Handling multi-objective optimization problems with a multi-swarm cooperative particle swarm optimizer

Yong Zhang, Dun-wei Gong, Zhong-hai Ding

A R T I C L E   I N F O

Keywords:
Multi-objective optimization
Multi-swarm
Particle swarm optimization
Species
Escape strategy

A B S T R A C T

This paper presents a new multi-objective optimization algorithm in which multi-swarm cooperative strategy is incorporated into particle swarm optimization algorithm, called multi-swarm cooperative multi-objective particle swarm optimizer (MC-MOPSO). This algorithm consists of multiple slave swarms and one master swarm. Each slave swarm is designed to optimize one objective function of the multi-objective problem in order to find out all the non-dominated optima of this objective function. In order to produce a well distributed Pareto front, the master swarm is developed to cover gaps among non-dominated optima by using a local MOPSO algorithm. Moreover, in order to strengthen the capability of locating multiple optima of the PSO, several improved techniques such as the Pareto dominance-based species technique and the escape strategy of mature species are introduced. The simulation results indicate that our algorithm is highly competitive to solving the multi-objective optimization problems.

1. Introduction

Due to the inherent conflicting nature among the objectives to be optimized, it is still a challenging to solve the multi-objective optimization problems (MOPs). Since the pioneering effort of Schaffer (1985), many well-known evolutionary techniques for the MOPs have been proposed with impressive success (Deb, Pratap, Agarwal, & Meyarivan, 2002; Jaimes & Coello Coello, 2007; Tan, Lee, & Khor, 2002; Zitzler, Laumanns, & Thiele, 2001).

Particle swarm optimization (PSO) (Kennedy & Eberhart, 1995) is a stochastic optimization technique which is inspired by the behavior of bird flock, and is considered as an evolutionary algorithm by its authors (Eberhart & Shi, 1998). Although PSO is relatively new, the relative simplicity, the fast convergence and the population-based feature (Reyes-Sierra & Coello Coello, 2006) have made it a high competitor in solving the MOPs. Some of the existing multi-objective particle swarm optimization (MOPSO) algorithms can be found in Coello Coello, Pulido, and Lechuga (2004), Leong and Yen (2008), Liu, Tan, Goh, and Ho (2007), Moore and Chapman (1999), Tripathi, Bandyopadhyay, and Pal (2007), Wang and Yang (2009), and Wickramasinghe and Li (2009). However, how to improve the diversity of swarm or overcome the local convergence of PSO is still a challenging to research the MOPs (Reyes-Sierra & Coello Coello, 2006).

The purpose of this paper is to develop a multi-swarm particle swarm optimization algorithm for the MOPs. On the one hand, the use of multi-swarm/population has at least two advantages (Jaimes & Coello Coello, 2007): (1) it can improve the population diversity; (2) it is easy to cooperate in hybrid with another search technique/strategy. On the other hand, for a continuous MOP, once we find out a solution set, whose neighbors have covered the true Pareto front of this MOP, it will be easy to produce a well-distributed Pareto optimal set by using a local search method. In fact, such a set can be formed by non-dominated optima of single objective functions of the MOP optimized. Herein, an optimum is called to be non-dominated if its neighbor contains Pareto optimal solutions. Moreover, there have been many effective heuristic algorithms, which are used to find multiple optima for multi-model objective function, such as (Brits, Engelbrecht, & Van den Bergh, 2007; Parrott & Li, 2006).

Inspired by the above idea, this paper presents a multi-swarm cooperative MOPSO algorithm based on the master-slave model. In this algorithm, each slave swarm corresponds to one objective function of the MOP optimized, and is designed to look for non-dominated optima of this objective function. The master swarm saves the non-dominated optima obtained by all the slave swarms into an external archive. And a local MOPSO algorithm is proposed to cover gaps among elements of the archive. Moreover, several improved techniques, such as the species technique based on the Pareto dominance relationship and the escape strategy of mature
species, are also incorporated into our algorithm to improve its search capability. By compared against three highly competitive multi-objective algorithms, the simulation results validate the efficiency of our algorithm.

This paper is organized as follows. Section 2 provides some basic concepts and reviews parts of the related work in the literatures. Section 3 is devoted to describe our proposed algorithm. The efficiency of our algorithm is provided in Sections 4 and 5. Finally, some main conclusions and open problems are given in Section 6.

2. Basic concepts and related work

2.1. Multi-objective optimization problems

For the benefits of readers, we introduce the basic concepts for the multi-objective optimization problems.

Definition 1 (MOPs). Find the vector \( \mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_n^*) \) satisfying

\[
\min \ f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}))
\]

where \( \mathbf{x} = (x_1, x_2, \ldots, x_n) \in \Omega \subset \mathbb{R}^n \) is called the decision variable, the set \( \Omega \) is called the feasible region.

Definition 2 (Pareto dominance). A vector \( \mathbf{u} = (u_1, u_2, \ldots, u_n) \) is said to dominate \( \mathbf{v} = (v_1, v_2, \ldots, v_n) \) (denoted by \( \mathbf{u} < \mathbf{v} \)) if \( \mathbf{u} \) is partially less than \( \mathbf{v} \), \( u_i < v_i \), \( \forall i \in \{1, 2, \ldots, M\} \), and there exists \( i \in \{1, 2, \ldots, M\} \) such that \( u_i < v_i \).

A feasible solution \( \mathbf{x} \) is said to be non-dominated with respect to the set \( \Omega \), if there does not exist another \( \mathbf{x} \in \Omega \) such that \( f(\mathbf{x}) < f(\mathbf{x}) \). Furthermore, the feasible solutions that are non-dominated within the entire search space are called the Pareto optimal solutions, and constitute the Pareto optimal set or the non-dominated set. Objective values of these Pareto optimal solutions constitute the Pareto front of MOP. Unless there is some preference information, the main goal of MOP is to find a Pareto-optimal set, instead of a single optimal solution. The detailed discussion of these basic concepts can be found from Coello Coello, Van Veldhuizen, and Lamont (2002).

2.2. The standard PSO

PSO is inspired of the social behavior of some biological organisms, especially the group's ability of some animal species to locate a desirable position in the given area. In PSO, a swarm consists of a set of particles, each particle represents a potential solution and moves through a multi-dimensional search space to look for a potential solution by tracking two positions. One position is the best position found by the particle itself so far, called the personal best position (Pbest) or the local leader, notified by \( \mathbf{p}_i = (p_{i1}, p_{i2}, \ldots, p_{in}) \). Another is the best position found by neighbors of the particle so far, called the global best position (Gbest) or the global leader, notified by \( \mathbf{g}_i = (g_{i1}, g_{i2}, \ldots, g_{in}) \).

At each iteration, the velocity and the position of a particle are updated by

\[
\begin{align*}
\mathbf{v}_{ij}(t) &= \mathbf{w}\mathbf{v}_{ij}(t) + c_1r_1(p_{ij}(t) - x_{ij}(t)) + c_2r_2(g_{ij}(t) - x_{ij}(t)) \\
x_{ij}(t) &= x_{ij}(t) + \mathbf{v}_{ij}(t)
\end{align*}
\]

where \( j = 1, 2, \ldots, n \), \( i = 1, 2, \ldots, |S| \), \( n \) and \( |S| \) are the number of decision variables and the size of swarm, respectively. The \( c_1 \) and \( c_2 \) are nonnegative constants called acceleration coefficients, \( r_1 \) and \( r_2 \) are two random numbers within \([0, 1]\), \( \mathbf{w} \) is an inertia weight to control particle's exploration capability in the search space (Eberhart & Shi, 1998).

2.3. Multi-objective particle swarm optimization

When extending PSO to the MOP, three design aspects are often considered in terms of the characteristics of PSO.

1. Update of the particles’ leaders. Since the solution of a multi-objective problem consists of a set of equally good solutions, it is evident that the concept of leader traditionally adopted in PSO has to be changed. As we know, the selection of a leader is a key component when designing a MOPSO algorithm. The most common approach is to consider every non-dominated solution as a new leader and then, just one leader has to be selected. Few of these existing methods include the sigma method (Mostaghim & Teich, 2003), the cross-searching strategy (Chiu, Sun, Hsieh, & Lin, 2007), and the diversity-based approaches (Li, 2003; Tripathi et al., 2007). Since the diversity-based approaches are able to prevent the particles from drifting towards some crowded areas, it is becoming more and more desirable in recent years.

2. Retaining the non-dominated solutions obtained. In order to report a good Pareto optimal set at the end of algorithm, an external archive with maximal capacity is often adopted to store those non-dominated solutions found along all the search process. Laumanns et al. (2002) introduced the \( \varepsilon \)-dominance method to control the archive size and help to reduce computational cost. Coello Coello et al. (2004) used a variation of the adaptive grid to prune the archive. Recently, Shubham, Panigrahi, and Kumar (2008) presented a fuzzy clustering-based particle swarm algorithm (FCPSO), where a fuzzy clustering technique and a niching mechanism were introduced to manage the archive size and to update the Gbests, respectively.

3. Promoting the diversities of particles. Leong and Yen (2008) presented a dynamic multiple-swarm MOPSO (DMOPSO), which incorporates multiple swarms to promote diversity of the particles, and introduces a population-growing strategy and a population-declining strategy to reduce computational cost. Tripathi et al. (2007) presented a time variant multi-objective particle swarm optimization (TV-MOPSO) that allows the vital parameters (inertia weight and acceleration coefficient) to change with iterations. In addition, the incorporation of genetic operators such as mutation (Coello Coello et al., 2004; Tripathi et al., 2007) and perturbation (Mostaghim & Teich, 2003) has greatly enhanced the exploration capability of the MOPSO algorithm.

3. Description of the proposed algorithm

This section describes our proposed algorithm, the multi-swarm cooperative MOPSO algorithm. In this algorithm, each slave swarm corresponds to one objective function of the MOP optimized, and is developed to find out non-dominated optima of this objective function. While a local MOPSO algorithm working in the master swarm concentrates its effort on covering gaps among the archive elements. Like most of the existing MOPSO algorithms, the external archive with maximal capacity is adopted to store non-dominated optima coming from the slave swarms and non-dominated solutions coming from the master swarm.

Fig. 1 shows the model of our algorithm. In each slave swarm, an improved PSO is run to find out the non-dominated optima of corresponding objective function. In the master swarm, the local MOPSO algorithm is run to produce a good Pareto optimal set for the MOP optimized. At each iteration, each slave swarm immigrates
non-dominated optima obtained to the archive. The archive receives these non-dominated optima, and then assigns its elements, which locate at areas with sparse solutions, as the Gbest of particles.

By using the above method, different swarms are able to concentrate their efforts on exploring different regions of the search space. In other words, each slave swarm concentrates its effort on looking for itself non-dominated optima. Based on those non-dominated optima and Pareto optimal solutions obtained, the local MOPSO algorithm is able to evolve a diverse and well-distributed nearly optimal Pareto front by covering gaps among the archive elements. Thus, the proposed algorithm leads to cooperation search among the swarms to the MOP optimized.

3.1. The improved PSO

In order to quickly find out all the non-dominated optima for each single objective function, an improved PSO is presented in this subsection. In the improved algorithm, a Pareto dominance-based species technique and an escape strategy for mature species are introduced to strengthen the capability locating multiple optima of PSO. Taking the mth slave swarm \( S_m(t) \) as example, where \( t \) is the current iteration times, the two improvements are described as follows.

(1) The Pareto dominance-based species technique

In order to help PSO locate multiple non-dominated optima simultaneously, the Pareto-based species technique inspired by the work from Parrott and Li (2006) is proposed to divide \( S_m(t) \) into multiple species. By this method, different species can evolve to different optima simultaneously. The Pareto-based species technique consists of two main parts: identifying species seeds and dividing species.

Fig. 2 summarizes steps for identifying the species seeds. This technique takes \( L_{sorted} \) as an input, a list containing all particles sorted in decreasing order of \( f_m \) values. In the beginning, the set \( PS \) that is used to store species seeds found is set to \( \emptyset \). All the particles are checked in turn (from best to least-fit) against the species seeds found so far. If a particle does not fall within the radius \( r_i \) of all the seeds of \( PS \), and is non-dominated with respect to the \( PS \), then it will become a new seed and be added to the \( PS \). In this paper we set

\[
r_i = k_0 \frac{\sum_{i=1}^{n} (ub(i) - lb(i))^2}{n}
\]

where \( ub(i) \) and \( lb(i) \) are the upper and the lower boundary of the \( i \)th decision variable, the parameter \( k_0 \) is a constant within [0, 1]. By applying the above technique, different species seeds can be identified for multiple species. Then, each particle in the \( S_m(t) \) is checked in turn against all the species seeds found. If a particle falls within the radius \( r_i \) of one seed, then this particle will be added to corresponding species of that seed. Finally, all the particles that do not belong to the existing species compose a pseudo species.

(2) The escape strategy of mature species

In the algorithm, it is common case that a species seed had converged to an optimum before other species converged to optimum. In other words, since a species converged to an optimum, it does not contribute further to exploring new optima. To overcome the above disadvantage, an escape strategy for the mature species is adopted herein. For a species seed, if its velocity (i.e., the velocity of its corresponding particle) is close to zero at sequential \( stop\_lim \) iterations, then all particles that belong to the same species as this species seed are forced to re-initialize their \( Pbest \) and \( Gbest \) randomly.

Based on the above two improvements, the generic steps of the improved PSO are described as follows:

Step 1: \( t = 0 \), initialize the swarm \( S_m(0) \) on the search space, and evaluate each particle in \( S_m(0) \).

Step 2: Initialize the velocity and the \( Pbest \) of each particle. For the \( i \)th particle \( x_i \), set \( P_{bi} = x_i \), \( V_{i} = 0 \), where \( i = 1, 2, \ldots, |S_m(0)| \), \( |S_m(0)| \) is the size of \( S_m(0) \).

Step 3: WHILE the maximum iteration \( T_{max} \) has not been reached, DO

Step 3.1: Identify species seeds and divide the swarm \( S_m(t) \) into multiple species via the Pareto dominance-based species technique.

Step 3.2: Migrate the above species seeds to the archive.

Step 3.3: Assign the \( Gbest \) for each particle. Since a species seed is the fittest solution compared to all of the particles in the species, all the particles within the same species can be made to follow the species seed as the \( Gbest \). This attracts particles within the same species to the areas that make them even fitter. In addition, for each particle in the pseudo species, a species seed is selected randomly as its \( Gbest \).

Step 3.4: If the velocity of a species seed is close to zero at sequential \( stop\_lim \) iterations, then the escape strategy is activated.

Step 3.5: Update the position and the velocity of every particle by using Eq. (2). In this paper, a linearly decreasing inertia weight \( w = 0.9 - 0.7 \times t/T_{max} \) is recommended, while the coefficients \( c_1 \) and \( c_2 \) take the same value of 1.5.
Step 3.6: Maintain the particles within the search space in case they go beyond their boundaries. When the particle position goes beyond its boundaries, it takes the value of its corresponding boundary (either the lower or the upper boundary).

Step 3.7: Update the Pbest for every particle. The winner between the new particle and the old Pbest with respect to the mth objective value will be the new Pbest.

Step 3.8: Increment the loop counter, \( t \rightarrow t + 1 \).

### 3.2. Update the external archive

Like most of the existing MOPSO algorithms, in this paper the external archive with maximal capacity is adopted to store non-dominated solutions obtained along all the search process. At each iteration, the archive gets updated with non-dominated optima from the slave swarms, and non-dominated solutions from the combined population of the master swarm and the archive. If the size of the archive exceeds the maximal capacity \( A_{\text{lim}} \), it is truncated on the basis of the density of elements. The crowding density of an element is defined as:

\[
\text{Crowding density} = \frac{\text{Distance to nearest non-dominated solution}}{\text{Distance to nearest dominated solution}}
\]

The proposed algorithm uses the crowding density to update the external archive.
The crowding distance (Deb et al., 2002) is adopted to estimate each element’s density in the proposed algorithm. By this approach, the most sparsely spread \( A_{lim} \) elements, i.e., \( A_{lim} \) elements with the largest crowding distance values, are retained in the archive.
3.3. The local MOPSO algorithm

The improved PSO above is designed to provide a MOP with an approximate Pareto front which is composed of non-dominated optima and part Pareto optimal solutions, but not a well-distributed Pareto optimal front. To cover gaps among the archive elements and produce a good Pareto front, in this subsection a local MOPSO algorithm is designed. A new method for updating the particle’s position and an established technique for updating the Pareto optimal front. To cover gaps among the archive elements and part Pareto optimal solutions, but not a well-distributed Pareto front which is composed of non-dominated optima and part Pareto optimal solutions, but not a well-distributed Pareto front.

(1) Update of the global best position. The global best position is the best solution obtained by neighbors of a particle so far. When solving a single-objective problem, it is completely determined once a neighborhood topology is established. However, in the case of MOPs, the conflicting nature of multiple objectives makes the choice of a single optimum solution difficult. To resolve this problem, we update the Gbest based on the crowding distance mentioned above. At every iteration, the density value (i.e., the crowding distance value) of every element in the archive is calculated first. Then, the binary tournament with these crowding distance values is done to select the Gbest for each particle from the archive. The higher crowding distance value signifies the better solution.

(2) Update of the particle position. In order to explore solutions that locate in areas with sparse solutions, a Gaussian sampling based on the Gbest is used to replace Eq. (2):

\[ x_i = \begin{cases} N(G_{bi}, |P_{bi} - G_{bi}|) & \text{if } U(0,1) < 0.5 \\ G_{bi} & \text{otherwise} \end{cases} \]  

(4)

Compared with Eqs. (2) and (4) generates a new position at neighbor of Gb by using the Gaussian sampling. Since the Gbest of each particle is selected from the archive at random, the exploiting to the Gbest does not result in losing diversity of the swarm. In addition, this method does not make use of the standard control parameters of PSO (i.e., inertia weight, acceleration coefficients, and velocity clamping) to update the particle’s position. This makes it unnecessary for the MOPSO to perform a fine tuning on these control parameters in order to pursue good performance.

Based on the above two methods, the local MOPSO algorithm is described as follows.

Step 1: Initialize the position, the velocity and the Pbest of each particle in the master swarm, and set the archive as ∅.
Step 2: Update the archive based on the method proposed in Section 3.2.
Step 3: Update the Gbest for each particle based on the method proposed in Section 3.3.
Step 4: Generate offspring particles via formula (4).
Step 5: Update the Pbest for each particle. The winner between the new particle and the old Pbest with respect to all the objective functions will be the new Pbest.
Step 6: Judge whether the termination condition is satisfied or not. If yes, stop this algorithm; otherwise, go to step 2.

4. Comparison of results

4.1. Test problems

In the context of multi-objective optimization, the benchmark problems must pose sufficient difficulty to impede the ability of MOPSO. In this paper seven benchmark problems (KUR, Quagliarella, ZDT1, ZDT2, ZDT3, ZDT4 and DTLZ1) that have some recognized features are selected. KUR has a disconnected Pareto optimal set and a disconnected Pareto front. It exploits the algorithm’s ability to search for all of the disconnected regions and to maintain a uniform spread on those regions. Difficulty of Quagliarella is to find out the whole Pareto front in many local segments. ZDT1 and ZDT2 have a convex Pareto front and a nonconvex Pareto front, respectively. ZDT3 has a Pareto front composed of five discontinuous convex regions. These three test problems challenge the capability of an algorithm to find and produce a quality spread of the Pareto front. ZDT4 has a continuous and convex Pareto front, but its search space contains \((21^{10-1} - 1)\) local Pareto-optimal fronts. The last one DTLZ1 is a three-objective optimization problem. Its Pareto front is a linear hyperplane in the first quadrant (i.e., an equilateral triangle in 3D space for three fitness functions with only positive values, 0.0–0.5). Its search space contains \((11^{10-2} - 1)\) local Pareto-optimal fronts, each of which can attract an MOPSO (Deb, Thiele, Laumanns, & Zitzler, 2001).

Many researchers such as the authors in Li (2003), Liu et al. (2007), Mostaghim and Teich (2003), and Tripathi et al. (2007) have applied these problems to examine their proposed algorithms. The number of decision variables is set to 3, 10, 30 and 7 for KUR, Quagliarella, ZDT4 and DTZL1, respectively, while set to 100 for the rest test problems instead of the standard number 30.
4.2. Performance metrics selected

In our comparative study, we adopt three performance metrics known to multi-objective optimizers. To measure the distribution of solutions throughout the Pareto optimal front found so far, we use the spacing metric (SP) (Schott, 1995). A value of zero for this metric indicates all members of the Pareto front currently available are equidistantly spaced. To evaluate the closeness of the Pareto front obtained and the true Pareto front (for short, the true PF), we use the generational distance (GD) (Van Veldhuizen, 1999). A value of zero for this metric indicates that all the solutions obtained belong to the true Pareto optimal set, while any other value indicates how far they are from the true Pareto front of test problem. To show the percentage of solutions (from the Pareto optimal front found so far) that are not members of the true PF, we use the error ratio (ERR) (Van Veldhuizen & Lamont, 1998). A value of zero for this metric indicates all the solutions obtained by an algorithm belong to the true Pareto front of test problem. Note that, if Euclidean distance between a solution and its nearest member in the true Pareto front is less than 0.01 in the objective space, then this solution is accepted as members of the true Pareto optimal set.

4.3. Selected algorithms and parameter settings

Two MOPSO algorithms and one MOEA are selected for the performance comparison. They are the MOPSO algorithm proposed by Coello Coello et al. (2004) (CMOPSO), the TV-MOPSO algorithm (Tripathi et al., 2007) and the NSGA-II algorithm (Deb et al., 2002). The parameter configurations for all the selected algorithms are summarized in Table 1. In our algorithm, the same species parameter $k_0$ is taken for all single objective functions.

4.4. Simulation results and analysis

The results with respect to different performance metrics are analyzed in this subsection. Table 2 shows comparison results of the four algorithms for seven test functions, while Fig. 3 shows the graphical results produced by the four algorithms where the
bold values in the Tables 2 and 3 are the best results obtained for each test problem.

For KUR with disconnected Pareto front, it can be seen from Table 2 that the average performance of our algorithm is the best with respect to all the three metrics. NSGA-II has the worst average performance with respect to all the three metrics. In addition, as Fig. 3(a) shows, a set solutions obtained by NSGA-II failed to converge to all the three true disconnected Pareto fronts.

For Quagliarella with disconnected Pareto optimal set and continuous Pareto front, NSGA-II failed to find the true Pareto front even its approximation as reflected by the ERR values. CMOPSO has the best performance with respect to GD and ERR, but our algorithm has the best performance with respect to SP. Moreover, a set solutions produced by our algorithm got the best extent as Fig. 3(b) shows.

5. Sensitivity analysis

This section performs an extensive analysis about the impact of the parameters on the performance of our algorithm.

5.1. Experiment 1

The experiment is designed to determine whether the local MOPSO adopted really played an important role in the proposed algorithm. Compared the algorithm with the local MOPSO to the one without the local MOPSO, Table 3 shows results with respect to the three metrics. It can be seen that the use of the local MOPSO produced an obvious improvement for solutions of KUR and ZDT3, with respect to all the three metrics. This behavior is due to the high exploitation capability of the local MOPSO to promising areas. Fig. 4 shows the graphical results produced by the algorithm without the local MOPSO.

5.2. Experiment 2

The proposed algorithm uses the specified species radius $r_s$ to determine species and species seeds, therefore, it is understandable that the species parameter $k_0$ plays a crucial role in the performance of our algorithm. This experiment performs an extensive analysis about the impact of $k_0$ on the performance of our algorithm, where the species parameter $k_0$ is allowed to vary from 0.02 to 1.

Fig. 5 shows results with different $k_0$ values. For KUR and ZDT3, it can be observed that the average of SP was not sensitive to the varying of $k_0$, but the averages of GD and ERR both got improvement by setting a large value for $k_0$. Explanation for these results is possibly that the more species signifies the better performance of our algorithm to tackle the problems with several disconnected Pareto fronts. For ZDT1, since it has a connected Pareto optimal set, the average performance of our algorithm was not sensitive to the varying of $k_0$ with respect to all the three metrics.

However, it is worth noting for DTZL1 that a large value of $k_0$ provided better average results than a small one with respect to the three metrics, as shown in Fig. 5. The proposed algorithm got a good performance when $k_0$ took a value not less than 0.6. Explanation for these results has to do with the characteristics of DTZL1. Since the search space of DTZL1 contains $11^5 - 1$ local Pareto fronts, it is very easy that a species with few particles is attracted to these local Pareto fronts. And, it is difficult to find the only global Pareto front from such many local Pareto fronts by re-initializing the mature species randomly (in this case, our algorithm is similar to a random search algorithm). On the contrary, a species with more particles is able to exploit sufficiently its neighbor before be-

Table 3

Results of experiment 1 for the test problems selected.

<table>
<thead>
<tr>
<th></th>
<th>KUR</th>
<th>ZDT3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The proposed algorithm (no)</td>
<td>The proposed algorithm (yes)</td>
</tr>
<tr>
<td>GD</td>
<td>0.0136</td>
<td>0.0088</td>
</tr>
<tr>
<td></td>
<td>0.0014</td>
<td>0.0007</td>
</tr>
<tr>
<td>SP</td>
<td>0.0858</td>
<td>0.0728</td>
</tr>
<tr>
<td></td>
<td>0.0175</td>
<td>0.0145</td>
</tr>
<tr>
<td>ERR</td>
<td>0.2951</td>
<td>0.1302</td>
</tr>
<tr>
<td></td>
<td>0.0423</td>
<td>0.0095</td>
</tr>
</tbody>
</table>
comes a mature species. So a large size $k_0$ may lead to the trade-off of the exploration and exploitation of our algorithm.

6. Conclusions

In this paper, a new multi-swarm cooperative multi-objective particle swarm optimization algorithm has been proposed. In order to improve its performance, several improved techniques such as the Pareto dominance-based species technique, the escape strategy of mature species and the local MOPSO algorithm have also been introduced. The comparative study showed that the proposed algorithm can produce solution sets that are highly competitive with respect to convergence, diversity, and distribution, for all the benchmark test problems.

One weakness of the proposed algorithm is about the settings of species radius $r_s$. As mentioned above, the species radius plays a critical role in the performance of our algorithm. However, in this paper we set the same species parameter $k_0$ for all the single objective functions that belong to the same MOP. It is certainly not an optimal choice, although our algorithm provided better simulation results. Deb and Goldberg (1989) have introduced a method to compute the value of niche radius, but it needs a prior knowledge about the number of optima. More studies are required to investigate the self-adaptation method to adjust species radius.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant No. 61005089 and the National Natural Science Foundation of Jiangsu Province Grant No. BK2008125.

References


