Control and Cybernetics

vol. **29** (2000) No. 4

Correlational parameter tuning by genetic meta-algorithm

by

P. Kieś[1] and W. Kosiński[1,2]

[1]Institute of Fundamental Technological Research,
Polish Academy of Sciences (IPPT PAN),
21 Świętokrzyska Str., 00-049 Warsaw, Poland
e-mail: pkies@ippt.gov.pl

[2]Polish-Japanese Institute of Information Technology,
86 Koszykowa Str., 02-008 Warsaw, Poland
e-mail: wkos@pjwstk.waw.pl

Abstract: The general problem of an off-line parameter tuning in the Binary Genetic Algorithm ($\mathcal{BGA}$) is introduced. An example of such a tuning: a class of Correlational Tuning Methods (CTMs) is proposed. The main idea of a CTM is that it uses a mapping called measurement function as an assessment of the $\mathcal{BGA}$'s efficiency. An example of a measurement function is described and two examples of CTMs: a modified "trials and errors" method and a modified genetic meta-algorithm (meta$\mathcal{BGA}$) are shown. Finally, experimental results with the meta$\mathcal{BGA}$ for four kinds of test fitness functions, where the code permutation is the tuned parameter, are presented.

Keywords: genetic algorithm, parameter tuning, adaptation, optimization, code permutation.

1. Introduction

As stated in Wolpert and Macready (1995), there is no universal optimization algorithm equally good for all possible fitness functions. So the only way to improve performance in optimization is to choose a suitable algorithm's variant and its parameters for a given problem.

In the case of the Binary Genetic Algorithm ($\mathcal{BGA}$) we do not want to modify the algorithm structure but we are looking for a method for adjusting the parameters of the $\mathcal{BGA}$ for a given problem. Of course, for real problems we are especially interested in such methods that do not need to know the global maximum of a fitness function. The existing parameter adjustment methods

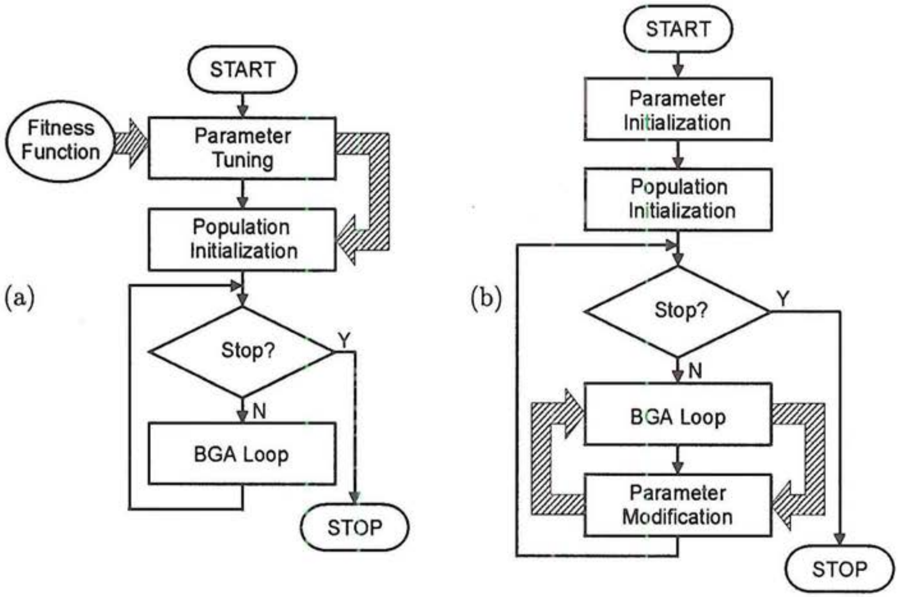

Figure 1. Typical schemes of parameter adjustment methods in the $\mathcal{BGA}$: (a)—off-line method, (b)—on-line method

- off-line methods (parameter tuning in Eiben et al., 1999)—parameters are adjusted according to measurements of the solved fitness functions before the $\mathcal{BGA}$ is run (see Kieś, 1998 and Kieś and Kosiński, 1998),
- on-line methods (adaptation or parameter control in Eiben et al., 1999)—parameters are adjusted while the algorithm is working, basing on measurements of the $\mathcal{BGA}$'s activity (see Mercer, 1977, Bäck, 1992, Grefenstete, 1986, Schaffer and Morishima, 1987, Spears, 1995, Jones, 1996 and other).

In this paper we describe the idea of parametr tuning methods and we propose a class of such methods called Correlational Tuning Methods (CTMs) which measures fitness functions (see Hordijk, 1996, Jones and Forrest, 1995, Altenberg, 1997) using measurement functions that will be introduced here. Additionally we will propose a new tuned $\mathcal{BGA}$'s parameter: a code permutation.

In Section 2 we introduce the general idea of the parameter tuning. In Section 3 we show the idea of CTMs and define the notion of a measurement function as well as present two examples of a CTM. In Section 4 we describe our experiments. We show the use of the code permutation as the tuned parameter, and we describe shortly an example of a measurement function. In Section 5 we

2. Overview of the parameter tuning problem

The aim of parameter tuning is to find better parameter values for an optimization algorithm (here the $\mathcal{BGA}$) before its execution. We usually do not tune all $\mathcal{BGA}$'s parameters but a certain subset of them only, while remaining parameters are fixed. A proper choice of the subset is one of the most important problems. For generality, in this section we will use a generalized tuned parameter with values from the set P. We will investigate neither its structure nor its properties.

We assume that the efficiency of the $\mathcal{BGA}$ is a certain desired feature (or a complex of features) of the algorithm and it can be expressed as a real number. Each problem solved by the $\mathcal{BGA}$ is fully defined by its fitness function, that is—a mapping from the family of functions defined on N-bit chromosomes $\mathcal{F} = \{f : \mathcal{B} \to \mathbb{R}\}$, where $\mathcal{B} = \{0,1\}^N$ is a set of N-bit chromosomes.

DEFINITION 2.1 *Let P be the set of values of the tuned parameters of the $\mathcal{BGA}$. Let $f \in \mathcal{F}$ denote the problem solved. By the* efficiency function *we will understand a function $\mathcal{E}_f : P \to \mathbb{R}$ for which fulfilling the condition $\mathcal{E}_f(p_1) > \mathcal{E}_f(p_2)$, where $p_1, p_2 \in P$, implies that the $\mathcal{BGA}$ usually solves the problem f more efficiently for the parameter value p_1 than for p_2.*

To better understand what we mean by more efficienctly we give an example.

EXAMPLE 2.1 *In this paper we assume that the efficiency of the $\mathcal{BGA}$ for a fixed fitness function $f \in \mathcal{F}$ is given by the following formula:*

$$\mathcal{E}_f(p) \equiv -\mathbf{E}(G_f(p)) \equiv -\mathcal{G}_f(p) \qquad (1)$$

where $\mathbf{E}(\cdot)$ is an expected value and $G_f(p) \in \mathbb{R}_+$ is a random variable whose values are numbers of generations necessary to reach the global optimum by the $\mathcal{BGA}$ for the parameter value $p \in P$, and where executions of the $\mathcal{BGA}$ are random events. The value $\mathbf{E}(G_f(p)) = \mathcal{G}_f(p)$ is estimated by the mean number of generations from repeated executions of the $\mathcal{BGA}$.

In the proposed parameter tuning method we are looking for two mappings:

$$\mathcal{M} : \mathcal{F} \to M \text{ and } \mathcal{P} : M \to P, \qquad (2)$$

where $\mathcal{M}$ is a measurement method for measuring fitness functions from the family $\mathcal{F}$, $\mathcal{P}$ is a method for calculating the tuned parameters from the measurement results, and M is a space of measurement results. We impose the following condition on the desired mappings $\mathcal{M}$ and $\mathcal{P}$:

$$p^* = \mathcal{P}(\mathcal{M}(f)) \Longrightarrow \mathcal{E}_f(p^*) = \max_{p \in P} \mathcal{E}_f(p), \qquad (3)$$

The desired mappings $\mathcal{M}$ and $\mathcal{P}$ should be universal, in the sense that the value p^* should be determined correctly for the biggest possible class of problems from the family $\mathcal{F}$. This aim is very difficult to reach, hence some parameter tuning methods could be promising here. One of them will be presented in the next section.

3. Correlational parameter tuning

3.1. Measurement function as an assessment of an efficiency function

Let us assume that an efficiency function $\mathcal{E}_f$ as in Def. 2.1 is given. Let the space of measurement results be the space of vectors of a length equal to the number of elements of P, i.e. $M = \mathbb{R}^{|P|}$. The components of the vector $\mathcal{M}(f) \in M$, as in (2), are defined in the following way:

$$\mathcal{M}(f) \equiv \left(m(f,p_1),\ldots,m(f,p_{|P|})\right) = (m_f(p_1),\ldots,m_f(p_{|P|})), \tag{4}$$

where p_k for $k = 1,\ldots,|P|$ takes succeeding values of all elements of P, and a function m is of type $\mathcal{F} \times P \to \mathbb{R}$. For fixed $f \in \mathcal{F}$ we write it briefly as

$$m_f : P \to \mathbb{R}. \tag{5}$$

The function m_f will be called a measurement function. One can see that the function (5) is of the same type as $\mathcal{E}_f$ (see Def. 2.1). Let us assume for a moment that the function m_f gives the efficiency as a result, i.e. $m_f \equiv \mathcal{E}_f$. In order to get the optimal parameter value p^* exactly the same as in (3), it is enough to take $\mathcal{P}$, consistently with (2), defined as follows:

$$\mathcal{P}\left(m_1,\ldots,m_{|P|}\right) = m_f^{-1}(\max_i m_i), \tag{6}$$

hence we can write the LHS of (3) equivalently as

$$p^* = \mathcal{P}(m_f(p_1),\ldots,m_f(p_{|P|})) = m_f^{-1}(\max_{p\in P} \mathcal{E}_f(p)). \tag{7}$$

Let us notice that we can define the mapping $\mathcal{E} : \mathcal{F} \to \mathbb{R}^{|P|}$, similarly to (4), as

$$\mathcal{E}(f) \equiv \left(\mathcal{E}_f(p_1),\ldots,\mathcal{E}_f(p_{|P|})\right), \tag{8}$$

where values p_k are the same as in (4). One can see that $\mathcal{P}$ can be treated here as a kind of a tuning method, and m_f can be treated as a kind of an assessment of $\mathcal{E}_f$. The vectors $\mathcal{M}(f)$ and $\mathcal{E}(f)$ are the graphs of m_f and $\mathcal{E}_f$ as functions of $k \in \{1,\ldots,|P|\}$. If we find such m_f that its graph is similar enough to the $\mathcal{E}_f$'s graph, we will be able to infer about values of $\mathcal{E}_f$ using the following rules:

$$\begin{aligned} &m_f(p_1) \le m_f(p_2) \Longrightarrow \mathcal{E}_f(p_1) \le \mathcal{E}_f(p_2), \text{ and} \\ &m_f(p_1) > m_f(p_2) \Longrightarrow \mathcal{E}_f(p_1) > \mathcal{E}_f(p_2), \end{aligned} \tag{9}$$

where $p_1, p_2 \in P$. Obviously the probability of correctness of these rules is

3.2. Correlation as a measure of graph similarity

In order to compare the quality of various measurement functions we need a statistical tool for determining similarity between two graphs. We assume that succeeding components of $\mathcal{M}(f)$ and $\mathcal{E}(f)$ are realizations of two random variables creating a 2D general population. As a measure of similarity of these variables we can take their correlation coefficient. One can notice that the correctness of the rules (9) is higher when the value of the correlation coefficient is higher. This idea is explained better when applied to graphs showed in Fig. 2.

(a) (b)

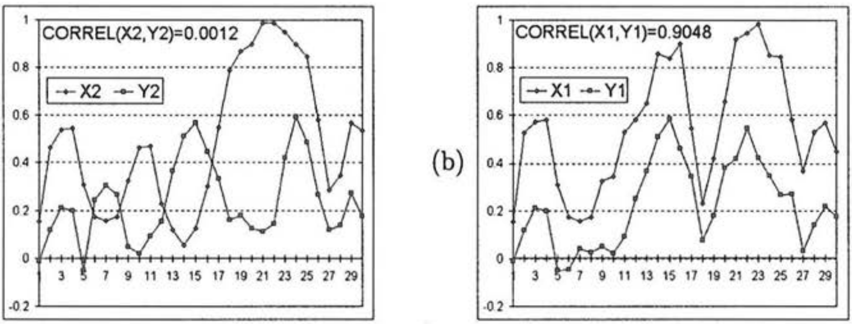

Figure 2. Comparison of (a) low and (b) high correlated graphs

It is usually difficult to calculate the correlation for $\mathcal{M}(f)$ and $\mathcal{E}(f)$ because P possesses a huge number of elements. So, in practical implementations one calculates the correlation as a statistic from a random sample from P.

Usefulness of CTMs depends on the existence of sufficiently universal measurement functions that are strongly correlated with the efficiency function for many classes of fitness functions. Additionally, the necessary condition is that the applied measurement function should be computationally simpler than the $\mathcal{BGA}$, because in the opposite situation it is better to use the $\mathcal{BGA}$ as a measurement function (see our considerations related to (6)). It appears that finding measurement functions that are good enough for certain problems is possible. An example will be described below in Subsection 4.2.

3.3. Examples of CTM's applications

3.3.1. The "trial and error" method

The simplest tuning method whose correlational version can be easily obtained is the "trial and error" method shown in Fig. 3. In the original method we execute a number of times the $\mathcal{BGA}$ for determining the efficiency $\mathcal{E}_f$ with parameter

parameter value p, for which $\mathcal{E}_f(p)$ was the highest. In the correlational version we evaluate $m_f(p)$ (in the frame) instead of $\mathcal{E}_f(p)$, so we do not need to execute the $\mathcal{BGA}$.

```
choose initial parameter value p from P.
let M_max := 0
while M_max is not high enough do
    | calculate M := m_f(p) |
    if M > M_max then let M_max := M and p_max := p
    choose next value of p from P
end
use BGA with the parameter value p_max
```

Figure 3. Correlational version of the "trial and error" method

3.3.2. Genetic meta-algorithm

The simple "trial and error" method is not enough for the effective search of the set P when no additional information about P's structure is available, because of a huge number of elements in P. A better solution is to use another $\mathcal{BGA}$, called binary genetic meta-algorithm (meta$\mathcal{BGA}$). This idea was originally proposed in the papers Mercer (1977) and Grefenstette (1986), where chromosomes encode values of the tuned parameter from the set P, and the fitness function has been taken as $f_{meta}(p) = \mathcal{E}_f(p)$ for $p \in P$. In our case of the modified meta$\mathcal{BGA}$, the fitness function is $f_{meta}(p) = m_f(p)$ (see Fig. 4).

```
choose randomly the initial population of metaBGA
execute W generations of metaBGA
let p_max be the best chromosome in Wth generation
execute BGA with the parameter value p_max
```

Figure 4. Correlational version of the genetic meta-algorithm (W—fixed parameter).

One can observe that after a certain number, say W, of generations of meta$\mathcal{BGA}$, we should get better parameter values, if, of course, m_f is a sufficiently good assessment of $\mathcal{E}_f$.

4. Description of the experiments

4.1. Code permutation as the tuned parameter

In order to use the $\mathcal{BGA}$ on a particular search space X we have to code each need to discretize it. Usually, one defines the coding function as a bijection $\Psi : X \to \mathcal{B}$, where X is the discrete search space, $\mathcal{B} = \{0,1\}^N$ is a set of N-bit chromosomes, and power of both sets is equal, i.e. $|X| = |\mathcal{B}|$. For chromosomes produced by the coding function we evaluate the fitness function $f \in \mathcal{F}$ (see Def. 2.1) during the evaluation stage of the $\mathcal{BGA}$.

In this paper we define the coding function differently:

$$\Psi^* : X \times \Omega \to \mathcal{B} \tag{10}$$

where Ω is the set of all permutations that could be applied to bits of an N-bit chromosome, and Ψ^* is a bijection with respect to the first argument. This special kind of the coding function will be called the coding with permutation.

EXAMPLE 4.1 *Let us assume that Ω is a set of all 8-element permutations. Let us take a permutation $\omega \in \Omega$ denoted as $\omega = (2,4,6,8,1,3,5,7)$ that is also a mapping $\omega : \mathcal{B} \to \mathcal{B}$. Thus, we can apply it to chromosomes, for example* $\omega(10101010) = 00001111$ *and* $\omega(11001100) = 10101010$.

It is easy to notice that code permutations affect only the crossover stage of the $\mathcal{BGA}$, because the remaining genetic operations, i.e. mutation and selection, are independent of the order of the bits. For better understanding of permutations the reader is referred to Dixon and Mortimer (1996) and Stadler (1995).

4.2. Fitness increment correlation as the measurement function

Measurement functions used in CTMs are usually designed heuristically, basing on experience of a researcher and on experiments. Here we describe shortly an example of such a measurement function (see (5)). We use code permutations from Ω (see Ex. 4.1) as the set of tuned parameter values. One can find exact descriptions of this and other measurement functions in Kieś (1999a), Kieś and Kosiński (1998), Kieś (1998) and Kieś (1999b).

The value of the proposed function is a correlation coefficient between a fitness and a fitness increment of the measured fitness function. The measurement function is called Fitness Increment Correlation and denoted by r_{FI}. It is calculated in the following way: we choose randomly a certain sequence of chromosomes from $\mathcal{B}$, and then we modify randomly a randomly chosen segment of each chromosome. We receive two sequences: a sequence of mean fitnesses of pairs of original and modified chromosomes, and a sequence of differences between their fitnesses. Next, treating both sequences as a two-dimensional random variable, we calculate the correlation coefficient, and its value is the value of the r_{FI} measurement function.

One can see that the value r_{FI} depends on the value of the tuned parameter, because of the use of coding with permutation (10). So it can be used as a

4.3. Experiment parameters

The efficiency of a CTM will be investigated basing on the CTM with a meta$\mathcal{BGA}$ (see Fig. 4). Let us assume that the meta$\mathcal{BGA}$ always generates a fixed number $W = 15$ of generations. Chromosomes in the meta$\mathcal{BGA}$ encode code permutations from Ω. We realize $V = 100$ identical experiments with the meta$\mathcal{BGA}$ with different random initial generations. For the i-th experiment we calculate the following coefficients:

- relative increment of efficiency: $\zeta_{Ei} = \frac{2(\mathcal{E}_{fW}-\mathcal{E}_{f0})}{\mathcal{E}_{fW}+\mathcal{E}_{f0}}$,
- relative increment of efficiency assessment: $\zeta_{Mi} = \frac{2(m_{fW}-m_{f0})}{m_{fW}+m_{f0}}$,
- tunability coefficient: $\kappa_{Ei} = \frac{\zeta_{Ei}}{\zeta_{Mi}}$,

where $\mathcal{E}_{fj}$ and m_{fj} are respectively: the expected number of $\mathcal{BGA}$'s generations and its assessment by the r_{FI} measurement function achieved after $j = 0$ or W generations of the meta$\mathcal{BGA}$ for the parameter value decoded from the best chromosome. Note that values m_{fj} and ζ_{Mi} do not depend on the run of the $\mathcal{BGA}$, nor on p_m, nor $PopSz$. Hence, they are shown once for each problem.

The calculated coefficients can be put in V-element sequences $\{\zeta_{Ei}\}$, $\{\zeta_{Mi}\}$ and $\{\kappa_{Ei}\}$. We denote their mean values by $\overline{\zeta}_E$, $\overline{\zeta}_M$ and $\overline{\kappa}_E$, respectively. The correlation coefficient of $\{\zeta_{Ei}\}$ and $\{\zeta_{Mi}\}$ is denoted by $\varrho(\zeta_E, \zeta_M)$. These values are shown in Table 1.

Problem	Params. of $\mathcal{BGA}$		$\varrho(\zeta_E,\zeta_M)$	$\overline{\zeta}_M$	$\overline{\zeta}_E$	$\overline{\zeta}_E/\overline{\zeta}_M$	$\overline{\kappa}_E$
	p_m	$PopSz$					
1	0.1	20	0.199	0.311	0.0465	0.150	0.152
	0.1	40	0.507		0.223	0.717	0.697
2	0.01	40	−0.0151	0.0248	−0.0025	−0.101	−0.455
3	0.01	20	0.157	0.0796	0.0476	0.598	1.247
	0.005	20	0.122		0.0320	0.402	0.891
4	0.01	20	0.225	0.0455	0.0019	0.0417	−1.619
	0.005	20	0.0039		0.0074	0.163	0.0366

Table 1. The results of measurements of the efficiency of a CTM with meta$\mathcal{BGA}$ (see Fig. 4), where r_{FI} is the measurement function and the code permutation Ω is the tuned parameter. Values for four problems solved by the $\mathcal{BGA}$ are shown with example settings for p_m and $PopSz$.

During our experiments we tune the code permutation in the standard $\mathcal{BGA}$ (see Michalewicz, 1996) with elitism (see Rudolph, 1994). The $\mathcal{BGA}$'s efficiency $\mathcal{E}_f$ is measured according to the Example 2.1 by the mean from $n_{\mathcal{E}_f} = 500$ executions of the $\mathcal{BGA}$. We assume that the $\mathcal{BGA}$ stops when $f(\underline{b}_{max}) > f(\underline{b}_{opt}) - \epsilon$, where $\underline{b}_{max}$ is the best chromosome in the current generation, $\underline{b}_{opt}$ is the global optimum, and ϵ is the maximum error. All the examined fitness functions,

4.4. Test fitness functions

4.4.1. Problem 1: Maximization of a function of a scalar argument

We maximize the following function of type $\mathbb{R} \to \mathbb{R}$:

$$f(x) = (0.1x + 1)(2sin^{40}(5x^2 + \pi/4) - 1)$$

for $x \in \langle -2^{31}/10^9, (2^{31}-1)/10^9 \rangle$, encoded in the standard way (see Michalewicz, 1996). The global maximum is $f(\sqrt{1.45\pi}) = 1.2134317$, and $\epsilon = 10^{-6}$.

4.4.2. Problem 2: The knapsack problem

We pack a knapsack by a number of objects of weights w_i and values v_i (one object per bit). We maximize the total value $V(\underline{b})$ of packed objects on the condition that the total weight $W(\underline{b})$ does not exceed the given maximum weight $W_{max} = 320$. The possible solutions are coded by chromosome bits: 1 means packed object and 0—not packed. We maximize the following function:

$$f(\underline{b}) = V(\underline{b}) - g(W(\underline{b}), W_{max}),$$

where $g : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is a nonnegative penalty function, and $\epsilon = 10^{-6}$.

4.4.3. Problem 3: Maximization of a function of a vector argument

We maximize the following function of type $\mathbb{R}^4 \to \mathbb{R}$:

$$f(x_1, x_2, x_3, x_4) = \sum_{i=1}^{4}(0.1x_i + 1)(2\sin^2(i+1)(x_i^2 + \pi/4) - 1)$$

for $x_1, x_2, x_3, x_4 \in \langle -2^7/64, (2^7 - 1)/64 \rangle$ encoded in the standard way (see Michalewicz, 1996), each on 8 bits. The global maximum is $f(1.9843, 1.8745, 1.8431, 1.8275) = 4.7476$, and $\epsilon = 10^{-2}$.

4.4.4. Problem 4: A simple classifier

We classify 16 two-dimensional vectors into maximum 4 clusters. A solution consists of 16 numbers of assignment clusters, two bits per vector. We maximize clustering quality rating J given by the following formula:

$$J = \frac{\sum_{j=1}^{L} \|\underline{x}_j - \underline{m}\|^2}{\sum_{i=1}^{c}(\sum_{j=1}^{N_i} \|\underline{x}_{ij} - \underline{m}_i\|^2 + \|\underline{m}_i - \underline{M}\|^2)} - 1,$$

where L—number of all vectors; $\underline{x}_j$—jth vector; $\underline{m}$—mean vector (centroid) of all clusters; $\underline{x}_{ij}$—jth vector from ith cluster; c—number of clusters; N_i—number of vectors in ith cluster; $\underline{m}_i$—mean vector (centroid) of ith cluster; $\underline{M}$—mean

4.5. Results of experiments

When interpreting the values shown in Table 1 one can see that the most important aim, considering the ability of increasing the $\mathcal{BGA}$'s efficiency $\mathcal{E}_f$ during tuning, is the maximization of $\overline{\zeta}_E$. But the maximization of $\varrho(\zeta_E, \zeta_M)$, $\overline{\zeta}_E/\overline{\zeta}_M$ and $\overline{\kappa}_E$ is the most important aim considering the exactness of the assessment of $\mathcal{E}_f$ by m_f.

One can see that r_{FI} is more suitable for Problems 1 and 3, where the fitness function is analytically defined, because of high values of $\varrho(\zeta_E, \zeta_M)$, $\overline{\zeta}_E/\overline{\zeta}_M$ and $\overline{\kappa}_E$. For the Problem 2, r_{FI} gives false results because the sign of $\overline{\zeta}_M$ is different from the remaining values, so we conclude that r_{FI} is not suitable for this kind of problems. The case given in Problem 4 is controversial. We suppose that the influence of the code permutation on the $\mathcal{BGA}$'s efficiency is very weak here, so the received results are open for an influence of errors and the experiment should be repeated with higher values of V and $n_{\mathcal{E}_f}$.

5. Conclusions

We suppose that the explanation of the dependence between $\mathcal{BGA}$ efficiency and code permutation is related to the building block hypothesis presented in Bagley (1967) and Rosenberg (1967). The cause is that the ability of creating good building blocks during crossover can be different for different code permutations.

Our experiments show that it is possible to find such a measurement function that can be used in CTMs and we conclude that the parameter tuning is an idea of not only the theoretical value. The development of CTMs depends on discovering other, more universal, measurement functions. All measurement functions demand also more tests in a well-defined environment. The idea of *NK-model* introduced by Kauffman (1989) can be very useful here, because it makes possible generation of various fitness functions that can be tuned from smooth to rugged.

Numerous new evolutionary algorithms have been proposed that are based strictly on the idea of coding solutions with binary chromosomes. Here the work of Jones (1996) can be a good example. Applying the code permutation can be an interesting way of extending the algorithms.

Undoubtedly the most exciting task is to discover a constructive method for finding the best code permutation for a given problem instead of comparing qualities of various permutations. This will be the subject of our further research.

References

ALTENBERG, L. (1997) Fitness distance correlation analysis: an instructive counter example. *Proceedings of the 7th International Conference on Ge-*

BÄCK, T. (1992) The Interaction of Mutation Rate, Selection, and Self-Adaptation within a Genetic Algorithm. *Parallel Problem Solving from Nature*, Vol. 2. North-Holland, Amsterdam, 85–94.

BAGLEY, J.D. (1967) The behavior of adaptive systems which employ genetic and correlation algorithms (doctoral dissertation, Univ. of Michigan). *Dissertation Abstracts International*, **28** (12), 5106B (Univ. Microfilms No. 68-7556).

DIXON, J.D. and MORTIMER, B. (1996) *Permutation Groups.* Graduate Texts in Mathematics 163, Springer-Verlag, New York.

EIBEN, A.E., HINTERDING, R. and MICHALEWICZ, Z. (1999) Parameter control in evolutionary algorithms. *IEEE Trans. on Evolutionary Computation*, **3**, No. 2, 124–141.

GREFENSTETTE, J.J. (1986) Optimization of control parameters for genetic algorithms. *IEEE Trans. on Systems, Man, and Cybernetics*, **16**, No. 1, 122–128.

HORDIJK, W. (1996) A Measure of Landscapes. *Evolutionary Computation*, **4**, No. 4, 335–360.

JONES, T. (1996) Crossover, Macromutation, and Population-based Search. *Proceedings of the 6th International Conference on Genetic Algorithms.* M. Kaufmann, San Mateo, CA.

JONES, T. and FORREST, S. (1995) Fitness distance correlation as a measure of problem difficulty for genetic algorithms. *Proceedings of the 6th International Conference on Genetic Algorithms.* M. Kaufmann, San Mateo, CA.

KAUFFMAN, S.A. (1989) Adaptation on Rugged Fitness Landscapes. In Stein, D. (ed.), *Lectures in the Sciences of Complexity.* Addison-Wesley, 527–618.

KIEŚ, P. (1998) Correlational method of an off-line adaptation. *Proc. of the 2nd Asia-Pacific Conf. on Simulated Evolution and Learning*, Vol. 1, Session 3, Univ. of New South Wales, Canberra, Australia, November 1998.

KIEŚ, P. (1999a) Adaptacyjny dobór metody krzyżowania w binarnym algorytmie genetycznym (Adaptational adjustment of crossover method in the binary genetic algorithm). Ph.D. thesis in Polish, Institute of Fundamental Technological Research PAS (IPPT PAN), Warsaw, June 1999.

KIEŚ, P. (1999b) Correlational adaptation method and genetic meta-algorithm. *Proc. of the VIII-th International Symposium on Intelligent Information Systems*, Ustroń n. Cieszyn, Poland, June 1999, 193–202.

KIEŚ, P. and KOSIŃSKI, W. (1998) On the correlational method of an off-line adaptation. *Colloquia in Artificial Intelligence Theory and Application, CAI'98*, Z. Gontar, Łódź, Poland, September 1998, 9–19.

KOLONKO, M. (1995) A Generalized Crossover Operation for Genetic Algorithms. *Complex Systems*, **9**, 177–191.

MERCER, R.E. (1977) Adaptive search using a reproductive meta-plan. Master's thesis, Univ. of Alberta, Edmonton.

MICHALEWICZ, Z. (1996) *Genetic Algorithms + Data Structures = Evolution*

QI, T. and PALMIERI, F. (1994) Theoretical Analysis of Evolutionary Algorithms with an Infinite Population Size in Continuous Space, Part II: Analysis of the Diversification Role of Crossover. *IEEE Trans. of Neural Networks*, **5**, No. 1.

ROSENBERG, R.S. (1967) Simulation of genetic populations with biochemical properties (Doctoral dissertation, Univ. of Michigan). *Dissertation Abstracts International*, **28** (7), 2732B (Univ. Microfilms No. 67-17836).

RUDOLPH, G. (1994) Convergence analysis of canonical genetic algorithms. *IEEE Trans. on Neural Networks*, Special Issue on Evolutionary Computation, **5**, No. 1.

SCHAFFER, J.D. and MORISHIMA, A. (1987) An adaptive crossover distribution mechanism for genetic algorithm. *Proc. of the 2nd International Conference on Genetic Algorithms*, Erlbaum Assoc., Hillsdale, NJ, 36–40.

SPEARS, W.M. (1995) Adapting Crossover in a Genetic Algorithm. *Artificial Intelligence Center Internal Report #AIC-94-025*, Naval Research Laboratory, Washington, DC 20375. *Evolutionary Programming IV*, McDonnell, J.R., Reynolds, R.G. and Fogel, D.B. (eds.), February 1995, 367–384.

STADLER, P.F. (1995) Towards a Theory of Landscapes. *Complex Systems and Binary Networks*, Lopez-Pena et al. (eds.), Springer-Verlag.

WOLPERT, D. and MACREADY, W. (1995) *No free lunch theorems for search.* Santa Fe Institute Technical Report SFI-TR-02-010.