

Introducing The Swingometer Crossover And Mutation Operators For Floating-Point Encoded Genetic Algorithms

Shane Lee and Hefin Rowlands

University of Wales, Newport
School of Computing and Engineering
Allt-yr-yn Campus
PO Box 180
Newport Gwent UK
Telephone (01633) 432441
Fax (01633) 432442
s.lee@newport.ac.uk
h.rowlands@newport.ac.uk

Abstract

Genetic algorithms that utilise floating point-encoded chromosomes instead of the traditional binary or Gray code have become wide spread in recent years. This paper introduces new floating-point crossover and mutation operators for such genetic algorithms. The operators are derived from examining the implicit constraints that traditional binary crossover and mutation operators impose on the values of the parameters being affected. It is shown by an example that a genetic algorithm using these operators does converge and they should be considered for general use in floating-point genetic algorithms.

Keywords Genetic Algorithm, Floating-point encoded, Real encoded

1. Introduction.

Genetic algorithms are now a well-known set of search algorithms based on the ideas and theories of natural selection [Goldberg 1989; Holland 1975; Workman & Reader 2004]. Traditionally, a population of binary or Gray coded candidate solutions are created at the initialisation stage of a genetic algorithm, the population ‘evolves’ by utilising a collection of selection, recombination and mutation operators.

However, several possible weaknesses have been identified with genetic algorithms that use binary or Gray code [Wright 1991; Oyama A Et al 2000]. An alternative encoding method using real numbers, often floating point numbers has become popular in recent years. One such weakness of binary encoding is the resolution. Consider a binary bit string of defined length l . The string can represent 2^l real numbers. The range of the real numbers that the string represent is bounded by the upper value R_U and lower value R_L , where $R_U - R_L = 2^l$. The resolution is restricted by the value of the least significant bit. Floating-point numbers are not subject to this restriction. Floating-point numbers in genetic

algorithms would also be free of the necessity of $R_U - R_L$ being a ‘round’ or normalized base two number. If a problem to be tackled by a binary genetic algorithm is not neatly bounded by a ‘round’ base two number in each dimension then new operators may need to be introduced in order to disqualify chromosomes that have parameters outside the required range.

Wright [1991] analyses the effect crossover has on the real values of parameters represented by the bit strings in binary and Gray encoded genetic algorithms by viewing the change in the ‘perturbations’ in the bit strings. However, implementing this view of crossover is necessarily conscious of the binary/Gray bit strings that might represent real numbers. But real floating-point numbers have a potentially infinite number of decimal places and therefore defy conversion to a limited length binary bit string without making approximations that would reintroduce the resolution weakness. Wright [1991] proposes an alternative crossover operator for real numbers named ‘Linear Crossover’ that is quite unlike the conventional crossover operators used in binary/Gray coded genetic algorithms and doesn’t implement the ‘perturbations’ revealed in his analysis. Djurišić et al [1997] use a crossover operator that restrict themselves to swapping complete floating-point encoded genes, as each real encoded gene is equivalent to many binary encoded genes the mixing in recombination does not produce new values for any parameter.

2 The Implicit Constraints of Binary Encoded Crossover.

2.1 Traditional Crossover

To illustrate the implicit constraints that traditional crossover imposes, it is wise to consider an example. Figure 1 shows an instance of single point crossover on a pair of binary encoded chromosomes. In the example, the chromosomes have two dimensions; each

To give these assertions algebraic form consider two parent chromosomes 'A' and 'B' each with 'n' parameters represented by floating-point values $(A_1, A_2, \dots, A_i, \dots, A_n)$ and $(B_1, B_2, \dots, B_i, \dots, B_n)$. The crossover point falls within parameter i . The child chromosomes are $(A_1, A_2, \dots, A_i - \Delta, \dots, B_n)$ and $(B_1, B_2, \dots, B_i + \Delta, \dots, A_n)$. Where ' Δ ' is the absolute value that modifies each affected parameter. Therefore the first constraint can be expressed as equation 1 where the left hand side of the equation is the affected parameters prior to crossover and the right hand side represents the new parameters post crossover.

$$(A_i + B_i) = (A_i - \Delta + B_i + \Delta) \dots\dots\dots 1$$

2.3 Difference between parameters, The Second Constraint.

Parameters with similar bit strings modify each other far less than those with dissimilar ones. The extreme case of identical bit strings cannot modify each other at all. In fact the magnitude of the modification is limited to the difference between them. Figure 3, the parameters differ by only one bit, no matter where the crossover point is, parameter A_i can change by no more than the value of the bit that differs from B_i .

Parent Chromosome parameter A_i	1 1 1 1 1 1	= 63
Parent Chromosome parameter B_i	1 1 0 1 1 0	= 54
Child Chromosome parameter A_i	1 1 1 1 1 0	= 62
Child Chromosome parameter B_i	1 1 0 1 1 1	= 53

Figure 2, Illustration of the difference between parameters constraint.

In general algebraic form the second constraint is

$$\Delta \leq |A_i - B_i| \dots\dots\dots 2$$

2.4 Boundary Restrictions, The Third Constraint.

The value of a parameter after crossover can be no more than the value of the upper boundary ' R_U ' no less than their lower boundary ' R_L '

allowed for that dimension. With brief inspection of figure 3, where the shown parameters have an upper boundary of 63 and lower boundary of 0, it can be seen that ' Δ ' cannot be greater than 63 or less than 0. In algebraic form these constraints for ' Δ ' using parameters A_i and B_i can be expressed as:-

$$\Delta \leq R_U - A_i \dots\dots\dots 3a$$

$$\Delta \leq A_i - R_L \dots\dots\dots 3b$$

$$\Delta \leq R_U - B_i \dots\dots\dots 3c$$

$$\Delta \leq B_i - R_L \dots\dots\dots 3d$$

3 The Swingometer Creating the Operator.

3.1 Creating the Operator.

The first step for the new floating-point crossover operator is to calculate the maximum value that each parameter can have subtracted without violating the constraints described in sections 2.3 and 2.4. These values, $\text{Max } \Delta_A$ and $\text{Max } \Delta_B$, can be derived by choosing the minimum value allowed as expressed by equations 4a and 4b.

$$\text{Max } \Delta_A = \min [|A_i - R_L|, |A_i - B_i|, |R_U - B_i|] \dots\dots 4a$$

$$\text{Max } \Delta_B = \min [|B_i - R_L|, |A_i - B_i|, |R_U - A_i|] \dots\dots 4b$$

Parameter A_i can have no more than Δ_A subtracted from it without violating the second and third constraints. Of course the argument is equally true for B_i . It follows that the maximum value allowed by the first constraint ' $\text{Max } \Delta$ ' is given by equation 5.

$$\text{Max } \Delta = \min [\text{Max } \Delta_A, \text{Max } \Delta_B] \dots\dots\dots 5$$

The value ‘ Δ ’ for the crossover is calculated by generating a random number between 0 and ‘Max Δ ’. The final step to implement the operator is to randomly choose to subtract ‘ Δ ’ from one parameter and add it to the other. This automatically imposes the first constraint (section 2.2).

3.2 Visualising the Operator.

The floating-point operator can be visualised as a ‘swingometer’ (figure 4). The arc of the swingometer is delimited by the values +Max Δ and -Max Δ , represented in figure 4 by points ‘a’ and ‘b’. The length of the arc is 2Max Δ . A value ‘ Δ ’ is selected randomly along the length of the arc. If the value ‘ Δ ’ is between ‘a’ and ‘ $\Delta = 0$ ’ it is added to A_i while simultaneously being subtracted B_i ; but if ‘ Δ ’ is between ‘ $\Delta = 0$ ’ and ‘b’ it is added to B_i while simultaneously being subtracted A_i .

4. The Swingometer Mutation Operator For Real Encoded Genetic Algorithms

The swingometer mutation operator has a very similar derivation to the equivalent swingometer crossover operator. However, the only constraint that applies for mutation is similar to the boundary restriction described in section 2.4. A parameter ‘ A_i ’ selected for mutation after the operation cannot be greater than ‘ R_U ’ or less than ‘ R_L ’. For mutation consider ‘ Δ ’ to be a number added to ‘ A_i ’ or subtracted from ‘ A_i ’. The maximum value that can be added to ‘ A_i ’ is given by equation 6a and the maximum value that can be subtracted from ‘ A_i ’ is given by equation 6b.

$$\begin{aligned} \text{Max } +\Delta &\leq R_U - A_i \dots\dots\dots 6a \\ \text{Max } -\Delta &\leq A_i - R_L \dots\dots\dots 6b \end{aligned}$$

For the swingometer, in figure 4, the limits ‘a’ and ‘b’ are Max + Δ and Max - Δ respectively. As for crossover, a random number is generated in along the arc ‘a’ to ‘b’ and the parameter A_i is modified accordingly.

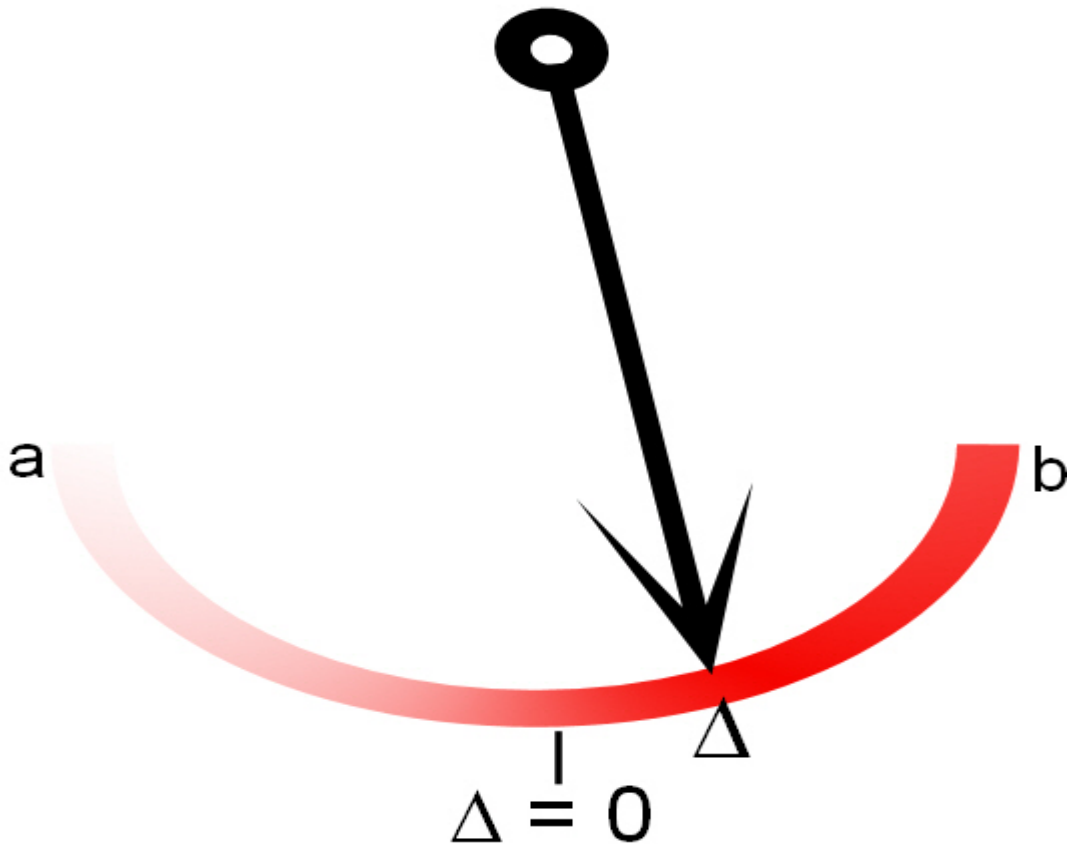


Figure 4, Visual representation of the swingometer crossover and mutation operators.
 For crossover a = Max + Δ and b = Max - Δ .
 For mutation a = +Max Δ and b = -Max Δ .

5. A Working Example.

This section shows the results of a set of 60 identical experiments that demonstrate the new operators in use within a genetic algorithm. The genetic algorithm used a population of 25, a crossover rate of 0.9 and a mutation rate of 0.2. The function that was optimised (equation 7) is the 5th of the Dejong suite[1975] normalised to give a global optimum fitness of 10,000. For each generation the fitness of the fittest chromosome is recorded and averaged for the 60 identical experiments.

$$F_5 = \left[0.002 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^2 (x_i - a_{ij})^6} \right]^{-1} \dots 7$$

Figure 5 shows the result; the genetic algorithm does perform as expected. The fitness of the fittest in the population rises generation by generation until a plateau of convergence is reached.

6 Discussion And Further Work.

This paper has set out to describe two new operators for floating-point encoded genetic algorithms and has shown that they do allow the algorithm to converge. The two operators have

been extensively and successfully used in [2002]. However, no attempt has been made to measure its relative efficacy when compared to other floating-point operators; experiments to fill this gap are a priority. Even though the new operators have yet to be fully tested they are simple to implement and do work and should be considered when designing a floating-point encoded genetic algorithm.

7. References.

- K. Dejong , 1975, An Analysis of the Behaviour of a Class of Genetic Adaptive Systems, Doctoral Thesis, University of Michigan.
- Djurišić A Et Al, 1997, Genetic Algorithms For Continuous Optimization Problems – A Concept Of Parameter-Space Adjustment, Journal of Physics A-mathematical & General. Pp 7849-7861.
- D. E. Goldberg, 1989, Genetic Algorithms in Search, Optimization & Machine Learning, Addison & Wesley.
- J. H. Holland, 1975, Adaptation In Natural And Artificial Systems, The University of Michigan Press.
- S. Lee, 2002, Genetic Algorithms, Orthogonal Arrays And Diploid Chromosomes, Doctoral Thesis, University of Wales

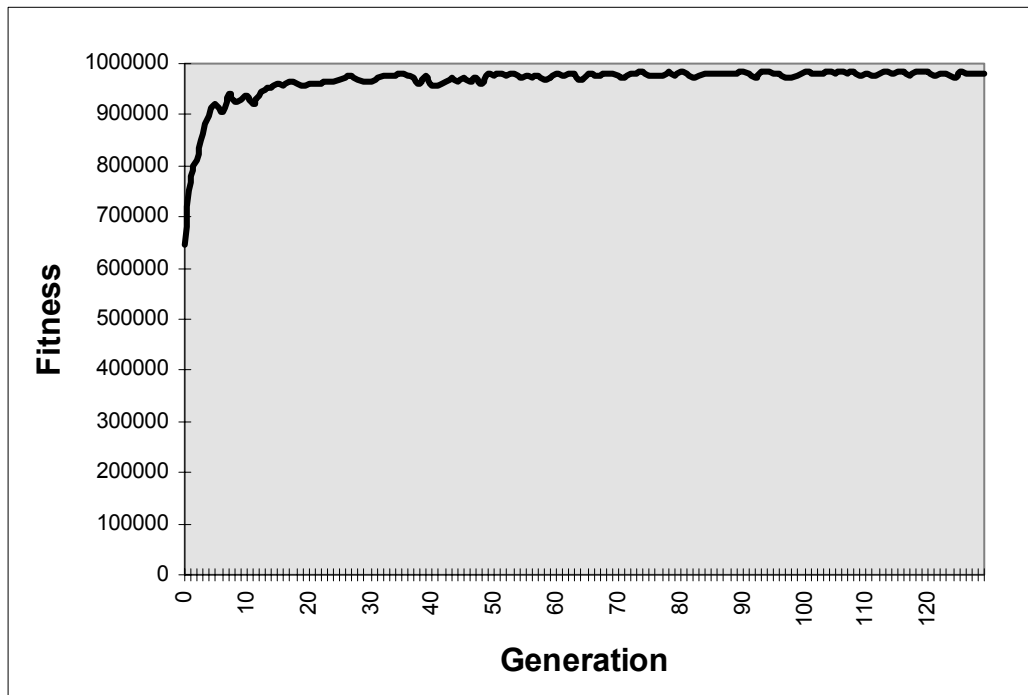


Figure 5, Result of the experiments to demonstrate that the new operators can work within a genetic algorithm.

Oyama A Et al, 2000, Real_Coded Adaptive Range Genetic Algorithm and Its Application to Aerodynamic Design and its Application, JSME International Journal Series A, Vol 43, Part 2, pp 125-129.

L.Workmen & W. Reader, 2004, Evolutionary Psychology: An Introduction, Cambridge University Press.

A. Wright , 1991, Genetic Algorithms for Real Parameter Optimization, Foundations of Genetic Algorithms , pp205-218.

8. Biographies



Dr Shane Lee

Senior Lecturer, School of Computing and Engineering –UW,N

His research field is computational intelligence in particular evolutionary computation and covers areas related to traditional engineering applications, computer games and art. He is the programme leader and creator of the BSc (hons) Games Development and Artificial Intelligence.

Outside academia he is a promoter and performer of stand-up comedy.



Professor Hefin Rowlands

Head of Research & Enterprise, School of Computing and Engineering –UW,N

Professor Hefin Rowlands is responsible in promoting, developing and leading research and enterprise activities in the University College's School of Computing and Engineering.

His research field, in the area of Quality Systems, covers research into Quality Management and Strategies including Six-sigma and Business Excellence Models.

Research interests also cover process modelling and developing an integrated model for business systems.

He has been involved in many successful TCS and College & Business Partnership (CBP) projects which enables local businesses to benefit from the expertise of a recently qualified graduate and the knowledge base of the School of Computing and Engineering at the University College.