Base Changing Strategies in an Adaptive Representation Evolutionary Algorithm

Crina Grosan*, Daniela Zaharie**

*Faculty of Mathematics and Computer Science, "Babes Bolyai" University, Kogalniceanu 1, 3400, Cluj-Napoca, e-mail: cgrosan@cs.ubbcluj.ro

**Faculty of Mathematics and Computer Science, West University of Timisoara, V. Parvan, 4, 300223 Timisoara, e-mail: dzaharie@info.uvt.ro

Abstract: The aim of this work is to analyze the influence of the representation's modification on the behaviour of an adaptive representation evolutionary algorithm based only on a mutation operator. A statistical analysis of the mutation properties suggests some strategies for the modification of the representation's base. The influence of these strategies is numerically analyzed on some classical test problems of single and multi-objective optimization.

Keywords: evolutionary algorithms, adaptive representation, single and multi-objective optimization

1 Introduction

The design of an evolutionary algorithm (EA) for a given problem involves many choices concerning: representation, fitness function, evolutionary operators (e.g. selection, mutation, recombination), control parameters etc. In the absence of some well-established design rules, the adaptation of the characteristics of an EA during the evolution becomes an important issue. Different adaptation approaches concerning parameters or representation have been proposed [3].

The basic idea of adaptive representation is that by changing the representation one obtains a reshaping of the search space ensuring in this way the use of multiple search heuristics [6], [7] . Such an approach has been applied for genetic algorithms in order to overcome the disadvantages of the binary encoding, either by using a Gray code [1] or by changing the base of the representation from 2 two other values [6]. The idea of changing the base of representation during the evolution has been introduced in [6] where some adaptive representation algorithms have been proposed. The main idea is to randomly change the

representation base after a given number of generations or when a random event occurs.

Starting from this idea, in [4] is introduced a new technique, called Adaptive Representation Evolutionary Algorithm (AREA) which can be applied for solving single and multiobjective optimization problems. The main particularity of AREA is that it relies only on mutation and after a number of generations in which no improvement occurs, the representation alphabet (i.e. the base) is changed by randomly modifying the representation base.

In this paper we analyze the influence of the representation on the mutation properties and try to find efficient strategies for changing the alphabet, starting from some theoretical insights. The next section presents the AREA technique. Section three contains a statistical analysis of the influence of the representation base on the mutation properties while section four presents results of numerical experiments on some classical single and multiobjective optimization problems. The ability of the AREA to solve a real-world problem of parameter estimation is compared with that of the differential evolution algorithm [8].

2 The AREA technique

In the AREA technique an individual consists of a pair (x, B) where x is a string encoding object variables and B specifies the base used for encoding x. B is an integer number, $B \geq 2$, and x is a string of symbols from the alphabet $\{0, 1,..., B\text{-}1\}$. If $B = 2$, the standard binary encoding is obtained. A chromosome encoded over a higher alphabet has a shorter length than a chromosome encoding the same value (point in the search space) at the same precision but over a lower alphabet.

The only transformation operator in AREA is mutation, thus AREA can be applied also for populations of a single individual. Mutation can modify the object variables as well as the last position (specifying the representation base). Thus the encoding is adaptive, i.e. may be changed during the search process as an effect of the mutation operator. When it is applied to a gene belonging to the object variable substring (x - part of the chromosome) the mutation consists in replacing the current value of the gene with a randomly one from the current alphabet. For each gene the mutation is applied with a given mutation probability, p_m. The offspring obtained by mutation will replace its parent only if it is better. A mutation generating an offspring worse than its parent is called a *harmful mutation*. Unlike the algorithms proposed in [6], in AREA the base is changed only when a predefined (consecutive) number of mutations applied to an individual do not improve its quality. Thus the representation changes in an adaptive manner being determined by the behaviour of the algorithm.

The structure of AREA follows the general structure of an evolutionary algorithm: *(i) initialisation* (during the initialisation, each AREA individual is encoded over a randomly chosen alphabet); *(ii)* iterative application of *mutation* (it is applied with probability p_m to each individual) and *selection* (if the offspring obtained by

mutation is better than its parent, then the offspring enters in the new population). If the number of successive harmful mutations for an individual exceeds a prescribed threshold (denoted by MAX_HARMFUL_MUTATIONS), then the individual representation is changed and it enters in the new population with this new representation. Otherwise, the individual (the parent) enters unchanged in the next generation. The reason behind this mechanism is to dynamically change the individual representation whenever it is necessary. If a particular representation has no potential for further exploring the search space, then the representation is changed. It is hoped that in this way the search space will be explored more efficiently.

This simple algorithm can be applied both for single objective and for multiobjective optimization (by considering in the selection step that the offspring is better if it dominates its parent). In implementing an adaptive representation algorithm at least two questions arise: *(i)* When the base should be changed? *(ii)* How should be chosen the new representation? While the first question has been answered in [4] the second is still an open one. Up to now mainly random base changes have been implemented. In the rest of this paper we will analyze different base changes strategies.

3 Influence of the representation on the mutation properties

In this section we present a statistical analysis of the influence of the representation on the exploration ability of the mutation operator. Such an analysis could be useful to find strategies for changing the representation.

Let us consider that each real value $x \in [a,b]$ is transformed into an integer value, $v \in \{0, 1, ..., V_{max}\}$ by the transformation $v=[(x-a)/(b-a)V_{max}]$ [1]. This integer value can be represented in a base $B \in \{2,...,B_{max}\}$ by a string of $q(B)=[\log_B V_{max}]$ symbols from $\{0, 1, ..., B-1\}$. Thus each real component of a vectorial individual is represented by a string, $S(v)$, of $q(B)$ integer values from $\{0, 1, ..., B-1\}$. When the base B is changed, the length of $S(v)$, $q(B)$, is also changed. Since mutation consists in modifying elements in the string $S(v)$, by changing the base the effect of the mutation can be altered. We analyze the influence of such a change on some statistical characteristics: the *expected number of mutated genes*, the *expected value obtained after mutation* and the *transition probability* between two values. If the probability of mutating a gene is p_m , then the expected number of mutated genes, $m(B)$, satisfies:

$$m(B) = \sum_{i=1}^{q(B)} (1 \cdot p_m + 0 \cdot (1 - p_m)) = p_m q(B) \tag{1}$$

[1] [.] denotes the upper integer part of a real number.

If p_m does not depend on $q(B)$ then by increasing B, $q(B)$ decreases, thus $m(B)$ also decreases. On the other hand, if $p_m=c/q(B)$, then $m(B)=c$, thus the expected number of mutated genes is constant. In the following we will consider that $p_m=c/q(B)$.

To compute the expected value after mutation, let us consider a given integer value $v \in \{0, 1, ..., V_{max}\}$ encoded in the base B by a string $S(v)=(v_0, v_1, ..., v_{q(B)-1})$.

By mutation, each element of $S(v)$ can be replaced with a randomly selected value from $\{0,1,...,B-1\}$. Denoting with W_i the random variable corresponding to the mutated ith component of $S(v)$, its probability distribution satisfies:

$$P(W_i = j) = p_m / B \quad \text{for all} \quad j \in \{0,1,..., B-1\},\ j \neq v_i \tag{2}$$

$$P(W_i = v_i) = (1 - p_m) + p_m / B \tag{3}$$

Thus the expected value of W_i is:

$$E(W_i) = \sum_{j=0}^{B-1} j \cdot P(W_i = j) = \frac{p_m}{B} \sum_{j=0}^{B-1} j + (1 - p_m) v_i = p_m \frac{B-1}{2} + (1 - p_m) v_i \tag{4}$$

The random variable, W, associated with the string $S(W)=(W_0,W_1, ..., W_{q(B)-1})$, verifies $W=W_{q(B)-1}B^{q(B)-1}+...+W_1B+W_0$, thus its expected value can be computed as follows:

$$\begin{aligned} E(W) &= \sum_{i=0}^{q(B)-1} E(W_i) B^i = \sum_{i=0}^{q(B)-1} (p_m (B-1)/2 + (1-p_m) v_i) B^i = \\ &= p_m \frac{B-1}{2} \sum_{i=0}^{q(B)-1} B^i + (1-p_m) \sum_{i=0}^{q(B)-1} v_i B^i = p_m \frac{B^{q(B)}-1}{2} + (1-p_m) v \end{aligned} \tag{5}$$

Thus if p_m does not depend on $q(B)$, then $E(W)$ is not significantly influenced by the value of B because $B^{q(B)} \approx V_{max}$. On the other hand, if p_m depends on $q(B)$ (for instance $p_m=1/q(B)$) then $E(W)$ varies when B varies. As B is greater the expected modification on v is greater.

The influence of the base value on the mutation properties can be also analyzed by the associated probability distribution function (*pdf*). To obtain information on the *pdf* we computed the transition probabilities between two values v and w. Let us consider that these values are represented in base B by $q(B)$ symbols: $S(v)=(v_0, v_1, ..., v_{q(B)-1})$ and $S(w)=(w_0, w_1, ..., w_{q(B)-1})$, respectively. The probability of transition of gene v_i into gene w_i is:

$$P(v_i \to w_i) = \begin{cases} p_m / B & \text{if } v_i \neq w_i \\ (1-p_m) + p_m / B & \text{if } v_i = w_i \end{cases} \tag{6}$$

Since the mutation is independently applied (with probability p_m) on each gene it follows that the transition probability between v and w is:

$$P(v \to w) = \prod_{i=0}^{q(B)-1} P(v_i \to w_i) = \left(\frac{p_m}{B}\right)^{q\neq} \left((1-p_m) + \frac{p_m}{B}\right)^{q=} \qquad (7)$$

where q denotes the number of identical genes in the encoding of v and w while $q_{\neq}$ denotes the number of different genes. As is illustrated in Fig. 1 and 2, the base value highly influences the transition probabilities, i.e. the probability distribution corresponding to mutation. This remark agrees with a largely accepted opinion that by changing the base one can change the search abilities of the algorithm. However there exist bases for which the corresponding transition probabilities are similar (e.g. bases 2, 4 and 8 in Fig. 1). Changing between such bases may not assure an increase of the search ability because some regions have low transition probability in all cases. On the other hand choosing successive values of the bases ensures a better cover of the search space through mutation (see Fig. 2). These remarks suggest that analysing the base changing strategy could be useful in practice. The current implementations of the adaptive representation algorithms [4], [6] are based on a random selection of the new representation base. Other strategies can be taken into consideration. Some examples are: *restricted random selection* (e.g. random powers of 2) or *deterministic selection* (e.g. successive values taken in circular manner). Unfortunately choosing the adequate strategy is a difficult problem, mainly due to the fact that the same strategy can be good for a given problem and bad for another one. Thus to find out some information on the influence of base changes strategies we have to perform some numerical experiments.

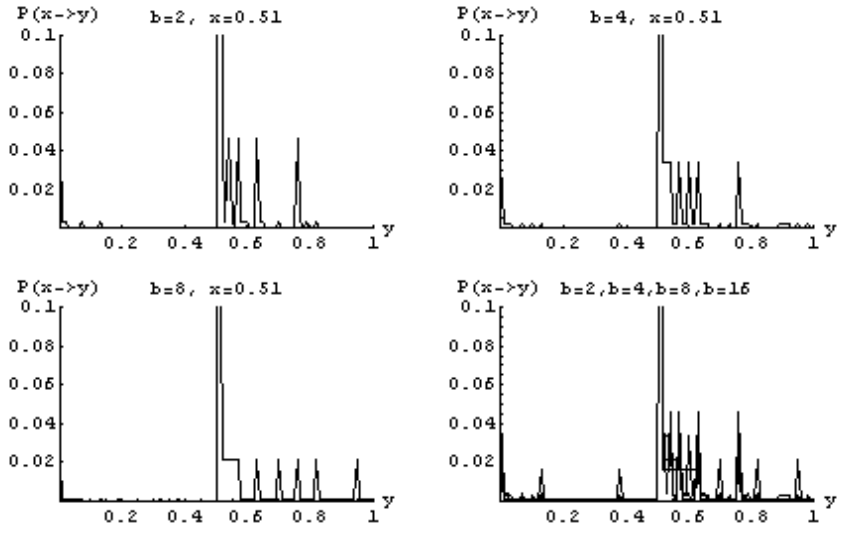

Fig. 1. Transition probabilities for bases that are powers of 2

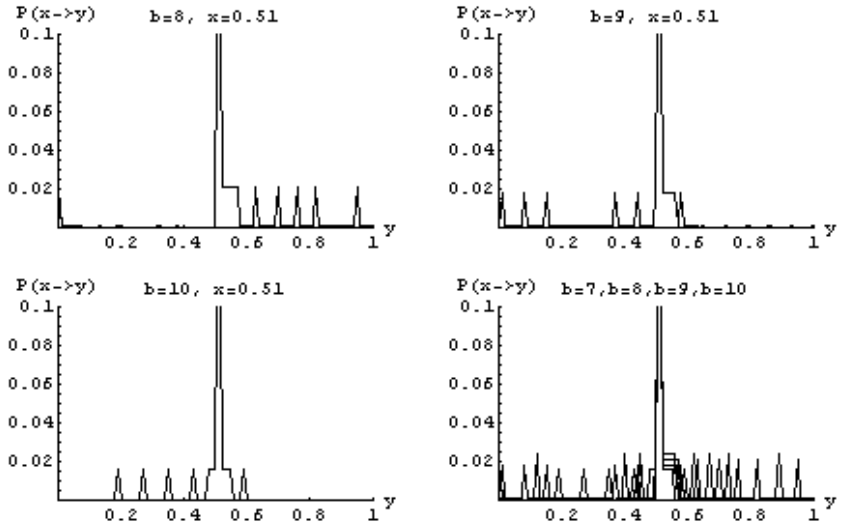

Fig.2. Transition probabilities for some consecutive values of the base (7, 8, 9, 10)

4 Experimental analysis

The first aim of this experimental analysis is to compare some base changing strategies. We started with the AREA technique proposed in [4] and we replaced the purely random strategy with some controlled ones. The goal is to get some insights on the influence of the base changing strategy starting from the results obtained for some classical optimization problems with single or multiple objectives. In all tests AREA used a single individual and the mutation probability was $p_m=c/(nq(B))$. All the results are averages obtained for 30 independent runs. The following base change strategies have been tested: *(i) Purely random*: the new base is randomly selected; *(ii) Random powers of 2*: the new base is a random power of 2; *(iii) Restricted random:* the new base is randomly selected such that the difference between the old and the new base is less than 5; *(iv) Deterministic:* the new base is obtained by incrementing the current one in a cyclic manner (e.g. B is replaced with $B+1$ and B_{max} is replaced with 2).

4.1. Single objective test problems

The parameters used in this case were: $p_m=1/(nq(B))$, B_{max}=32, $V_{max}=2^{30}-1$, MAX_HARMFUL_MUTATIONS=5. The single objective test functions are presented in Table 1. Test function f_1, also known as Griewangk's function has many widespread local minima. The locations of the local minima are regularly distributed. Test function f_2, also known as Rosenbrock's valley is a classic optimization problem, also known as Banana function. The global optimum is

inside a long, narrow, parabolic shaped flat valley. To find the valley is trivial, however convergence to the global optimum is difficult.

Test functions	Domain	Global minimum
$f_1(x) = \frac{1}{4000} \cdot \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-500,500]^n$	$x^0 = (0,\ldots,0) \quad f(x^0) = 0$
$f_2(x) = \sum_{i=1}^{n-1} 100 \cdot (x_{i+1} - x_i^2)^2 + (1 - x_i)^2$	$[-5,5]^n$	$x^0 = (0,\ldots,0) \quad f(x^0) = 0$
$f_3(x) = 10 \cdot n + \sum_{i=1}^{n} (x_i^2 - 10 \cdot \cos(2 \cdot \pi \cdot x_i))$	$[-2,2]^n$	$x^0 = (0,\ldots,0) \quad f(x^0) = 0$

Table 1. Single objective test functions used in experiments.

Test function f_3, also known as Rastrigin's function is based on unimodal function proposed by DeJong with the addition of a cosine modulation to produce many local minima. Thus, the test function is highly multimodal. However, the locations of the minima are regularly distributed. The results in Table 2 contain the value of the objective function obtained after the given number of function evaluations.

Test functions	Base changing strategy			
	Function evaluations	Purely random	Random powers of 2	Deterministic
f_1	100000	0.06535	0.10452	**0.05118**
f_2	500000	**5.62107**	47.8586	8.09402
f_3	100000	0.01127	0.08238	**0.00000053**

Table 2. Results for the single objective problems

These results suggest that the base changing strategy influences the AREA behavior for some test functions. As the theoretical analysis implies using random powers of 2 one obtains worse results than for the other strategies.

4.2. Multi-objective test problems

Multiobjective test functions used in these experiments have been introduced in [2]. These functions are built by using three functions f_1, g, h. The bi-objective function T considered here is: $T(x) = (f_1(x), f_2(x))$. The optimization problem is:

$$\begin{cases} Minimize\ T(x),\ where\ f_2(x) = g(x_2,\ldots x_m)\, h(f_1(x_1), g(x_2,\ldots x_m)), \\ \qquad\qquad\qquad\qquad x = (x_1,\ldots,x_m) \end{cases}$$

The first problem is the test function ZDT_4. This function contains 21^9 local Pareto optimal fronts and, therefore, it tests the EA ability to deal with multimodality. The involved functions are defined by:

$$f_1(x) = x_1; \quad g(x_2,\ldots,x_m) = 1+10(m-1)+\sum_{1=2}^{m}(x_i^2-10\cos(4\pi x_i)); \quad h(f_1,g) = 1-\sqrt{f_1/g}$$

where m = 10, $x_1 \in [0,1]$ and $x_2,\ldots,x_m \in [-5,5]$. Global Pareto optimal front is characterized by the equation $g(x) = 1$. The best local Pareto optimal front is described by the equation $g(x) = 1.25$. Note that not all local Pareto optimal sets are distinguishable in the objective space. The second problem is test function ZDT_6. This function includes two difficulties caused by the nonuniformity of the search space. First, the Pareto optimal solutions are nonuniformly distributed along the global Pareto optimal front (the front is biased for solutions for which $f_1(x)$ is near one). Second, the density of the solutions is lowest near the Pareto optimal front and highest away from the front. This test function is defined by:

$$f_1(x_1) = 1-\exp(-4x_1)\sin^6(6\pi x_1); \quad g(x_2,\ldots x_m) = 1+9\cdot(\sum_{i=2}^{m}x_i/(m-1))^{0.25}; \quad h(f_1,g) = 1-(f_1/g)^2$$

where $m = 10$, $x_i \in [0,1]$, $i = 1,2,\ldots m$. The Pareto optimal front is characterized by the equation $g(x) = 1$, and is nonconvex.

In order to analyze the results obtained for multiobjective test functions a distance metric proposed in [5] is used. Assume that the Pareto front is known. Let us denote by P a set of Pareto optimal solutions. For each individual i from the final population, the FP distance (Euclidean distance or another suitable distance) d_{ij} to all points j of P is computed. The minimum distance: $mindist_i = \min_{j\in P} d_{ij}$ is kept for each individual. The average of these distances:

$$DM = \frac{\sum_{i\in FP} mindist_i}{|FP|} \tag{8}$$

represents a measure of convergence to the Pareto front. Lower values for DM indicate a better convergence. The values of the distance metric for the test functions ZDT_4 and ZDT_6 are presented in Table 3. These values have been obtained for: $p_m=1/(nq(B))$, B_{max}=15, MAX_HARMFUL_MUTATIONS=50, number of function evaluations: 20000. From Table 3 we can see that for the test function ZDT_4 the result obtained by using only powers of two is almost similar to the result obtained by standard AREA (purely random). On the other hand in the case of ZDT_6 this strategy seems to be not appropriate. Using the restricted random strategy better results are obtained for ZDT_4. Increasing the maximal value of harmful mutations the results obtained by AREA is very good for both considered test functions.

Test function	Base changing strategy			
	Purely random	**Random powers of 2**	**Restricted random**	**AREA with MAX_HARM_MUT= 5000**
ZDT_4	0.97518	0.97229	**0.43411**	0.5556
ZDT_6	0.1042	0.69992	0.10491	**0.0024**

Table 3. Results obtained for test functions ZDT_4 and ZDT_6.

4.3. A parameter estimation application

The second aim of the experimental analysis was to compare AREA and an evolutionary algorithm based on a real valued representation. We considered the problem of fitting some experimental data with a model based on six kernel functions of bigaussian type. Each bigaussian has three parameters and the goal was to find the parameters of the involved bigaussians and also the coefficients of the linear combinations of the bigaussians (thus 24 parameters have to be determined). We tried to solve this problem by using both a "differential evolution" (DE) algorithm [8] and the AREA technique. The differential evolution algorithm proved to be efficient in parameter estimation problems but its behaviour is highly dependent on its control parameter values. To compare the behavior of the two algorithms we computed the mean squared error (MSE) obtained by using the same number of function evaluations. The main remark is that when its parameters are carefully chosen, DE performs better than AREA, but AREA depends on fewer parameters and the differences between results obtained using different base changing strategies are not as large as in the case of DE, when a control parameter value (e.g. CR) is changed.

DE (gen.: 500, population size: 100)		**AREA (function evaluations: 50000)**	
Control parameters	**MSE**	**Base changing strategy**	**MSE**
CR=0.9, F=0.5	**0.00086**	Purely random	0.00559
CR=0.2, F=0.5	0.13370	Random powers of 2	0.01895
Adaptive DE [9]	0.02084	Deterministic	**0.00441**

Table 4. Results for the parameters estimation problem

Conclusions

The strategy of changing the base in AREA technique may influence the search ability of the algorithm but this influence depends on the problem. For a parameter estimation problem, AREA proved to be competitive with respect to the DE algorithm.

Taking into account the No Free Lunch Theorems (NFL) [10], we cannot say that one of the strategies used for changing the representation is better than the others for all optimisation test problems. One can suggest the adequate strategy only for particular problems.

References

[1] L.Barbulescu, J.P.Watson, L.D. Whitley: Dynamic Representations and Escaping Local Optima: Improving Genetic Algorithms and Local Search, 17th National Conference on Artificial Intelligence, 2000, pp. 879-884

[2] K. Deb: Multi-objective genetic algorithms: Problem difficulties and construction of test functions. *Evolutionary Computation*, 7(3), 1999, pp. 205-230.

[3] A.E. Eiben, R. Hinterding, Z. Michalewicz: Parameter Control in Evolutionary Algorithms, IEEE Trans. on Evoliutionary Algorithms, 3(2), 1999, pp. 124-141.

[4] C. Grosan, M. Oltean: Adaptive Representation Evolutionary Algorithm – a new technique for single objective optimization. In Proceedings of First Balcanic Conference in Informatics (BCI), Thessaloniki, Greece, 2003, pp. 345-355.

[5] C. Grosan: How to compare the multiobjective evolutionary algorithms performances? Academicals Days of Babes-Bolyai University, 2003.

[6] J. Kingdon , L. Dekker: The Shape of Space, Proceedings of the First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA '95) IEE, London, 1995, pp. 543-548.

[7] N. Schraudolph, R. Belew: Dynamic Parameter Encoding for Genetic Algorithms, CSE TR #CS 90-175, 1990.

[8] R.Storn, K. Price: Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continous Spaces, TR-95-012, ICSI, 1995.

[9] D. Zaharie: Control of Population Diversity and Adaptation in Differential Evolution Algorithms, in R. Matousek, P. Osmera (eds), Proc. of Mendel 2003, 9th Intern. Conf. on Soft Computing, 2003, pp. 41-46

[10] D.H. Wolpert, W.G. Macready: No free lunch theorem for search. Technical Report SFI-TR-95-02-010. Santa Fe Institute, USA 1995.