Extensions of Differential Evolution Algorithms for Multimodal Optimization

Daniela Zaharie

West University of Timişoara,
Faculty of Mathematics and Computer Science
Bv. V. Pârvan, no. 4, 300223 Timişoara
dzaharie@info.uvt.ro

Abstract. Recently, some extensions of differential evolution to multimodal optimization have been proposed. The aim of this paper is to provide a comparative study of them and to propose a hybrid version in order to overpass some of their deficiencies. The comparative analysis is based on numerical experiments on some test functions.
AMS Subject Classification(2000): 90C56, 68W20
Keywords and phrases: differential evolution, multimodal optimization, sharing, crowding, island model

1 Introduction

Multimodal optimization deals with locating all global (or even local) optima of an objective function. From a practical viewpoint, multimodal optimization is motivated by at least two reasons [10]: (i) by encouraging the location of multiple optima the chances of locating the global one may be improved; (ii) in a design context, identifying a set of high-quality solutions can suggest to the decision maker some viable alternatives. Due to their work with populations, the evolutionary algorithms (EAs) are good candidates for multimodal optimization. However their classical structure based on successively applying mutation, recombination and selection operators without encouraging the simultaneously search for different potential optima leads to find only the global optimum. Therefore different techniques having the aim of locating and maintaining different potential optima have been proposed in the last years.

The basic idea of these methods is to impose the formation of different so-called species into the population such that each one will identify an optimum. This speciation can be an implicit one (based on some techniques, different species emerges into the population) or an explicit one (the population is explicitly divided into subpopulations).

Implicit speciation can be assured by the so-called niching methods. Two well-known niching techniques are: *sharing* [5] and *crowding* [13]. The sharing technique interferes with the selection step and consists in derating the individual fitness depending on the number of similar individuals. The crowding technique interferes with the replacement of the generation of parents with the offspring's one by favoring replacements between similar individuals.

Explicit speciation by subpopulation models attracted the attention of EAs researchers due to its relationship with the parallelization models of EA. Two of these models are proposed in [4] and [10].

If the first multimodal evolutionary approaches were based mainly on the genetic algorithms paradigm (binary encoding, classical crossover and mutation operators) recently other evolutionary paradigms have been extended to multimodal optimization. Thus in [2] genetic algorithms with real encoding are extended to multimodal optimization and in [7] is proposed an extension of evolution strategies to locate multiple optima.

Following the same idea on extending to multimodal optimization other EAs successfully applied in global optimization, recently some variants of differential evolution (DE) [14] have been proposed [6],[15],[18]. The motivation of these extensions is a natural one if we take into account the fact that differential evolution is one of the simplest and in the same time most efficient global optimization methods.

The aim of this paper is to compare the abilities of some DE extensions to locate and maintain multiple optima and to propose a hybridization between the crowding DE [15] and multi-population DE [18]. The motivation of this hybridization is to reduce the cost of the global crowding process used in original crowding DE and to increase its parallelization potential.

The paper is organized as follows. The next section summarizes the particularities of DE algorithms. In section three are presented the current DE extensions for multimodal optimization and is introduced the multipopulation crowding DE. Section four is devoted to comparative numerical results and the last section concludes the work.

2 Differential Evolution

Differential evolution has been proposed in [14] as an heuristic, inspired by simplex methods, able to efficiently solve difficult optimization problems on continuous domains. Since its invention it has been extended to solve: discrete optimization problems [8], constrained optimization problems [9], multi-objective optimization problems [1],[12] and, recently, multi-modal optimization problems [6],[15], [18].

Its particularity consists in its search operator based on an internal perturbation scheme not on an external one as is usual in classical mutation operators. To summarize the particularities of the DE algorithm let us consider a simple unconstrained maximization problem of a function $f : D \to R$: find $x^* \in D \subset R^n$ such that $f(x^*) \geq f(x)$ for all $x \in D$. Let us denote by $X = (x_1, \ldots, x_m)$ the current generation, by x_* the best element of X ($f(x_*) \geq f(x_i)$ for all $i \in \{1, \ldots, m\}$) and by $Y = (y_1, \ldots, y_m)$ the offspring population (y_i is considered the offspring of x_i).

Different schemes of constructing $y_i = (y_i^1, \ldots, y_i^n)$ starting from the elements of X have been proposed. We exemplify here only two of them, those used in the next sections. The first one is that which is most used:

$$y_i^j = \begin{cases} x_{r_3}^j + F \cdot (x_{r_1}^j - x_{r_2}^j), & \text{with probability } p_c \\ x_i^j, & \text{with probability } 1 - p_c \end{cases} \qquad j = \overline{1,n} \qquad (1)$$

In relation (1) r_1, r_2, r_3 are distinct indices randomly selected from $\{1, \ldots, m\}$, $F \in (0, 2)$ is a parameter which controls the magnitude of the perturbation and $p_c \in [0, 1]$ is a probability value which controls the ratio of new components in the offspring.

Another scheme is characterized by replacing x_{r_3} with the best element of the population, x_*. The variant used in section 3 is characterized by $p_c = 1$ thus:

$$y_i^j = x_*^j + F \cdot (x_{r_1}^j - x_{r_2}^j), \qquad j = \overline{1,n} \qquad (2)$$

The DE behavior is highly influenced by the values of the control parameters p_c and F. If no diversity preserving techniques are applied, the most frequent problem of DE is premature convergence. Different techniques of avoiding premature convergence have been proposed. One of them consists in adjusting the values of parameters F and p_c such that the perturbation step allows keeping the population diversity to an acceptable level [17].

The selection step is a simple one: an offspring, y_i, replaces its parent, x_i, if it is better ($f(y_i) \geq f(x_i)$). The general structure of DE algorithm is similar to that of classical evolutionary algorithms. Either generational (synchronous) or steady state (asynchronous) strategies can be used (Fig. 1).

```
Population initialization
REPEAT
   FOR i = 1,m
      construct y_i using (1) or (2)
   FOR i = 1,m
      IF f(y_i) ≥ f(x_i) replaces x_i with y_i
UNTIL a stopping condition is satisfied
```

(a)

```
Population initialization
REPEAT
   FOR i = 1,m
      construct y_i using (1) or (2)
      IF f(y_i) ≥ f(x_i) replaces x_i with y_i
UNTIL a stopping condition is satisfied
```

(b)

Fig. 1. The structure of a DE algorithm. (a) generational strategy; (b) steady-state strategy.

3 Differential Evolution extensions for multimodal optimization

Crowding based DE [15]. The starting idea in developing the crowding based DE (CDE) was to apply to the classical steady state DE algorithm (Fig.1(b))

(based on a type 1 perturbation) some niching techniques in order to increase its ability to locate and maintain multiple optima. Both sharing and crowding have been tried but as is reported in [15] the crowding version outperforms the sharing one when tested on a set of test functions.
Algorithm's structure. The particularity of the CDE consists in a global crowding, i.e. when a new offspring, y_i is generated it is compared not with its parent, x_i, but with the most similar element from the current population, $z \in X$ ($d(z, y_i) \leq d(x, y_i)$ for all x in X). As measure of similarity is used the euclidian distance. If y_i is better than z then y_i replaces z. This approach imposes the computation at each step of the distances between the offspring and all the elements of X. The general structure of CDE algorithm is described in Fig. 2.

Population initialization
`REPEAT`
  `FOR` $i = \overline{1, m}$
    construct y_i using (1)
    find the element $z \in X$ most similar to y_i
    `IF` $f(y_i) \geq f(z)$ replaces z with y_i
`UNTIL` a stopping condition is satisfied

Fig. 2. The structure of crowding DE (CDE)

Pros. The most important advantage of CDE is its simplicity since it can be obtained from DE by only a few modifications.
Cons. A possible disadvantage of CDE could be the global character of crowding. Besides the fact that for large populations this could increase the computational effort it also precludes CDE from efficient parallel implementations.

Multiresolution multipopulation DE [18]. The main idea of multiresolution multipopulation DE (MMDE) is to divide the population into equally sized subpopulations. In each subpopulation a generational based DE (Fig. 1 (a)) is applied for a given number of generations. After this, a migration process can occur. When a subpopulation converged, the optima that it found are collected into an archive. Since each subpopulation locates at most one optimum, when many unequally spaced optima have to be found it may be useful to reiterate the search process through a few *search epochs*. This idea of reusing a subpopulation after that it found an optimum is similar to that proposed in roaming algorithms [11]. The particularity of MMDE consists in the controlled subpopulations (re)initialization. This is based on a *resolution factor* concept and has the aim of ensuring the search space exploration and avoiding the redundant search.
Algorithm's structure. Let us consider that the population is divided in s subpopulations $X_1, \ldots, X_s$ and the search domain is $D = \prod_{j=1}^{n} [a_j, b_j]$.

Initialize the archive: $A := \emptyset$
Initialize the epoch counter: $e := 1$
REPEAT
  Compute the resolution factors: $r_j^e = (b_j - a_j)/(s \cdot e)^{1/n}, \qquad j = \overline{1,n}$
  (Re)initialize the subpopulations $X_1, \ldots, X_s$
  REPEAT
    Apply DE to $X_1, \ldots, X_s$ for τ generations
    Apply migration
  UNTIL all subpopulations converged
  Add the best elements of subpopulation to the archive
  $e := e + 1$
UNTIL a stopping condition is satisfied

Fig. 3. The structure of multiresolution multipopulation DE (MMDE)

The initialization is based on a decomposition of D in subdomains: each subpopulation will be initialized with random elements selected from a subdomain. The subdomains are defined based on discretization steplengths called resolution factors, r_j. For instance $r_j = (b_j - a_j)/s^{1/n}$ and the subpopulation X_i will be initialized in $D_i = [a_1^i, b_1^i] \times \ldots \times [a_n^i, b_n^i]$, where $a_j^i = a_j + r_j k_j^i$, $b_j^i = a_j^i + r_j$, with k_j^i randomly selected from $\{0, 1, \ldots, [s^{1/n}] - 1\}$ independently for each subpopulation and for each component. During the evolution, the subpopulations are not restricted to the subdomain affected in the initialization step. During the evolution the subpopulations are not necessarily non-overlapping, therefore different subpopulations could find the same optimum. When passing from a search epoch to another one, the subpopulations are reinitialized based on a finer discretization of the domain, i.e. on smaller resolution factors. A simple rule to choose the resolution factors in epoch e is: $r_j^e = (b_j - a_j)/(s \cdot e)^{1/n}$, $j = \overline{1,m}$. For each of the s subpopulations, a subdomain is randomly chosen from a number of $s \cdot e$ possible subdomains. The elements of the subpopulations are selected using a uniform distribution only in the first epoch. In the next epochs the selection distribution is influenced by the elements already placed in the archive. To be more specific, let us consider that at epoch e the archive contains the elements $\alpha_1, \alpha_2, \ldots, \alpha_k$ and the subdomain affected to subpopulation X_i is D_i. During the initialization of X_i, an element x, randomly selected from D_i is accepted with a probability P_a determined using a sharing function:

$$P_a(x) = \frac{1}{1 + \sum_{i=1}^{k} \sigma(x, \alpha_i)}, \qquad \sigma(x, \alpha) = \begin{cases} 1 - \frac{d(x,\alpha)}{r_e/2} & \text{if } d(x, \alpha) < r_e/2 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

with $r_e = (\sum_{j=1}^{n} r_j^e)/n$ and d denotes the euclidian distance. The sharing function is computed with respect to the elements belonging to the current archive, the resolution factors playing the role of a niche radius. The acceptance probability is high only for elements that are not too close to the optima already

placed in the archive. This idea is similar to that of derating the fitness function used in sequential niching [3] but here one derates the acceptance probability. If in the initialization of a subpopulation element a given number of successive rejections occur, the subdomain is replaced with a new randomly selected one.

The migration used in MMDE is a random one characterized by the fact that each element of a subpopulation can be swapped with a given migration probability with a randomly selected individual from a randomly selected subpopulation. Due to the change of information between subpopulations, the migration avoids the premature convergence but in the same time it can guide different subpopulations toward the same optimum. Thus, as results reported in [18] suggest when the aim is to locate multiple optima the migration probability should be small, even zero, which means no migration. At each search epoch the best element of each subpopulation is added to the optima archive. To avoid redundancy in the archive (presence of multiple copies of the same optimum) two techniques are used:

(i) A new optimum is added to the archive only if it is sufficiently dissimilar from the already stored optima. A candidate is added to the archive only if its distance to the other optima is greater than a threshold depending on the current resolution factor, r_e (e.g. $r_e/4$).

(ii) A sufficiently dissimilar candidate is added to the archive only if it belongs to a different hill (in the case of a maximization problem) than the other elements of the archive. The decision procedure is based on the idea of hill-valley function introduced in [16]. To decide if there is a valley between two elements x and y a given number of convex combinations of x and y are generated. If for at least one $z = cx + (1 - c)y$ the relation $f(z) < \min\{f(x), f(y)\}$ holds then one can decide that there exists a valley between x and y. If for a candidate optimum there is found an element in the archive such that no valley is detected between it and the candidate then the candidate is not added to the archive.

Pros. The main advantage of MMDE is the fact that it does not use a global processing step (e.g. clustering) allowing to be efficiently implemented in parallel. The communication between subpopulations is assured, even in the absence of migration, through the archive.

Cons. Each subpopulation locates at most an optimum, thus many subpopulations or a multiresolution approach are needed when multiple optima have to be located. The multiresolution approach, despite the fact that avoids the specification of a niche radius, increases the complexity of DE. It is obvious that MMDE is not as simple as CDE is. The numerical results in [18] suggest that MMDE is sensitive to the values of m (subpopulations size), s (number of subpopulations) and $e_{\max}$ (the maximal number of epochs).

MultiDE [6]. The approach in MultiDE is similar to that of MMDE since it also considers equally sized subpopulations. However the number of subpopulations is variable (subpopulations appear or disappear). A structure similar to an archive, called "population 0" is also used. When an element of a subpopulation is similar to an element from "population 0" then it is dropped from further computations. The similarity is established based on a so-called *precision parameter* controlled

by the user. This is somewhat similar to the resolution factor in MMDE. MultiDE uses a mechanism to encourage the search for different optima based on a *minimum spanning* distance. For each new trial offspring the distance between it and the other subpopulations is computed and if this distance is smaller than a user established parameter then the offspring is perturbed by moving it in a opposite direction. Over generations, the parameter corresponding to minimum spanning distance is slowly decreased but it is reset to the initial value every time the number of subpopulations is increased. Another specific parameter is the *expiration time*, i.e. the number of generations after that a subpopulation is eliminated if it did not discover a new optima.
Pros. As in the case of MMDE the subpopulations approach opens the possibility of parallel implementations. However the computation of distances between the trial offspring and all subpopulations could an impediment be for an efficient parallelization.
Cons. The main disadvantage of MultiDE is the presence of some supplementary control parameters (precision parameter, minimum spanning distance and expiration time).

A multipopulation crowding DE. Starting from the main deficiency of CDE, that of the presence of a global processing step we propose a multipopulation variant of CDE. The basic idea is to use in a multipopulation DE a niching technique based on crowding.

The proposed technique have the following characteristics: (i) since the crowding technique allows each subpopulation to locate many optima the subpopulation reinitialization is no more necessary; (ii) the crowding computation is limited to subpopulations, thus a global processing step is avoided.
Algorithm's structure. The multipopulation crowding DE (MCDE) is characterized by applying CDE for each subpopulation:

Initialize the subpopulations $X_1, \ldots, X_s$
`REPEAT`
    Apply CDE to $X_1, \ldots, X_s$ for τ generations
    Apply migration
`UNTIL` a stopping condition is satisfied
Collect the optima found by all subpopulations

Fig. 4. The structure of multipopulation crowding DE (MCDE)

The migration strategy can be the random one used in MMDE. After the subpopulations converged, the optima they found are collected. Since different subpopulations can found the same optimum only those sufficiently dissimilar or belonging to different hills are collected. This process is similar to that of archiving used in MMDE.

4 Numerical results

The aim of the numerical tests is twofold: (i) to compare the ability of crowding DE and of multiresolution multipopulation DE in solving multimodal optimization problems; (ii) to analyze the behavior of multipopulation crowding DE.

Due to the similarity of the approach in MMDE and MultiDE we conducted the comparative study toward CDE, MMDE and the proposed hybrid variant.

The measures used to evaluate the analyzed algorithms are: (i) *success rate - SR* (the ratio between the number of cases when all optima have been located with the desired accuracy and the total number of runs); (ii)*averaged number of found optima-*$\langle P \rangle$; (iii) *averaged number of objective function evaluation-*$\langle nfe \rangle$ (computed in case of success). For MMDE the averaged number of epochs, $\langle e \rangle$, has been also computed. All results have been obtained by processing the results of 30 independent runs.

The DE specific parameters were chosen as follows: $p_c = 0.9$ and $F = 0.5$ for CDE and MCDE while for MMDE they were adapted according to the adaptation rules proposed in [17]. In MMDE the evolution of a subpopulation is stopped when it has lost its diversity. The migration interval (number of iterations between migrations) is set to $\tau = 20$. The test functions are summarized in Table 1 and are discussed in the following.

Function	Expression	Domain
Himmelblau	$f_1(x,y) = 200 - (x^2 + y - 11)^2 - (x + y^2 - 7)^2$	$[-6,6]^2$
Six-Hump Camel Back	$f_2(x,y) = -[(4 - 2.1x^2 + x^4/3)x^2 + xy + 4(y^2 - 1)y^2]$	$[-1.9, 1.9]\times$ $[-1.1, 1.1]$
Shubert	$f_3(x,y) = -\sum_{i=1}^{5} i\cos((i+1)x + i) \cdot \sum_{i=1}^{5} i\cos((i+1)y + i)$	$[-10,10]^2$
Schaffer	$f_4(x,y) = -0.5 - (\sin^2(\sqrt{x^2+y^2}) - 0.5)/(1 + 0.001(x^2+y^2))^2$	$[-100,100]^2$
Ursem	$f_5(x,y) = \sin(2.2\pi x + 0.5\pi)\frac{(2-\lvert y\rvert)(3-\lvert x\rvert)}{4} + \sin(0.5\pi y^2 + 0.5\pi)\frac{(2-\lvert y\rvert)(2-\lvert x\rvert)}{4}$	$[-2,2]\times$ $[-1.5,1.5]$

Table 1. Test functions

The stopping criterion has been established depending on the problem. The results are presented in Tables 2-6.

Himmelblau: the aim is to locate all the four global optima ($f_1(x^*) = 200$) with an accuracy $\epsilon = 10^{-6}$ with respect to the optima value. As can be seen in Table 2 almost similar results have been obtained by CDE, MMDE and MCDE. In the case of multipopulation variants the subpopulation size has to be at least 10.

Six-hump camel back: the aim is to find the two global optima ($f_2(x^*) = 1.03163$) with accuracy $\epsilon = 10^{-5}$ and to locate other four local optima. This is a difficult problem for CDE and MCDE. When a hill-valley approach is used (as is

CDE and MCDE						MMDE							
m	s	p_m	SR	$\langle P\rangle$	$\langle nfe\rangle$	m	s	p_m	$\langle e\rangle$	SR	$\langle P\rangle$	$\langle nfe\rangle$	DE eq.
15	1	0	96%	4	6605	10	5	0	2.03	100%	4	7553	(1)
20	1	0	100%	4	8033	5	10	0	2	96%	4	8081	(1)
10	*2*	*0.1*	*100%*	*4*	*7486*	*10*	*5*	*0*	*2.76*	*100%*	*4*	*6069*	*(2)*
10	2	0.5	100%	4	7653	5	10	0	5.86	3%	4	10647	(2)

Table 2. Results for Himmelblau's test function

suggested in [15]) CDE finds all optima but MCDE has not enough explorative power. On the other hand MMDE behaves better especially when the perturbation of type (2) is used.

CDE and MCDE						MMDE							
m	s	p_m	SR	$\langle P\rangle$	$\langle nfe\rangle$	m	s	p_m	$\langle e\rangle$	SR	$\langle P\rangle$	$\langle nfe\rangle$	DE eq.
100	*1*	*0*	*100%*	*6*	*62645*	10	10	0	5.43	90%	5.86	37538	(1)
120	1	0	100%	6	71790	12	10	0	5.80	93%	5.93	47318	(1)
20	5	0.1	70%	5.56	48177	*10*	*10*	*0*	*3.43*	*100%*	*6*	*14610*	*(2)*
30	4	0.1	83%	6.13	53151	12	10	0	3.40	100%	6	16822	(2)

Table 3. Results for "six hump camel back" test function

Shubert: the aim is to locate all eighteen global maxima ($f_3(x^*) = 186.731$) with accuracy $\epsilon = 10^{-3}$. In this case the best results are obtained by MCDE. A possible explanation is that the aim was to locate only global maxima. However CDE locates more local optima than MCDE. On the other hand MMDE needs more function evaluations to locate all global optima and the number of found local optima is smaller than in the case of CDE.

CDE and MCDE						MMDE							
m	s	p_m	SR	$\langle P\rangle$	$\langle nfe\rangle$	m	s	p_m	$\langle e\rangle$	SR	$\langle P\rangle$	$\langle nfe\rangle$	DE eq.
50	1	0	93%	17.9/49.9	60800	15	20	0	5.73	96%	17.9/40.83	136833	(1)
75	1	0	100%	18/74.9	107200	20	20	0	4.90	100%	18/33.73	172967	(1)
25	*2*	*0.1*	*100%*	*18/23.7*	*29283*	15	20	0	7.96	96%	17.9/78.2	110076	(2)
25	3	0.1	100%	18/25.4	38325	*20*	*20*	*0*	*5.63*	*100%*	*18/47.33*	*109703*	*(2)*

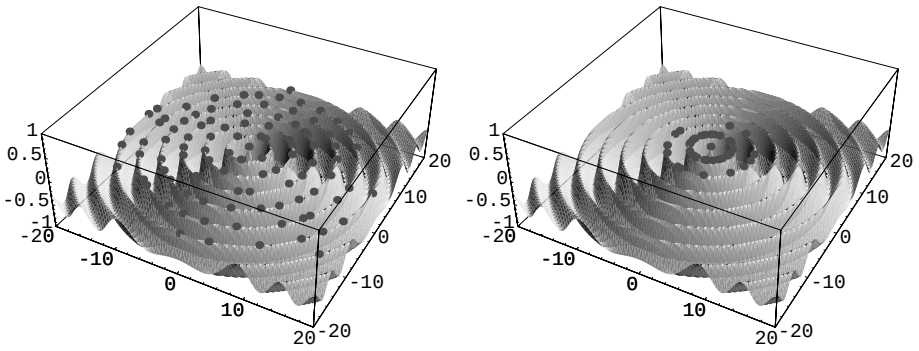
Table 4. Results for Shubert's test function

Schaffer: the aim is to locate the global optimum ($f_4(x^*) = 0$) with accuracy $\epsilon = 10^{-4}$. The difference between the value of the global optimum and that of the local ones is very small. In this case MMDE (based on perturbation of the first type) outperforms CDE and MCDE. Also MCDE behavior is better than

that of CDE with respect to the number of the number of function evaluations but as is expected the explorative power is lower (see also Fig. 5).

CDE and MCDE						MMDE							
m	s	p_m	SR	$\langle P \rangle$	$\langle nfe \rangle$	m	s	p_m	$\langle e \rangle$	SR	$\langle P \rangle$	$\langle nfe \rangle$	DE eq.
20	1	0	53%	19.96	60570	*10*	*2*	*0.1*	*2.83*	*100%*	*1*	*21748*	*(1)*
50	1	0	46%	49.93	263835	10	4	0.1	1.66	100%	1	24786	(1)
10	2	0.1	100%	13.36	44006	10	2	0.1	10	13%	1.33	9655	(2)
10	*2*	*0.5*	*100%*	*12.9*	*31566*	10	4	0.1	9.13	26%	1.4	19308	(2)

Table 5. Results for Schaffer's test function



(a) (b)

Fig. 5. The population after 100 iterations - Schaffer test function. (a)CDE ($m = 100$, $s = 1$, $p_m = 0$) (b) MCDE ($m = 10$, $s = 10$, $p_m = 0.1$, $\tau = 10$)

Ursem: the aim is to approximate the global optimum ($f(x^*) = 2.5$) with accuracy $\epsilon = 10^{-5}$ and to locate other four local optima. With respect to the number of function evaluations MMDE behaves similar to CDE. When the population size is at least ten and there are enough subpopulations, MCDE behaves better than CDE. The population distribution after 100 iterations is illustrated both for CDE and MCDE in Fig. 6. The population distribution suggests that in the case of MCDE the population concentrates quicker on the optima than in the case of CDE.

5 Conclusions and further work

The numerical results suggest that crowding DE and multiresolution multipopulation DE behaves almost similar. However CDE behaves better when the number of optima to be found is large (e.g. Shubert's test function) due to the

CDE and MCDE						MMDE							
m	s	p_m	SR	$\langle P \rangle$	$\langle nfe \rangle$	m	s	p_m	$\langle e \rangle$	SR	$\langle P \rangle$	$\langle nfe \rangle$	DE eq.
25	1	0	96%	5	5766	*5*	*5*	*0*	*4*	*100%*	*5*	*6429*	*(1)*
50	1	0	100%	5.1	13016	10	5	0	4.86	100%	5	16291	(1)
5	5	0.1	43%	4.30	4948	5	5	0	9.43	26%	5.06	4429	(2)
10	*5*	*0.1*	*100%*	*5.03*	*8650*	10	5	0	3.7	100%	5	8268	(2)

Table 6. Results for Ursem's test function

presence of crowding process. The multipopulation version of CDE is able to identify and to maintain multiple optima as long as the subpopulation size is at least ten. It converges quicker than CDE but, since the crowding technique is applied only at the subpopulations level, it has a lower explorative power. Further analysis on the hybrid technique is needed in order to establish its properties.

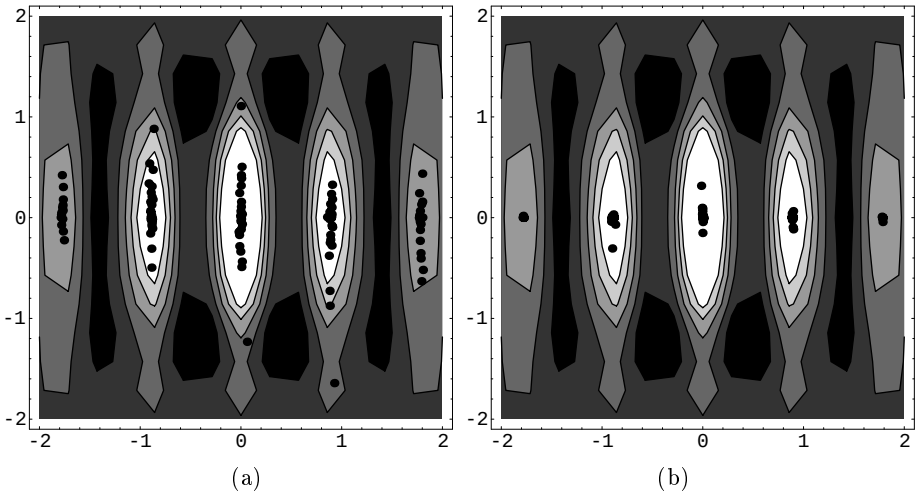


Fig. 6. The population after 100 iterations - Ursem test function. (a)CDE ($m = 100$, $s = 1$, $p_m = 0$) (b) MCDE ($m = 10$, $s = 10$, $p_m = 0.1$, $\tau = 10$)

References

1. H. A. Abbass, R. Sarker and C. Newton; PDE: A Pareto-frontier Differential Evolution Approach for Multi-objective Optimization Problems. In *IEEE Proc. of the Congress on Evolutionary Computation 2001* (CEC'2001), vol. **2**, pp. 971–978, 2001.
2. P.J. Ballester, J.N. Carter; Real-Parameter Genetic Algorithms for Finding Multiple Optimal Solutions in Multi-modal Optimization, in E. Cantú-Paz et al (eds.), *GECCO 2003*, LNCS 2723, pp.706-717, 2003.

3. D. Beasley, D.R. Bull, R.R. Martin; A Sequential Niche Technique for Multimodal Function Optimization, *Evolutionary Computing*, 1(2), pp. 101-125, 1993.
4. M. Bessaou, A. Ptrowski, P. Siarry; Island Model Cooperating with Speciation for Multimodal Optimization, *Parallel Problem Solving from Nature (PPSN 2000)*, Paris, France, September 16-20, 2000.
5. D.E. Goldberg, J. Richardson; Genetic Algorithms with Sharing for Multimodal Function Optimization, in J.J. Grefenstette (ed.), *Proc. of 2nd Int. Conf. of Genetic Algorithms*, Hillsdale, NJ:Lawrence Erlbaum Associates, pp. 41-49, 1987.
6. Z. V. Hendershot; A Differential Evolution Algorithm for Automatically Discovering Multiple Global Optima in Multidimensional, Discontinuous Spaces, *Proc. of MAICS 2004, Fifteenth Midwest Artificial Intelligence and Cognitive Sciences Conference*, Chicago IL, April 16-18, pp. 92-97, 2004.
7. C.H. Im, H.K. Kim, H.K. Jung; A Novel Algorithm for Multimodal Function Optimization Based on Evolution Strategy, *IEEE Trans. on Magnetics*, 40(2), pp. 1326-1329, 2004.
8. J. Lampinen, I. Zelinka; Mixed Integer-Discrete-Continuous Optimization by Differential Evolution, in P. Ošmera (ed.), *Proc. of Mendel'99*, Brno, June 9-12, pp. 71-84, 1999.
9. J. Lampinen; Solving Problems Subject to Multiple Nonlinear Constraints by the Differential Evolution, in R. Matoušek, P. Ošmera (eds.), *Proc. of Mendel'01*, Brno, June 6-8, pp.50-57, 2001.
10. J.P. Li, M.E. Balasz, G.T. Parks, P.J. Clarkson; A Species Conserving Genetic Algorithm for Multimodal Function Optimization, *Evolutionary Computation*, 10(3), pp. 207-234, 2002.
11. R.I.Lung, D. Dumitrescu; Roaming Optimization: a new Evolutionary Techique, in D.Petcu et al. (eds), *Proc. of SYNASC'03*, Timişoara, October 1-4, pp. 149-156, 2003.
12. N. K. Madavan; Multiobjective Optimization using a Pareto Differential Evolution Approach, *IEEE Proc. of Congress on Evolutionary Computation* (CEC'2002), vol. **1**, pp. 1145-1150, 2002.
13. S.W. Mahfoud; Crowding and Preselection Revisited, in R. Männer and B. Manderick (eds.),*Parallel Problem Solving from Nature*, vol. **2**, Elsevier, pp. 27-36, 1993.
14. Storn, K. Price; Differential Evolution; A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Technical Report TR-95-012, ICSI, 1995.
15. R. Thomsen; Multimodal Optimization Using Crowding-Based Differential Evolution, *Proc. of the 2004 Congress on Evolutionary Computation*, Portland, June 20-23, 2004.
16. K. Ursem; Multinational Evolutionary Algorithms, in *Proc. of Congress of Evolutionary Computation*, vol.**3**, pp. 1633-1640, 1999.
17. D. Zaharie; Control of Population Diversity and Adaptation in Differential Evolution Algorithms, in R. Matouek, P. Omera (eds.), *Proc. of Mendel'03*, Brno, June 6-8, pp. 41-46, 2003.
18. D. Zaharie; A Multipopulation Differential Evolution Algorithm for Multimodal Optimization, in R. Matoušek, and P. Ošmera (eds.), *Proc. of Mendel'04*, Brno, June 16-18, pp. 17-22, 2004.