Adaptive Crossover in Genetic Algorithms Using Pattern Based Method

A. Hamzeh, A. Rahmani
Department of Computer Engineering
Iran University of Science and Technology
{hamzeh, rahmani}@iust.ac.ir

Abstract

Genetic Algorithms (GAs) emulate the natural evolution process and maintain a population of potential solutions to a given problem. Through the population, GA implicitly maintains statistics about the search space. This implicit statistics can be used explicitly to enhance GA's performance. Inspired by this idea, a pattern-based adaptive uniform crossover (PAUX) has been proposed. PAUX uses the statistical information of the alleles in each locus to adaptively calculate the swapping probability of that locus for crossover operation. In this paper PAUX is introduced and examined in some benchmark tests. Experimental results show that using PAUX improves the performance of traditional GAs.

Keywords: Genetic Algorithms, Stochastic Genetic Algorithms, Crossover Operators, Adaptive Genetic Algorithms

1. Introduction

Genetic Algorithms (GAs) emulate the natural evolution process and maintain a population of potential solutions to a given problem, which are evaluated by a problem-specific fitness function. This population is evolved by randomly selecting relatively fit members and performing genetic operations, especially recombination and mutation, on them to generate a new population [1]. With the progress of the GA, the average fitness of the population increases, hopefully leading to the optimal solution(s) to the problem. Through the population GAs implicitly maintain statistics about the search space. That is, useful materials or building blocks permeate in the population. GAs uses the selection, crossover and mutation operators to explicitly extract the implicit statistics from the population to reach the next set of points in the search space. In fact, this implicit statistics in the population can be used explicitly to enhance GA’s performance. Inspired by this idea, a pattern-based adaptive uniform crossover, called PAUX, is introduced in this paper. Pattern is a bit string derived from population using observer schema fitness which guides GA entire overall progress to produce new offspring using two parents with respect to population genotype structure.

2. Related Work on Crossover Operators

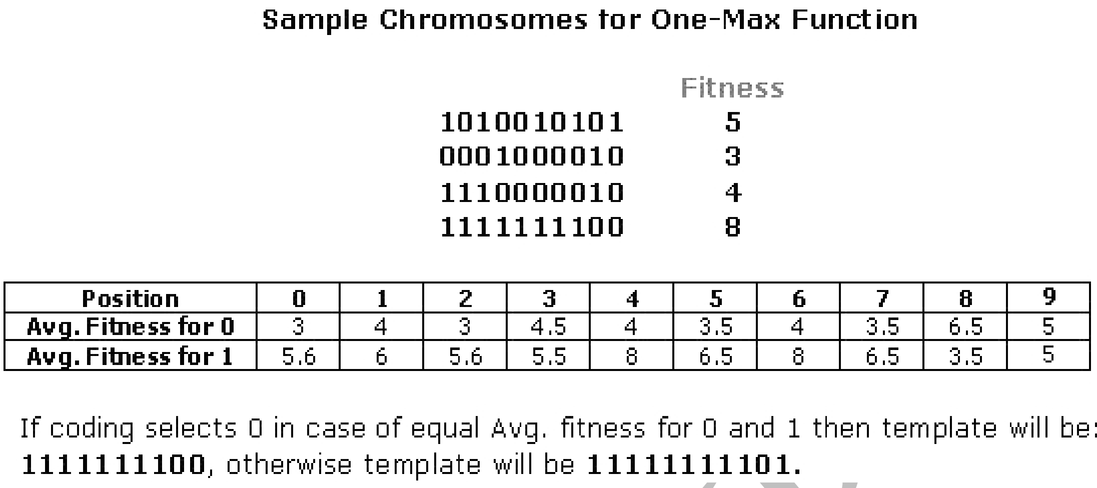
Traditionally, GAs have used one-point or two-point crossover [2, 3]. Researchers have also carried out experiments with multi-point crossover: n-point crossover [4] and uniform crossover [5]. With the n-point crossover, n cut points are randomly chosen within the strings and the n+1 segment between the n cut points of the two parents are exchanged. Uniform crossover is the generalization of n-point crossover. It creates offspring by swapping each bit of two parents with probability p_s of 0.5. [6] Proposes the parameterized uniform crossover where the decision of whether to swap for each locus is made by a biased coin flipping, i.e., the swapping probability P_u could be other than 0.5. Traditional crossover operators are inherently all based on uniformly randomization mechanism, i.e., generating cut points (n-point crossover) or swapping points (uniform crossover) uniformly randomly across the chromosome. This situation is not very true with natural evolution because it is intrinsically dynamic and adaptive. Recently, researchers have applied adaptation techniques to crossover to enhance GAs capabilities [7]. According to [8], adaptation in crossover happens in three levels from top to bottom. In the top level, crossover operators are themselves adapted during a run of the GA. In the medium level, the rate or probability of crossover is altered during a run of the GA. In the bottom level, the position of crossing or swapping probability in each locus is adapted during a run of the GA.

3. Pattern-Based Uniform Crossover operator (PAUX)

According to above information and with respect to this fact that adaptive crossover operators can help GA to achieve better performance, PAUX is designed as an adaptive crossover operator which is based on statistical pattern of entire population. PAUX is created using both selection mechanism and crossover methods. It means that in the case of crossover action, both fitness and crossover patterns are considered to produce offspring. PAUX is based on uniform crossover operator. The Uniform Crossover) uses a P_u as probability of bit swapping between two parents to produce offspring [5]. The first modification to this operator occurs in computing P_u. In PAUX P_u is a variable probability and is computed with respect to parents' fitness. Formula 1 is used by PAUX to compute P_u.

$$P_u = f1 / (f1+f2) \qquad (1)$$

where f1 is fitness of first parent and f2 is second ones'. This value is used as the probability of exchanging bits between parents. This mechanism allows parents with higher fitness to contribute more of their genetic makeup to their offspring. Besides, PAUX uses a new concept called *Template.* Template is used to reflect population statistical information in behavior of crossover operator. In PAUX, template is a chromosome constructed using the entire population with respect to fitness and genetic distribution of population in the following manner: in each position, the entire population is traced and average fitness between all chromosomes with the same gene at that position is calculated, then the gene with highest average fitness, as winner gene, is inserted at the same position in template. For example consider Fig. 1:



Sample Chromosomes for One-Max Function

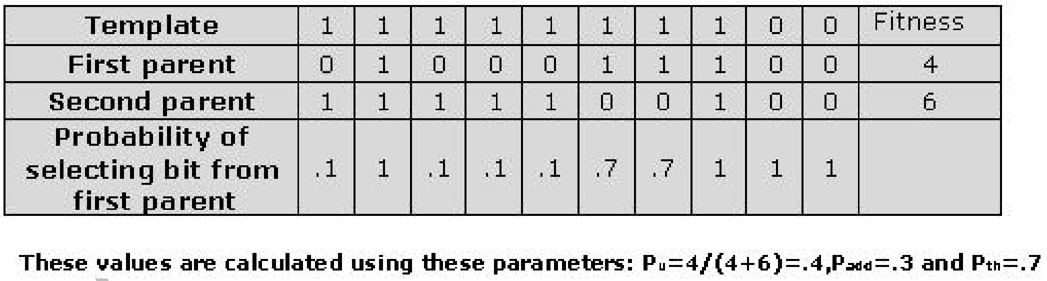
	Fitness
1010010101	5
0001000010	3
1110000010	4
1111111100	8

Position	0	1	2	3	4	5	6	7	8	9
Avg. Fitness for 0	3	4	3	4.5	4	3.5	4	3.5	6.5	5
Avg. Fitness for 1	5.6	6	5.6	5.5	8	6.5	8	6.5	3.5	5

If coding selects 0 in case of equal Avg. fitness for 0 and 1 then template will be: **1111111100**, otherwise template will be **1111111101.**

Fig. 1: Sample template finding in PAUX

After finding Template, real crossover action begins. In PAUX, both parents and template are involved in crossover mechanism and offspring are generated by their interaction. PAUX crosses two chromosomes using following method: after selecting two parents for crossover and finding suitable crossover probability using formula 1, both genes at first position in both parents are compared with each other. If they are not the same, a constant value called P_{add} is added to probability of selecting parent which its first gene agrees with first gene in template. if both genes in parents agree with each other and are not equal to first gene in template, then with probability of P_{th} first gene of offspring comes from template, and if both genes in first position of parents are the same and agree with first gene of template then first gene of offspring will be first gene of template without any consideration. For example see Fig. 2:



Template	1	1	1	1	1	1	1	1	0	0	Fitness
First parent	0	1	0	0	0	1	1	1	0	0	4
Second parent	1	1	1	1	1	0	0	1	0	0	6
Probability of selecting bit from first parent	.1	1	.1	.1	.1	.7	.7	1	1	1	

These values are calculated using these parameters: P_u=4/(4+6)=.4,P_{add}=.3 and P_{th}=.7

Fig. 2: Sample crossover probability calculation in PAUX

Using this method, statistical information of population is also involved to produce new generations. This involvement can help to improve GA performance in many occasions. Also, with respect to this fact that bit selection is biased using Avg. fitness of chromosomes with the same gene, premature convergence can be avoided or recognized rapidly. This is due to this fact that only one chromosome with different gene structure and higher fitness than the local minima can produce many offspring similar to it and can also change template in such a way that to produce more same offspring. In the following sections PAUX is comprised with other crossover operators for some benchmark problems.

4. Test Problems

In this experiment, we use two test functions: One-Max and Modified Royal Road functions. First, we explain these problems:

4.1. One-Max Problem

This problem is one of the well-known GA problems, for a binary string x of length l, this is the problem to maximizing

$$\sum_{i=1}^{l} x_i, \; x_i \in \{0,1\}^l. \tag{2}$$

One-Max is a classical test to confirm that one is able to artificially evolve to a solution starting from a given initial population. The One-Max problem has been extensively studied [9, 10, 11], but with assumptions about selection that do not seem to hold in our situation.

4.2. Royal Road Problem; RR (JH) Version

The `Royal Road' function(s). In this problem each chromosome is regarded as being composed o regularly-spaced non-overlapping blocks of bits, separated by a number of irrelevant bits. Various parameters control the scoring: b, g, and m* are integers and u*, u and v are real numbers. The low-level blocks are of size b bits, with g irrelevant bits making up a gap between each. Each low-level block scores 0 if completely filled or *mv* if it contains m bits and m ≤ m*, or *–mv* if there are more than m* bits set but the whole block is not filled. Thus the low-level blocks are mildly deceptive. In addition there is a hierarchy of completed blocks which earn bonus points. The hierarchy has a number of levels; level 1 is the lowest, and level (j + 1) has half the number of blocks that level j has - the first and second blocks in level j form the first block of level j + 1, the third and fourth from level j form the second and level j +1 and so on. If level j has nj > 0 'filled' blocks then it earns a bonus score of u* + (nj - 1) u; thus u* is a special bonus for getting at least one filled at that level. The topmost layer of the hierarchy has one block, which is filled if and only if every block in the lowest level is filled. [12, 13] report that such `royal road' functions are very hard for some GAs and analyze why. See also the challenge issued by John Holland in the GA list, vol. 7 no. 22. This challenge is used in our experiment; the pseudo code is given in table 1:

Table 1: pseudo code for RR(JH), GAList vol. 7 no. 22

```
Royal Road (JH):
j indexes levels in hierarchy (1 is lowest level).
i indexes target schemata (1 is at left).
There are 2**k target schemata at level 1, and 2**(k-j) target
```

Table 1 (cont.): pseudo code for RR(JH), GAList vol. 7 no. 22

```
schemata at level j+1 (compounded of adjacent pairs of schemata
from the next lower level); each target schema is defined over b loci.
BONUS(j) = u*+(n(j)-1)u, n(j)>0
 = 0, n(j) = 0
where n(j) is the number of found targets at level j and u* and u are
parameters, u*>u.
PART(i)=contribution to overall score from m(i) correct alleles in
target
 schema i at the lowest level, 0<i<1+2**k,
 = m(i)v, if m(i)<m*+1,
 = -(m(i)-m*)v if m*<m(i)<b,
 = 0 otherwise.
(PART introduces simple nonlinearities: The score actually
decreases if there are more than m* correct bits in the target area).
SCORE = Sumj[BONUS(j)]+Sumi[PART(i)].
```

5. Experimental Results

In the above problems, a simple GA using PAUX as crossover operator is compared with simple GAs using one-point, two-point and uniform crossover operators. Fig.3 shows PAUX versus other operators in one-max problem. Results show that pattern-based method is faster than the other operators to evolve population to reach global optimum and also achieves better average fitness.

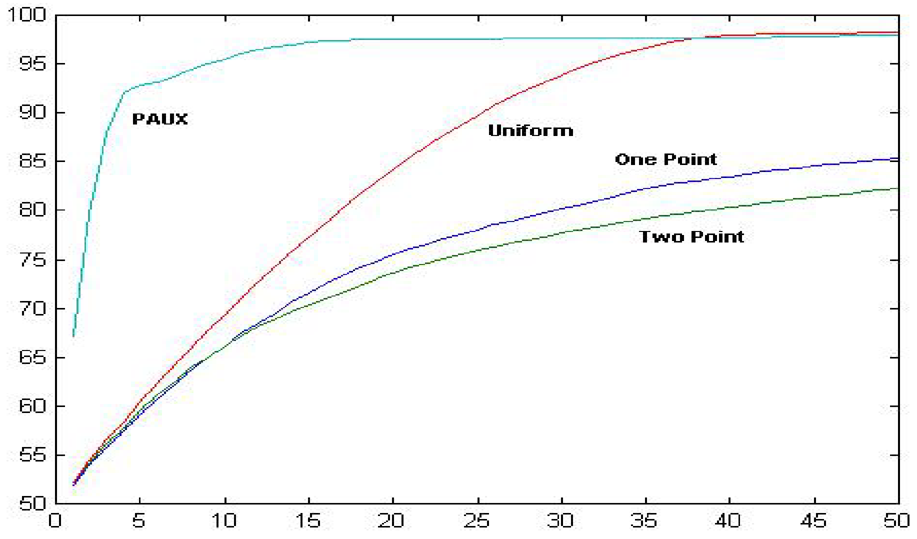
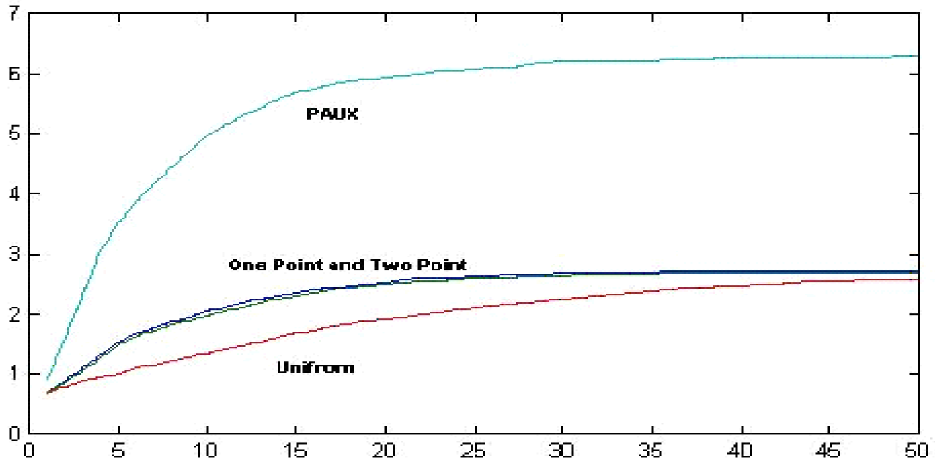


Fig. 3: PAUX versus other crossover operators in one-max problem (horizontal axis is generation and vertical axis is average fitness of population)

This test is done over 100 independent runs with population of size 100 and chromosome length of 100. Fig. 4 shows PAUX versus other operators in RR problem. As expected, results of PAUX are significantly better than other operators. It reaches best performance and achieves best average fitness in every generation. These results are averaged over 100 independent runs with chromosome length of 246 with the following parameters: b=8, g=7, m*=4 and u*=1.0.



4: PAUX versus other crossover operators in RR problem (horizontal axis is generation and vertical axis is average fitness of population)

6. Future Works

It seems that pattern-based methods can improve AGAL performance. Further theoretical analysis is required to investigate the reason behind the astonishing performance of PAUX and its general applicability.

7. Conclusion

In this paper, we propose a new crossover operator, the PAUX that can be used as a crossover operator instead of traditional ones to produce new generations in traditional or modified GAs. This operator increases GA potential in constructing new building blocks while preserving the old ones. The experimental results show that PAUX performs much better than other commonly used crossover operators on the test problems. Our experiment results indicate that PAUX is a very good candidate crossover operator for GAs.

8. References

[1] Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning. Reading*, MA: Addison-Wesley, 1989.

[2] De Jong, K. A. An Analysis of the Behavior of a Class of Genetic Adaptive Systems. PhD Thesis, University of Michigan, Ann Abor., 1975

[3] Holland, J. H. *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: University of Michigan Press 2nd Edition, 1992.

[4] Eshelman, L. J.; Caruana, R. A.; and Schaer J. D. Biases in the crossover landscape. In Scha er, D., ed.. *Proc. of the 3rd Int. Conf. on Genetic Algorithms,* 1989,10-19.

[5] Syswerda, G.. Uniform crossover in genetic algorithms. In Schaer, J. D., ed., *Proc. of the 3rd Int. Conf. on Genetic Algorithms, 1989.*

[6] Spears, W. M., and De Jong, K. A.. On the virtues of parameterized uniform crossover. In Belew, R. K., and Booker, L. B., eds., *Proc. 4th Int. Conf. on Genetic Algorithms, 230-236. Morgan Kaufmann.* 1991.

[7] Eiben, A. E.; Hinterding, R.; and Michalewicz, Z. Parameter control in evolutionary algorithms. *IEEE Trans. on Evolutionary Computation 3(2)*, 1999,p124-141.

[8] Yang, S. Adaptive non-uniform crossover based on statistics for genetic algorithms. In Langdon, W. B., et al, eds., *Proc. of the Genetic and Evolutionary Computation Conference in Artificial Life VIII*, 2002. pp 182{185

[9] Philippe Gigure and David E. Goldberg. Population sizing for optimum sampling with genetic algorithms: A case study of the Onemax problem. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David E. Gold-berg, Hitoshi Iba, and Rick Riolo, editors, Genetic Programming 1998: *Proceedings of the Third Annual Conference,* 1993, pages 496

[10] David E. Goldberg and Kalyanmoy Deb. A comparative analysis of selection schemes used in genetic algorithms. *In Foundations of Genetic Algorithms, FOGA'91, proceedings.* 1991.

[11] David E. Goldberg, Kalyanmoy Deb, and James H. Clark. Genetic algorithms, noise, and the sizing of populations. *Complex Systems, 1992.*

[12] M.Mitchell, S.Forrest and J. Holland, The royal road for genetic algorithms" in (eds) F.J.Varela and P.Bourgine, Towards a Practice of Autonomous Systems: *Proceedings of the First European Conference on Artificial Life, MIT Press, 1992.*

[13] S. Forrest and M. Mitchell., What makes a problem hard for a genetic algorithm? Some anomalous results and their explanation, *Machine Learning, vol. 13,* 1993, pp. 285--319.