International Journal of Computational Intelligence Systems, Vol.3, No. 5 (October, 2010), 590-600

A scalable coevolutionary multi-objective particle swarm optimizer

Xiangwei Zheng[1,2)*], **Hong Liu**[1,2)]
1) School of Information Science and Engineering, Shandong Normal University, Jinan 250014, P.R. China
2) Shandong Provincial Key Laboratory for Distributed Computer Software Novel Technology, Jinan 250014, P.R. China
E-mail: xwzhengcn@gmail.com, hongliu@sdnu.edu.cn
*Corresponding author

Received: 16-07-2009
Accepted: 15-10-2010

Abstract

Multi-Objective Particle Swarm Optimizers (MOPSOs) are easily trapped in local optima, cost more function evaluations and suffer from the curse of dimensionality. A scalable cooperative coevolution and ε-dominance based MOPSO (CEPSO) is proposed to address these issues. In CEPSO, Multi-objective Optimization Problems (MOPs) are decomposed in terms of their decision variables and are optimized by cooperative coevolutionary subswarms, and a uniform distribution mutation operator is adopted to avoid premature convergence. All subswarms share an external archive based on ε-dominance, which is also used as a leader set. Collaborators are selected from the archive and used to construct context vectors in order to evaluate particles in a subswarm. CEPSO is tested on several classical MOP benchmark functions and experimental results show that CEPSO can readily escape from local optima and optimize both low and high dimensional problems, but the number of function evaluations only increases linearly with respect to the number of decision variables. Therefore, CEPSO is competitive in solving various MOPs.

Keywords: Multi-objective optimization; Scalable; Cooperative coevolution; ε-dominance; MOPSO(multi-objective particle swarm optimizer)

1. Introduction

Multi-objective Optimization Problems (MOPs) are a class of problems frequently encountered in various fields of science and technologies. Such problems can be very complex when certain pragmatic functions and specific model constraints come into place. Traditional methods, such as mathematical programming, are robust and have been proved their effectiveness in handling a variety of common MOPs. However, such techniques have been found to encounter difficulties such as easily getting trapped in local optima, intolerable computational complexity, and inapplicable to certain kinds of objective functions.[1] To overcome these shortcomings, heuristic optimization techniques have been developed, among which Multi-Objective Particle Swarm Optimizers (MOPSOs) are especially promising.[2-3]

Researches on MOPSO is grounded on the successful application of Particle Swarm Optimizer (PSO)[4], a population based stochastic optimization technique developed by Kennedy and Eberhart in 1995[5], and the fact that several improved PSOs are proven to produce very good results at a very low computational cost. Moore and Chapman proposed the first extension of PSO strategy for solving multi-objective problems in an unpublished manuscript in 1999[6], which is recognized as the origin of MOPSOs. Since this earliest attempt, special interest in extending PSO have bursted among researchers, and over 30 different proposals of MOPSOs have been reported in literature thus far. However, all these MOPSOs are more or less suffered from severe drawbacks including being easily trapped in local optima, unbearable number of function evaluations and the curse of dimensionality.

Recently coevolution has been studied by many researchers and shown success to complex and unstructured problems[7], and ε-dominance based archive strategy has also demonstrated satisfactory convergence and distribution properties[8], which inspires our research.

In this paper, a scalable Cooperative Coevolution and ε-dominance based multi-objective Particle Swarm Optimizer (CEPSO) is proposed to counter the above mentioned MOPSO disadvantages. In CEPSO, the MOPs are decomposed with respect to their decision variables and are optimized by cooperative coevolutionary subswarms. A uniform distribution mutation operator is adopted to avoid premature convergence. All subswarms share an external archive based on ε-dominance, which is also served as a leader set. Collaborators are selected from the archive and used to construct context vectors in order to evaluate particles in a subswarm. CEPSO is tested on several classical MOP benchmark functions and experimental results show that CEPSO can readily escape from local optima and optimize both low and high dimensional problems while the number of function evaluations just increases linearly with respect to the number of decision variables.

The remainder of this paper is organized as follows: Section 2 describes basic concepts of MOPs; Section 3 reviews MOPSOs and related work on coevolutionary evolutionary algorithms; Section 4 details the proposed CEPSO, including problem decomposition, selection of collaborators, flow chart and the algorithm in pseudo code; In Section 5, experimental results on several benchmark functions are presented and discussed; And finally Section 6 concludes the paper and points out possible future work.

2. Basic concepts

In MOPs, commonly a set of non-dominated solutions are generated instead of a single recommended solution. According to the concept of non-dominance, also referred to as Pareto optimal, a solution to a multiobjective problem is nondominated, if no objective can be improved without worsening at least one other objective. Without loss of generality, we define the MOPs in eq. (1).

$$\textbf{Min } f(x) = [f_1(x), f_2(x), \ldots, f_k(x)] \tag{1}$$

Where x is the decision vector and f_i is the *ith* objective function, where $i=1,\ldots,k$. To describe the concept of Pareto optimal in MOPs, several definitions must be given first.

Definition 1 (Pareto dominance). A decision vector x_0 dominates x_1, denoted as $(x_0 \prec x_1)$, iff

$f_i(x_0) \le f_i(x_1)$, $\forall i = 1, 2, \ldots, k$

$f_i(x_0) < f_i(x_1)$, $\exists i \in \{1, 2, \ldots, k\}$

Definition 2 (Pareto optimal or Pareto nondominance). A decision vector x_0 is Pareto optimal iff $\neg\exists x_1$: $x_1 \prec x_0$

Definition 3 (Pareto optimal set). Pareto optimal set is a set containing all the Pareto optimal vectors. $P_s = \{x_0 \mid \neg\exists x_1 \prec x_0\}$

Definition 4 (Pareto Front). It is defined as $PF=\{f(x)=(f_1(x), f_2(x), \ldots, f_k(x)) \mid x \in P_s\}$

3. Related work

3.1. *MOPSO*

PSO is a relatively new population-based stochastic optimization technique developed by Kennedy and Eberhart in 1995.[5] PSO emulates swarm behavior of insects, animals herding, birds flocking, and fish schooling, in which collaborative search for food exhibits a potential computational model. Each member in the swarm adapts its search patterns by learning its own experiences and other members'. These phenomena have been studied scientifically and a relevant computing model, the so-called Partical Swarm Optimization or PSO has been constructed, which can be described informally as follows.

A member in the swarm, called a particle, represents a potential solution which is a point in the search space. The global optimum is regarded as the location of food. Each particle uses both an adaptable fitness value and an adaptable velocity to adjust its flying direction according to the best experiences of the swarm to search for the global optimum in a *n*-dimensional solution space.

The trajectory of each individual in the search space is adjusted by dynamically altering its velocity according to its own flying experiences as well as the others'. The position vector and the velocity vector of the *ith* particle in the *n*-dimensional search space can be represented as $x_i=(x_{i1}, x_{i2}, \ldots, x_{in})$ and $v_i=(v_{i1}, v_{i2}, \ldots, v_{in})$ respectively. Given a user defined fitness function, denote the best position obtained by the best fitness value of a particle at time t as $p_i=(p_{i1}, p_{i2}, \ldots, p_{in})$, which is named as *pbest*, further denote the fittest particle found so far at time t as $p_g=(p_{g1}, p_{g2}, \ldots, p_{gn})$, which is named as *gbest*, then, the new velocity and position of particle i at time t+1 can be calculated using eq. (2).

$$\begin{cases} x_i(t+1) = x_i(t) + v_i(t+1) \\ v_i(t+1) = v_i(t) + c_1 r_1 (p_i - x_i(t)) + c_2 r_2 (p_g - x_i(t)) \end{cases} \tag{2}$$

Where c_1 and c_2 are constants known as acceleration coefficients, and r_1 and r_2 are two separately generated uniformly distributed random numbers in range [0,1].

Successful applications of PSO encouraged its introduction to the field of MOPs. Up till now, there have been dozens of representative MOPSOs reported in the literature. Moore and Chapman developed the first MOPSO based on Pareto dominance[6], in which they especially emphasized on the importance of performing both an individual and a group search. The individual best (*pbest*) of all the non-dominated solutions found in the trajectory of a particle is put into a list, and the *pbest* selection is performed on the list. Mostaghim and Teich proposed the *sigma* method for finding the best local guides for each particle of the population[9], which has been implemented and compared with the method that uses the strategy of an existing MOPSO for finding the local guides. Fieldsend et al. employed an order applied to members of non-dominated sets according to the recent dominated tree data structure to facilitate the selection of the best global individual for each member of the swarm, in order to direct their velocities.[10] Coello et al. incorporated Pareto dominance into particle swarm optimization in order to allow this heuristic to handle problems with several objective functions.[11] The algorithm uses a secondary repository of particles that is later used by other particles to guide their own flight. A special mutation operator is incorporated to enrich the exploratory capabilities of the algorithm. Li X. proposed Non-dominated Sorting Particle Swarm Optimizer (NSPSO) for multi-objective optimization.[12] NSPSO extends the basic form of PSO by making a better use of particles' individual bests and offspring and comparing all particles' individual bests and their offspring in the entire population. Sierra et al.

proposed a new multi-objective particle swarm optimizer based on Pareto dominance and a crowding factor to filter out the list of available leaders.[13] They use different mutation operators to act on different subdivisions of the swarm, and they also incorporate the ε-dominance concept to fix the size of the set of final solutions produced by the algorithm.

3.2. *Coevolution*

Generally speaking, the first attempt to study coevolution is attributed to the cooperative GA by Potter and De Jong in 1994. They proposed a general model for coevolution of cooperative species.[14] The model was instantiated and tested in the domain of function optimization and compared with a traditional GA-based function optimizer, and the results were encouraging. They also suggested ways to improve the performance of GA and other EA-based optimizers and a new approach to evolving complex structures such as neural networks and rule sets.

Research on coevolutionary MOEAs based on GA has also been reported. Tan KC et al. proposed a cooperative coevolutionary algorithm (CCEA) for multi-objective optimization[7], which applies the divide-and-conquer approach to decompose decision vectors into smaller components and evolves multiple solutions in the form of cooperative subpopulations. CCEA is capable of maintaining archive diversity in the evolution and distributing the solutions uniformly along the Pareto front. Iorio AW and Li X proposed a cooperative coevolutionary algorithm into which a novel collaboration formation mechanism was incorporated.[15] This approach encouraged rewarding of components participating in successful collaborations from each sub-population. Sierra MR et al proposed a multiobjective coevolutionary algorithm[16], which concentrated the search effort on promising regions that arose during the evolutionary process as a product of a clustering mechanism applied to the set of decision variables corresponding to the known Pareto front. Tan TG et al. proposed a new algorithm which integrates cooperative coevolutionary and the Strength Pareto Evolutionary Algorithm 2 (SPEA2).[17] They conducted comprehensive empirical tests for cooperative coevolution using an evolutionary multi-objective algorithm for 3-dimensional problems.

Coevolution in PSO has also attracted a number of researchers. van den Bergh et al. proposed a cooperative particle swarm optimizer (CPSO) for single-objective optimization problems[18], which employed cooperative behavior to significantly improve the performance of the original algorithm. This is achieved by using multiple swarms to optimize different components of the solution vector cooperatively. Application of the new PSO algorithm on several benchmark optimization problems showed remarkable improvement on performance over the traditional PSO. Furthermore, multipopulations have been applied to improved PSO to locate multi-optima. Iwamatsu et al. presented further simplification and improvement of a modified particle swarm optimizer called the Multi-Species Particle Swarm Optimizer (MSPSO).[19] MSPSO divided the particle swarm spatially into multiple clusters, called species, in a multi-dimensional search space. Each species was responsible for exploring a specific area of the search space and trying to find out the global or local optima of that area. Seo et al. proposed a new algorithm for the multimodal function optimization[20], and named it as multigrouped particle swarm optimization (MGPSO), which had a unique advantage in searching superior peaks of a multimodal function when the number of groups is N.

Best to our knowledge, no research on cooperative coevolutionary MOPSOs for high dimensional problems has been found in literature up till now. However, researches on cooperative algorithms for large scale optimization problems were conducted several years ago. Xin Yao et al. proposed the coevolution of FEP for single-objective optimization, namely FEPCC.[21] FEPCC showed that coevolution could be used quite simply for enhancing the performance of existing algorithms. The time used by FEPCC to find a near optimal solution appeared to increase linearly with respect to the dimensionality. Zhenyu Yang et al. proposed two new efficient DE variants, named DECC-I and DECC-II[22], for high-dimensional optimization (up to 1000 dimensions), which were based on a cooperative coevolution framework incorporated with several novel strategies.

4. A scalable Cooperative Coevolution and ε–dominance based multi-objective Particle Swarm Optimizer

4.1. *Problem decomposition and selection of collaborators*

In our study, problem decomposition is conducted with respect to decision variables, which is simple and easy for automatic decomposition. Each decision variable corresponds to a subswarm, and these subswarms coevolve individuals they contain. For,

$$\min F(x) \tag{3}$$

Where $x=(x_1, x_2, \ldots, x_n)$, i.e. x includes n decision variables, n subswarms are involved in the coevolution.

Frans van den Bergh et al. pointed out that if some of the components in the vector were correlated, they should be grouped in the same subswarm, since the independent changes made by the different subswarms would have a detrimental effect on correlated decision variables.[18] This results in consequence that some subswarms have m decision variables and others having l decision variables, which can be easily accommodated in the framework presented above. Unfortunately, it is not always known in advance how the components are related, so the simple method above is often used and the total particle number is exactly the sum of particles in each subswarm.

On the other hand, collaboration method selection for CEPSO is among the most important tasks, because collaboration method influences algorithm's performance on convergence, diversity, premature and so on. Although the available choices are usually problem specific, two methods are frequently used in practice. The first is simply to select the best individual of the other subswarms and construct context vectors(individual) to evaluate particles in a subswarm, which is greedy and might not be satisfactory in some cases. The second is to select individuals randomly from other subswarms and construct context vectors, which may slow the convergence rate because the selective pressure is reduced.

In the proposed CEPSO, the first method is adopted, since certain experiments showed that it is more robust when no information about problems is available.[23]

4.2. *Velocity update equation and Mutation operator*

By analyzing the random variable in velocity update equation, Maurice Clerc and Kennedy pointed out that some modifications could improve the performance of PSO[24], so a new equation is proposed in this paper:

$$\begin{cases} r = r_1 + r_2 \\ vel = iwt(i) \times vel + c_1 \dfrac{r_1}{r}(p_i - x_i) + c_2 \dfrac{r_2}{r}(p_g - x_i) \end{cases} \tag{4}$$

That means that

$$c_1 \frac{r_1}{r} + c_2 \frac{r_2}{r} = c_1 \tag{5}$$

When $c_1 = c_2$, this equation can be applied in more algorithms and can also improve the performance of MOPSO.[25]

Tournament selection is employed for the leader selection to keep uniform distribution of the Pareto solutions. Because all the solutions in ε-dominance based external archive are nondominated, only crowding distance (CD) [26] is compared to determine which particle becomes a leader when tournament size is set to 2.

PSO converges relatively rapidly in the first part of the search and then slows down or stagnates. This behavior has been attributed to the loss of diversity in the swarm, which can lead to the whole swarm being trapped in a local optimum and hard to escape. Therefore mutation operator is often introduced in PSO to change the stagnation. Because the global best individual attracts all members of the swarm, it is possible to lead the swarm away from a current location by mutating a single individual if the mutated individual becomes the new global best.[27] This mechanism potentially provides a means that can readily escape from local optima and speed up the search remarkably. In CEPSO, a simple mutation operator based on uniform distribution is incorporated, which randomly generate certain new particles in the range of the decision variable and replace some current particles, that is

$$pop_i = uniform(range(pop_i)) \tag{6}$$

Where pop_i is the *ith* subswarm, *uniform()* is a uniform distribution function and *range()* is a function to calculate ranges of each decision variable.

The mutation operator is applied with probability p_m.

4.3. *ε-dominance based archive strategy*

Convergence and diversity are two main metrics for MOEAs. However, existing MOEAs either focus on convergence or focus on a good distribution of solutions but can not achieve both goals simultaneously. Based on the concept of ε-dominance, Laumanns et al. proposed a new archive strategy that led to MOEAs with desired convergence and distribution properties.[28] The ε-dominance based archive strategy has a two-level concept. On the coarse level, the search space is discretized into boxes by a division according to eq. (7), where each vector uniquely belongs to one box.

$$b_i = \left\lfloor \frac{\log f_i}{\log(1+\varepsilon)} \right\rfloor \tag{7}$$

Using a generalized dominance relation on these boxes, the algorithm always maintains a set of non-dominated boxes, thus guaranteeing the ε-approximation property. On the fine level one element is kept in each box at most. Within a box, each representative vector can only be replaced by a dominating one, thus guaranteeing convergence with a bounded size according to eq. (8). Mostaghim et al. investigated the role of ε-dominance in MOPSOs and showed that the ε-dominance method can find solutions much faster than the clustering technique with comparable and even in some cases better convergence and diversity.[7] Therefore, ε-dominance based archive strategy is adopted in the proposed algorithm.

$$|A| \le \left(\frac{\log K}{\log(1+\varepsilon)} \right)^{(m-1)} \tag{8}$$

After updating external archive, crowding distance is calculated for each particle. The tournament selection of a leader particle(*gbest*) is based on crowding distance. [26]

4.4. *The pseudo and flow chart of CEPSO*

Two definitions introduced in CEPSO are given firstly for generalization, which can satisfy different evolutionary generations of subswarms and coevolutionary process, and lead to easy adjustment to different problem features.

Definition 5 (generation). It refers one complete pass through the fly or the mutation, the updates of all p_i and the update of the p_g.

Definition 6 (cycle). It refers to evolve all subswarms one time.

In this subsection, the pseudo of CEPSO in Matlab is given in Algorithm CEPSO, whose parameters are presented as follows.

maxgen: Maximum generation to evolve for each subswarm

maxcycle: Maximum cycle to coevolve

p_m: mutation probability

ps: particle number of each subswarm

pop_i: the *ith* subswarm

pop_{ij}: the *ith* particle in pop_i

The flow chart and cooperation procedure of CEPSO is described in Figure 1.

Algorithm CEPSO

```
initialize $n$ and $ps$;
initialize all subswarms $pop_i$, $i$=1, 2, …, $n$;
initialize all subswarms velocity $vel_{ij}$, $i$=1, 2, …, $n$, $j$=1, 2, …, $ps$;
$pbest_{ij}=pop_{ij}$, $i$=1,2,…, $n$, $j$=1, 2, …, $ps$;
construct an archive $A$ baesd on $pop_i$, $i$=1, 2, …, $n$;
calculate CD of particles in $A$;
for $c=1:maxcycle$;
    for $i$=1: $n$
        for $k=1:maxgen$
            if $rand<p_m$
                $pop_i=uniform(range(pop_i));$
            else
                for $m$=1:size($pop_i$)
                    select a leader from the archive $A$ based
                    on tournament;
                    calculate velocity $vel_{ij}$ according eq. (2);
                    limit velocity $vel_{ij}$;
                    update position $pop_{ij}$ according eq. (2);
                    limit position $pop_{ij}$;
                end
            end
            select collaborators from other subswarms;
            construct context vectors;
            evaluate each particle in subswarm $pop_i$;
            update each $pbest$ in subswarm $pop_i$ ;
            update the archive $A$ of swarm based on ε-dominance;
            calculate CD of particles in $A$;
        end
    end
end
output $A$
```
□

5. Experiments and discussion

5.1. *Benchmark functions*

The benchmark functions designed by K Deb et al. have been widely accepted[29], because they feature test-necessary properties: convexity, nonconvexity, discreteness and nonuniformity. The selected benchmark functions in this paper are ZDT1, ZDT2, ZDT4 and ZDT6, each is structured in the same manner and consists of three functions shown in eq. (9).

$$\begin{aligned} &\text{Minimize} && T(x) = (f_1(x_1), f_2(x)) \\ &\text{Subject to} && f_2(x) = g(x_2,\ldots,x_m)h(f_1(x_1), g(x_2,\ldots,x_m)) \\ &\text{where} && x = (x_1,\ldots,x_m) \end{aligned} \tag{9}$$

The definitions of benchmark functions are listed in Table 1.

Table 1. Benchmark functions

Function name	Feature	Definition	Parameters
ZDT1	ZDT1 has a convex Pareto-optimal front	$f_1(x_1) = x_1$ $g(x_2\ldots,x_D) = 1+9\cdot\sum_{i=2}^{D} x_i/(D-1)$ $h(f_1,g) = 1-\sqrt{f_1/g}$	$x=(x_1,\ldots,x_D)$ $x_i \in [0,1]$ i=1,…,D
ZDT2	ZDT2 has a nonconvex Pareto-optimal front	$f_1(x_1) = x_1$ $g(x_2\ldots,x_D) = 1+9\cdot\sum_{i=2}^{D} x_i/(D-1)$ $h(f_1,g) = 1-(f_1/g)^2$	$x=(x_1,\ldots,x_D)$ $x_i \in [0,1]$ i=1,…,D
ZDT4	ZDT4 has 21^9 local optima in its Pareto front	$f_1(x) = x_1$ $f_2(x) = g(x)[1-\sqrt{x_1/g(x)}]$ $g(x) = 1+10(D-1)+\sum_{i=2}^{D}[x_i^2-10\cos(4\pi x_i)]$	$x=(x_1,\ldots,x_D)$ $x_1 \in [0,1]$ $x_i \in [-5,5]$, i=2,…,D
ZDT6	The pareto front of ZDT6 is nonuniform	$f_1(x_1) = 1-\exp(-4x_1)\sin^6(6\pi x_1)$ $g(x_2\ldots,x_D) = 1+9\cdot(\sum_{i=2}^{D} x_i/(D-1))^{0.25}$ $h(f_1,g) = 1-(f_1/g)^2$	$x=(x_1,\ldots,x_D)$ $x_i \in [0,1]$ i=1,…,D

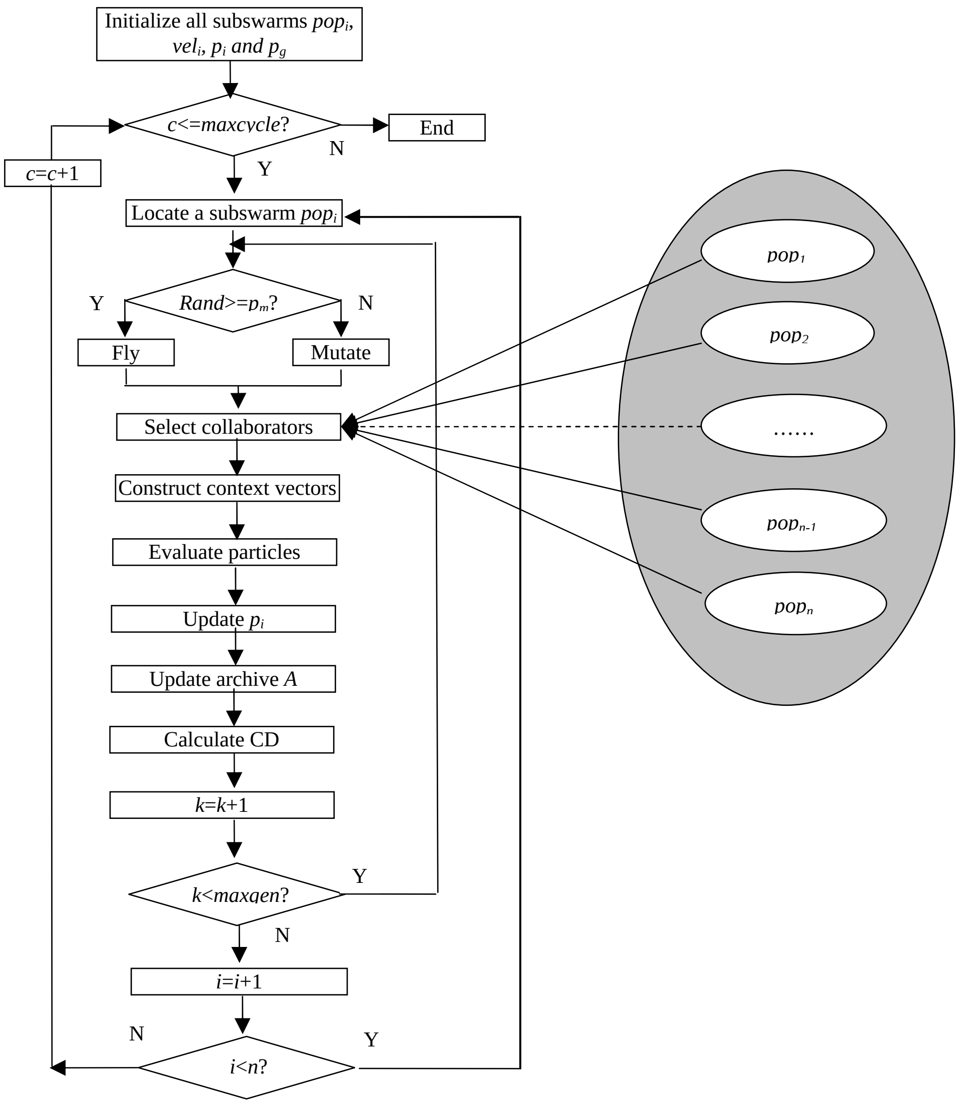

Fig. 1. Flowchart and cooperation procedure of CEPSO

5.2. *Metrics*

In this paper, three metrics for MOPs are employed and described in the following.

(1) Solution Number in Archive (SNA)

SNA is defined as the reserved solution number in external archive. This measure could embody the ability of MOPSOs to find more solutions as possible. So the bigger the SNA is, the better the MOPSO is.

(2) Generational Distance (GD)

The metric of generational distance represents how "far" the known Pareto front (PF_{known}) is away from the true Pareto front (PF_{true}) [11]:

$$GD = \left(\frac{1}{n} \sum_{i=1}^{n} (d_i{}^p) \right)^{1/p} \tag{10}$$

Where n is the number of members in PF_{known}, d_i is the Euclidean distance (in the objective domain) between the member in PF_{known} and its nearest member in PF_{true}.

(3) Spacing

Spacing is a metric measuring range (distance) variance of neighboring vectors in the nondominated vectors found so far. Since the "beginning" and "end" of current Pareto front found are known, the metric judges how well the solutions in such front are distributed[11]. This metric is defined in eq. (11).

$$S = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (\bar{d} - d_i)^2} \tag{11}$$

Where

$$\begin{cases} d_i = \min_j \left(\left| f_1^i(x) - f_1^j(x) \right| + \left| f_2^i(x) - f_2^j(x) \right| \right) \\ i, j = 1 \ldots n; \bar{d} = \frac{1}{n} \sum_{k=1}^{n} d_i \end{cases} \tag{12}$$

$\bar{d}$ is the mean of all solutions found so far, and n is the number of nondominated vectors. A value of zero for this metric indicates all members of the Pareto front currently available are equidistantly spaced. The metric addresses the second issue previously mentioned, namely good distribution of Pareto solutions.

5.3. *Results and discussion*

For the results presented in the following, the function evaluations needed by CEPSO are increased linearly with respect to the number of decision variables (*D*) and can be calculated by eq. (13).

$$functionevaluations = ps \times D \times maxgen \times maxcycle \qquad (13)$$

The main parameters of the experiments are set as follows:

- ps=10(subswarm size)
- p_m=0.3(mutation probability)
- $c_1=c_2$=2
- ω=1.0~0.4
- $maxgen$=1
- $maxcycle$=50

CEPSO is independently run 30 times for each benchmark function and the experimental results are summarized in Table 2~5 and Figure 3~18.

Table 2. Statistical results for ZDT1

D	SNA			GD			Spacing		
	max	mean	min	max	mean	min	max	mean	min
250	42	40	37	3.00E-18	8.59E-19	0.00E+00	1.99E-02	8.49E-03	4.10E-04
500	42	40	38	2.85E-18	6.77E-19	0.00E+00	1.52E-02	8.99E-03	2.63E-03
750	42	40	38	2.92E-18	7.55E-19	0.00E+00	1.94E-02	8.69E-03	1.20E-03
1000	42	40	38	2.85E-18	1.07E-18	0.00E+00	1.91E-02	8.24E-03	5.54E-04

Table 3. Statistical results for ZDT2

D	SNA			GD			Spacing		
	max	mean	min	max	mean	min	max	mean	min
250	22	22	21	5.05E-18	1.68E-18	0.00E+00	4.96E-03	1.80E-03	2.48E-04
500	22	22	22	5.05E-18	1.18E-18	0.00E+00	1.11E-02	1.83E-03	1.45E-04
750	22	22	22	5.05E-18	5.89E-19	0.00E+00	6.37E-03	1.78E-03	9.04E-05
1000	22	22	22	5.05E-18	5.89E-19	0.00E+00	4.58E-03	1.48E-03	7.73E-05

Table 4. Statistical results for ZDT4

D	SNA			GD			Spacing		
	max	mean	min	max	mean	min	max	mean	min
250	42	40	38	2.78E-18	9.48E-19	0.00E+00	1.85E-02	8.56E-03	1.32E-03
500	42	40	39	2.64E-18	6.69E-19	0.00E+00	1.56E-02	7.27E-03	1.16E-04
750	42	40	38	2.85E-18	7.40E-19	0.00E+00	1.67E-02	9.13E-03	6.92E-04
1000	42	39	38	3.89E-18	1.26E-18	0.00E+00	2.35E-02	9.73E-03	6.33E-04

Table 5. Statistical results for ZDT6

D	SNA			GD			Spacing		
	max	mean	min	max	mean	min	max	mean	min
250	20	19	16	5.84E-18	1.18E-18	0.00E+00	2.73E-02	1.68E-02	1.07E-02
500	19	18	16	6.94E-18	1.15E-18	0.00E+00	1.97E-02	1.50E-02	1.06E-02
750	20	19	16	5.84E-18	4.01E-19	0.00E+00	3.68E-02	1.66E-02	1.11E-02
1000	20	18	16	6.53E-18	1.22E-18	0.00E+00	2.96E-02	1.57E-02	1.21E-02

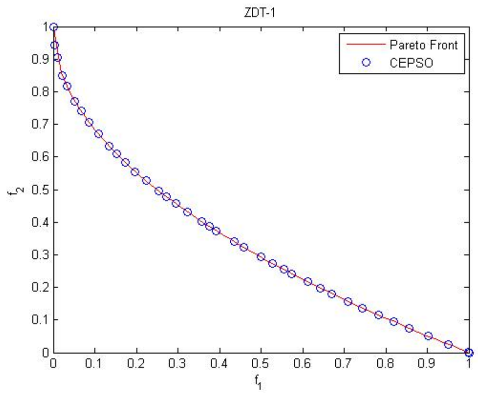

Figure 2 Pareto front of ZDT1(D=1000)

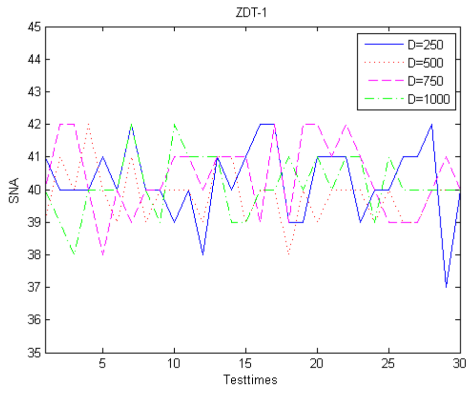

Figure 3 SNA of ZDT1

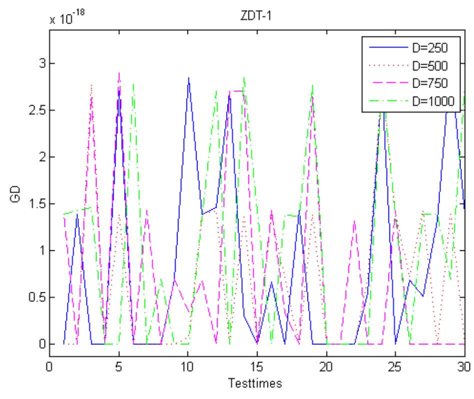

Figure 4 GD of ZDT1

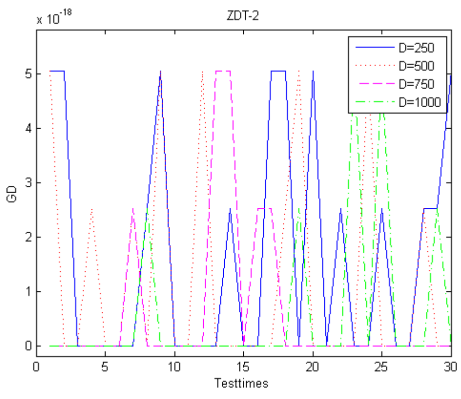

Figure 8 GD of ZDT2

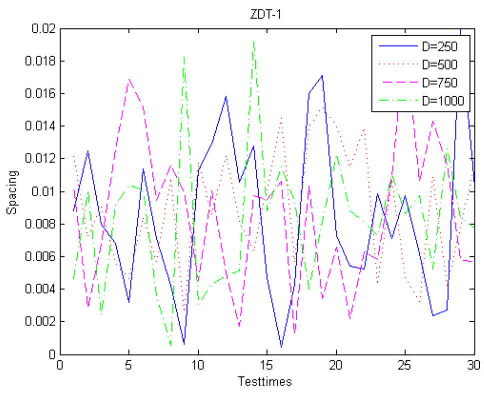

Figure 5 Spacing of ZDT1

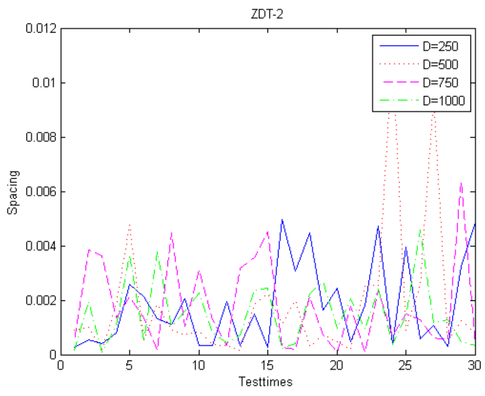

Figure 9 Spacing of ZDT2

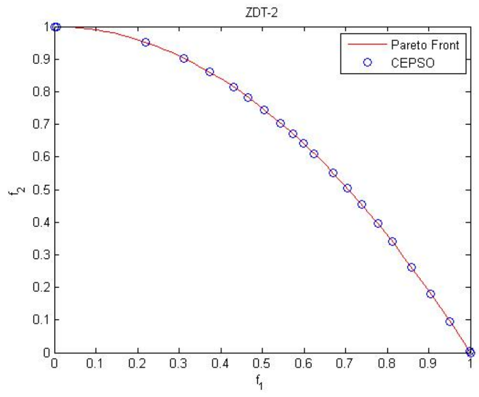

Figure 6 Pareto front of ZDT2(D=1000)

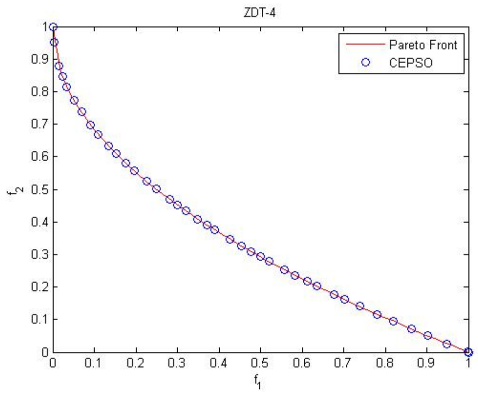

Figure 10 Pareto front of ZDT4(D=1000)

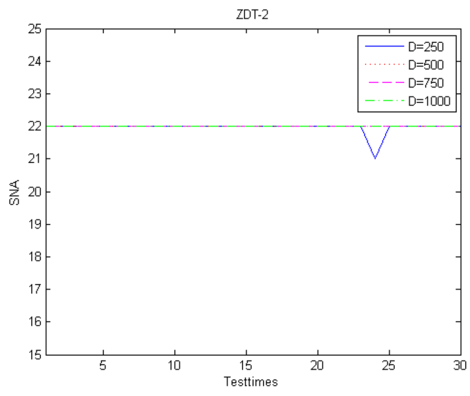

Figure 7 SNA of ZDT2

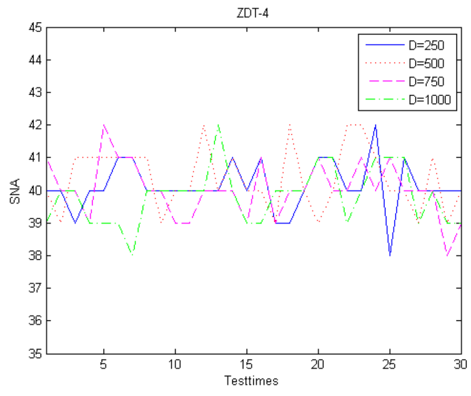

Figure 11 SNA of ZDT4

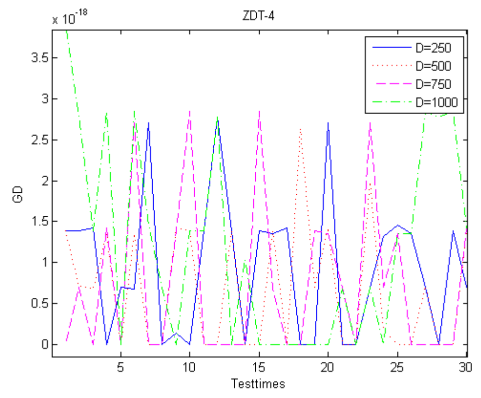

Figure 12 GD of ZDT4

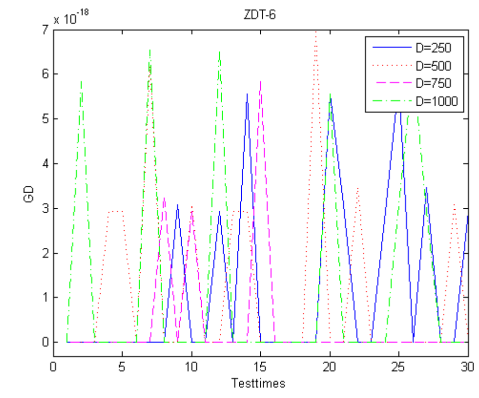

Figure 16 GD of ZDT6

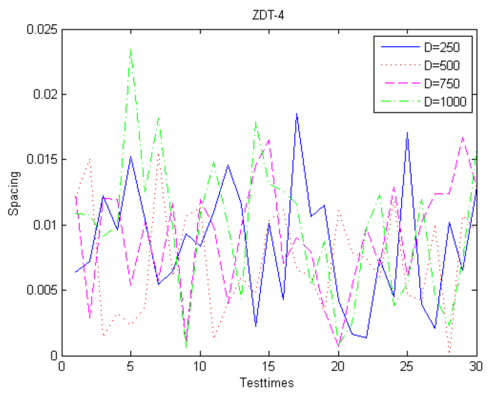

Figure 13 Spacing of ZDT4

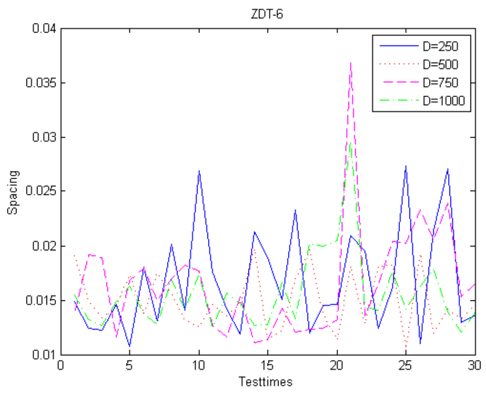

Figure 17 Spacing of ZDT6

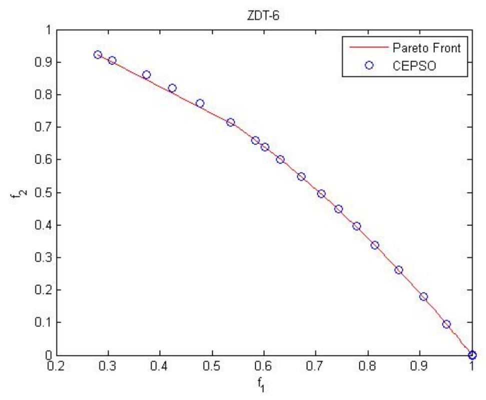

Figure 14 Pareto front of ZDT6(D=1000)

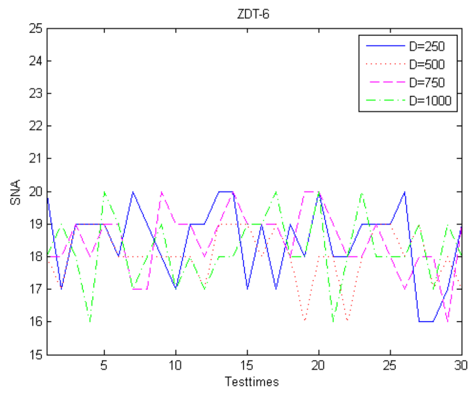

Figure 15 SNA of ZDT6

The benchmark function ZDT1 and ZDT2 are relatively simple. Most MOPSOs can solve the two problems with 30 decision variables (Tables and figures for 10~100D are omitted in this paper). Not only can CEPSO solve ZDT1 and ZDT2 with 30 decision variables but also problems with up to 1000 decision variables. The simulation results for the high dimensional problems are given in Table 2~3 and Figure 2~9. As can be seen from these tables and figures, for the metric of SNA, there is a little change in number. The GD is negligibly tiny to 10^{-19}, which means that PF_{known} is almost unlimitedly near PF_{true}. On the other hand, the Spacing is stable and there is no difference among various dimensions, but the Spacing of ZDT1 is better than that of ZDT2.

The benchmark function ZDT4 and ZDT6 are more difficult than ZDT1 and ZDT2. Again, not only can CEPSO solve ZDT2 and ZDT4 with 30 decision variables but also solve problems with up to 1000 decision variables. The benchmark function ZDT4 has 21^9 local optima and most MOPSOs are easily trapped in them. The simulation results for the high dimensional problems are given in Table 4 and Figure 10~13. As can be seen, for the metric of SNA, there is some changes in number. However, the least is 38, which is a reasonable result. The GD is very tiny and is about 10^{-19}. Most runs can obtain true Pareto front. On the other hand, the Spacing is stable and there is no difference among various dimensions.

ZDT6 is also a difficult problem for its solutions are not uniformly distributed in its Pareto front. The simulation results for the high dimensional problems are given in Table 5 and Figure 14~17. As can be seen, for the metric of SNA, there is little changes in number. In fact, ZDT2 and ZDT6 have the same Pareto front. However, the metric of SNA for ZDT2 is better than ZDT6, while GD and Spacing are similar to ZDT2.

These simulation results demonstrated that CEPSO can solve MOPs with up to 1000 decision variables but the numbers of function evaluations are only linear with respect to them, which means that CEPSO has good scalability for dealing with various MOPs.

6. Conclusion and future works

A scalable cooperative coevolution and ε-dominance based MOPSO is proposed and delineated. The experimental results show that CEPSO can not only solve MOPs with 30~100 decision variables but also MOPs with up to 1000 decision variables with fair scalability. Besides, function evaluation numbers are only linear with respect to the number of decision variables. Considering the fact that most MOPSOs can only solve MOPs with 10~30 decision variables and their performance will deteriorate when decision variables are 100 or above, the proposed approach is much promising.

The future work will focus more on theoretical analysis and mathematical formulation and proof on one hand. On the other hand, we will attempt to reduce the run time to a certain magnitude. We will also try parallel implementation of CEPSO, which can reduce the run time other way around. And finally, we will apply CEPSO to practical applications, which will realize the true value of our research work.

Acknowledgement

We are grateful for the support of National Natural Science Foundation of China (60970004) and A Project of Shandong Province Higher Educational Science and Technology Program (J10LG08). We also thank the anonymous reviewers for their helpful advices.

References

1. C.A.C. Coello, Evolutionary multi-objective optimization: A historical view of the field, *IEEE Computational Intelligence Magazine*, **1**(1) (2006) 28–36.
2. Enrique H. Ruspini, Soft Computing: Coping with Complexity, *International Journal of Computational Intelligence Systems*, **3**(2) (2010) 190-196.
3. M.R. Sierra and C.A.C. Coello, Multi-Objective particle swarm optimizers: A survey of the state-of-the-art, *Int'l Journal of Computational Intelligence Research*, **2**(3) (2006) 287–308.
4. L. Hu, X. Che, X. Cheng, Bandwidth Prediction based on Nu-Support Vector Regression and Parallel Hybrid Particle Swarm Optimization, *International Journal of Computational Intelligence Systems*, **3**(1) (2010) 70-83.
5. J. Kennedy and R.C. Eberhart, Particle swarm optimization, in *proc. of the IEEE Conf. on Neural Networks*, (1995), pp.1942–1948.
6. J. Moore, R. Chapman, Application of particle swarm to multi-objective optimization, *Department of Computer Science and Software Engineering*, (Auburn University, 1999).
7. K.C. Tan, Y.J. Yang and C.K. Goh, A distributed cooperative coevolutionary algorithm for multiobjective optimization, *IEEE Trans. on Evolutionary Computation*, **10**(5) (2006) 527–549.
8. S. Mostaghim, J. Teich, The role of ε-dominance in multi objective particle swarm optimization methods, in *proc. of 2003 Congress on Evolutionary Computation*, (2003), pp.1764–1771.
9. S. Mostaghim, J. Teich, Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO), in *proc. of IEEE Swarm Intelligence Symposium*, (*2003*), pp.26–33.
10. J. Fieldsend and S. Singh, A multiobjective algorithm based upon particle swarm optimization, an efficient data structure and turbulence, in *proc. of the 2002 U.K. Workshop on Computational Intelligence*, (2002), pp.37–44.
11. C.A.C. Coello, G. Pulido, and M. Lechuga, Handling multiple objectives with particle swarm optimization, *IEEE Transactions on Evolutionary Computation*, **8**(3) (2004) 256–279.
12. X. Li, A non-dominated sorting particle swarm optimizer for multi-objective optimization, in *proc. of the Genetic and Evolutionary Computation Conference*, (2003), pp.37–48.
13. M.R. Sierra and C.A.C. Coello, Improving PSO-based multi-objective optimization using crowding, mutation and ε-dominance, in *proc. of Third International Conference on Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science*, **3410** (2005) 505–519.
14. M.A. Potter, K.A. De Jong, A cooperative coevolutionary approach to function optimization, in *proc. of the 3rd Parallel Problem Solving from Nature*, (1994), pp.249–257.
15. A.W. Iorio, and X. Li, A Cooperative Coevolutionary Multiobjective Algorithm Using non-dominated Sorting, in *proc. of the Genetic and Evolutionary Computation Conference, Lecture Notes in Computer Science,* **3102** (2004) 537-548.
16. M.R. Sierrra and C.A.C. Coello, Coevolutionary Multiobjective Optimization using Clustering Techniques, *Advances in Artificial Intelligence, Lecture Notes in Artificial Intelligence*, **3789** (2005) 603–612.
17. T.G. Tan, H.K. Lau, J. Teo, Cooperative coevolution for pareto multiobjective optimization: An empirical study using SPEA2, in *proc. of 2007 IEEE Region 10 International Conference on TENCON*, (2007), pp.1-4.
18. F. van den Bergh and A.P. Engelbrecht, A cooperative approach to particle swarm optimization, *IEEE Transaction on Evolutionary Computation*, **8**(3) (2004) 225–239.
19. M. Iwamatsu, Locating all global minima using multi-species particle swarm optimizer: The inertia weight and the constriction factor variants, in *proc. of the 2006 IEEE Congress on Evolutionary Computation*, (2006), pp.816–822.
20. J.H. Seo, C.H. Im, Heo et al. Multimodal function optimization based on particle swarm optimization, *IEEE Trans. on Magnetics*, **42**(4) (2006) 1095–1098.
21. X. Yao, Y. Liu, J. Li, J. He and C. Frayn, Current developments and future directions of bio-inspired computation and mplications for ecoinformatics, *Ecological informatics*, **1** (2006) 9-22.
22. Z. Yang, K. Tang, and X. Yao, Differential Evolution for high-dimensional function optimization, *Proc. IEEE*

Congress on Evolutionary Computation, (2007), pp.3523-3530.

23. P. Wiegand, W. Liles and K. De Jong, An empirical analysis of collaboration methods in cooperative coevolutionary algorithms, in *proc. of Genetic and Evolutionary Computation Conference*, (2001), pp. 1235-1242.
24. M. Clerc, J. Kennedy, The particle swarm: Explosion, stability, and convergence in a multi-dimensional complex space, *IEEE Trans. on Evolutionary Computation*, **6**(1) (2002) 58–73.
25. X. Zheng and H. Liu, A hybrid vertical mutation and self-adaptation based MOPSO, *Computers and Mathematics with Applications*, **57**(3/4) (2009) 2030-2038.
26. K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II, *IEEE Transactions on Evolutionary Computation*, **6**(2) (2002) 182–197.
27. P.S. Andrews, An investigation into mutation operators for particle swarm optimization, in *proc. of the 2006 Congress on Evolutionary Computation*, (2006), pp.1044-1051.
28. M. Laumanns, L. Thiele, K. Deb and E. Zitzler, Combining convergence and diversity in evolutionary multi-objective optimization, *Evolutionary Computation*, **10**(3) (2002) 263–282.
29. E. Zitzler, K. Deb and L. Thiele, Comparison of multi-objective evolutionary algorithms: empirical results, *Evolutionary Computation*, **8**(2) (2000) 173–195.