Real-Coded Genetic Algorithms Based on Mathematical Morphology

Dolores Barrios, Daniel Manrique, Jaime Porras, and Juan Ríos

Facultad de Informática, Universidad Politécnica de Madrid
jrios@fi.upm.es

Abstract. The goal of this work is to propose a general-purpose crossover operator for real-coded genetic algorithms that is able to avoid the major problems found in this kind of approach such as the premature convergence to local optima, the weakness of genetic algorithms in local fine-tuning and the use of real-coded genetic algorithms instead of the traditional binary-coded problems. Mathematical morphology operations have been employed with this purpose adapting its meaning from other application fields to the generation of better individuals along the evolution in the convergence process. This new crossover technique has been called mathematical morphology crossover (MMX) and it is described along with the resolution of systematic experiments that allow to test its high speed of convergence to the optimal value in the search space.

1 Introduction

Since its genesis genetic algorithms (GA) paradigm is being increasingly used in search and optimization problems. GAs are generally represented as a set (or a population) of one-dimensional strings, called individuals, each of which contains a given number of chromosomes and maps a possible solution to the problem. Given an evaluation function, GAs approach the optimal solution by applying various operators over the individuals of the population. Such operators as reproduction, crossover and mutation are the most frequently used [1].

Several improvements have been made to GAs during the last decade in order to avoid some of the major inconvenients found in this kind of approach. Such problems, object of attention in this work are: the premature convergence to local optima, the weakness of GAs in local fine-tuning and the using of real-coded GAs instead of the traditional binary-coded GAs.

The problem of premature convergence of the GA to local optima of the objective function is tightly related with the loss of genetic diversity of the population, being the cause of a decrease on the quality of the solutions found. Several techniques have been proposed in order to avoid this problem: half greedy crossover was introduced [2] based on delaying the convergence process making it slower. Other approaches are related to changing dynamically the probability of mutation: at regular intervals in time, the value of the standard deviation of the population is tested, if it is lower than a

F.J. Ferri et al. (Eds.): SSPR&SPR 2000, LNCS 1876, pp. 706-715, 2000.

predefined value, the probability of mutation is increased, getting new random individuals and so, more diversity [3]. This approach has the inconvenience of the high computational cost derived from the calculation of the standard deviation. Generalized binary crossover (GBX) [4] has the important feature that extends dynamically the schemata defined by the two parents in order to pick the offspring. The problem here is that this expansion may be uncontrolled as it depends on the binary coding of the values of the parents. BLX-α [5] is an example of a crossover operator for real-coded GAs that has the ability of expand the interval defined by the two parents, but depending on the previously fixed parameter α, not on the diversity of the population.

The weakness of GAs in performing fine tuned local search is widely recognized. Although GAs exhibit fast convergence at first to a point of approximate solution in a search space; when a population reaches a state where it is dominated by the best individual, finding a better solution is very difficult, resulting in very inefficient search. There are several ways to solve this problem, one of them is to have larger population, however it requires extensive computation for each generation. Other solution would be combining GAs and other approaches as descent gradient methods: first GAs are used to locate a point near the solution. Then descent gradient leads to the final solution. The problem here is to know the best point in time to change from GA to descent gradient in order to be very efficient and the premise that the derivated function of the fitness must exist near the optimal solution.

Finally a growing number of researchers have come to champion real-coded (or floating point) chromosomes as opposed to binary-coded ones. The primary advantages offered by the real-coded GAs are mainly three. First, real-coded GAs eliminate the worry that there is inadequate precision so that good values are representable in the search space. Second, the range of parameters does not have to be a power of two. Third, real-coded GAs have the ability to exploit the gradualness of functions of continuous variables [6]. Several new crossover operators have been designed to work with real numbers such as Radcliffe's flat crossover (RFX) [7]. This operator chooses parameters for an offspring by uniformly picking parameter values between (inclusively) the two parents parameter values. Of course this approach has the premature convergence problem. Other important, but also quite computationally expensive, crossover technique for real-coded GAs is the UNDX [8] that can optimize functions by generating the offspring using the normal distribution defined by the three parents.

A general-purpose method of optimizing functions using real-coded GAs is described throughout this paper. The GA proposed to accomplish this task employs a new crossover technique based in mathematical morphology [9] [10] as so it is called mathematical morphology crossover (MMX). In particular MMX, employs the morphological gradient very used with segmentation purposes [11] and adapting its meaning from gray-scale images to the generation of better individuals to reach the optimal solution very quickly. The objective when MMX was being designed was to dynamically expand or make narrower the interval defined by the parents depending on the genetic diversity of the population, allowing this way fine local-tuning capabilities while avoiding the premature convergence to a local optimum.

Several optimization problems are devised to test this new approach ranging from the optimization of some functions to even the task of training artificial neural net-

works (ANN). These have been chosen to allow us to test its accuracy, high speed of convergence and fine-tuning capabilities without being trapped in any local minima that these problems present.

2 MMX: Mathematical Morphology Crossover

MMX works with a population of *m* individuals with *l* chromosomes each coded with real numbers, so each point in the search space $D_{\Re}$ is defined by:

$$s=(a_0, a_1, ..., a_{l-1}), \text{ where } a_i \in \Re. \quad (1)$$

The operator MMX works with each gene in the parents independently to obtain the corresponding two genes in the offspring. Let $s_1, ..., s_n$ be an odd number of strings chosen from the actual population to be crossed, the $(n \times l)$gene-matrix is defined as:

$$G = \begin{pmatrix} a_{10} & a_{11} & ... & a_{1l-1} \\ a_{20} & a_{21} & ... & a_{2l-1} \\ ... & ... & ... & ... \\ a_{n0} & a_{n1} & ... & a_{nl-1} \end{pmatrix} \text{ where } s_i = (a_{i0}, a_{i1}, ..., a_{i,l-1}),\ i = 1 ,..., n \quad (2)$$

The crossover operator works with each column $f_i=(a_{1i},a_{2i}, ...,a_{ni})$ in matrix G obtaining genes $o_i \in \Re$ and $o_i' \in \Re$. The result of applying the operator to matrix G is, therefore, the two new descendants $o =(o_0,o_1, ...,o_{l-1}) \in D_{\Re}$ and $o'= (o_0',o_1', ...,o_{l-1}') \in D_{\Re}$. The procedure employed by this crossover to generate the new offspring string o from the parents $s_1, ..., s_n$ in matrix G is shown in figure 1. The descendant o' is obtained from o as it will be seen from formula 7.

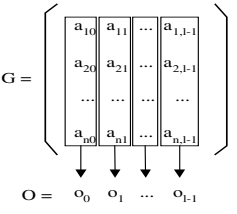


Fig. 1. Generating the new descendant *o* from the parents

2.1 The Morphological Gradient Operator

In the first step of the algorithm, the morphological gradient operator usually employed on digital images *f* is now applied on each vector f_i with i = 0, 1, .., l-1. In this case, f_i may be considered as a function $f_i: D_{fi} \bullet \Re$, being $D_{fi} = \{1, 2, ..., n\}$ and $f_i(j) = a_{ji}$.

The structuring element taken to build the crossover operator is also a function $b:D_b \bullet \Re$ defined as:

$$b(x) = 0, \forall x \in D_b \text{ , } D_b = \{-E(n/2), ..., 0, ..., E(n/2)\} \quad (3)$$

being E(x) the integer part of x.

The morphological gradient function applied on f_i with the structuring element b is defined by $g_b(f_i)$. From this function g_i is obtained as the value:

$$g_i = g_b(f_i)\ (E(n/2)+1)\ \ i \in \{0, 1, ..., l-1\} \quad (4)$$

The morphological gradient applied to images returns high values when sudden transitions in gray levels values are detected, and low values if the pixels covered by the window (structuring element) are similar. A new interpretation of the morphological gradient has been given when applied on GAs: g_i gives a measure of the heterogeneity of gene i in the individuals chosen to be crossed. If value g_i is high, the population is scattered, while if it is low, that means the values of that gene are converging.

2.2 The Crossover Intervals

This step determines *n crossover intervals* for each of the *n* chromosomes of the individuals of the population: $\{C_1, C_2, ..., C_i, ..., C_n\}$. Each of the *n* pairs o_i and o_i' of the offspring will be taken from the crossover interval C_i. In order to obtain the edges of each crossover interval C_i, let us define φ as the function $\varphi: \Re \rightarrow \Re$, and the maximum gene g_{imax} as:

$$g_{imax} = \max(f_i) - \varphi(g_i) \quad (5)$$

While the minimum gene g_{imin} is defined as:

$$g_{imin} = \min(f_i) + \varphi(g_i) \quad (6)$$

The maximum gene and the minimum gene finally determine de edges of the crossover interval C_i as: $C_i = [g_{imin}, g_{imax}]$.

2.3 Obtaining the Offspring

The final result of MMX is the generation of two new descendants: $o = (o_0, o_1, ..., o_{l-1})$ and $o' = (o_0', o_1', ..., o_{l-1}')$. o_i is obtained by randomly picking a value inside the crossover interval C_i, while o_i' is obtained as:

$$oi' = (\min(fi) + \max(fi)) - oi \quad (7)$$

This way, the following formula is satisfied:

$$o_i + o_i' = \min(f_i) + \max(f_i) = g_{imin} + g_{imax} \quad (8)$$

3 How Does MMX Work?

Function φ, used to obtain the maximum and the minimum genes in formulas 5 and 6, is employed to determine the crossover intervals where the offspring is taken from. This function carries out an important feature that allows MMX to work successfully: the crossover interval may be dynamically extended or narrowed with respect to the *reference interval* $[\min(f_i),\max(f_i)]$ determined by the parents depending on the genetic diversity of the population. When the individuals to be crossed are diverse (which implies a high value of the gradient) the crossover interval is made narrower according to the reference interval, thus allowing to explore its interior searching for the optimum much faster. On the other hand, if the individuals to be crossed are very similar (gradient close to zero), which means that the population is converging, then it is advisable to expand the interval $[\min(f_i),\max(f_i)]$ to allow the exploration of new points in the domain, thus avoiding the possible convergence to a local optimum. This possibility of expanding or narrowing the crossover interval depending on the value of the gradient g_i, must be given by the chosen function φ, which must satisfy the following premises:

- It should have low computational cost because on each application of the operator, φ will be calculated *l* times (the dimension of the individuals of the population).
- Its domain must fit within the range of chromosomes values.
- It is necessary that $g_{imin} \leq g_{imax}$ in order to build the crossover interval C_i. From (5) and (6):

$$\min(f_i) + \varphi(g_i) \leq \max(f_i) - \varphi(g_i) \quad (9)$$

Leaving $\varphi(g_i)$ alone:

$$\varphi(g_i) \leq \tfrac{1}{2}[\max(f_i) - \min(f_i)] \quad (10)$$

From (4), and using the structuring element defined in (3), it may be obtained that:

$$g_i = \max(f_i) - \min(f_i) \quad (11)$$

So we have from (9) and (10):

$$\varphi[\max(f_i) - \min(f_i)] \leq \tfrac{1}{2}[\max(f_i) - \min(f_i)] \quad (12)$$

Obtaining the premise that φ must finally satisfy to assure that $g_{imin} \leq g_{imax}$:

$$\varphi(x) \leq \frac{x}{2}, \ \forall x \in D_\varphi \quad (13)$$

- A range in which function φ returns positive values must exist in order to make the crossover interval narrower. It is also necessary another range in which the function returns negative values in order to expand the crossover interval.
- If $g_i = 0$ then all the values of chromosome i of the individuals to be crossed are equal:

$$g_i = 0 \rightarrow g_b(f_i)(E(n/2)+1) = 0 \rightarrow f_i(j) = k, \ \forall j \in D_{fi}, k \in \Re \quad (14)$$

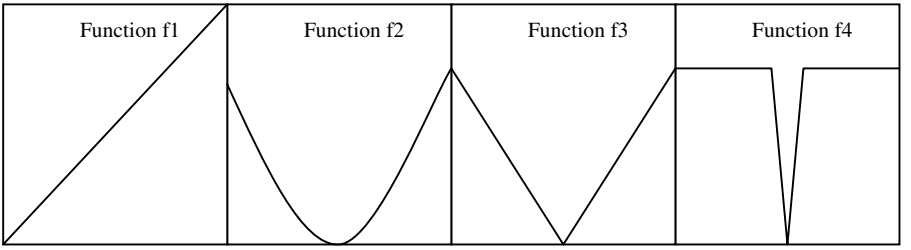


Fig. 2. Test functions f1 to f4

Due to this property, $\varphi(g_i)$ must satisfy the condition: $\varphi(0) \neq 0$; otherwise, it would happen that:

$$g_{imax} = \max(f_i) - \varphi(0) = k$$
$$g_{imin} = \min(f_i) + \varphi(0) = k \quad (\mathbf{15})$$

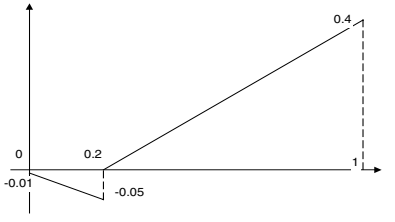
Then the crossover interval is $C_i = [k , k]$, so the value of the offspring gene o_i will be always k. In these conditions, if k is not the optimal value for gene i, the algorithm will be stalled, never reaching an optimal solution.

4 Experimental Results

Six different optimization problems have been used to test MMX. These are the optimization of five different functions (f1,f2, …, f5) and the task of training an artificial neural network to solve the two-spirals problem. Our purpose in choosing functions f1 to f4 (shown in figure 2) was not to come up with challenging problems, but to choose a set of simple functions that enable us to show the major problems presented in other approaches that are solved by MMX. Function f5 and the ANN training test are larger problems that will show how MMX performs high speed of convergence without being trapped in any of the great amount of local optima that these examples present. All the crossover operators employed produces two children from two parents, excepting MMX, which uses a matting pool of five individuals. In each case, a population of 50 individuals has been used and halted the search when either the minimum is reached (with an error less or equal to 10E-4) or the population converged. In order to place the emphasis on the effects of each crossover, no mutation was used in either algorithm. Parents to be crossed have been taken from the population by the roulette-wheel method, the new offspring replaces the worst two individuals of the population.

In case of MMX, the function φ described in section 3 has been empirically chosen, and it is shown in Figure 3. This function, which satisfies all premises previously reported, is defined in domain [0,1], so the population had to be normalized in the same range. This function only performs one floating point multiplication, so its application in the crossover operator described is very efficient, as it allows the generation of a new individual with only l multiplications. $\varphi(g_i)$ is positive when g_i is strictly greater than 0.2, making the crossover interval narrower in this case. On the other

hand $\varphi(g_i)$ takes negative values if $g_i \leq 0.2$, that is, the crossover interval is expanded when the population is converging.



$$\varphi(g_i) = \begin{cases} -(0.2 \cdot g_i) - 0.01 & \text{if} \quad g_i \leq 0.2 \\ (0.5 \cdot g_i) - 0.1 & \text{otherwise} \end{cases}$$

Fig. 3. Function φ employed by MMX

4.1 Failure Tests

Results given by MMX optimizing functions f1 to f4 have been compared to two real-coded crossover operators: RFX and BLX-• with •=0.5; and also compared to the results reported by the binary-coded crossover operator GBX. Function f1 is a simple incline problem with the optimum at one extreme: f(x) = x. The second, f2, is a parabola with the minimum at the center: $f(x) = x^2$. The third, f3, has the shape of a V: f(x) = |x|. Finally, function f4 has the shape of function f3 between points x = -0.1 and x = 0.1, taking the minimum value of 0 at x = 0, and being 1 for all other values of x. For these for functions, x ranges from -2^{29} to $+2^{29}-1$.

Results are reported in table 1 where it can be seen for each of the crossover operators tested the percentage of successful trials over 1500 trials run. RFX is trapped in local minima in most of the cases due to it does not extend the interval outside the extrema determined by the parents. RFX can not even solve f1 problem, but also, RFX is the fastest algorithm when it is able to reach the optimum, which confirm the assumption that it is better to make the crossover interval narrower in the first stages of the convergence process. BLX-0.5 is similar to RFX, but BLX extends the reference interval, which gives better results than RFX except for function f4. GBX finds some problems in function f4 due to most of the points of the domain have the same fitness and the algorithm can not find out which ones are closer to the optimum. Finally, as it was supposed, MMX is never trapped in any local optima. One of the main reasons why RFX, BLX and GBX converge before reaching the optimum is because when the population is converging, there is a high probability of getting a matting pool of two identical parents, then, the children are also the same individuals, and the population quickly converges. There is no way to obtain a child different to its ancestors due to mutation probability is set to 0. This effect does not occurs to MMX, which gives different offspring for identical parents, so it can explore other points, searching for the best solution.

Table 1. Results for functions f1 to f4

	f1 succ. %	f2 succ. %	f3 succ. %	f4 succ. %
MMX	100%	100%	100%	100%
BLX-0.5	65%	91%	83%	44%
RFX	0 %	82%	68%	62%
GBX	100%	100%	60%	43%

4.2 Performance Tests

Function f5 and the task of training an ANN to solve the two-spirals problem have chosen as performance tests. f5 is a two variables sine envelope sine wave function very frequently used to measure the performance of GAs: f5 is cylindrically symmetric about the z axis, the global optimum is located at coordinates (0,0), having infinite local suboptima in concentric circles if seen from above.

MMX has been compared to BLX-0.5 and RFX optimizing function f5. These three crossover operators have similar computational cost, so performance has been measured in terms of the number of crossover operations needed to reach the convergence. Results for this test case, obtained from 1500 trials run, are reported in table 2 that shows the averaged number of iterations needed to reach the optimum value (with the error required), and the percentage of failed trials in the last two columns (the population has converged to a local optimum). The three operators take more or less the same iterations to reach the optimum value. The main differences are that MMX never fails and it is much faster that the other algorithms in the first stages of the convergence process due to MMX is the only algorithm that makes the crossover interval narrower when there is high population diversity.

Table 2. Results for function f5

	Averaged iterations		% Failed trials	
	error•10E-4	error•10E-3	error•10E-4	error•10E-3
MMX	16.234	4.972	0%	0%
BLX-0.5	17.480	6.157	83%	88%
RFX	16.824	5.356	94%	98%

Training an ANN with the new crossover operator proposed has also been tested using the two-spirals problem, where for each of 100 training points belonging to one of two interwined spirals, the network must tell in which spiral a given point belongs to. This is a hard task for descent gradient and GA methods due to its complicated error landscape. In this problem, each of the floating point numbers of the individuals in the population representing the weights and biases of the network are randomly initialized within the interval [-25.0,+25.0]. The size of population is 30, which is quite small, allowing higher efficiency in computational terms. To evaluate the fitness of each individual, feed forward computation is performed by presenting the patterns of one epoch and the mean square error (MSE) is calculated. Comparisons between

MMX and other approaches based on GAs are not shown due to all other approaches were computationally much more expensive that MMX or fell in local optima most of the times. However, in this case, MMX has been compared to backpropagation with quick-drop (BPQ), one of the fastest variants of backpropagation [12].

Table 3 shows the averaged number of epochs needed by BPQ and MMX to reach each level of error (It has been calculated that one BPQ epoch is approximately equivalent to 220 MMX iterations). Finally, last column shows the percentage of failed trails. It can be seen from this table how, although the error landscape is very complicated, MMX does not fall in any optima while performing very high speed of convergence even when approaching to the optimum.

Table 3. Results from the two-spirals problem

MSE	Averaged epochs		% Failed trials	
	BPQ	MMX	BPQ	MMX
10E-3	326	3,74	47,5%	0%
10E-4	401	4,03	52,1%	0%

5 Conclusions

A new general purpose crossover operator called MMX has been proposed. This operator works with real-coded individuals and is based on morphological techniques allowing faster convergence speed than other approaches based on GAs. MMX solves the major problems with GAs: Convergence speed is very high for small and larger problems as different experiments demonstrated. It has also to be noticed that MMX has been designed to avoid the loss of genetic diversity of the whole population in order to prevent the premature convergence to local optima. At the same time, and dynamically with the convergence process, MMX performs local search giving strong local fine-tuning capabilities to the algorithm allowing high speed of convergence not only at the initial stage, but also later by focusing the search. Another important novel feature of MMX is that at the initial stages of the convergence process and when the there is high genetic diversity, MMX makes the interval crossover narrower, which focuses its search, increasing the speed of convergence.

The experiments shown in this paper also demonstrate that GAs with MMX clearly outperforms other crossover operators and other gradient descent methods with a relatively small population avoiding the problem of falling in a local minimum although the landscape error is very complicated.

References

1. Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley (1989).
2. Kureichick, V., Melikhov, A. N., Miaghick, V. V., Savelev, O. V., Topchy, A. P.: Some New Features in the Genetic Solution of the Traveling Salesman Problem. Proceedings of the 2nd International Conference of the Integration of Genetic Algorithms and Neural Network Computing and Related Adaptive Computing with Current Engineering Practice, ACEDC'96, Plymouth (1996) 231-247.
3. Rocha, M., Neves, J.: Preventing Premature Convergence to Local Optima in Genetic Algorithms via Random Offspring Generation. Proceedings of the 12th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Springer-Verlag, Cairo (1999) 127-136.
4. Barrios, D.: Generalized Crossover Operator for Genetic Algorithms. PhD Thesis, Facultad de Informática, Universidad Politécnica de Madrid, Madrid (1991).
5. Eshelman, L. J., Schaffer, J. D.: Real-Coded Genetic Algorithms and Interval-Schemata. Foundations of Genetic Algorithms, 2 (1993).
6. Wright, A.: Genetic Algorithms for Real Parameter Optimization. Foundations of Genetic Algorithms, Morgan Kaufmann, San Mateo, CA (1991) 205-218.
7. Radcliffe, N. J.: Genetic Neural Networks on MIMD Computers. PhD Dissertation, Dept. of Theoretical Phisics, University of Edinburgh, Edinburgh, UK (1990).
8. Ono I., Kobayashi, S.: A Real-Coded Genetic Algorithm for Function Optimization Using Unimodal Normal Distribution Crossover. Proceedings of 7th International Conference on Genetic Algorithms (1997) 246-253.
9. Crespo, J.: Morphological Connected Filters and Intra-Region Smoothing for Image Segmentation. PhD thesis, Georgia Institute of Technology, Atlanta (1993).
10. D'Alotto, L. A., Giardina, C. R.: A Unified Signal Algebra Approach to Two-Dimensional Parallel Digital Signal Processing. Marcel Dekker, New York (1998).
11. González, R. C., Woods, R. E.: Digital Image Processing. Addison Wesley, New York, (1993).
12. Mars, P., Chen, J. R., Nambiar, R.: Learning Algorithms: Theory and Applications in Signal Processing, Control and Communications. CRC Press, New York (1996).