Nat Comput (2015) 14:655–672
DOI 10.1007/s11047-014-9465-2

Dynamic cluster in particle swarm optimization algorithm

Abbas El Dor · David Lemoine · Maurice Clerc · Patrick Siarry · Laurent Deroussi · Michel Gourgand

Published online: 8 October 2014

Abstract Particle swarm optimization is an optimization method based on a simulated social behavior displayed by artificial particles in a swarm, inspired from bird flocks and fish schools. An underlying component that influences the exchange of information between particles in a swarm, is its topological structure. Therefore, this property has a great influence on the comportment of the optimization method. In this study, we propose *DCluster*: a dynamic topology, based on a combination of two well-known topologies viz. *Four-cluster* and *Fitness*. The proposed topology is analyzed, and compared to six other topologies used in the standard PSO algorithm using a set of benchmark test functions and several well-known constrained and unconstrained engineering design problems. Our comparisons demonstrate that *DCluster* outperforms the other tested topologies and leads to satisfactory performance while avoiding the problem of premature convergence.

Keywords Particle swarm optimization · Neighborhood topologies · Social networks · Global best · Local best · Dynamic cluster

A. El Dor · M. Clerc · P. Siarry (✉)
Université de Paris-Est Créteil, LiSSi (E.A. 3956), 61 Avenue du Général de Gaulle, 94010 Créteil, France
e-mail: siarry@u-pec.fr

D. Lemoine
Ecole des Mines de Nantes, IRCCyN (CNRS UMR 6597), 4 Rue A. Kastler, 44307 Nantes, France

L. Deroussi · M. Gourgand
Université Blaise Pascal, LIMOS (CNRS UMR 6158), Campus des Cézeaux, 63173 Aubière, France

1 Introduction

Particle swarm optimization, introduced by Eberhart and Kennedy (1995), inspired from bird flocking behavior, is a class of population-based global optimization algorithms for solving complex problems. It uses coordinated particles to explore the search space where each particle position corresponds to a point of this space. Each particle can fly through the search space with a certain velocity. Towards this purpose, each particle is initialized with a random position and a random velocity. Later, its position and its velocity are updated, as part of the search process intelligently, based on the particle's learning experience and information exchanged with its neighbors.

Since its inception in 1995, PSO has attracted a great deal of attention from researchers because of its many advantages; it is simple, fast and can be implemented in few lines of code. The PSO optimization paradigm has become significantly popular among swarm intelligence-based algorithms (Engelbrecht 2006). Many ideas were introduced thereafter for trying to improve the original version, known to show a fast convergence that may lead the algorithm to be trapped in a local optimum. Kennedy (1999) demonstrated that the choice of neighbors for each particle has a predominant influence on the efficiency of PSO. A lot of research has been conducted in the design of such sets of neighborhood particles and has led to the definition of a variety of topologies for Particle Swarm Optimization. In this paper, we will focus on the concept of neighborhood (topology) in PSO and we propose an efficient topology called *DCluster*. We test *DCluster* on a set of well-known benchmarks in literature and we compare our results with those from popular topologies for Particle swarm optimization.

This article is organized as follows: the particle swarm optimization paradigm is briefly presented in Sect. 2. In Sect. 3, the concept of topology for PSO is introduced and the commonly used topologies are discussed. In Sect. 4, the proposed topology is described. An evaluation of seven different topologies (including *DCluster*) and their comparison are reported in Sect. 5 and finally, we conclude the study in Sect. 6.

2 Particle swarm optimization

Particle swarm optimization (PSO) is a swarm intelligent algorithm which has been motivated by the flocking behavior of birds and fish schools. This population-based metaheuristic had been initially designed to solve continuous optimization problems. Although PSO has been applied to discrete optimization, the present application deals with a continuous PSO version. Therefore, we consider a continuous optimization problem that can be written according to the following manner $\underset{x\in\Omega}{Optimize}\, f(x)$ where Ω is a convex set of $\mathbb{R}^n$ and f is a function from Ω to $\mathbb{R}$. In the present PSO framework, we consider a minimization problem, without loss of generality.

In PSO, there is a set of particles $\mathcal{S} = \{P_i\}_{i=1,\ldots,n}$ called swarm. A particle is a simple software agent which flies in the search space Ω. It may exchange information with other particles in the swarm to find an optimal solution. Thus, at each iteration t, a particle P_i is characterized by four components:

- its position $x_{it} \in \Omega$ which is a candidate solution to the optimization problem,
- its velocity $\overrightarrow{v_{it}}$ which represents the flight direction of the particle,
- its best position $pbest_i$ according to the objective function f,
- a set of particles $N_{it} \subset S$ with which it can communicate. This set is usually called the neighborhood of P_i and comprises the concerned particle too. Each particle of this neighborhood is also called a neighbor of P_i.

For each particle and at each iteration $t+1$ of the algorithm, a new velocity is computed based on a combination of the following three properties, which define the paradigm of the PSO:

- the physical property. The particle P_i tends to keep its previous direction of displacement $\overrightarrow{v_{it}}$,
- the cognitive property. The particle P_i tends to move towards its best position $pbest_i$,
- the social property. The particle uses information obtained from its current neighbors in N_{it+1}.

We can notice that, usually, the information given by the neighborhood of the particle is the best *pbest* position of all the particles in the neighborhood and is denoted by $lbest_i$:

$$lbest_i = \left\{ pbest_k \text{ where } P_k \in \underset{P_j\in N_{it}}{\operatorname{argmin}} \left[f(pbest_j) \right] \right\} \quad (1)$$

Thus the new velocity $\overrightarrow{v_{it}}$ is computed based on the Eq. 2:

$$\overrightarrow{v_{it+1}} = \overrightarrow{\mathcal{L}}\left(\overrightarrow{v_{it}}, pbest_i - x_{it}, lbest_i - x_{it}\right) \quad (2)$$

where $\overrightarrow{\mathcal{L}}$ is a vectorial function of $\mathbb{R}^n$ usually defined by:

$$\begin{aligned}&\overrightarrow{\mathcal{L}}\left(\overrightarrow{v_{it}}, pbest_i - x_{it}, lbest_i - x_{it}\right)\\ &= \overrightarrow{v_{it}} + c_1 r_{1t} \cdot [pbest_i - x_{it}] + c_2 r_{2t} \cdot [lbest_i - x_{it}]\end{aligned} \quad (3)$$

where

- r_{1t} and r_{2t} are two random vectors of $\mathbb{R}^n$ uniformly chosen in $[0,1]^n$ and called *acceleration coefficients*,
- c_1 and c_2 are two real constants called *cognitive* and *social* coefficients, respectively (and, therefore, which are fixed throughout the optimization algorithm).
- $\cdot$ is the product between two vectors, element by element.

In order to control the influence of the velocity of the particle in the determination of its new position, Shi and Eberhart (1999) have extended this formulation by introducing an *inertia weight* $w \in \mathbb{R}$. Thus, by using the same notation, the vectorial function becomes:

$$\begin{aligned}&\overrightarrow{\mathcal{L}}\left(\overrightarrow{v_{it}}, pbest_i - x_{it}, lbest_i - x_{it}\right)\\ &= w\overrightarrow{v_{it}} + c_1 r_{1t} \cdot [pbest_i - x_{it}] + c_2 r_{2t} \cdot [lbest_i - x_{it}]\end{aligned} \quad (4)$$

Finally, the new position of the particle P_i is calculated according to the following expression:

$$x_{it+1} = x_{it} + \overrightarrow{v_{it+1}} \quad (5)$$

The PSO algorithm in its synchronous version, which is used in this paper, is summarized in Fig. 1.

As pointed out by Kennedy (1999), the design of the neighborhood of each particle has a great influence on the performance of PSO. Therefore, several kinds of neighborhood designs have been proposed. In the next section, we present previously reported work on neighborhood topological structures for PSO.

3 Topologies in particle swarm optimization

As described in the previous section, a particle P_i of the swarm $\mathcal{S}$ can exchange some information with other particles in the swarm. This property can be modeled using a graph in which the vertices are the particles, and the arcs are the communication links between two particles. If the

Algorithm 1

Initialization:
for each $P_i \in \mathcal{S}$ **do**
  randomly choose the position $x_{i0} \in \Omega$, the velocity $\overrightarrow{v_{i0}}$ and set $pbest_i = x_{i0}$
  evaluate $f(x_{i0})$
end
set iteration $t = 0$.
Algorithm:
while the stopping criterion is not satisfied **do**
  for each $P_i \in \mathcal{S}$ **do**
    determine the set N_{it} of the neighbors of P_i
    update $lbest_i$ according to equation (1)
    compute $\overrightarrow{v_{it+1}}$ using relation (2)
    update position x_{it+1} using equation (5)
  end
  for each $P_i \in \mathcal{S}$ **do**
    evaluate $f(x_{it+1})$ and update $pbest_i$ if necessary
  end
  set $t = t + 1$
end

Fig. 1 Pseudo-code of synchronous PSO

communication is unidirectional, the graph is directed, otherwise it is undirected. Sometimes, a particle may communicate with itself, thus the corresponding vertex may have a loop.

Here, we focus on bidirectional communication, thus the considered graph is undirected and denoted by $G = (\mathcal{S}, E)$ where E is the set of the edges. Thus, the neighborhood of a particle P_i is the set of particles which are adjacent to P_i in the graph G. The set of edges between each particle P_i and its neighbors N_{it} forms the communication graph, also called the topology.

In addition, while the PSO algorithm progresses, the communication links between the particles may also change. Therefore, at each iteration, the topology can evolve. Accordingly, there exist two kinds of topologies:

- static topologies. Communications between particles are fixed at the beginning of the algorithm. Thus the topology (and, consequently the neighborhood of each particle) will not change. Therefore, we can denote the neighborhood of each particle P_i by N_i instead of N_{it}.
- dynamic topologies. The communication between particles may change at each iteration and the topology will keep getting dynamically updated.

We present in a first subsection, some of the main static topologies used in literature (Bastos-Filho et al. 2008), and some of the main dynamic topologies in a second subsection.

3.1 The static topologies

In Kennedy and Mendes (2002), the authors focused on static topologies for the PSO. They used metrics defined by Watts (1999) and Watts and Strogatz (1998) to classify topologies according to their effectiveness and based on some properties of the objective function such as modality. These metrics are:

- $\mathcal{K}$ the cardinal number of the neighborhood of the particles,
- the amount of clustering $\mathcal{C}$ which is the average number of neighbors-in-common for the particles.

Using a huge number of benchmarks, they showed that random graphs can be efficient for PSO when $\mathcal{K} = 5$, but they also noticed that regular structures as in standard PSO work well too. Some of the commonly used static topologies suggested in previous literature for standard PSO include:

- *Gbest*. In the original paper of PSO, Eberhart and Kennedy (1995) used a topology called *Gbest* in which each particle can communicate with all the other ones. Therefore, in this case, the graph $G = (\mathcal{S}, E)$ is complete (for each particle P_i, $N_i = \mathcal{S}$). Consequently, all the particles P_i have the same $lbest_i$ defined by (1): this common position is also called *Gbest* (for Global best position) and the Eq. (3) becomes:

$$\overrightarrow{\mathcal{L}}\left(\overrightarrow{v_{it}}, pbest_i - x_{it}, Gbest - x_{it}\right) = \overrightarrow{v_{it}} + c_1 r_{1t} \cdot [pbest_i - x_{it}] + c_2 r_{2t} \cdot [Gbest - x_{it}] \quad (6)$$

  As opposed to the *Gbest* topology, the other ones are called *Lbest* (local best position): indeed, in these cases, particles use the best position of their neighbors and not the best position in the whole swarm.

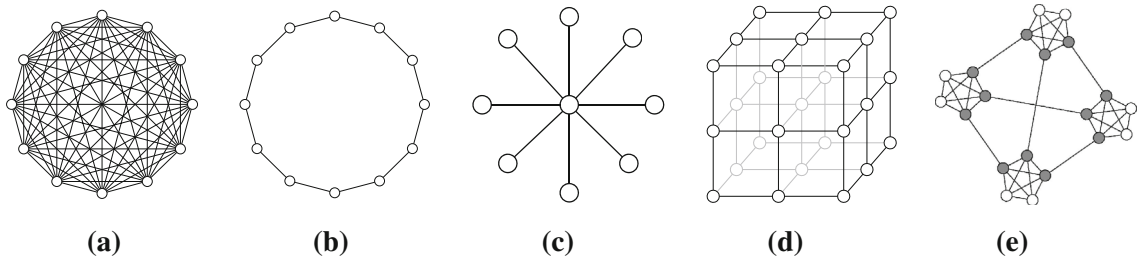

Fig. 2 Static topologies: (**a**) *Gbest* topology, (**b**) *Ring* topology, (**c**) *Wheel* topology, (**d**) *von Neumann* topology and (**e**) *Four-clusters* topology

- *Ring*. In Mendes et al. (2004), a ring is defined as a graph in which every vertex is connected to two others.
- *Wheel* also called *Focal* or *Star*. In Kennedy and Mendes (2002), this topology is defined as follows: one central node influences, and is influenced by, all other members of the population.
- *Von Neumann*. In Mendes et al. (2003), Von Neumann graph is defined as a square lattice whose extremities connect as a torus.
- *Four-clusters*. According to Mendes et al. (2004), the topology is defined as a graph with four cliques connected among themselves by gateways.

These static topologies are represented in Fig. 2.

In Kennedy and Mendes (2002), the authors conclude that no static topology can be efficient for all the continuous optimization problems and the efficiency of such topology largely depends on the landscape of the objective function. For instance, *Gbest* which instanciates the most immediate communication possible, is usually not a good topology for complex function landscape (this is due to its propensity to converge too quickly to a local optimum), whereas *Ring* topology which converges very slowly, usually exhibits good performances in complicated landscapes. *Von Neumann* and *Four-clusters* seem to be good compromises between quality of the obtained solution and speed of convergence. Therefore, they might be the most efficient standard static topologies.

As mentioned before, a static topology can be efficient for one kind of optimization problems (for instance *Gbest* for unimodal functions) but can be quite the opposite on a large set of other types of landscapes. This is due to its inability to evolve during the discovery of the optimization landscape. Dynamic topologies were consequently designed, to overcome this limitation.

3.2 The dynamic topologies

In this section, we present some dynamic topologies. They can be classified into two groups based on the regularity of the structure used, to define the topology.

3.2.1 *Dynamic topologies without regular structure*

In Clerc (2006), the author proposed a randomized topology, named here R2006. According to the distribution probability chosen in order to determine which particle is informed by whom, the size of neighborhood for each particle follows a Binomial distribution where the mean value of this distribution is just greater than K, the preferred size of particle's neighborhood. If the current iteration does not improve the objective function, the topology is redesigned according to the same scheme at the next iteration. This dynamic topology has been used in several versions of PSO, as in SPSO 2006, SPSO 2007 and SPSO 2011 (PSC 2012).

3.2.2 *Dynamic topologies with regular structure*

One of the most natural ways to define such dynamic topology is to allow a particle to communicate with its Kth closest particles in the search space Ω according to the Euclidean distance. Hence, the underlying graph is a K-regular Graph. In this paper, this topology is called Geometric. This approach has been generalized by using spatial subdivisions of Ω by using for instance Delaunay triangulation (Lane et al. 2008) or Voronoi diagram (Safavieh et al. 2009). Such topologies are called *Geographical topologies* as opposed to *Social topologies* which do not use these geographical considerations but usually define static topologies.

In Janson and Middendorf (2005), the authors propose a dynamic topology, named here Hierarchy. This topology is defined by using a regular tree as the communication graph. This tree is characterized by its branching degree $K - 2$ and its depth h (therefore neighborhoods of particles which are not placed on a leaf or on the root of the tree have all the same size K). The position of each particle evolves according to the following rule: using a breadth-first search, particle on a node is compared to all particles which are on its child nodes according to their best solutions. If a particle on a child node outperforms the particle on the parent node, then they swap their positions in the tree. As the authors noticed the great influence of the

branching degree, they proposed a procedure to dynamically change it (by decreasing it).

Richards and Ventura (2003) propose a dynamic topology which tries to exploit the idea that the *ring* topology is more adapted to multimodal landscape whereas *Gbest* topology converges quickly in a monomodal context. Thus, beginning with a *ring* topology, new edges are periodically added in order to obtain a *Gbest* topology at 80 % of the maximum number of iterations. Therefore, at the beginning, the topology allows a survey of the search space whereas particles are focusing on the most promising area when topology converges to *Gbest*.

In Wang and Xiang (2008a), the authors propose the *Fitness* topology. In *Fitness*, the authors also use a *ring* topology but on this ring, particles are positioned in a non decreasing order of their personal best values, in a clockwise direction. Then, they only consider monodirectional communication links in an anticlockwise direction (of course, the authors consider an undirected one). Therefore, a particle can only receive informations from a better particle except the best one which is connected to the worst particle. In order to improve this topology, they designed a rolling window on the ring: a particular set of contiguous particles (the window) is assigned to each particle and when we move from a particle to the next one on the ring, the window is shifted forward one particle. If for one particle, the corresponding window is composed with at most one better particle, then a "Learn Far From Better Ones" strategy (Wang and Xiang 2008b) is applied, otherwise a "centroid of mass" strategy inspired from Kennedy (2000) is used.

In Liang and Suganthan (2005), the authors proposed the Dynamic Multi-Swarm Particle Swarm Optimizer (DMS-PSO), which is based on the local version of PSO with a new neighborhood topology. In DMS-PSO, the sub-swarms are dynamic and the sub-swarms' size is very small. The whole population is divided into a large number of sub-swarms, these sub-swarms are regrouped frequently by using various regrouping schedules and information is exchanged among the particles in the whole swarm. Zhao, Liang, Suganthan, Nasir and Li proposed also other sophisticated topologies in Zhao et al. (2011), Liang and Suganthan (2006), Nasir et al. (2012) and Li et al. (2011).

Dynamic topologies are more adapted to complicated landscapes thanks to their ability to take into account parts of search space already discovered. Then, a static topology may not be suitable for all situations. Therefore, we proposed a new dynamic topology called *DCluster* which combines *Four-clusters* and *Fitness* topologies in order to exploit the salient properties of their underlying topologies. *DCluster* topology is explained in the next section.

4 The proposed topology

As described earlier, static topologies may suffer from many limitations, which affect the convergence of the standard PSO algorithm. Considering the fact that a dynamic topology may play a significant role in complex optimization problems, we propose *Dcluster* - a swarm that is segmented into several sub-swarms having the same size (called Clusters), based on the following rule: the particles will be grouped according to the values of their current solutions in order to create level clusters. Thus, the algorithm tends to employ better clusters to aid clusters having comparatively poorer solutions. Such a topology may be named as *DCluster* for Dynamic Cluster.

At each iteration t, the design of the topology follows four steps:

1. For each particle, evaluate its position according to the objective function and sort particles according to their current values in a non decreasing order, as in *Fitness* topology.
2. Divide the sorted set of particles into several subsets (with the same size) of particles with respect of the non decreasing order. The number of subsets has to be chosen in order to be equal to the number of particles in a subset plus one.
3. In each subset, connect all the particles together (the obtained graph for each subset is complete). Note that each particle is also connected to itself. At the end of this step, the clusters are created and can be sorted in a non decreasing order according to the sum of the values of their own particles. Therefore, the cluster no. 1 will be composed of the worst particles of the swarm, the cluster no. 2 with the worst particles by removing particles of cluster no. 1 from the swarm and so on. Therefore, the last cluster is composed of the best particles. The three steps are summarized in Fig. 3.
4. Cluster no. 1 is placed in the center of the topology (and is called "central cluster"). By sorting particles of this cluster in a non decreasing order, its first particle is linked to the worst particle of Cluster no. 2, its second particle to the worst particle of Cluster no. 3, and so one. At the end, each particle of the central cluster is linked to the worst particle of another cluster. Hence, all clusters in this topology, including the central cluster, have a fully connected neighborhood, as shown in Fig. 4

The reason why the central cluster is linked to each cluster by only one gateway and with the worst particle of the latter is to avoid a premature convergence to a local optimum by slowing down the propagation of the informations in the whole swarm. Another point to notice is that

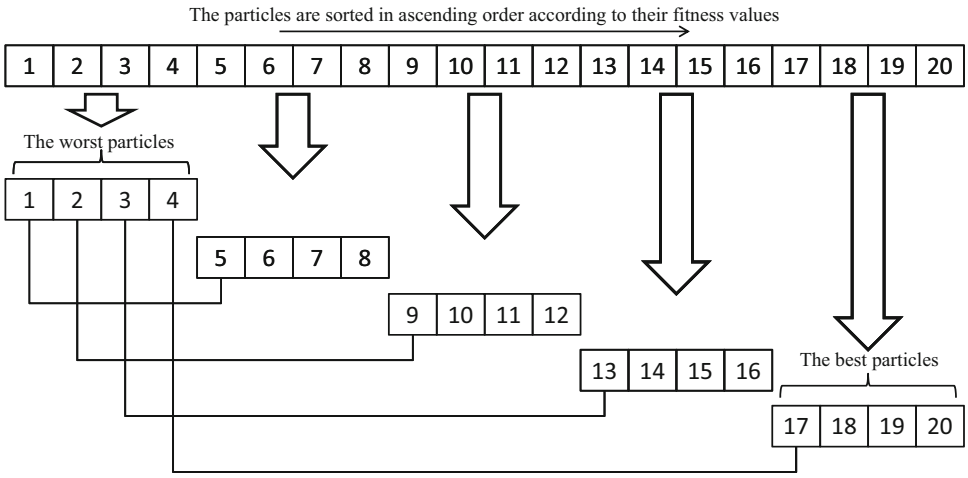

Fig. 3 The partitioning of the table into clusters

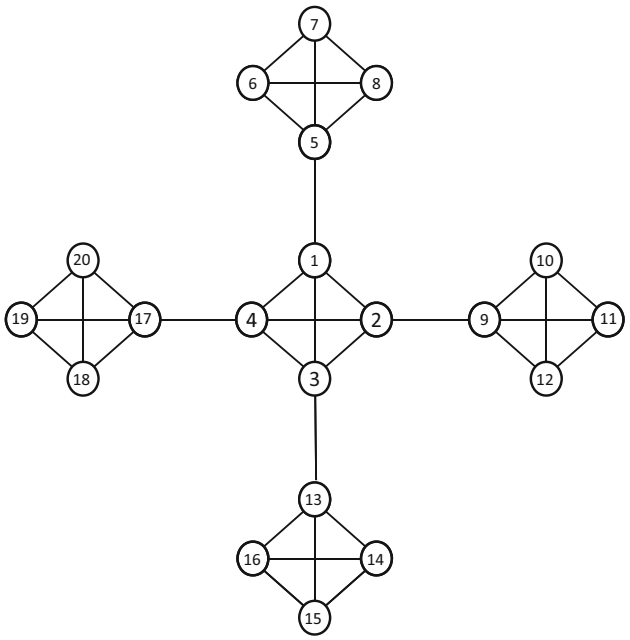

Fig. 4 The structure of *DCluster* topology

the use of such topology imposes a particular size of the swarm. Indeed, due to the step 2 of the design of the topology, the size of the swarm has to be computed according to the Eq. (7):

$$|\mathcal{S}| = N \times (N + 1) \tag{7}$$

where N is the size of each cluster.

Using previous notations, pseudocode of PSO using *DCluster* topology is given in Fig. 5

In the following section, we present results obtained with our topology and we compare them with results obtained with other topologies found in the literature and known to be efficient.

5 Experiments and discussion

In this section, we present a comparison between *DCluster* topology and six other topologies taken from the literature of PSO. In order to achieve this comparison, sets of benchmark functions and real-life problems [twelve functions from CEC2005 (Suganthan et al. 2005), and twelve real-life problems] are used in our experiments that are detailed in el Dor et al. (2012), Ali et al. (2010), Clerc (2006), El-Saleh et al. (2009), Liu et al. (2010) and Cagnina et al. (2008). In Shifted functions, the coordinates were shifted by using an offset vector $O = (o_1, o_2, \ldots, o_n)$. Thus the global optimum is translated, and it is expected that this operation makes the problem more difficult to solve. In Rotated functions, the Salomon's method (Salomon 1996) is used to rotate the functions. This method generates an orthogonal matrix M to get a new variable y ($y = M \times x$), where x is the original variable. The used offset vectors and orthogonal matrices are taken from the proceedings of the Conference CEC 2005. These functions and the real-life problems are presented in the Appendix and are used to compare the results obtained by all PSO versions with different topologies.

5.1 Experimental setting

The seven different topologies including the proposed one [R2006 (Clerc 2006), *DCluster*, *Fitness* (Wang and Xiang 2008a), *Four-clusters* (Mendes et al. 2004), Geometric, Hierarchy (Janson and Middendorf 2005), and *Gbest*], were implemented in the standard version of PSO SPSO2006[1] (PSC 2012), and their performances were compared. The *cognitive* and *social* coefficients c_1 and c_2 are chosen similar ($c_1 = c_2 = 1.19$) and the inertia weight w is fixed to 0.72 for each algorithm [the default values used in the *Standard* PSO 2007 (PSC 2012) and in Hsieh et al. (2009)].

[1] SPSO2006 uses the random topology R2006.

Algorithm 2

Initialization:
for each $P_i \in \mathcal{S}$ **do**
  randomly choose the position $x_{i0} \in \Omega$, the velocity $\overrightarrow{v_{i0}}$ and set $pbest_i = x_{i0}$
  evaluate $f(x_{i0})$
end
set iteration $t = 0$.
Algorithm:
while the stopping criterion is not satisfied **do**
  /* Design of the topology */
  /* Step 1 */
  Sort the set of particles according to their current values in a non decreasing order: let $\{P_{\sigma[1]}, \ldots, P_{\sigma[N\times(N+1)]}\}$ this ordered set.
  /* Step 2 */
  for $i = 1$ **to** $N + 1$ **do**
    Set $Cluster(i) = \emptyset$
    for $j = 1$ **to** N **do**
      $Cluster(i) = Cluster(i) \cup \{P_{\sigma[(i-1)*N+j]}\}$
    end
  end
  /* Step 3 */
  for $i = 1$ **to** $N + 1$ **do**
    Connect all the particles in $Cluster(i)$ together
  end
  /* Step 4 */
  for $j = 1$ **to** N **do**
    Connect $P_{\sigma[j]}$ to $P_{\sigma[j\times N+1]}$
  end
  /* End of the design */
  for each $P_i \in \mathcal{S}$ **do**
    determine the set N_{it} of the neighbors of P_i
    update $lbest_i$ according to equation (1)
    compute $\overrightarrow{v_{it+1}}$ using relation (2)
    update position x_{it+1} using equation (5)
  end
  for each $P_i \in \mathcal{S}$ **do**
    evaluate $f(x_{it+1})$ and update $pbest_i$ if necessary
  end
  set $t = t + 1$
end

Fig. 5 Pseudo-code of the *DCluster* PSO

The population size is set to 20 for the seven versions of PSO. The size of the neighborhood for *Fitness* and Geometric is set to five, where it is variable between one and 20 for R2006, with a high probability to be close to five. For the Hierarchy topology, the depth h is set to four and the branching degree $\mathcal{K} - 2$ is set to two. In these experiments, *DCluster* uses a regular clustering form as in Fig. 4, where the number of clusters is equal to five and the number of particles in each cluster N is set to four. The maximum number of function evaluations (FEs) is set to 50,000 in 10-D, for the twelve benchmark functions $(f_1, f_2, \ldots, f_{12})$ given in the section 'Benchmark functions' in the Appendix , and is set to 150,000 in 30-D. For the twelve real problems presented in the section 'Real life problems' in the Appendix , the maximum number of function evaluations (FEs) for each problem is given in Table 3. The KISS (Keep it Simple Stupid) Random Number Generator (RNG) (Marsaglia and Zaman 1993) is used for the seven topologies. The experiments are performed using a computer equipped with a 64-bit processor.

5.2 Experimental results

The simulation results in 10-D and 30-D, on the twelve benchmark functions presented in the section 'Benchmark functions' in the Appendix , are reported in Tables 1 and 2, respectively. More specifically, the mean best value and the standard deviation obtained in 100 experiments for each algorithm and each function using 10 and 30 dimensions are reported in these tables. We added a column showing the P value (which expresses the probability for a hypothesis to be true) given by the Friedman statistical test, in order to confirm that the results are significantly different (the 100 best found solutions of each algorithm). The best results among those obtained by the seven algorithms are shown in bold. From these experiments, we can notice

Table 1 Results on the 10-D problems

f	R2006	*DCluster*	*Fitness*	*Four-clusters*	Geometric	Hierarchy	*Gbest*	P value
f_1	2.81e+004*	**4.99e+000**	1.29e+001	44e+000	8.12e+003	2.25e+001	1.55e+005	1.92e−30
	2.79e+005	**1.55e+001**	4.77e+001	3.10e+001	8.07e+004	6.27e+001	1.29e+006	
f_2	1.16e−002	1.79e−014	1.16e−002	1.14e−001	1.62e+000	**1.70e−014**	8.23e−001	1.67e−50
	1.15e−001	1.54e−014	1.15e−001	3.67e−001	1.48e+000	**1.50e−014**	2.10e+000	
f_3	5.92e−002	**2.71e−002**	3.78e−002	7.15e–002	1.03e−001	4.84e−002	1.75e−001	2.35e−27
	3.38e−002	**2.08e−002**	2.84e−002	5.02e−002	6.89e−002	4.11e−002	3.75e−001	
f_4	6.06e+000	**2.81e+000**	3.83e+000	9.49e+000	1.57e+001	4.08e+000	1.76e+001	1.31e−59
	3.07e+000	1.62e+000	1.76e+000	5.31e+000	6.94e+000	1.77e+000	8.45e+000	
f_5	**3.55e+002**	7.62e+002	1.53e+003	5.16e+002	2.51e+003	2.56e+003	5.053e+002	3.37e−14
	1.28e+003	2.63e+003	4.14e+003	1.90e+003	1.69e+004	5.02e+003	1.96e+003	
f_6	2.33e+001	**2.28e+001**	2.31e+001	2.30e+001	2.29e+001	2.33e+001	5.93e+003	1.06e−14
	8.50e−002	**7.93e−002**	1.05e−001	8.93e−002	8.21e−002	1.06e−001	3.02e+001	
f_7	1.12e−001	**1.04e−002**	8.78e−002	1.87e−001	6.86e−001	5.27e−002	3.18e−001	1.89e−56
	7.68e−002	**8.86e−002**	7.79e−002	1.17e−001	7.02e−001	3.61e−002	2.71e−001	
f_8	1.14e+001	**4.96e+000**	7.87e+000	1.35e+001	1.23e+001	5.61e+000	1.99e+001	3.86e−35
	5.81e+000	**3.40e+000**	4.00e+000	4.99e+000	3.78e+000	4.29e+000	8.79e+000	
f_9	2.02e−001	**1.49e−001**	4.35e−001	2.01e−001	3.57e+000	8.17e−001	3.68e−001	1.26e−37
	8.70e−001	**2.86e−001**	1.04e+000	8.70e−001	1.12e+000	1.55e+000	1.14e+000	
f_{10}	3.96e−002	**3.66e−015**	4.19e−015	5.11e−002	1.14e+000	3.73e−015	6.90e−001	3.21e−84
	2.29e−001	**1.23e−015**	1.55e−015	2.54e−001	1.21e+000	1.22e−015	8.57e−001	
f_{11}	5.02e−002	**2.48e−002**	3.04e−002	6.45e−002	9.24e−002	4.11e−002	1.22e−001	5.56e−35
	3.02e−002	**2.00e−002**	2.28e−002	3.78e−002	5.95e−002	2.92e−002	7.25e−002	
f_{12}	6.00e+000	**2.64e+000**	3.65e+000	8.07e+000	1.13e+001	4.08e+000	1.39e+001	9.86e−45
	2.52e+000	**1.82e+000**	2.15e+000	4.39e+000	4.51e+000	2.10e+000	7.10e+000	

* The reason of the bad average is due to a single run whose best found solution is 2.80e+006, knowing that the average of the other 99 runs is 7.87e+000

that the *Gbest* topology yields the worst results on most functions. For the other topologies, the results are mitigated with better ones for the *DCluster* topology, thus outperforming the other topologies for the majority of the tested functions. From the Tables 1 and 2, one can notice that the obtained P-value is less than the significance level α (threshold chosen to reject the hypothesis checked by the statistical test), which is often 0.05, for all tested functions. Therefore, we can conclude that the obtained results by each algorithm are significantly different from the others, with a confidence level of 95 % (P value ≤ 0.05).

In Table 1, the algorithm using *DCluster* topology performs better than the others for $f_1, f_3, f_4, f_6, f_7, f_8, f_9, f_{10}, f_{11}$ and f_{12}, but not for function f_2 and f_5. *DCluster* and Hierarchy show noticeable results for f_2 with Hierarchy having the best result of this function. Indeed, *DCluster* outperforms the other topologies for ten functions out of twelve, and it is outperformed by Hierarchy on one function and by R2006 on one other. The simulation results for 10-D Shifted Ackley, 10-D Shifted Griewank, 10-D Shifted Rastrigin and 10-D Shifted Rosenbrock are shown in Fig. 6. As we can see in this figure, *DCluster* does not lead to the best solution for each problem compared to other topologies. However, there is no premature convergence, and the best solution found still improves at the end of the execution of PSO. The good convergence of *DCluster* can be explained by the use of five sets of particles dynamically linked. Hence, each set can explore a different region of the search space, that can accelerate the convergence to the global optimum, and prevent the algorithm from being trapped in a local optimum. Figure 6 demonstrates also the fast converging ability of PSO using Geometric and *Four-clusters* topologies in comparison with the others.

Our experimental results for the 30-D functions are shown in Table 2. We compare the mean best values and the standard deviations oven 100 runs of the seven PSO algorithms given in Table 2. Despite the fact that the 30-D functions are more difficult than the 10-D ones, PSO using *DCluster* topology still obtains good performances and scales up well with the problem dimension. It is noticeable that the results obtained by *DCluster* in 30-D are indeed not significantly different than those obtained in 10-D, where *DCluster* topology outperforms the others in eight functions out of twelve. PSO using *DCluster* topology performs

Table 2 Results on the 30-D problems

f	R2006	*DCluster*	*Fitness*	*Four-clusters*	Geometric	Hierarchy	Gbest	P value
f_1	2.83e+001	5.72e+001	8.21e+001	**1.50e+001**	6.70e+007	9.10e+001	3.99e+006	6.00e−64
	6.66e+001	9.32e+001	1.37e+002	**3.32e+001**	9.17e+007	1.65e+002	3.07e+007	
f_2	1.73e+000	**9.00e−014**	2.00e−001	2.66e+000	1.47e+001	5.98e−001	8.42e+000	1.10e−86
	1.03e+000	**2.45e−014**	4.92e−001	1.64e+000	2.26e+000	1.30e+000	2.46e+000	
f_3	6.26e−002	**4.43e−003**	1.50e−002	1.10e−001	1.39e−001	4.87e−002	4.05e+000	1.03e−57
	3.14e−001	**9.38e−003**	2.18e−002	3.44e−001	1.12e+001	8.43e−002	9.18e+000	
f_4	5.65e+001	**3.20e+001**	4.51e+001	9.49e+001	1.55e+002	4.12e+001	1.27e+002	2.49e−78
	1.85e+001	**1.14e−001**	1.57e+001	2.53e+001	3.15e+001	1.59e+001	4.28e+001	
f_5	1.87e+003	2.13e+003	1.82e+005	**1.32e+003**	5.67e+008	3.07e+003	9.35e+002	3.89e−46
	4.62e+003	4.72e+003	1.78e+006	**3.88e+003**	5.69e+008	6.36e+003	2.35e+003	
f_6	2.92e+001	**2.83e−001**	2.91e+001	2.90e+001	2.90e+001	2.92e+001	2.09e+001	8.49e−9
	7.09e−002	**5.98e−002**	7.85e−002	6.92e−002	8.23e−002	6.85e−002	9.93e−002	
f_7	2.11e−002	**1.04e−002**	1.63e−002	2.37e−002	2.50e+001	1.95e−002	2.37e−002	1.79e−46
	1.65e−002	**1.49e−002**	1.36e−002	2.20e−002	1.50e+001	1.89e−002	2.11e−002	
f_8	7.82e+001	1.03e+002	5.66e+001	7.21e+001	1.09e+002	**3.63e+001**	1.06e+002	3.21e−54
	4.69e+001	2.08e−001	3.80e+001	2.01e+001	2.44e+001	**2.53e+001**	2.93e+001	
f_9	3.96e+000	1.94e+000	1.90e+001	**8.84e−001**	6.04e+001	1.72e+001	3.26e+000	5.44e−19
	1.91e+000	1.54e+000	1.09e+001	**1.57e+000**	2.73e+001	1.66e+001	3.91e+000	
f_{10}	1.47e+000	**2.08e−002**	1.92e−001	2.45e+000	1.07e+001	4.84e−001	6.10e+000	5.87e−55
	9.92e−001	**1.47e−001**	4.85e−001	1.09e+000	1.86e+000	7.82e−001	1.98e+000	
f_{11}	2.71e−002	**3.56e−003**	9.83e−003	2.94e−002	2.91e+000	1.05e−002	1.35e−001	8.42e−25
	4.99e−002	**7.30e−003**	1.45e−002	3.80e−002	2.30e+000	1.35e−002	2.48e−001	
f_{12}	5.13e+001	**3.22e+001**	5.08e+001	6.32e+001	7.93e+001	4.61e+001	7.54e+001	1.58e−33
	1.41e+001	**1.28e+001**	1.62e+001	1.67e+001	1.63e+001	1.62e+001	2.01e+001	

significantly better than the other algorithms for f_2. PSO using *Four-cluster* topology performs better for f_1, f_5 and f_9 (*Shifted Rosenbrock, Rotated Rosenbrock* and *Rosenbrock*), in which the global minimum is inside a long, narrow, parabolic shaped flat valley, and it is difficult to find it. *DCluster* and the other topologies are outperformed by Hierarchy topology on f_8. In the last column in Table 2, the P values are reported to ensure that the obtained results of the seven topologies are significantly different. Figure 6 illustrates a representative set of experiments for the same four functions as discussed above in 10-D. The graphs show that PSO using *DCluster* topology achieves better results and avoids the premature convergence. The convergence speed of *DCluster* topology is close to the convergence speed of the other topologies, except for the Geometric topology, in 30-D shifted Rosenbrock function. Geometric topology is trapped in a local optimum early during the search for all functions. Reader can also notice that in the Shifted Ackley function case, *DCluster* significantly outperforms the other topologies: this is probably due to the special characteristic of this function. Indeed, the later includes many local optima, and their number rapidly increases with the dimension D. For $D = 10$, the Hierarchy topology still can cope with them, but this does not remain true when the dimension increases ($D = 30$).

To ensure the performance of *DCluster* topology, we evaluate the six topologies (R2006, *DCluster*, *Fitness*, *Four-clusters*, Geometric, Hierarchy) in twelve real-life problems (presented in the section 'Real life problems' in the Appendix) having different numbers of dimensions. The results in Table 4 are calculated for these problems in the same way as in 10-D and 30-D on 100 runs. In this table, we added, for each problem, a column showing the number of dimensions and another showing the maximum number of fitness evaluations (Nb. Evals.). The P values given by the Friedman statistical test are also reported in this table.

In order to preserve feasibility of the solutions for the constrained problems used here, the update of the best positions of the particles is performed using a static penalty approach (Homaifar et al. 1994), in which the penalty factors do not depend on the current generation number, and therefore, remain constant during the entire search process. More specifically, the best position of a particle is updated only if the new candidate best position is feasible, otherwise, it remains unchanged. In this approach, the user

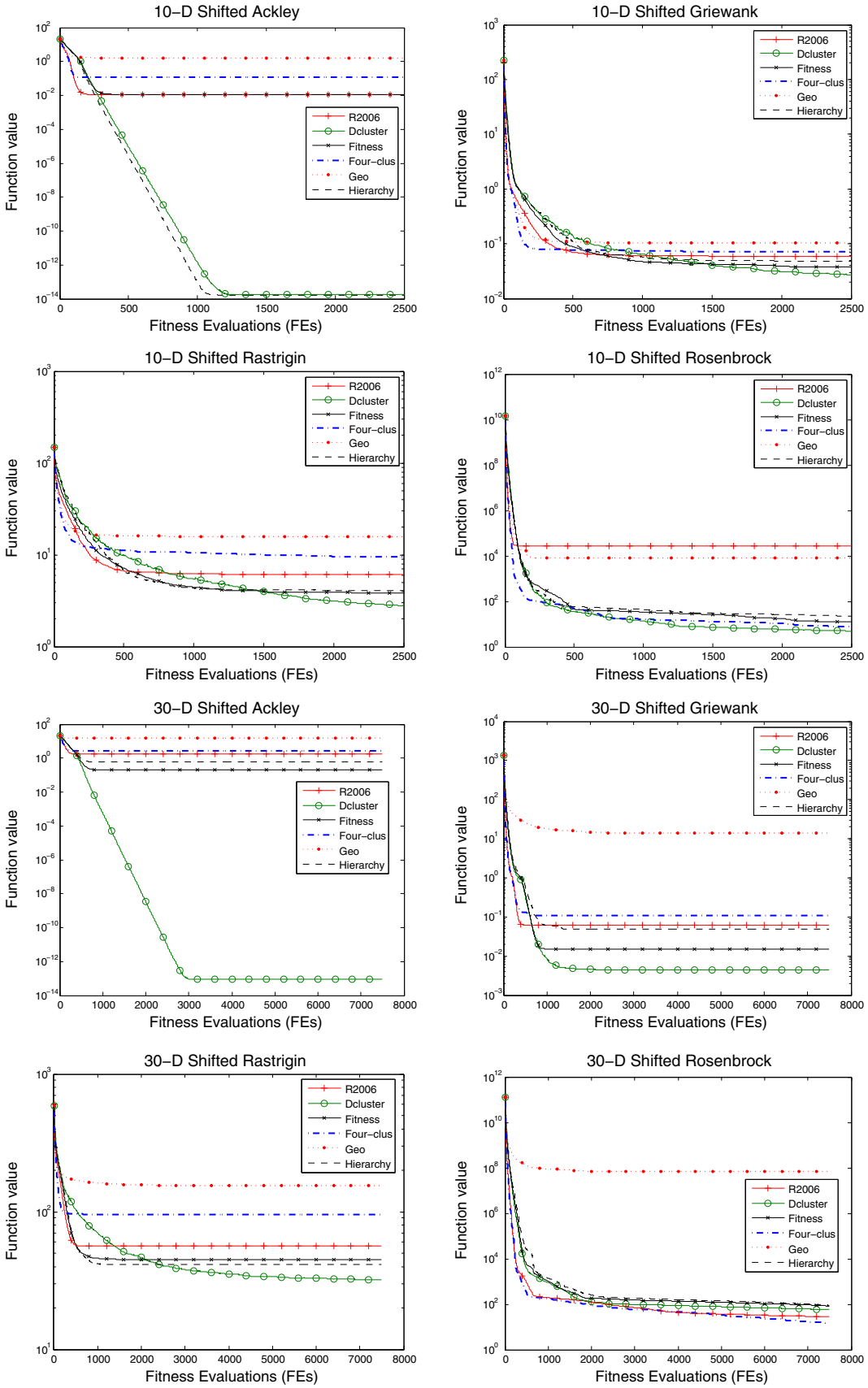

Fig. 6 Algorithm convergence for the selected problems. Results are averaged over 100 runs

Table 3 *D* and *FEs* for the twelve real-life problems

	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}
D	27	4	6	10	3	5	42	4	4	7	13	10
FEs	150,000	50,000	150,000	240,000	50,000	50,000	150,000	300,000	50,000	50,000	15,000	100,000

Table 4 Results on the real-life problems

F	R2006	*DCluster*	*Fitness*	*Four-cluster*	Geometric	Hierarchy	*P* value
F_1	1.26e+000	6.63e+000	1.49e+000	1.08e+000	5.74e+000	**1.00e+000**	1.99e−71
	1.26e+000	1.83e+000	1.57e+000	6.98e−001	1.97e+000	**6.19e−001**	
F_2	3.61e−010	2.89e−010	3.17e−010	9.20e−010	4.37e−009	**2.56e−010**	8.50e−34
	7.79e−010	6.27e−010	6.01e−010	2.87e−009	7.69e−009	**4.91e−010**	
F_3	8.89e+000	**2.21e+000**	4.50e+000	1.14e+001	5.74e+000	6.53e+000	2.40e−22
	6.36e+000	**3.98e+000**	4.97e+000	6.73e+000	1.97e+000	6.14e+000	
F_4	4.69e−001	**3.90e−001**	4.23e−001	6.34e−001	7.21e−001	4.17e−001	1.07e−48
	1.64e−001	**1.22e−0001**	1.41e−001	1.26e−001	9.03e−002	1.62e−001	
F_5	8.54e−003	**6.53e−005**	2.67e−004	2.21e−002	5.29e−003	2.34e−003	1.04e−8
	2.99e−002	**6.50e−004**	1.28e−003	5.36e−002	3.23e−002	1.19e−002	
F_6	5.29e+002	**1.14e+002**	1.62e+002	5.99e+002	6.69e+002	1.98e+002	2.04e−34
	5.03e+002	**1.96e+002**	2.41e+002	5.49e+002	5.66e+002	2.92e+002	
F_7	1.04e+002	**9.06e+001**	9.49e+001	1.09e+002	1.12e+002	9.79e+001	3.68e−31
	1.37e+001	**1.02e+001**	1.23e+001	1.81e+001	2.90e+001	9.65e+000	
F_8	6.68e+000	**3.59e−001**	3.03e+000	9.92e+000	3.86e−001	5.37e+000	1.00e−7
	2.68e+001	**3.58e+000**	1.90e+001	4.25e+001	3.84e−001	2.30e+001	
F_9	1.15e−002	**3.08e−007**	5.51e−004	6.98e−002	1.00e−001	**3.08e−007**	1.97e−83
	5.52e−002	**6.48e−017**	5.25e−003	1.30e−001	1.33e−001	**3.79e−017**	
F_{10}	3.92e−001	**0.00e+00**	**0.00e+00**	1.08e+000	3.92e−001	**0.00e+00**	1.36e−10
	3.91e+000	**0.00e+00**	**0.00e+00**	5.71e+000	3.91e+000	**0.00e+00**	
F_{11}	6.17e+000	**2.00e−002**	3.57e−001	6.06e+000	5.06e+000	5.97e−001	2.76e−84
	2.28e+000	**1.99e−001**	8.29e−001	2.55e+000	2.15e+000	9.68e−001	
F_{12}	7.06e−001	**3.26e−001**	4.11e−001	1.21e+000	7.43e+000	5.93e−001	4.39e−52
	6.83e−001	**2.92e−001**	3.86e−001	1.12e+000	8.97e+000	5.69e−001	

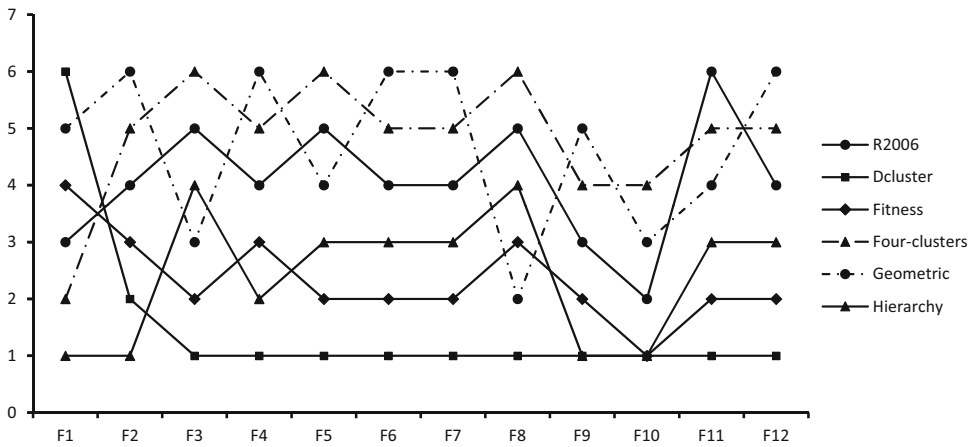

Fig. 7 The classification of the six algorithms on twelve real problems

defines several levels of violation, and a penalty coefficient is chosen for each in such a way that the penalty coefficient increases as we reach higher levels of violation. This approach starts with a random population of individuals (feasible or infeasible). An individual is evaluated using:

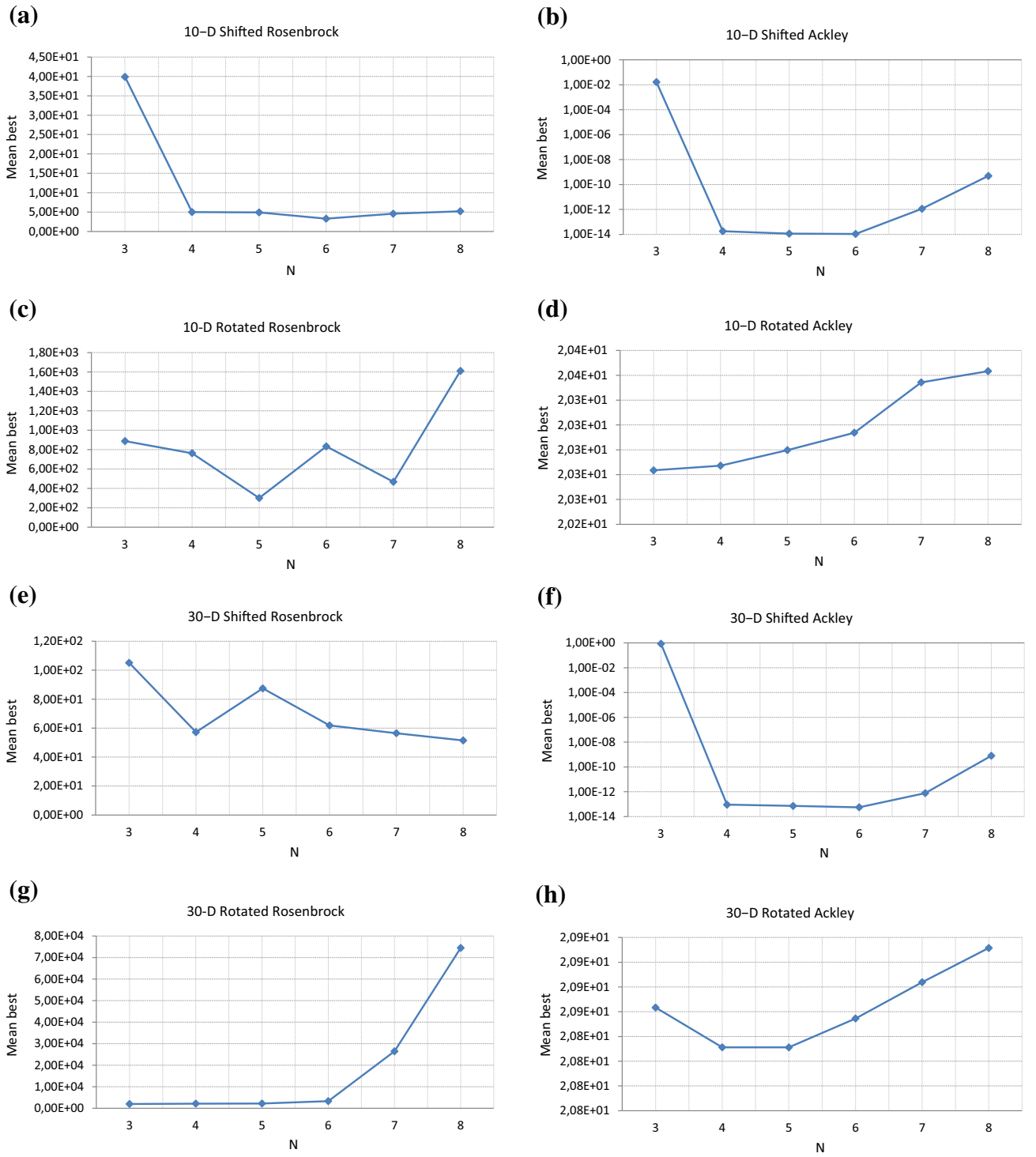

Fig. 8 Sensitivity analysis of *DCluster* topology, the evolution of the parameter N for: (**a**) 10-D shifted rosenbrock, (**b**) 10-D shifted Ackley, (**c**) 10-D rotated Rosenbrock, (**d**) 10-D rotated Ackley, (**e**) 30-D shifted Rosenbrock, (**f**) 30-D shifted Ackley, (**g**) 30-D rotated Rosenbrock and (**h**) 30-D rotated Ackley

$$fitness(\mathbf{x}) = f(\mathbf{x}) + \sum_{i=1}^{m} \left(R_{k,i} \times \max[0, g_i(\mathbf{x})]^2 \right) \quad (8)$$

where $R_{k,i}$ are the penalty coefficients used, m is the total the number of constraints, $f(\mathbf{x})$ is the unpenalized objective function, and $k = 1, 2, \ldots, l$, where l is the number of levels of violation defined by the user.

According to our experiments, the best results are obtained when PSO uses *DCluster* topology on most problems, except F_1 and F_2. The results obtained using the other

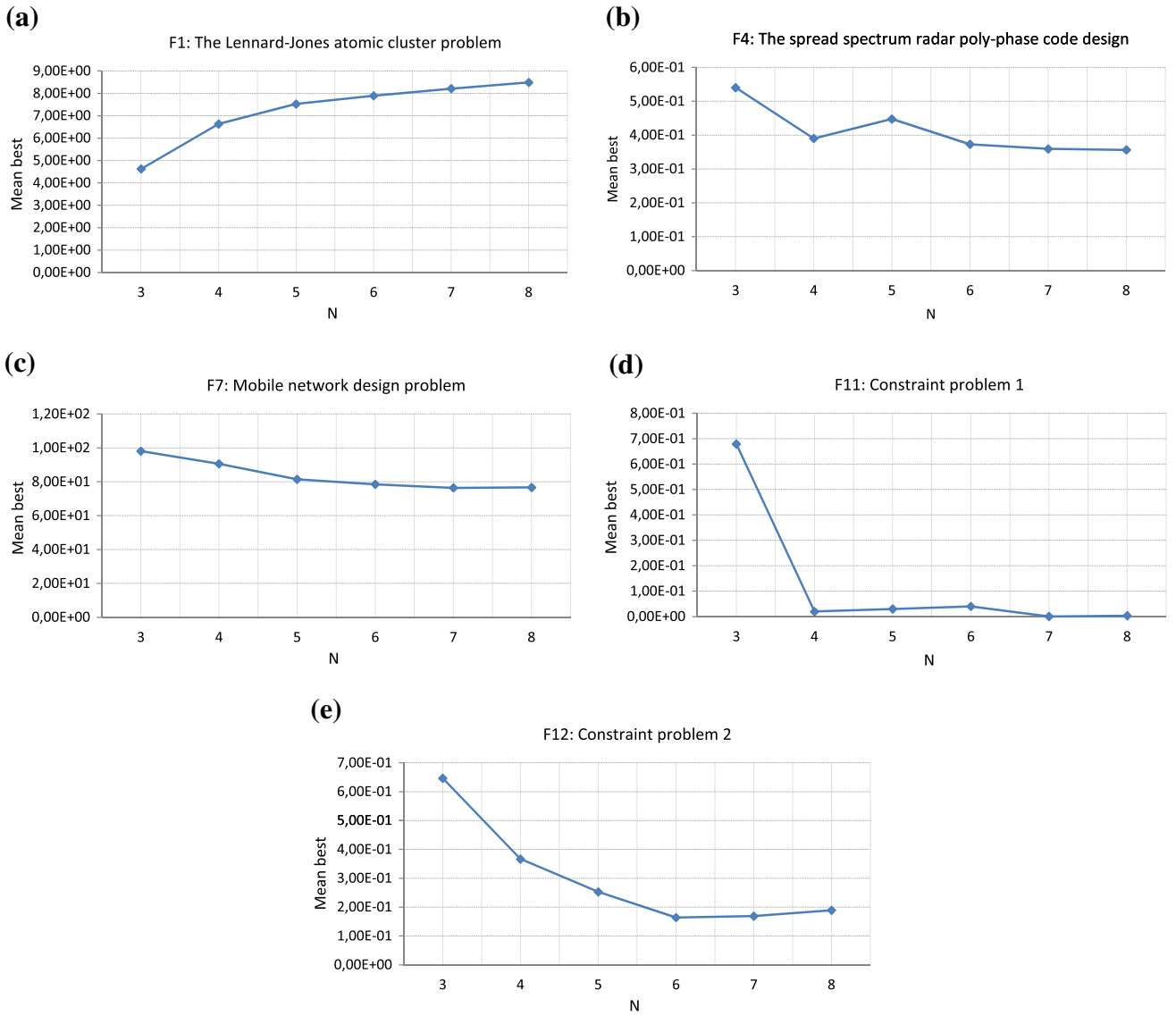

Fig. 9 Sensitivity analysis of *DCluster* topology, the evolution of the parameter N for: (**a**) F1: the Lennard-Jones atomic cluster problem, (**b**) F4: the spread spectrum radar poly-phase code design, (**c**) F7: mobile network design problem, (**d**) F11: constraint problem 1, (**e**) F12: constraint problem 2

topologies are mitigated and we cannot determine which topology performs better than the others. The Hierarchy topology outperforms our proposed topology on F_1 and F_2 and it obtains the same results on F_9 and F_{10}. F_1 is well known to be very discriminant: indeed, this function has not only many local optima but also, and it is the most important point, an attraction basin of the global one which is quite small. However, for fair comparison we have used the same population size (20) for all topologies, but in this precise case it may be not enough with *DCluster*. Thus, Finding a way to automatically adapt the population size to the difficulty of the problem will be one of the topics of our future work.

Figure 7 clearly shows the classification of the six algorithms on the twelve problems. One can notice that our proposed topology outperforms the other topologies on most tested problems, where it was ranked at the first place for 10 problems among 12. From this figure, we can observe that the other topologies are unstable, i.e. their results are mitigated.

5.3 Sensitivity analysis of *DCluster* topology

The sensitivity of *DCluster* topology was studied by varying the value of the number of particles in each cluster N and hence the number of clusters which is equal to $N + 1$, while the other parameters were left fixed to their default values as in Sect. 5.1. The parameter N was studied accordingly, by applying PSO using *DCluster* topology on four benchmark functions [Shifted Rosenbrock(f_1), Shifted

Ackley(f_2), Rotated Rosenbrock(f_5) and Rotated Ackley(f_6)] in 10-D and 30-D, and on five real problems ($F1$, $F4$, $F7$, $F11$ and $F12$, see the section 'Real life problems' in the Appendix). For these test problems, the Mean best was computed from 100 runs of the algorithm.

The parameter N varies between three and eight. Figures 8 and 9 show the value of the best solution found in y-axis, for each value of the parameter N in x-axis. As it can be seen, the choice of the parameter N can influence significantly the performance of the algorithm. Indeed, N has to be fitted according to the problem at hand. We can notice that the results obtained, using different values of N, are mitigated and thus it cannot be determined for which value of N, *DCluster* achieves the best overall performance. However, the analyzed functions achieve the worst results with small or large values of N ($N = 3$ and $N = 8$). Therefore, setting N to four or five is a compromise to obtain reasonable results in all functions, taking into consideration the fixed numbers of evaluations. In fact, increasing the value of N demands an increase in the number of evaluations, according to the convergence graphs that show the slow convergence of *DCluster*.

6 Conclusion

As mentioned by Kennedy (1999), the design of the neighborhood of each particle, and consequently of the topology used, has a great influence on the performance of the PSO. In this paper, we present a new dynamic topology called *DCluster*, based on a combination between the *Four-cluster* and *Fitness* topologies. *DCluster* has achieved good results on literature benchmarks (benchmark functions, unconstrained and constrained real-life problems) in comparison to the commonly used topologies found in literature. Indeed, the experiments showed that none of the tested topologies completely dominates the others, but the proposed topology proved to have the best performances for most of the tested problems. In addition, we showed that *DCluster* gives a good convergence graph for most of the tested problems, for premature convergence of PSO is less probable.

Appendix

See Table 5.

Table 5 Benchmark functions

f	Function name	Test function	Search space
f_1	*Shifted*	$f_1(x) = \sum_{i=1}^{n-1}\left(100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2\right) + f_{bias_1}$	$[-100, 100]^N$
	Rosenbrock	$z_i = x_i - o_{2i}, f^* = 390$	
f_2	*Shifted*	$f_2(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} z_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi z_i)\right) + 20 + e + f_{bias_2}$	$[-32, 32]^N$
	Ackley	$z_i = x_i - o_{3i}, f^* = -140$	
f_3	*Shifted*	$f_3(x) = \sum_{i=1}^{n}\frac{z_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{z_i}{\sqrt{i}}\right) + 1 + f_{bias_3}$	$[-600, 600]^N$
	Griewank	$z_i = x_i - o_{4i}, f^* = -180$	
f_4	*Shifted*	$f_4(x) = \sum_{i=1}^{n}\left(z_i^2 - 10\cos(2\pi z_i) + 10\right) + f_{bias_4}$	$[-5, 5]^N$
	Rastrigin	$z_i = x_i - o_{5i}, f^* = -330$	
f_5	*Rotated*	$f_5(x) = \sum_{i=1}^{n-1}\left(100(y_i^2 - y_{i+1})^2 + (y_i - 1)^2\right)$	$[-100, 100]^N$
	Rosenbrock	$y = M_1 * x, f^* = 390$	
f_6	*Rotated*	$f_6(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} y_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi y_i)\right) + 20 + e$	$[-32, 32]^N$
	Ackley	$y = M_2 * x, f^* = -140$	
f_7	*Rotated*	$f_7(x) = \sum_{i=1}^{n}\frac{y_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{y_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^N$
	Griewank	$y = M_3 * x, f^* = -180$	
f_8	*Rotated*	$f_8(x) = \sum_{i=1}^{n}\left(y_i^2 - 10\cos(2\pi y_i) + 10\right)$	$[-5, 5]^N$
	Rastrigin	$y = M_4 * x, f^* = -330$	
f_9	*Rosenbrock*	$f_9(x) = \sum_{i=1}^{n-1}\left(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2\right)$	$[-100, 100]^N$
f_{10}	*Ackley*	$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	$[-32, 32]^N$
f_{11}	*Griewank*	$f_{11}(x) = \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^N$
f_{12}	*Rastrigin*	$f_{12}(x) = \sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$	$[-5, 5]^N$

Real life problems

F1: The Lennard-Jones atomic cluster problem

The Lennard-Jones (LJ) problem is the simplest form of the molecular conformation problem, which consists of finding a configuration of atoms in a cluster or molecule whose potential energy is minimum.

We present the LJ problem as follows: we denote by K the number of clusters of atoms, $x^i = (x_1^i, x_2^i, x_3^i)$ represents the coordinates of atom i in the three-dimensional space, and let be $X = ((x^1), \ldots, (x^K))$. The LJ potential energy function $v(r_{ij})$ of a pair of atoms (i,j) is given by $v(r_{ij}) = \frac{1}{r_{ij}^{12}} - \frac{1}{r_{ij}^{6}}, 1 \le i,j \le K$, where $r_{ij} = \|x^i - x^j\|$.

We can notice that, with a single pair of neutral atoms, the LJ potential function is classified as a simple unimodal function. In a cluster with a high number of atoms, the interaction between atoms increases. Then we need to calculate the LJ potential function for each pair of atoms in a cluster. Many local minima are found in the resulting rugged energy landscape. The function to be minimized in this application is as follows:

$$\begin{aligned} \text{Min } V_k(x) &= \textstyle\sum_{i<j} v\left(\left\|x_i - x_j\right\|\right) \\ &= \textstyle\sum_{i=1}^{K-1} \sum_{j=i+1}^{K} \left(\frac{1}{\left\|x_i - x_j\right\|^{12}} - \frac{1}{\left\|x_i - x_j\right\|^{6}} \right). \end{aligned} \tag{9}$$

In this work, we use nine atoms for this problem, hence the search space is $[-2,2]^{27}$. The value of the global optimum is $f^* = -24.113360$.

F2: Gear train design problem

A gear train is a set or system of gears arranged to transfer rotational torque from one part of a mechanical system to another. The problem of gear train is a common problem in the field of mechanical engineering, to be designed such that the gear ratio is as close as possible to 1/6.931. For each gear the number of teeth must be between 12 and 60. Since the number of teeth is to be an integer, the variables must be integers. The mathematical model of gear train design is given by:

$$\text{Min} f(x) = \left\{ \frac{1}{6.931} - \frac{T_d T_b}{T_a T_f} \right\}^2 = \left\{ \frac{1}{6.931} - \frac{x_1 x_2}{x_3 x_4} \right\}^2. \tag{10}$$

Bounds: $12 \le x_i \le 60, i = 1,2,3,4$ and x_i should be integers. Where T_a, T_b, T_d and T_f are the numbers of teeth on gears A, B, D and F respectively.

F3: Frequency modulation sound parameter identification

The frequency modulation sound parameter identification is a complex multimodal optimization problem. The problem is to satisfy six parameters a_1, w_1, a_2, w_2, a_3 and w_3 of the frequency modulation sound model given below:

$$\begin{aligned} y(t) = a_1 \times sin(w_1 \times t \times \theta + a_2 \times sin(w_2 \times t \times \theta + a_3 \\ \times sin(w_3 \times t \times \theta))) \end{aligned} \tag{11}$$

with $\theta = (2.\pi/100)$. The fitness function is defined as the sum of square errors between the evolved data and the model data, as follows:

$$f(a_1, w_1, a_2, w_2, a_3, w_3) = \textstyle\sum_{t=0}^{100} (y(t) - y_0(t))^2$$

where the model data are given by the following equation:

$$\begin{aligned} y_0(t) = 1.0 \times sin(5.0 \times t \times \theta + 1.5 \times sin(4.8 \times t \times \theta \\ + 2.0 \times sin(4.9 \times t \times \theta))). \end{aligned} \tag{12}$$

Bounds: $-6.4 \le a_i, w_i \le 6.35$, $i = 1,2,3$.

F4: The spread spectrum radar polyphase code design problem

A famous problem of optimal design arises in the field of spread spectrum radar poly-phase codes. Such a problem is very well suited for the application of global optimization algorithms. The problem under consideration is modeled as a min–max nonlinear non-convex optimization problem in continuous variables and with numerous local optima. It can be expressed as follows:

$$\text{Min} f(X) = \max\{f_1(X), \ldots, f_{2m}(X)\}. \tag{13}$$

where:

$X = \left\{ (x_1, \ldots, x_n) \in R^n \mid 0 \le x_j \le 2\pi, j = 1,2,\ldots,n \right\}$ and $m = 2n - 1$,

with:

$$f_{2i-1}(x) = \sum_{j=i}^{n} \cos\left(\sum_{k=|2i-j-1|+1}^{j} x_k \right) \quad i = 1,2,\ldots,n;$$

$$f_{2i}(x) = 0.5 + \sum_{j=i+1}^{n} \cos\left(\sum_{k=|2i-j|+1}^{j} x_k \right) \quad i = 1,2,\ldots,n-1;$$

$$f_{m+i}(X) = -f_i(X), i = 1,2,\ldots,m.$$

Here the objective is to minimize the module of the biggest among the samples of the so-called autocorrelation function which is related to the complex envelope of the compressed radar pulse at the optimal receiver output, while the variables x_k represent symmetrized phase differences. This problem belongs to the class of continuous min–max global optimization problems. They are characterized by the fact that the objective function is piecewise smooth.

F5: Compression spring design problem

This is a simplified version of a more difficult problem (see Clerc 2006). The function to minimize is:

$$\text{Min} f(x) = \pi^2 \frac{x_2 x_3^2 (x_1 + 1)}{4}, \tag{14}$$

where $x_1 \in \{1, \ldots, 70\}$ with a granularity 1, $x_2 \in [0.6, 3.0]$ and $x_3 \in \{0.207, \ldots, 0.5\}$ with a granularity 0.0001. The best known solution is (7, 1.386599591, 0.292) which gives the fitness value $f^* = 2.6254214578$. This problem has five constraints (Clerc 2006), and a penalty method is used to take them into account.

F6: Perm function

The function to be minimized is (PSC 2012):

$$\text{Min} f(x) = \sum_{k=1}^{5} \left[\sum_{i=1}^{5} (i^k + \beta) \left\{ (x_i/i)^k - 1 \right\} \right]^2. \tag{15}$$

In the domain $x \in [-5, 5]$, the global optimum is (1, 2, 3, 4, 5) which gives the fitness value $f^* = 0$. The value of β is fixed to 10.

F7: Mobile network design problem

In mobile network design, the challenge is to efficiently determine the locations of base control stations, mobile switching centers, and their connecting links for given locations of base transceiver stations. This problem is complex to describe, and the reader can refer to El-Saleh et al. (2009) for more information. The search space is made of binary variables (38 variables) and of continuous ones (four variables). Hence, the problem has 42 dimensions.

F8: Pressure vessel design problem

This problem was proposed by Kannan and Kramer (1994). The objective is to minimize the total cost, including the cost of the material, forming and welding. The mathematical model of pressure vessel optimization problem can be described as follows (Clerc 2006):

$$\begin{aligned} \text{Min} f(x) = {} & 0.6224 x_1 x_3 x_4 + 1.7781 x_2 x_3^2 + 3.1661 x_1^2 x_4 \\ & + 19.84 x_1^2 x_3. \end{aligned} \tag{16}$$

Subject to:

$$g_1(x) = 0.0193 x_3 - x_1 \le 0, \quad g_2(x) = 0.00954 x_3 - x_2 \le 0$$
$$g_3(x) = 750 \times 1728 - \pi x_3^2 \left(x_4 + \frac{4}{3} x_3\right) \le 0$$

where $x_1 \in \{0.0625, \ldots, 12.5\}$ with a granularity 0.0625, $x_2 \in \{0.625, \ldots, 12.5\}$ with a granularity 0.0625, $x_3 \in]0, 240]$ and $x_4 \in]0, 240]$. The best solution is (1.125, 0.625, 58.2901554, 43.6926562) which gives the fitness value $f^* = 7,197.72893$.

F9: Welded beam design problem

The problem is to design a welded beam for minimum cost, subject to some constraints (Ragsdell and Phillips 1976). The problem can be mathematically formulated as follows:

$$\text{Min} f(x) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2). \tag{17}$$

Subject to:

$$g_1(x) = \tau(x) - 13000 \le 0 \quad g_2(x) = \sigma(x) - 30000 \le 0$$
$$g_3(x) = x_1 - x_4 \le 0$$
$$g_4(x) = 6000 - P_c(x) \le 0 \quad g_5(x) = 0.125 - x_1 \le 0$$
$$g_6(x) = \delta(x) - 0.25 \le 0$$
$$g_7(x) = 0.10471 x_1^2 + 0.04811 x_3 x_4 (14.0 + x_2) - 5.0 \le 0$$

where:

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau'' \frac{x_2}{2R} + (\tau'')^2} \quad M = 6000\left(14 + \frac{x_2}{2}\right)$$
$$J = 2\left\{ \sqrt{2} x_1 x_2 \left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2 \right] \right\}$$
$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2} \quad \sigma(x) = \frac{504000}{x_4 x_3^2}$$
$$\delta(x) = \frac{2.1952}{x_3^3 x_4} \quad \tau' = \frac{6000}{\sqrt{2} x_1 x_2}$$
$$\tau'' = \frac{MR}{J} \quad P_c(x) = \frac{4.013(30 \times 10^6)\sqrt{\frac{x_3^2 x_4^6}{36}}}{196}$$
$$\left(1 - \frac{x_3 \sqrt{\frac{30 \times 10^6}{4(12 \times 10^6)}}}{28}\right).$$

With: $x_1, x_4 \in [0.1, 2.0]$ and $x_2, x_3 \in [0.1, 10.0]$.

Best solution: $x^* = (0.205730, 3.470489, 9.036624, 0.205729)$ where $f^* = 1.724852$.

F10: Speed reducer design problem

This problem was modeled by Golinski (1973) as a single-level optimization. The objective is to minimize the speed reducer weight while satisfying a number of constraints imposed by gear and shaft design practices. Mathematically, the problem is specified as follows:

$$\begin{aligned} \text{Min} f(x) = {} & 0.7854 x_1 x_2^2 (3.3333 x_3^2 + 14.9334 x_3 - 43.0934) \\ & - 1.508 x_1 (x_6^2 + x_7^2) \\ & + 7.4777 (x_6^3 + x_7^3) + 0.7854 (x_4 x_6^2 + x_5 x_7^2). \end{aligned} \tag{18}$$

Subject to:

$$g_1(x) = \frac{27}{x_1 x_2^2 x_3} \le 0 \quad g_2(x) = \frac{397.5}{x_1 x_2^2 x_3^2} \le 0$$

$$g_3(x) = \frac{1.93 x_4^3}{x_2 x_3 x_6^4} \le 0 \quad g_4(x) = \frac{1.93 x_5^3}{x_2 x_3 x_7^4} \le 0$$

$$g_5(x) = \frac{1.0}{110 x_6^3} \sqrt{\left(\frac{745 x_4}{x_2 x_3}\right)^2 + 16.9 \times 10^6} - 1 \le 0$$

$$g_6(x) = \frac{x_2 x_3}{40} - 1 \le 0 \quad g_7(x) = \frac{5 x_2}{x_1} - 1 \le 0$$

$$g_8(x) = \frac{1.0}{85 x_7^3} \sqrt{\left(\frac{745 x_5}{x_2 x_3}\right)^2 + 157.5 \times 10^6} - 1 \le 0$$

$$g_9(x) = \frac{x_1}{12 x_2} - 1 \le 0$$

$$g_{10}(x) = \frac{1.5 x_6 + 1.9}{x_4} - 1 \le 0$$

$$g_{11}(x) = \frac{1.1 x_7 + 1.9}{x_5} - 1 \le 0.$$

With: $2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 7.3 \le x_4 \le 8.3, 7.8 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9,$ and $5.0 \le x_7 \le 5.5$. Best solution: $x^* = (3.5, 0.7, 17, 7.3, 7.8, 3.350214, 5.286683)$ where $f^* = 2,996.348165$.

F11: Constraint problem 1

The function to be minimized is:

$$\text{Min} f(x) = 5 \sum_{i=1}^{4} x_i - 5 \sum_{i=1}^{4} {x_i}^2 - \sum_{i=1}^{13} x_i. \tag{19}$$

Subject to:

$$g_1 = 2x_1 + 2x_2 + x_1 0 + x_1 1 - 10 \le 0 \quad g_4 = -8x_1 + x_{10} \le 0 \quad g_7 = -2x_4 - x_5 + x_{10} \le 0$$
$$g_2 = 2x_1 + 2x_3 + x10 + x12 - 10 \le 0 \quad g_5 = -8x_2 + x_{11} \le 0 \quad g_8 = -2x_6 - x_7 + x_{11} \le 0$$
$$g_3 = 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \le 0 \quad g_6 = -8x_3 + x_{12} \le 0 \quad g_9 = -2x_8 - x_9 + x_{12} \le 0$$

Bounds: $0 \le x_i \le 1 \quad (i = 1, \ldots, 9, 13)$, and $0 \le x_i \le 100$ $(i = 10, 11, 12)$. The global optimum is $(1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1)$, where $f^* = -15$.

F12: Constraint problem 2

The function to be minimized is:

$$\begin{aligned}\text{Min} f(x) = {} & x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 \\ & + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 \\ & + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 \\ & + (x_{10} - 7)^2 + 45. \end{aligned} \tag{20}$$

Subject to:

$$g_1 = -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \le 0$$

$$g_4 = -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \le 0$$

$$g_2 = 10x_1 - 8x_2 - 17x_7 + 2x_8 \le 0 \quad g_5 = 5x_1^2 + 8x_2 + (x3 - 6)^2 - 2x_4 - 40 \le 0$$

$$g_3 = -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \le 0$$

$$g_6 = x_1^2 + 2(x_2 - 2)^2 - 2x_1 x_2 + 14x_5 - 6x_6 \le 0$$

$$g_7 = 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \le 0$$

$$g_8 = 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \le 0$$

Bounds: $-10 \le x_i \le 10 \quad (i = 1, 2, \ldots, 10)$. The optimum solution is : $x^* = (2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430574, 1.321644, 9.828726, 8.280092, 8.375927)$, where $f(x^*) = 24.3062091$.

References

Ali M, Pant M, Singh VP (2010) Two modified differential evolution algorithms and their applications to engineering design problems. World J Model Simul 6(1):72–80

Bastos-Filho CJA, Carvalho DF, Caraciolo MP, Miranda PBC, Figueiredo EMN (2009) Multi-ring particle swarm optimization. In: Wellington Pinheiro dos Santos (ed) Evolutionary computation. InTech, Vienna

Cagnina LC, Esquivel SC, Coello CA (2008) Solving engineering optimization problems with the simple constrained particle swarm optimizer. Informatica (Slovenia) 32(3):319–326

Clerc M (2006) Particle swarm optimization. ISTE (International Scientific and Technical Encyclopaedia), London

Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science, MHS'95Nagoya, Japan, pp 39–43

El Dor A, Clerc M, Siarry P (2012) A multi-swarm PSO using charged particles in a partitioned search space for continuous optimization. Comput Optimiz Appl 53(1):271–295

El-Saleh AA, Ismail M, Viknesh R, Mark CC, Chan ML (2009) Particle swarm optimization for mobile network design. IEICE Electron Express 6(17):1219–1225

Engelbrecht AP (2006) Fundamentals of computational swarm intelligence. Wiley, Chichester

Golinski J (1973) An adaptive optimization system applied to machine synthesis. J Eng Ind Trans ASME 8(4):419–436

Homaifar A, Qi CX, Lai SH (1994) Constrained optimization via genetic algorithms. Simulation 62(4):242–253

Hsieh ST, Sun TY, Liu CC, Tsai SJ (2009) Efficient population utilization strategy for particle swarm optimizer. IEEE Trans Syst Man Cybernet B: Cybernet 39(2):444–456

Janson S, Middendorf M (2005) A hierarchical particle swarm optimizer and its adaptive variant. IEEE Trans Syst Man Cybernet B 35(6):1272–1282

Kannan BK, Kramer SN (1994) An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. J Mech Des 116(2):318–320

Kennedy J (1999) Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. In: Proceedings of the 1999 congress on evolutionary computation, vol 3, CEC'99DC USA, Washington, pp 1931–1938

Kennedy J (2000) Stereotyping: improving particle swarm performance with cluster analysis. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp 1507–1512. IEEE

Kennedy J, Mendes R (2002) Population structure and particle swarm performance. In: Proceedings of the 2002 IEEE Congress on Evolutionary Computation, CEC'02Honolulu, HI, USA, pp 1671–1676

Lane J, Engelbrecht A, Gain J (2008) Particle swarm optimization with spatially meaningful neighbours. In: Proceedings of the 2008 IEEE Swarm Intelligence Symposium. Piscataway, NJ, IEEE, pp 1–8.

Li C, Yang S, Nguyen TT (2011) A self-learning particle swarm optimizer for global optimization problems. IEEE Trans Syst Man Cybernet B 42(3):627–646

Liang JJ, Suganthan PN (2005) Dynamic multi-swarm particle swarm optimizer. In: Proceedings of the 2005 IEEE Swarm Intelligence Symposium, pp 124–129, Pasadena, CA, USA

Liang JJ, Suganthan PN (2006) Dynamic multi-swarm particle swarm optimizer with a novel constraint-handling mechanism. In Proceedings of the IEEE Congress on Evolutionary Computation, pp 9–16, Canada

Liu H, Cai Z, Wang Y (2010) Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. Appl Soft Comput 10(2):629–640

Marsaglia G, Zaman A (1993) The KISS generator. Technical report. Department of statistics, Florida State University, Tallahassee, FL, USA

Mendes R, Kennedy J, Neves J (2004) The fully informed particle swarm: simpler. May be better. IEEE Trans Evolut Comput 8(3):204–210

Mendes R, Kennedy J, Neves J (2003) Watch thy neighbor or how the swarm can learn from its environment. In: Proceedings of the 2003 IEEE Swarm Intelligence Symposium, pp 88–94, Indianapolis, Indiana, USA

Nasir M, Das S, Maity D, Sengupta S, Halder U, Suganthan PN (2012) A dynamic neighborhood learning based particle swarm optimizer for global numerical optimization. Inf Sci 209:16–36

Particle swarm central (2012) http://particleswarm.info. Accessed 1 Oct 2014

Ragsdell K, Phillips D (1976) Optimal design of a class of welded structures using geometric programming. J Eng Ind Trans ASME 98(3):1021–1025

Richards M, Ventura D (2003) Dynamic cociometry in particle swarm optimization. In: Proceedings of the joint conference on information sciences, pp 1557–1560, Cary, North Carolina USA

Safavieh E, Gheibi A, Abolghasemi M, Mohades A (2009) Particle swarm optimization with Voronoi neighborhood. In Proceedings of the 14th international CSI computer conference (CSICC2009), pp 397–402, Tehran, Iran

Salomon R (1996) Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions. BioSyst 39:263–278

Shi Y, Eberhart RC (1999) Empirical study of particle swarm optimization. In Proceedings of the 1999 congress on evolutionary computation, CEC'99, vol 3, pp 1945–1950, Washington, DC USA

Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical report. Nanyang Technological University, Singapore

Wang YX, Xiang QL (2008a) Particle swarms with dynamic ring topology. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp 419–423. IEEE

Wang YX, Xiang QL (2008b) Exploring new learning strategies in differential evolution algorithm. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp 204–209. IEEE

Watts DJ (1999) Small worlds : the dynamics of networks between order and randomness. Princeton University Press, Princeton

Watts DJ, Strogatz SH (1998) Collective dynamics of 'small-world' networks. Nature 393:440–442

Zhao SZ, Suganthan PN, Pan QK, Tasgetiren MF (2011) Dynamic multi-swarm particle swarm optimizer with harmony search. Expert Syst Appl 38(4):3735–3742