SELF ADAPTATION OF PARAMETERS FOR MCPC IN GENETIC ALGORITHMS

ESQUIVEL S. C., LEIVA H. A., GALLARD R.H[*].

Abstract

As a new promising crossover method, multiple crossovers per couple (MCPC) deserves special attention in evolutionary computing field. Allowing multiple crossovers per couple on a selected pair of parents provided an extra benefit in processing time and similar quality of solutions when contrasted against the conventional single crossover per couple approach (SCPC). These results, were confirmed when optimising classic testing functions and harder (non-linear, non-separable) functions.

Despite these benefits, due to a reinforcement of selective pressure, MCPC showed in some cases an undesirable premature convergence effect. In order to face this problem, the present paper attempts to control the number of crossovers, and offspring, allowed to the mating pair in a self-adaptive manner.

Self-adaptation of parameters is a central feature of evolutionary strategies, another class of evolutionary algorithms, which simultaneously apply evolutionary principles on the search space of object variables and on strategy parameters. In other words, parameter values are also submitted to the evolutionary process. This approach can be also applied to genetic algorithms.

In the case of MCPC, the number of crossovers allowed to a selected couple is a key parameter and consequently self-adaptation is achieved by adding to the chromosome structure "labels" describing the number of crossover allowed to each individual. Labels, which are bit strings, also undergo crossover and mutation and consequently evolve together with the individual. During the stages of the evolution process, it is expected that the algorithm will return the number of crossovers for which the current population exhibits a better behaviour.

Descriptions of different self-adaptation methods used, experiments and some of the results obtained are shown.

Keywords: genetic algorithms, self-adaptation, crossover, function optimisation.

[*] Proyecto UNSL-338403[1]. Departamento de Informática. Universidad Nacional de San Luis (UNSL). Ejército de los Andes 950 - Local 106. 5700 - San Luis, Argentina. E-mail: {esquivel, aleiva, rgallard}@unsl.edu.ar. Phone: + 54 2652 420823. Fax : +54 2652 430224.

The Research Group is supported by the Universidad Nacional de San Luis and the ANPCYT (National Agency to Promote Science and Technology).

1. INTRODUCTION

MCPC as proposed in [1] allows multiple children per couple by replicated application of crossover operator. In those experiments the number of crossover allowed to a couple remain fixed during a single run. The rationale behind this implementation was to isolate MCPC effects to obtain a set of preliminary results. Previous experiments showed a quality of results as good as under SCPC, and sometimes better, when 3 and 4 crossovers per couple were allowed. Also an extra benefit in processing time was detected; this benefit was achieved by making a lesser number of selections. Despite these benefits, due to a reinforcement of selective pressure by increasing the production of similar individual, MCPC showed in some cases an undesirable premature convergence effect.

Parameter control is a new field in evolutionary computation that proposes to dynamically tune algorithm parameters by adjusting their values while the algorithm is running trying to solve a particular problem. Inside this field there are three categories, the one concerned here is called *self-adaptation of the parameter values* which evolves them as part of the chromosome structure. Previous work of Spears [2] suggested self-adaptive approaches to select the type of crossover operator to be applied to each couple during a genetic algorithm execution. In this paper we propose a self-adaptive approach to determine the number of crossovers to be applied to a selected couple under MCPC. A parameter control general classification scheme, the criteria to implement self-adaptation when optimizing hard testing functions, a description of experiments and results are shown in the following sections.

2. PARAMETER CONTROL: A CLASSIFICATION SCHEME

Today a great interest exists in methods including mechanisms to control parameters used by evolutionary algorithms during execution. Eiben, Hinterding and Michalewicz [3] gave the following main categories of parameter control:

- *Deterministic Parameter Control*: This is the case in which the parameter value is modified according with a deterministic rule, without taking care of any feedback of the searching process performed by the strategy. For example, the crossover probability rate value can be updated as a function of the number of generations reached.
- *Adaptive Parameter Control*: In this case some feedback information of the searching process is used to determine the direction and magnitude of the change in the parameters. Rather than being changed by a deterministic rule, here a parameter value can be changed by considering the search convergence degree for example.
- *Self-adaptive Parameter Control*: Here the parameters to be adapted are codified within the chromosomes and undergo genetic operations. The best individuals of the population have better chances of survival and reproduction. Hence it is expected that better parameter values be more intensively propagated.

As the number of crossovers to be applied, to a couple, in MCPC is one of the parameters of the algorithm that is included as a part of an individual, our present approach appertains to the last above-mentioned category.

3. SELF-ADAPTIVE PARAMETER CONTROL IMPLEMENTATION

As we previously said, we attempt to self-adapt the number of crossovers per couple in MCPC. Because we are using a binary representation of chromosomes, the number of crossovers allowed for an individual is codified in a field at the rightmost positions of the bit string. Let us call it the *ncross_field*. In some experiments we allowed a maximum of three and in others a maximum of seven crossovers per couple. So, two or three extra bits were enough for that purpose. More generally the last $log_2(max\_cross + 1)$ bits of each individual are used to find an expected optimum number of crossovers.

In that way the individuals of each new population are exploring two searching spaces: one corresponding to the objective function and other associated to the parameter value, the number of crossovers to apply.

Each individual preserves information about the number of crossovers originally applied to their parents. In this way it is expected that, based on the *survival-of-the-fittest* principle, good solutions carry information about the number of crossover applied to their ancestors and that this number would be an appropriate one.

According to Spears [2] there are two possible criteria for determining the way by which the change will take place, *local* or *global*. With *global* adaptation the choice of how many crossover operations should be applied is not tied to a particular couple but to the population as a whole. For example, if we have the *ncross_field content* distributed through the whole population as shown in the following table:

Number of cross.	*Prob. Distribution*
1	0.25
2	0.45
3	0.30

That means the 25% of the individuals were created using one crossover operation, the 45% were generated using two and the remaining 30% using three crossovers per couple. It is expected also that this remains unchangeable unless any of the following things happen:

- the *ncross_field* is disrupted by crossover or mutation or,
- some of the couple, appertaining to one of the above shown classes, produces more children than before.

Here in this paper we analyse the behaviour of a modified GA using only a *local* criterion of self-adaptation. Different individuals are traversing different trajectories through the search space, thus possibly requiring different amount of recombination. The bits in the *ncross_field* with each individual are used to choose the times the crossover operator will be applied on the selected pair. The later is made by checking the corresponding number of crossover carried by each parent and;

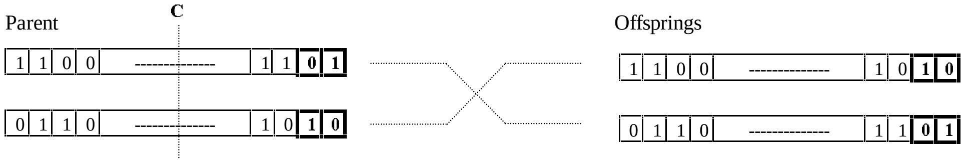
- If they match, then we apply the recombination operator a number of times specified by the *ncross_field.*
- Otherwise we choose a random number in the permitted range.

In the second situation and following the Spears' approach, when decoded numbers of crossovers are different, we are violating our attempt to preserve information because the children do not keep the number of crossover by which them were created. If the crossover point does not disrupt the *ncross_field* (and this event has low probability of occurrence) then children retain information from either parent, but they do not preserve information about how they were created.

In order to either retain information about how an individual was created or how their parents were created, we devised two different approaches for experimentation.

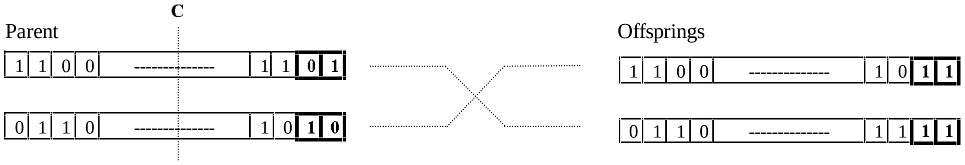
- *E1* (see the illustration in the next page): In any situation, exchange of information from parents to children is done in he traditional way applying the genetic operators with their corresponding probabilities. Parent's chromosomes are mated and undergo crossover a certain number of times

according to the specified values in *ncross_field* if they match, or to a random allowed value otherwise. After recombination, mutation is applied to the children.
In the *don't match* situation, this approach, preserving parent's information, enforces population diversity in the parameter searching space, because most of the time one child inherits characteristics from one of the parent and the other child inherits features from the other parent.



C: Crossover
Random Crossover Number: **3**

- *E2* (see the illustration beneath): If the values specified in *ncross_field* do not match then the new random value for the number of crossovers is inserted first in the parent's *ncross_field,* and afterwards crossover is performed for the number of times specified by this random value. This approach by preserving individual information creates more similar individuals in the parameter searching space and increases loss of genetic diversity.



C: Crossover
Random Crossover Number: **3**

4. EXPERIMENTAL TESTS

Experiments E1 and E2, to verify the parameter control mechanisms were designed. For this we choose two hard testing functions: Easom's [5] and Schweffel's [6] functions (see table 1 for a detailed description of them). A modified version of the non-canonical genetic algorithm (Goldberg [4]) was implemented.

Series of many runs were performed on each function, with randomised initial population of size fixed to 1000 individuals, using binary representation, proportional selection, elitism, one point crossover and bit flip mutation. The number of generations was fixed to 500 and probabilities for crossover and mutation were fixed to 0.65 and 0.05, respectively.

The relatively big population size of 1000 individuals was chosen in order to allow a significant contribution of selected individuals to the evolution process when high number of crossovers are allowed. For example in the case of six crossovers per couple only 16.5% of the available individuals in the old population will intervene when building the new generation (i.e. 165 individuals of 1000 in this case).

Notation	Description		Characteristics
f5: Easom's Function	$f_5(x_1, x_2) = -\cos(x_1)\cos(x_2)\, e^{-((x_1-\pi)^2+(x_2-\pi)^2)}$, $x_1, x_2 \in [-100,\ 100]$		unimodal, the global minimum has a small area relative to the search space
f7: Schwefel's Function 7	$f_7(\vec{x}) = \sum_i -x_i \cdot \sin\left(\sqrt{\lvert x_i \rvert}\right),\ for\ i = 1:n$ $x_i \in [-500, 500]$	Dim. $N = 5$	highly multimodal, the global minimum is geometrically distant from the next best local minima.

Table 1. Objective functions

The following variables were chosen for the analysis

Name	***Description***
Quality	Is the ratio $\left({best\_value}/{opt\_value} \right)$ between the best value and the optimal value. It measures the quality of a solution, how good is a solution contrasted against the optimum value known or estimated..
CrAvg	Mean number of crossover allowed per selected couple. This is the variable subject of this study.
Dtime	Defined as in [1]. Running time difference. It is the percentage of time reduction when compared with classic crossover (single crossover per couple).

5. RESULTS

Results concerning *Quality* were similar under E1 and E2, but different when *CrAvg* is considered. Hence the following figures alternatively show *Quality* results under experiments E1 or E2. Results in regard to *CrAvg* are summarised in the same graph for both experiments.

Function f5

We started optimizing *f5*, and for this unimodal function two bits were used to code the *ncross_field* , allowing a maximum of three crossovers (six children per couple).

In figure 1 *Quality* values for the Easom's function, show a slightly slower convergence of MCPC when compared with SCPC, but after 80 generations the former results are better. In fact during the simulation, values of *Quality* reach 1.0 under MCPC and 0.9998 under SCPC.

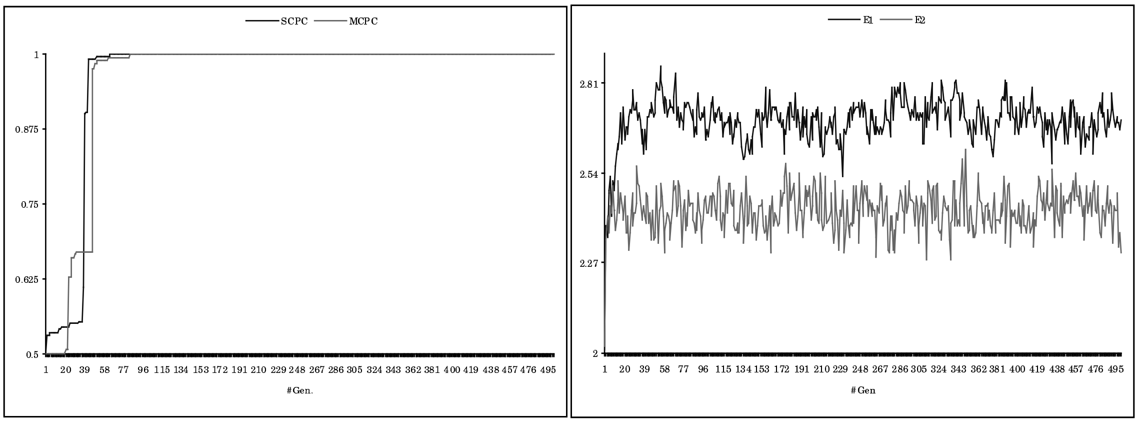


Fig. 1: ***Quality*** and ***CrAvg*** values for function f_5

Regarding *CrAvg* values, it can be observed that they oscillate between 2.5 and 2.8 after the few first generations under E1 and between 2.3 and 2.6 under E2.

Here the behaviour of the self adaptive parameter control mechanism is clear: when genetic diversity in the parameter searching space is low then lesser number of crossovers are allowed and viceversa. This behaviour favours the evolutionary process.

Concerning to *Dtime* values they were 37,7% and 33,6% under E1 and E2 approaches respectively.

Function f7

When optimizing *f7*, we decided to use two and also three bits to code the *ncross_field* , allowing a maximum of three and seven crossovers respectively.

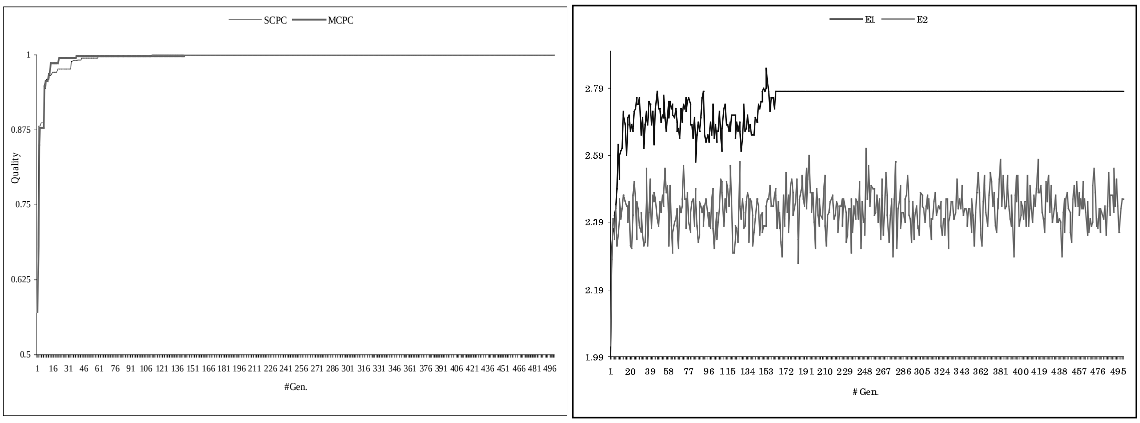


Fig. 2: ***Quality*** and ***CrAvg*** values for function f_7

In figure 2 *Quality* values for the Schwefel's function with two bits for *ncross_field*, show a slightly faster convergence of MCPC when compared with SCPC, but both converge to the optimum reaching a *Quality* value of 1.0.

Regarding *CrAvg* values, it can be observed that they oscillate between 2.6 and 2.8 remaining stable after 152 generations under E1, and oscillate between 2.3 and 2.6 under E2.

Here again the behaviour of the self-adaptive parameter control mechanism is shown. Parameter setting is adapted to the population diversity in the parameter searching space. In this case this behaviour prevent the evolutionary process of being trapped in a local optima.

Concerning to *Dtime* values they were 22.1% and 21.3% under E1 and E2 approaches respectively.

Another set of experiments studied the behaviour of the control mechanism for the Schwefel's function with three bits for the *ncross_field*.

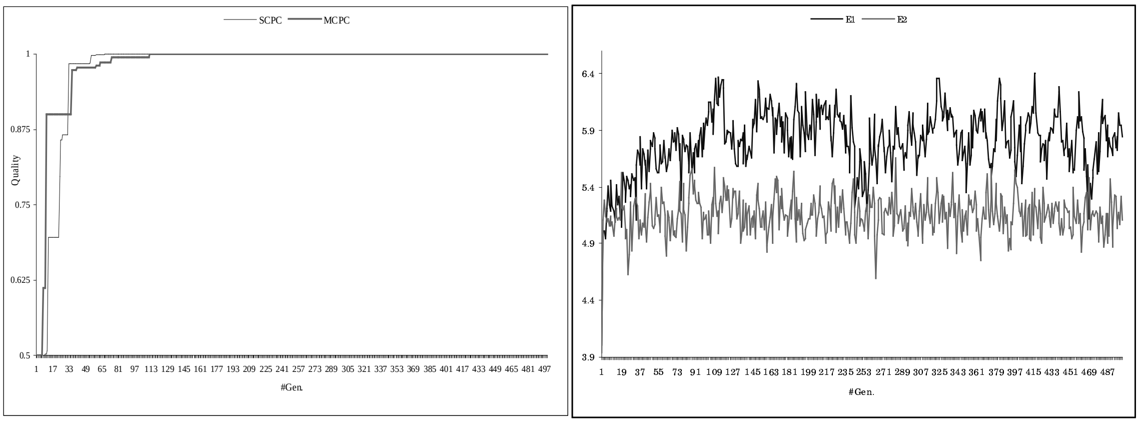


Fig. 3: ***Quality*** and ***CrAvg*** values for function f_5

In figure 3 *Quality* values for the Schwefel's function with three bits for *ncross_field*, are shown. Here we cannot clear differences on convergence velocity. At the beginning MCPC shows to be faster and after that SCPC is faster 110 generations both reach the optimum.

Regarding *CrAvg* values, it can be observed that they oscillate between 4.5 and 5.6 under E1, and between 5.2 and 6.3 under E2.

Here again the behaviour of the self-adaptive parameter control mechanism is shown. Parameter setting is adapted not only to the population diversity but to the maximum number of crossovers allowed. Concerning to *Dtime* values they were 50% and 48% under E1 and E2 approaches respectively.

6. CONCLUSIONS

The present paper proposes an alternative approach to assign the number of crossovers allowed for a selected couple. Instead of doing that based on the couple fitness as in [8], here we suggest to use a self-adaptive parameter control approach. The parameter considered is coded in a *ncross_field* of two or three bits in the chromosome structure, and further submitted to genetic operations in the same way as any evolutionary technique does. Within this approach two searching processes are carried out simultaneously: one on the problem (objective function) space and the other on the parameter space. In this way it is expected an adaptive parameter setting, retaining best settings through the survival of the fittest individuals in the problem space.

Two different strategies were approached in order to overcome loss of information about offspring or parents creation. As they were conceived, approach *E1* maintains population diversity in the parameter searching space while approach *E2* leads to a loss of diversity.

Being consequent with this situation the control mechanism adapts the number of crossovers for exploration (under E1) or exploitation (under E2) accordingly. And this behaviour is preserved for diverse maximum number of crossover allowed.

Self-adaptation of MCPC parameters has been addressed to find the most appropriate number of crossover operation to be applied to a mating pair and, from the above shown results, we can conclude that this can not be a fixed number. Usually, when locally self-adapting, this number oscillates between couples and runs. And this oscillation seems to show best results compared against the cases where we allow a fixed number of crossover per couple.

It is also remarkable that on each experiment MCPC outperforms SCPC most of the time on quite different fitness landscapes.

Future work will consider more biased methods, tied to the fitness of individuals in the couple, to choose the number of crossovers in the *don't-match* case.

7. ACKNOWLEDGEMENTS

We acknowledge the cooperation of the project group for providing new ideas and constructive criticisms. Also to the Universidad Nacional de San Luis the CONICET, and the ANPCYT from which we receive continuous support.

8. BIBLIOGRAPHY

[1] Esquivel S., Leiva A., Gallard R.: *Multiple Crossover per Couple in Genetic Algorithms*. Proceedings of the Fourth IEEE Conference on Evolutionary Computation (ICEC'97), Indianapolis, USA, April 1997.

[2] William M. Spears: *Adapting Crossover in Evolutionary Algorithms*. Proceedings of the Evolutionary Programming Conference, 1995.

[3] Agoston E. Eiben, Robert Hinterding, Zbigniew Michalewicz: *Parameter Control in Evolutionary Algorithms*. Technical Report, UNC - Charlotte, 1998.

[4] Goldberg D.E.: *Genetic Algorithms in Search Optimization and Machine Learning*. Addison Wesley Publishing, 1989.

[5] Easom, E.: *A survey of global optimization techniques*. M. Eng. Thesis, Univ. Louisville, Louisville, KY, 1990. In [7].

[6] Schwefel, H. P.: *Numerical optimization of computer models*. Chichester: Wiley & Sons, 1981. In [7].

[7] Hartmut Pohlheim: *Genetic and Evolutionary Algorithm Toolbox for use with Matlab (GEATbx)*. Copyright © 1996, Germany. All rights reserved.

[8] Esquivel S., Leiva A., Gallard R.: *Couple Fitness Based Selection with Multiple Crossover per Couple in Genetic Algorithms*. Proceedings of the International Symposium on Engineering of Intelligent Systems (EIS´98), La Laguna, Tenerife, Spain, February 1998.