Diversity Enhancing Mechanisms for Evolutionary Optimization in Static and Dynamic Environments

Daniela Zaharie, Flavia Zamfirache

West University of Timişoara, bv. Vasile Pârvan, no. 4, 300322,
e-mail: dzaharie@info.uvt.ro, zflavia@info.uvt.ro

Abstract: The population diversity is an important issue in evolutionary optimization, playing the main role in exploring the search space. The aim of this paper is to analyze the influence of some diversity enhancing mechanisms on the ability of two algorithms related to evolutionary optimization, differential evolution and particle swarm optimization, to solve static and dynamic optimization problems.

Keywords: static optimization, dynamic optimization, differential evolution, particle swarm optimization, population diversity.

1 Introduction

1.1 Particularities of Evolutionary Optimization in Static and Dynamic Environments

Evolutionary algorithms are population-based search methods inspired by natural evolution which are able to solve optimization problems. The success of the evolutionary search is highly dependent on a good balance between two antagonist processes: exploration and exploitation. Exploration allows searching the entire search space by ensuring the redirection of the search toward new regions, while exploitation favours a quick convergence toward the optimum. In many population-based algorithms the explorative power is influenced by the population diversity (a population consisting of almost identical elements has a low exploration power). During the evolution, the population usually loses its diversity, thus without a diversity enhancing mechanism the search process is not able to efficiently explore the search space. In the case of static optimization problems a low diversity could induce premature convergence, i.e. the search is stuck in a local optimum. In dynamic optimization, a population with a low diversity is not able to track the changing optima.

Optimization in dynamic environments is characterized by the variation in time of the parameters involved in the objective function or in the constraints. Real-world optimization problems are frequently of this type (e.g. dynamic scheduling, dynamic resource allocation, peptide identification, dynamic risk minimization [1]). While a static optimization problem consists of finding $x^* \in D \subset R^n$ such that $f(x^*) \geq f(x)$ for all $x \in D$ (in the case of a maximization problem) a dynamic one consists of finding for each time moment t the value $x^*(t)$ which satisfies $\tilde{f}(t, x^*(t)) \geq \tilde{f}(t, x)$ for all $x \in D$. Solving dynamic optimization problems is more challenging than solving static optimization problems and the difficulty of the problem is influenced by the severity of the optimum change.

There are two main types of optimum changes: *continuous* and *discontinuous*. In the continuous case the optimum follows a continuous trajectory in the search space. An example of a simple dynamic optimization problem with continuously changing optima can be defined starting from a static base function, *f*, as follows [2]: $\tilde{f}(t, x) = f(x - \Delta(t))$. If x^* is the optimum of f then $x^*(t) = x^* + \Delta(t)$ is the optimum of $\tilde{f}$ corresponding to moment *t*. The movement vector $\Delta(t)$ controls the optimum trajectory. If, for instance, $\Delta(t) = \delta \cdot t$ then the trajectory of the optimum is linear and $x^*(t + h) = x^*(t) + \delta \cdot h$ for any $h>0$. On the other hand, if $\Delta(t) = \delta \cdot \sin(2\pi \cdot t / T)$ then the optimum's trajectory is a periodic one. The severity of the change is determined by the value of $\delta > 0$. In the discontinuous case the optimum's trajectory is a discontinuous one meaning that the optimum can jump from one position to another one. Such behaviour could be obtained by using a multimodal base function and by allowing the random modification of the optima positions and values. Such an approach was used by Branke in developing the Moving Peaks Benchmark [4], a well-known generator of dynamic optimization test problems.

While in static optimization the main aim is to estimate as accurate as possible the optimum, in dynamic optimization the aim is to track as closely as possible the changing optimum. Problems with discontinuous changes are usually more difficult than those with continuous changes. If the change in the optimum position is too severe the only approach is to start a new search process. When the change is rather small the search process can benefit from the info gathered in the past avoiding the necessity of starting the search from scratch. The population-based search methods are good candidates for such a task because of their ability to explore the search space. This explains the large number of evolutionary approaches in dynamic optimization [5].

1.2 Diversity Enhancing Mechanisms in Dynamic Optimization

In the last decade a plethora of mechanisms having the aim of improving the behavior of population-based methods especially in the case of dynamic optimization have been proposed (for a review see [6]).

There are two main approaches in enhancing the population diversity necessary for an optima tracking process: a *reactive* one and a *proactive* one. The first approach consists in reacting to a change by triggering a diversity increasing mechanism (e.g. hypermutation, random immigrants). The main advantage is that between changes the search process is not altered allowing the algorithm to approach the current optimum. The main disadvantage is the necessity of implementing a change detection mechanism and of establishing the amount of diversity which should be introduced (e.g. mutation probability in the case of hypermutation or the percent of random immigrants). Such an approach is appropriate for discontinuous changes.

The proactive approach consists in maintaining the population diversity throughout the entire run of the algorithm. This means avoiding the convergence by diminishing the selection pressure (through sharing or crowding) or by directly stimulating the diversity (through random immigrants in each generation). The main advantage of such an approach is that it does not need a change detection mechanism while the main disadvantage is the slow convergence. Usually the algorithm will be able to follow the optimum but not too close. This approach is appropriate for continuous small changes in the objective function.

Another diversity preserving mechanism is based on using multiple populations. If each population maintain information about a different promising area of the search space the search process can easily follow the optimum. This approach can be of reactive type if a population is reinitialized when a change is detected or of proactive type when populations are always enforced to search different areas. The multi-population approach is appropriate for multi-modal objective functions.

The existence of many different mechanisms raises the problem of chosing the right mechanism for a given problem. Besides its effectiveness, a mechanism should be as simple as possible in order to not increase the computational cost of the algorithm. In the same time a given mechanism could have different impact on different population-based optimization methods.

The aim of this paper is to analyze some simple diversity enhancing mechanisms applied to two population-based optimization algorithms: differential evolution (DE) [10] and particle swarm optimization (PSO) [7]. The problem of diversity enhancing in DE and PSO algorithms is discussed in sections 2 and 3, respectively. Experiments on some test problems are presented in section 4.

2 Diversity Enhancing in Differential Evolution

Differential evolution (DE) introduced in [10] is a population-based optimization method based on a recombination operator which uses randomly selected parents and a simple selection procedure based on a competition between a parent and its offspring. The generational version of DE consists of constructing a new population $z = \{z_1, ..., z_m\}$ starting from the current population $x = \{x_1, ..., x_m\}$ by computing a set of candidates $y = \{y_1, ..., y_m\}$ as follows:

$$y_i^j = \begin{cases} \lambda x_*^j + (1-\lambda)x_{r1}^j + F_j \cdot (x_{r2}^j - x_{r3}^j), & \text{with probability } p_j \\ x_i^j, & \text{with probability } 1 - p_j \end{cases}, i = \overline{1,m},\ j = \overline{1,n} \tag{1}$$

where x_* is the best element of the current population, *r1*, *r2*, *r3* are random indices from $\{1,...,m\}$ and $\lambda \in [0,1]$, $F_j \in (0,2]$, $p_j \in (0,1]$ are control parameters. For a static maximization problem, the elements of the new generation are selected from the populations *x* and *y* depending on their quality with respect to the objective function *f*: $z_i = y_i$ if $f(y_i) > f(x_i)$ and $z_i = x_i$ otherwise. Frequently used variants are obtained for $\lambda = 0$ (DE/rand/1) and $\lambda = 1$ (DE/best/1).

This simple scheme proved to be efficient and robust in solving static optimization problems as long as the problem of premature convergence is avoided. Premature convergence means that the search is stuck in a local optimum because of a too low diversity in the population. Two mechanisms which proved to be successful for static optimization are [11]: (i) adapt the parameters p_j and F_j such that the recombination step avoid a too quick decrease of the population diversity; (ii) use multiple populations and a random migration strategy.

As diversity measure we used the variance computed for each component of the population elements: $Var(x^j) = Var(\{x_1^j, ..., x_m^j\})$. When $\lambda = 0$ we obtained a relation between the population variance before and after recombination:

$$Var(y^j) = (1 - 2p_j/m + p_j^2/m + 2p_jF_j^2)Var(x^j), \quad j = \overline{1,n} \tag{2}$$

In order to keep the diversity to almost the same level, at each generation, *t*, the ratio $c_j = \gamma Var(x^j)/Var(z^j)$ is computed and the parameters for the next generation are chosen such that

$$1 - 2p_j/m + p_j^2/m + 2p_jF_j^2 = c_j, \quad j = \overline{1,n}\,. \tag{3}$$

The parameter $\gamma > 0$ allows us to control the variance level (values larger than one favour an increase of diversity). At each generation, either the probability parameters, p_j, or the parameters F_j, are adjusted according to eq. (3). This diversity controlling process by parameters adaptation proved to be beneficial in avoiding premature convergence. Moreover since the parameters p_j and F_j can be randomly initialised it completely avoids the task of parameters tuning.

The multi-population approach is characterized by the fact that in each population an adaptive DE algorithm is applied and periodically a random migration process is started: any element of each subpopulation is, with a given migration probability, interchanged with a randomly selected element from an arbitrary population.

Despite their benefits for static problems these mechanisms are not powerful enough for tracking changing optima. Recently, in [9] some schemes for enhancing the DE behavior for dynamic optimization problems have been proposed. The main analyzed mechanisms are: (i) allowing some elements of the population to oscillate around the best element of the current population (e.g. brownian elements, quantum elements); (ii) using multiple populations and an exclusion method which forces each population to search for a different peak.

The problem which we analyze in this paper is to find the simplest mechanism which is still powerful enough to allow for tracking a changing optimum. In the case of continuous changes of the optimum position (without large jumps) it should be enough to perturb the current best element in order to arrive in the region of the new position of the optimum. Thus, using a “cloud” of elements wandering around the currently best element could ensure the ability of the algorithm to track the changing optimum.

In the DE with brownian elements the population is divided in two parts: a set $\{x_1,\dots,x_\mu\}$ of elements which evolves according to DE rules and another set of elements which just wander around the best element, x_*, of the previous generation:

$$y_i^j = x_*^j + K_j \cdot N(0,1),\quad i = \overline{\mu+1,m},\ j = \overline{1,n} \qquad (4)$$

where *N(0,1)* is a random value with a standard normal distribution and $K_j > 0$.

Numerical results obtained by applying these mechanisms for static and dynamic problems are presented in section 4.

3 Diversity Enhancing in Particle Swarm Optimization

Particle Swarm Optimization (PSO) [7] has been proposed almost in the same time as Differential Evolution. Its basic idea is to use a population of elements called particles, each one being characterized by a position vector, x_i, and by a velocity vector, v_i. It is a generational algorithm and at each generation the velocity is updated based on the relative positions of the particle with respect to its best position and with respect to the best position found by the entire population (called swarm). Unlike DE, the PSO algorithms do not use selection. The adjustments of the particles velocity and position satisfy:

$$\begin{cases} v_i^j(t+1) = \gamma(v_i^j(t) + \varphi_1 U(0,1)(p_i^j - x_i^j(t)) + \varphi_2 U(0,1)(p_g^j - x_i^j(t))) \\ x_i^j(t+1) = x_i^j(t) + v_i^j(t+1) \end{cases}, \ i=\overline{1,m},\ j=\overline{1,n} \tag{5}$$

where p_i is the best position of particle *i*, p_g is the best position found by the entire population, U(0,1) denotes a random value uniformly distributed in [0,1], $\gamma, \varphi_1, \varphi_2$ are positive coefficients which influence the particles dynamics. In order to ensure the convergence, the parameters $\gamma, \varphi_1, \varphi_2$ should be carefully chosen. Theoretical results on PSO convergence properties [5] give hints to choose appropriate values, e.g. $\varphi_1 + \varphi_2 > 4$ and $\gamma = 2/(\varphi - 2 + \sqrt{\varphi^2 - 4\varphi})$, $\varphi = \varphi_1 + \varphi_2$. Premature convergence can appear when the velocities become too small and the particles are frozen. A natural idea to solve premature convergence problems in PSO is to give an impulse to particles which stagnate either by implicitely increasing their velocity or by perturbing their position. Such an approach has been recently proposed in [8] where velocities smaller than a threshold, v_c, are replaced with a random value $U(-1,1)V_{\max}/\rho$. This variant is called Turbulent Particle Swarm Optimization (TPSO). We propose the use of a simpler approach based on perturbing the position of frozen particles instead of their velocities:

$$x_i^j(t+1) = \begin{cases} x_i^j(t) + v_i^j(t+1) & \text{if } v_i^j(t+1) \geq v_c \\ x_i^j(t) + K_j \cdot N(0,1) & \text{if } v_i^j(t+1) < v_c \end{cases} \tag{6}$$

Since any change in the position has an influence also on velocity, the particle can escape from the local optimum. Comparative results for these two diversity enhancing strategies both in the case of static optimization and in the case of dynamic optimization are presented in the next section.

4 Numerical Results and Discussion

The aim of the numerical experiments was to compare the diversity enhancing schemes presented in sections 2 and 3 for DE and PSO algorithms in the case of some classical static and dynamic test functions.

4.1 Static optimization

One of the functions for which both DE and PSO algorithms prematurely converges if their parameters are not very carefully chosen is Rastrigin's function:

$$f(x_1,...,x_n)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10),\quad x_i\in[-5.12,5.12] \tag{7}$$

This is a multimodal function and the evolutionary algorithms are easily trapped in its local minima. The global minimum is in (0,…,0) and its value is 0. We used this function for n=30 and tested the ability of some diversity enhancing algorithms to approach the optimum value within a precision of $\varepsilon=10^{-5}$. In order to collect the statistics (average and standard deviation of the number of generations until convergence) each algorithm has been independently run for 30 times. A run is considered to be successful if the optimum is approximated within the above specified precision in at most 5000 generations.

For DE algorithms the following diversity enhancing mechanisms have been analyzed: (i) parameters adaptation starting from random values of parameters (based on eq. (3)); (ii) multi-population approach with random migration (the migration process is started at each 100 generations and the migration probability is $p_m=0.1$; (iii) adaptive parameters plus brownian elements. These variants have been compared with DE having fixed parameters and randomly chosen parameters, respectively. The results presented in Table 1 illustrates the fact that the best behaviour is obtained using the adaptive variant and the fact that by introducing brownian elements the convergene is slow down. In all tests the population has 50 elements and the value of λ was 0. On the other hand using multiple populations (5 populations of size 10) and a random migration has beneficial influence even in the case when the parameters p and F are randomly chosen.

The classical PSO algorithm with recommended values for the parameters [5]: $\varphi_1=\varphi_2=2.833$, $\gamma=0.6$ prematurely converges for Rastrigin's function. As results presented in Table 2 suggest, both the Turbulent Particle Swarm Optimization (TPSO) introduced in [8] and the simple perturbed PSO proposed in Section 3 are effective. On the other hand by combining them with a multi-population approach the convergence is significantly slow down. The results in

Table 2 have been obtained for populations of 50 elements (in multi-population variant 5 populations of size 10 have been used). A run is considered to be successful if the algorithm found the optimum in at most 10000 generations.

By analyzing the results presented in Tables 1 and 2 it follows that DE is slightly faster than PSO.

	p=0.5 F=0.5	**Random parameters**	**Adaptive parameters** ($\gamma = 1.025$)	**Random parameters + migration**	**Adaptive parameters + brownian elements**
% of succ.	100%	6%	100%	76,6%	100%
< gen>	1448.3	3543	1042.53	2290.91	3923.8
stdev(gen)	23.006	839.5	42.79	570.184	399.154

Table 1. Results of DE variants for Rastrigin test function.

	TPSO ($V_c = 10^{-7}, \rho = 2$)	**Perturbed PSO** ($V_c = 10^{-7}, K = 1$)	**TPSO+ migration**	**Perturbed PSO + migration**
% of succ.	100%	100%	49%	50%
< gen>	2689	2706	8010	7094.4
stdev(gen)	783.19	769	1229	1879.8

Table 2. Results of PSO variants for Rastrigin test function.

4.2 Dynamic optimization

To analyze the impact of the diversity enhancing schemes for dynamic optimization problems we used as test functions a dynamic variant of Ackley's function and a test function from the moving peaks benchmark [4].

The dynamic Ackley's function is obtained by linearly moving the optimum:

$$f(t, x_1, \ldots, x_n) = 20 + e - \exp\left(-\frac{0.2}{\sqrt{n}}\sum_{i=1}^{n}(x_i - s_i(t))^2\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi(x_i - s_i(t)))\right)$$

$$x_i \in [-32,32],\quad s_i(t+1) = s_i(t) + \delta,\ \delta \in R$$

At each generation the optimum coordinates are increased with δ. When the optimum reaches the boundary of the domain the sign of δ is changed. This is similar with the linear trajectory proposed in [2]. In numerical tests the absolute value of δ is set to 0.1 and $n=30$. The evolution of the averaged distance between the best element in the population and the current optimum is illustrated in Figure 1 for some variants of the DE algorithm. The analysis is made for 5000 generations, a population of size 50, a moving step $\delta = 0.1$ and 10 independent runs of the algorithms. The main remarks are: if the parameter γ of the adaptive scheme in DE is high enough ($\gamma = 1.25$) the behaviour is better than in the non-adaptive case (p=0.5, F=0.5). For smaller values of γ (e.g. $\gamma = 1$) the adaptive DE is not able to track the optimum (Figure 1 (d)). On the other hand, using a percent between 25% and 50% of brownian elements the behaviour of the algorithm is significantly improved. It should be remarked that in order to track the optimum it is necessary to reevaluate at each generation the elements of the current population.

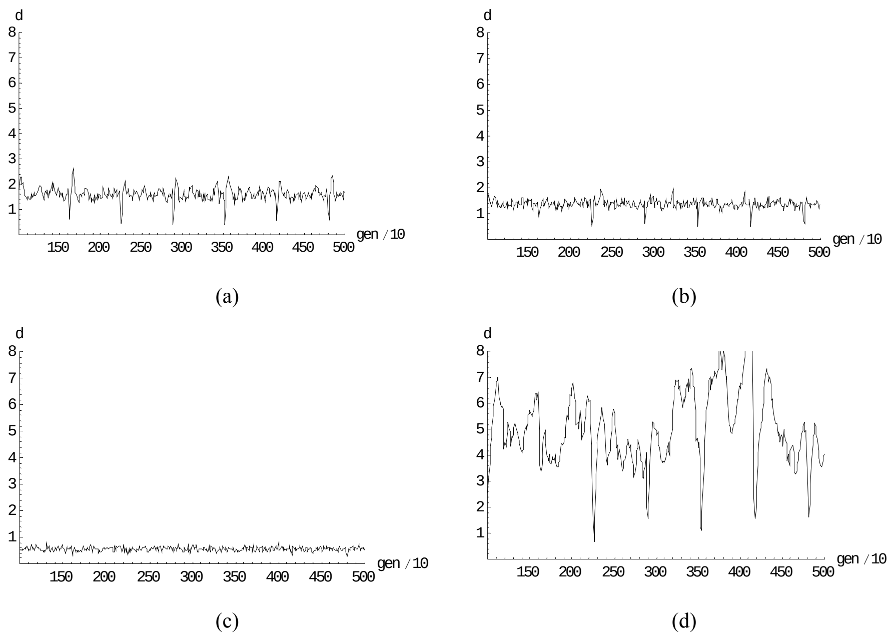

Figure 1

Behaviour of DE for dynamic Ackley function. (a) DE with fixed parameters (p=0.5,F=0.5) ; (b) Adaptive DE with $\gamma = 1.25$; (c) DE with fixed parameters (p=0.5, F=0.5) and 50% brownian elements; (d) Adaptive DE with $\gamma = 1$

The Moving Peaks Benchmark [4] offers a flexible way to generate dynamic test functions. The test function which we used is of the following form:

$$f(x) = \max_{j=1,k} f_j(x), \quad f_j(x_1,...,x_n) = \frac{h_j}{1 + w_j \sum_{i=1}^{n} (x_i - p_j^i)^2} \tag{8}$$

The tests have been conducted for a particular case characterized by *n=5* and only one moving peak (*k=1*). At each 10000 function evaluations the position (vector *p*), the height (*h*) and the width (*w*) are randomly modified. This is similar to a discontinuous trajectory of the optimum. The measures which we used to evaluate the algorithms behavior are the current error, *e(t)*, and the offline error, *o(T)*:

$$e(t) = f(t, x^*(t)) - f(t, x_*(t)), \quad o(T) = \frac{1}{T} \sum_{t=1}^{T} (f(t, x^*(t)) - f(t, x_*(t))) \tag{9}$$

where $x^*(t)$ is the true optimum and $x_*(t)$ is the currently best element of the population.

For DE algorithms we analyzed the impact of parameters adaptation and of brownian elements. As is illustrated in Figure 2 the adaptive DE has an acceptable behavior but not as good as if a small percent of brownian elements is used.

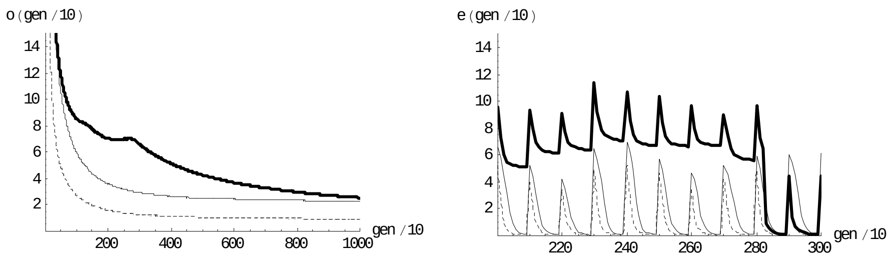

Figure 2

Behaviour of DE for one moving peak: offline error (left), current error (right). DE variants: adaptive DE $(\gamma = 1)$ + 50% brownian elements (thick line) , adaptive DE $(\gamma = 1)$ + 25% brownian elements (dashed line), adaptive DE $(\gamma = 1.25)$ without brownian elements (thin line).

Since the convergence of PSO algorithms is slower than the convergence of DE algorithms it would be expected that classical PSO algorithms are able to track slowly moving optima. These is illustrated in Figure 3 which also suggest that for simple dynamic problems there is no need for supplementary diversity enhancing schemes (as position/velocity perturbation or brownian elements).

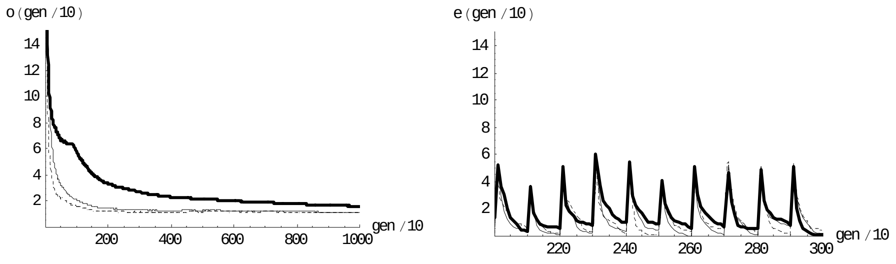

Figure 3
Behaviour of PSO for one moving peak. offline error (left), current error (right). PSO variants: classical (dashed line), perturbed PSO (thin line), PSO + 25% brownian elements (thick line).

Conclusions

For static problems, the ability of DE to avoid premature convergence is significantly improved by the parameter adaptation scheme and the multi-population approach. Using brownian elements in DE algorithms slows down the convergence but used without other mechanisms cannot prevent premature convergence. On the other hand it proved to be a valuable mechanism in tracking changing optima.

Both the Turbulent PSO introduced in [8] and the simple perturbed PSO proposed in Section 3 proved to be effective in avoiding premature convergence. Concerning the multi-population approach the numerical experiments suggest that it slows down the convergence of PSO but it is not able to avoid premature convergence by itself.

In the case of relatively slowly changing optima good results are obtained using simple diversity enhancing mechanisms (e.g. parameter adaptation which favor large values of the variance or brownian elements in the case of DE algorithms) or even no supplementary mechanisms (as in the case of PSO algorithms). However when the severity of change is higher supplementary schemes (e.g. exclusion [9], charged PSO [3]) should be used.

References

[1] Peter J. Angeline: Tracking Extrema in Dynamic Environments, in Proc. 6th International Conference of Evolutionary Programming, Indianapolis, Indiana, USA, April 13-16, 1997 (ed. P.J. Angeline at al.) LNCS 1213, pp. 335-345.

[2] Marcus Andrews, Andrew Tuson: Dynamic Optimization: An Investigation of Practitioners Requirements, in Proceedings of the The 24th Annual Workshop of the UK Planning and Scheduling Special Interest Group (PlanSIG 2005), London, UK.

[3] T.M. Blackwell, P.J. Bentley: Dynamic Search with Charged Swarms, Proc. of Genetic and Evolutionary Computation Conference, 2002, pp. 19-26.

[4] Jurgen Branke: Moving Peaks Benchmark, www.aifb.uni-karlsruhe.de/~jbr/MovPeaks/movpeaks/

[5] Maurice Clerc, James Kennedy: The Particle Swarm: Explosion, Stability, and Convergence in a Multi-Dimensional Complex Space, IEEE Transactions on Evolutionary Computation, 2002, vol. 6, pp. 58-73.

[6] Yaochu Jin, JurgenBranke: Evolutionary Optimization in Uncertain Environments, IEEE Transactions on Evolutionary Computation, vol. 9, no. 3, 2005, pp. 303-317.

[7] James Kennedy, Russell Eberhart: Particle Swarm Optimization, Proc. IEEE Int'l. Conf. on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, 1995, pp. 1942-1948.

[8] Hongbo Liu, Ajith Abraham: Fuzzy Adaptive Turbulent Particle Swarm Optimization , Proc. of fifth International Conference on Hybrid Intelligent Systems, Rio de Janeiro, November 6-9, 2005, 445-450.

[9] Ruy Mendes, Arvind Mohais: DynDE: a Differential Evolution for Dynamic Optimization Problems, Proc. of CEC'2005. Edinburgh, Scotland, September 2-5, 2005, 2808-2815.

[10] Rainer Storn, Keneth Price: Differential Evolution – A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces, Technical Report TR-95-012, ICSI, March, 1995.

[11] Daniela Zaharie: Control of Population Diversity and Adaptation in Differential Evolution Algorithms, Proc. of 9th Intern. Conference of Soft Computing – Mendel 2003, June 4-6, 2003, pp. 41-46.