DOES BCGA SIMPLE CROSSOVER HASTEN RCGA CONVERGENCE?

C. Lucas [a], A. Nayyeri [b], and M. Yabandeh [b]

[a] *Center of Excellence on Control and Intelligent Processing, ECE Dept., University of Tehran; and SIS, IPM*
lucas@ipm.ir
[b] *University of Tehran, Faculty of Engineering, Institute of Electrical and Computer Engineering*
{a.nayyeri, m.yabandeh}@ece.ut.ac.ir}

Abstract:

Real Coded Genetic Algorithm is the type of GA which operates on chromosomes with real valued parameters. Different mutation and crossover operations are defined for RCGA. One usable crossover for this kind of GA is to consider its chromosomes simply as bit strings and utilize same operations as Binary Coded GA.
In this paper we attempt to show that this kind of crossover can not hasten the convergence process unless the break points are the boundaries of parameters in the chromosome.

1. Introduction:

Genetic Algorithm is a search methodology invented by Holland [1], which is inspired by the natural genetic theory. Today GA is a very popular and effective method for solving different problems. The most important reasons of this success are rooted in its simplicity and performance.

The main idea of this technique is to generate diverse chromosomes and selecting the most appropriates to continue. In fact in this method we have an initial population of chromosomes which are produced randomly or by a particular tactic. Then, iteratively, we try to generate new generations of population out of the previous ones using mutation, crossover and selection. Mutation is designed to generate a new chromosome out of an existing one by randomly changing it. In the crossover two existing chromosomes are combined to generate new chromosomes. Finally selection will help us to select the new population from the previous population and the new chromosomes which are generated with the help of mutation and crossover so that average fitness will tend to increase. In this paper we wish to point out a very common [2, 3, 9] misuse of the algorithm when decision variables are continuous, but are represented via corresponding binary numbers. In particular, we want to discuss the usual crossover operator.

Each chromosome in the population usually consists of different parameters concatenated to each other. In many theoretical and practical problems this parameters are real numbers. We call the GA working with real parameters in its chromosome RCGA (Real Coded Genetic Algorithm). RCGAs have been used widely for optimization purposes by different people [5, 6, 7].

Crossover is a powerful technique to improve the speed of convergence of the GA. There are different techniques for RCGA crossover [8]. In the most popular one, two chromosomes are broken down into pieces and new chromosomes are configured by random replacement of these pieces. The place of breaking chromosomes plays important role in the performance of the GA. As a matter of fact random selection of this point can decrease the speed of convergence.

In this paper we try to show that breaking the RCGA's chromosomes from any point other than the boundary of their chromosomes can ruin the performance of the GA. In other words breaking a real

parameter from its middle not only does not improve the convergence time of the algorithm, but also can result in an algorithm slower than simple GA without crossover.

In section two we show that there is a theory behind simple crossover and that this theory does not work for the RCGA. Section three provides an empirical study which shows the advantage of simple GA over using crossover with middle parameter breaking point. The last section is devoted to conclusion of the paper.

2. Theory of crossover:

Crossover is the process of combination of two chromosomes with the hope of generating fitter offsprings. Crossover is not a necessary part of an evolutionary strategy; it just hastens the process. An intuition on this matter can be seen in the schema theorem [1]. When two chromosomes are mixed up, new genes with combined attributes are created. If the good part of each chromosome participates in this combination then the new chromosome can be fitter than its parents. So combining different attributes of the genes can cause the emergence of fitter individuals in a population.

There is no reason to believe that by breaking the concatenated strings of chromosomes from an arbitrary point in the middle of the binary representation of a real number useful combination of previously found building blocks could be obtained. Unless the crossover point happens to be in the boundary of the real numbers, it is more plausible to assume that crossing over the binary digits will ruin what has been already found during the succession of previous generations. Factually the second kind of breaking can cause a flaw in the process of searching because it ruins one of the parameters which probably we are already near its wanted value.

Holland [1] represents a theory for the GA stating that the number of fitter schemas is increasing during the process of the Genetic Algorithm. His lower bound for the expected value of the schema H in the generation t is:

$$\mathbf{E}[m(\boldsymbol{H},t+1)] \geq \frac{m(\boldsymbol{H},t)\,f(\boldsymbol{H},t)}{\overline{f}(t)}\left\{1-p_c\,\frac{\delta(\boldsymbol{H})}{1-l}\,p_{diff}(\boldsymbol{H},t)-o(\boldsymbol{H})\,p_m\right\}$$

It can be clearly seen that the values of δ, schema defining length, and o, schema order, play important roles in the formula. In fact the roles of these attributes can be summarized to the building block hypothesis which states that the GA works well when short, low-order, highly-fit schemas (“building blocks”) recombine to form even more highly fit higher-order schemas. In Goldberg's words, “. . . we construct better and better strings from the best partial solutions of past samplings"[2]. However it should be considered that in an RCGA the importance of these parameters are different due to the place of them in the sequence. For example the order and length of the schemas 101*** and ***101 should be interpreted differently; although the length and order of them are equal their values in the process of the solution are extremely different. To be clearer, imagine the value for which we are searching is 101101. Therefore the worst value in the first schema is 101010 which is a much better estimation for the answer than 010101, the worst bit string in second schema. It is also worth noting that because GA usually finds a close estimation of the answer rather that the exact value of it, the arithmetic distance from the exact solution is more important than the number of bits in which our estimation is different from it. This problem can be seen more clearly in some formats of storing real numbers, like floating point, where the bit strings of two arithmeticaly close values maybe absolutely different. In fact the concept of schema cannot be defined similarly in BCGA and RCGA.

In the next session we show an experience that demonstrates this matter.

3. Practical Test

The first example of the Goldberg’s book [2] is finding the maximum of a function. Goldberg, in his book, used one parameter chromosome and the fitness value was generated by the output of the function.

He breaks the only gene in the middle simulating the behavior of the crossover operation. For the first test we chose the tangent function; because it has a deceptive nature; its maximum and minimum are near each other making the search process difficult. The function e^x is considered for another test in the [0, 20] interval; it is a simple growing function and the optimum is obviously at the end of the interval. Both of the tests are done twice, with and without the crossover operator. As parameters for comparison, we choose the number of generations and the required time to reach the answer. Tests were done on a Pentium IV machine and in the same condition one thousand times each. The average results are shown in the following tables.

	Average time (ms)	Average number of generation
With crossover	83.705	318.79
Without crossover	68.285	297.366

Results of the GA on the tan (x) in the [0, 2л] interval

	Average time (ms)	Average number of generation
With crossover	4.343	21.614
Without crossover	3.102	20.491

Results of the GA on the e^x in the [0, 20] interval

4. Conclusion:

The paper has presented a critique of binary coding of real numbers in GA, when the usual genetic operators are used. Although mutations can also be argued to be wasteful, because right hand side binary digits are less significant than the left hand side ones, the paper aimed to show the wasteful nature of the crossover operator in particular. Our conjecture that crossover operations that break down a gene in the middle not only fails to improve the speed of the search process, but also slows it down, has been tested via a benchmark example. This decrease can be the consequence of two notable matters. First, this kind of crossover has a destructive nature, which is explained in the previous session. Second, the consumed time by the crossover operation can be significant. The empirical result of this paper showed that both of these are important enough to be considered in running a GA.

5. References:

[1] J.H. Holland. Adaptation in Natural and Artificial Systems. The University of Michigan Press, 1975.

[2] D.E. Goldberg. Genetic Algorithms in Search, Optimization, and Machine Learning, 1989, ADDISON-WESLEY PUBLISHING COMPANY, INC.

[3] C.A. Perez, C.A. Salinas, P.A. Estévez, and P.M. Valenzuela. Genetic Design of Biologically Inspired Receptive Fields for Neural Pattern Recognition. IEEE TRANSACTIONS ON SYSTEMS, 2003, MAN, AND CYBERNETICS—PART B: CYBERNETICS, VOL. 33, NO. 2, APRIL 2003

[4] C. Michael, G. McGraw, M. Schantz. Generating Software Test Data by Evolution. IEEE Transactions on SOFTWARE TESTING, 2001, Vol. 27, No. 12.

[5] L. Davis. Handbook of Genetic Algorithms, 1991, Van Nostrand Reinhold, New York.

[6] L.J. Eshelman, J.D. Schaffer. Real Coded Genetic Algorithms and Interval-Schemata. *Foundation of Genetic Algorithms 2, 1993,* L.Darrell Whitley (Ed.), (Morgan Kaufmann Publishers, San Mateo), 187-202.

[7] F. Herrera, M. Lozano, J.L. Verdegay. Tuning Fuzzy Logic Controllers by Genetic Algorithms. *International Journal of Approximate Reasoning, 1995,* 12, 299-315.

[8] F. Herrera, M. Lozano, J. L. Verdegay. Tackling Real Coded Genetic Algorithm: Operations and Tools for Behavioral Analysis. Artificial Inteligence Review, 1998, (12):265--319, 1998. Kluwer Academic Publisher. Printed in Netherlands.

[9] G. Grewal, T. Wilson, A. Morton. An EGA approach to the compile-time assignment of data to multiple memories in digital-signal processors. ACM SIGARCH Computer Architecture News, 2003, Volume 31 , Issue 1.

[10] S. Forrest, M. Mitchell. Relative Building Block Fitness and the Building Block Hypothesis. Foundations of Genetic Algorithms 2, 1993.

[11] C. Stephens, H. WaelBroeck. Schemata Evolution and Building Blocks. Evolutionary Computation, Volume 7, 1999, Issue 2.