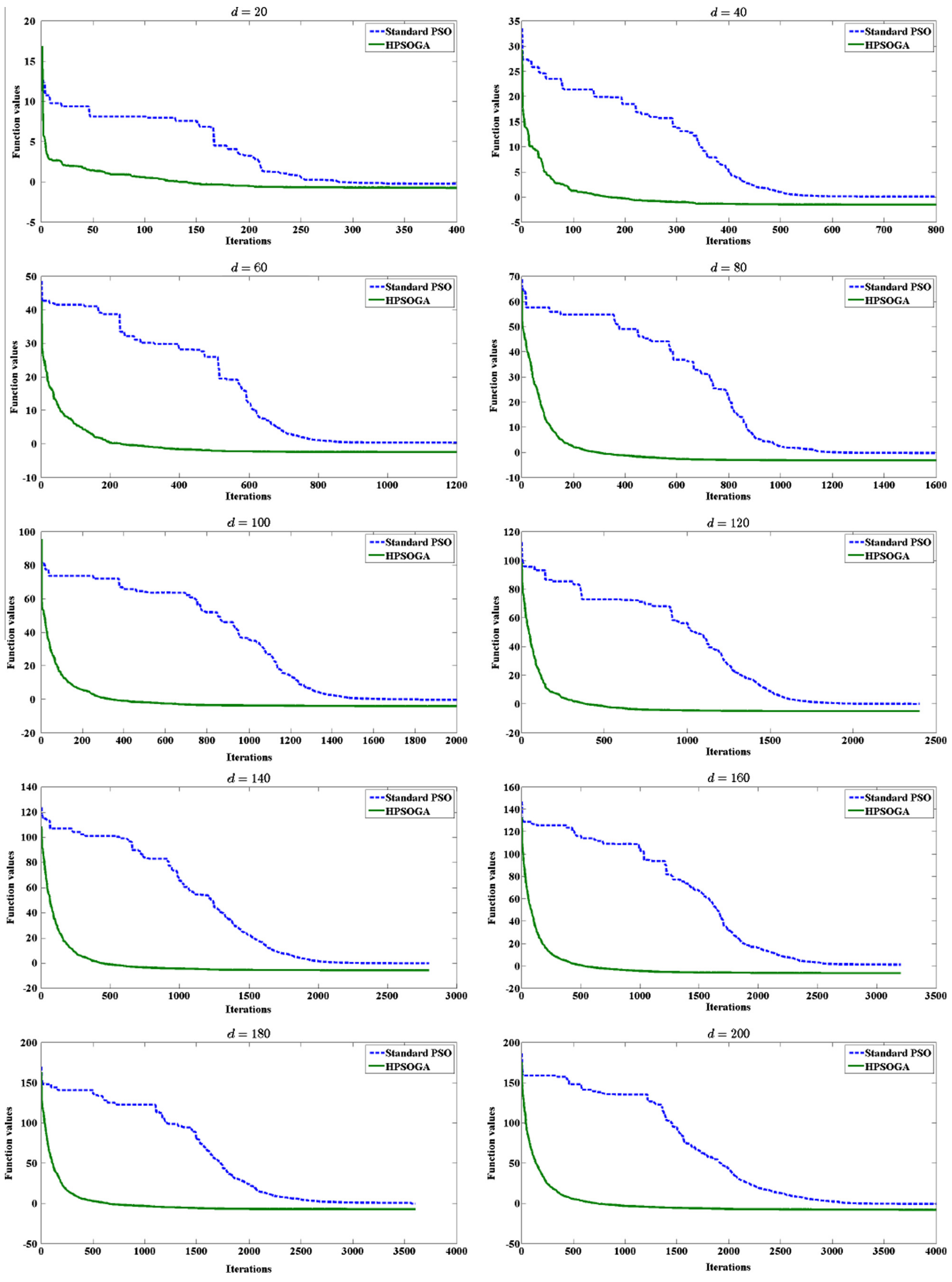

Figure 7 The efficiency of HPSOGA for minimizing the molecular potential energy function.

the test is returned in $\rho < 0.05$ indicates a rejection of the null hypothesis, while $\rho > 0.05$ indicates a failure to reject the null hypothesis. The R^+ is the sum of positive ranks, while R^- is the sum of negative ranks.

The results of the Wilcoxon test are shown in Table 11. Since, the test is not valid when the number of samples is less than 6, Wilcoxon test is applied on the proposed algorithm and other two methods VNS-123 and VNS-3. The statistical analysis of the Wilcoxon test on the data in Table 2 shows that the proposed algorithm is a promising algorithm.

6. Conclusion

In this paper, a new hybrid particle swarm optimization and genetic algorithm with population partitioning has been proposed in order to minimize the energy function of a simplified model of the molecule. The problem of finding the global minimum of the molecular energy function is difficult to solve since the number of the local minima increases exponentially with the molecular size. The proposed algorithm is called Hybrid Particle Swarm Optimization and Genetic Algorithm (HPSOGA). The solutions are updated by the proposed algorithm where the particle swarm optimization and the population partitioning mechanism are applied to reduce the dimensionality problem of the molecular potential energy function, while the arithmetical crossover operator is applied in each sub-population in order to increase the diversity of the search in the proposed algorithm. The mutation operator is applied in order to avoid the premature convergence of the solutions and escape from trapping in local minima. The proposed algorithm is tested on 13 unconstrained benchmark functions in order to investigate its performance on the large scale functions, and then it has been applied to minimize the potential energy function with different sizes up to 200 dimensions and compared against 9 benchmark algorithms in order to verify its efficiency. The experimental results show that the proposed algorithm is a promising algorithm and can obtain the optimal or near optimal global minimum of the molecular energy function faster than the other comparative algorithms.

Acknowledgments

We thank the reviewers for their thorough review and highly appreciate the comments and suggestions, which significantly contributed to improving the quality of the paper. The research of the 2nd author is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC). The postdoctoral fellowship of the 1st author is supported by NSERC.

References

[1] Wales DJ, Scheraga HA. Global optimization of clusters, crystals and biomolecules. Science 1999;285:1368–72.
[2] Floudas CA, Klepeis JL, Pardalos PM. Global optimization approaches in protein folding and peptide docking. DIMACS series in discrete mathematics and theoretical computer science. American Mathematical Society; 1999.
[3] Pardalos PM, Shalloway D, Xue GL. Optimization methods for computing global minima of nonconvex potential energy function. J Global Optim 1994;4:117–33.
[4] Troyer JM, Cohen FE. Simplified models for understanding and predicting protein structure. Rev Comput Chem 1991;2:57–80.
[5] Lavor C, Maculan N. A function to test methods applied to global minimization of potential energy of molecules. Numer Algor 2004;35:287–300.
[6] Zhao J, Tang HW. An improved simulated annealing algorithm and its application. J Dalian Univ Technol 2006;46(5):75–780.
[7] Barbosa HJC, Lavor C, Raupp FM. A GA-simplex hybrid algorithm for global minimization of molecular potential energy function. Ann Oper Res 2005;138:189–202.
[8] Deep K, Shashi, Vinod K, Katiyar, Atulya K, Nagar. Minimization of molecular potential energy function using newly developed real coded genetic algorithms. Int J Optim Control: Theor Appl (IJOCTA) 2012;2(1):51–8.
[9] Hedar A, Ali AF, Hassan T. Genetic algorithm and tabu search based methods for molecular 3D-structure prediction. Numer Algebra, Control Optim (NACO) 2011;1(1):191–209.
[10] Kovačević-Vujčić V, čangalović M, Dražić M, Mladenović N. VNS-based heuristics for continuous global optimization. In: Hoai An LT, Tao PD, editors. Modelling. Computation and optimization in information systems and management sciences. Hermes Science Publishing Ltd; 2004. p. 215–22.
[11] Dražić M, Lavor C, Maculan N, Mladenović N. A continuous variable neighborhood search heuristic for finding the three-dimensional structure of a molecule. Eur J Oper Res 2008;185:1265–73.
[12] Bansal JC, Shashi, Deep K, Katiyar VK. Minimization of molecular potential energy function using particle swarm optimization. Int J Appl Math Mech 2010;6(9):1–9.
[13] Agrawal Shikha, Silakari Sanjay. Fletcher–Reeves based particle swarm optimization for prediction of molecular structure. J Mol Graph Model 2014;49:11–7.
[14] Pogorelov A. Geometry. Moscow: Mir Publishers; 1987.
[15] Kennedy J, Eberhart RC. Swarm intelligence. San Mateo: Morgan Kaufman; 2001.
[16] Holland JH. Adaptation in natural and artificial systems. Ann Arbor, MI: University ofMichigan Press; 1975.
[17] De Jong KA. Genetic algorithms: a 10 year perspective. In: International conference on genetic algorithms. p. 169–77.
[18] Goldberg DE. Genetic algorithms in search, optimization, and machine learning. Addison-Wesley; 1989.
[19] Deep K, Thakur M. A new mutation operator for real coded genetic algorithms. Appl Math Comput 2007;193(1):211–30.
[20] Deep K, Thakur M. A new crossover operator for real coded genetic algorithms. Appl Math Comput 2007;188(1):895–912.
[21] Garcia S, Fernandez A, Luengo J, Herrera F. A study of statistical techniques and performance measures for genetics-based machine learning, accuracy and interpretability. Soft Comput 2009;13:959–77.
[22] Sheskin DJ. Handbook of parametric and nonparametric statistical procedures. Boca Raton: CRC Press; 2003.
[23] Zar JH. Biostatistical analysis. Englewood Cliffs: Prentice Hall; 1999.

Mohamed A. Tawhid got his PhD in Applied Mathematics from the University of Maryland Baltimore County, Maryland, USA. From 2000 to 2002, he was a Postdoctoral Fellow at the Faculty of Management, McGill University, Montreal, Quebec, Canada. Currently, he is a full professor at Thompson Rivers University.

His research interests include nonlinear/sto chastic/heuristic optimization, operations research, modelling and simulation, data analysis, and wireless sensor network. He has published in journals such as Computational Optimization and Applications, J. Optimization and Engineering, Journal of Optimization Theory and Applications, European Journal of Operational Research, Journal of Industrial and Management Optimization, Journal Applied Mathematics and Computation.

Mohamed Tawhid published more than 50 referred papers and edited 5 special issues in J. Optimization and Engineering (Springer), J. Abstract and Applied Analysis, J. Advanced Modeling and Optimization, and International Journal of Distributed Sensor Networks. Also, he has served on editorial board several journals. Also, he has worked on several industrial projects in BC, Canada.

Ahmed F. Ali received the B.Sc., M.Sc. and Ph. D. degrees in computer science from the Assiut University in 1998, 2006 and 2011, respectively. Currently, he is a Postdoctoral Fellow at Thompson Rivers University, Kamloops, BC Canada. In addition, he is an Assistant Professor at the Faculty of Computers and Informatics, Suez Canal University, Ismailia, Egypt. He served as a member of Computer Science Department Council from 2014 to 2015.

He worked as director of digital library unit at Suez Canal University; he is a member in SRGE (Scientific Research Group in Egypt). He also served as a technical program committee member and reviewer in worldwide conferences.

Dr. Ali research has been focused on meta-heuristics and their applications, global optimization, machine learning, data mining, web mining, bioinformatics and parallel programming. He has published many papers in international journals and conferences and he has uploaded some meta-heuristics lectures in slidshare website.