The New Negative Slope Coefficient Measure

STJEPAN PICEK
Ring Datacom d.o.o.
Trg J. J. Strossmayera 5, 10000 Zagreb
CROATIA
stjepan@ring.hr

MARIN GOLUB
Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb
CROATIA
marin.golub@fer.hr

Abstract: When is a problem easy or difficult for a genetic algorithm? This work focuses on unitation functions as tests for the efficiency of a genetic algorithm in reaching an optimal solution. We research the effectiveness of the Negative Slope Coefficient Measure (NSC measure) in finding difficult problems and present flaws of such a measure. In summary, we present a new measure for defining the hardness of a problem, the new NSC, based on the Fitness Landscape; experimentally we demonstrate the efficacy of the method and compare it with the performance measure achieved by real runs. Finally we propose new steps for development of the method.

Key–Words: Genetic Algorithm, Unitation, Fitness Landscape, Negative Slope Coefficient, Hardness

1 Introduction

Genetic algorithms (GA) are successfully applied to a variety of problems. However, they have also shown disappointing results. Considering the wide-spread use of genetic algorithms it is necessary to know in which cases they will be successful, and in which they will not find an optimal solution, that is, which problems are difficult for them to solve and which are not. Based on the works of Bethke (1980), Goldberg (1987) has introduced the term deception in order to help understand the situations that will create a problem to the genetic algorithm when performing optimisation tasks. Today there are a number of explanations that describe problems that are difficult for a GA. One important concept is the Fitness Landscape. The Fitness Landscape presents a powerful metaphor for global optimisation. It presents a visualisation of the link between the genotype or phenotype in the given population and their respective probability of reproduction.

Because it is impossible to define a fitness landscape in practice because the solution space is simply too large, over recent years and in many different ways researchers have been looking for ways of defining interesting characteristics of a fitness landscape. Among others, one should certainly mention as a measure of the ruggedness of a fitness landscape the works of Weinberger, Jones and Forrest, and Vanneschi. Weinberger introduced the autocorrelation function and the correlation length for random walks. Jones and Forrest proposed the fitness distance correlation (FDC) - the correlation of the fitness of an individual and its distance from the global optimum. FDC presents a very reliable measure of problematic difficulty for a GA. However, it has several flaws. Among others, the need to know the optimal solution upfront is significant. Vanneschi et al. first introduced the Negative Slope Coefficient measure (NSC measure), which could be considered as an extension of Altenberg's evolvability measure [21]. Vanneschi also introduced the Fitness-Proportionate Negative Slope Coefficient measure as a supplement to his NSC method. A big advantage of these methods is that they are predictive, that is, it is not necessary to know an optimal solution prior to an experiment itself [20].

In section 2, there is background information necessary for understanding this work, section three defines the parameters used in experiments and sets up performance measure for the comparison measurement. Section 3 contains experiments conducted on unitation functions and the results achieved for the NSC measure; Section 4 repeats the experiments, but this time for the new measure, and finally Section 5 draws a conclusion and future guidelines.

2 Background

This section addresses the information necessary for a complete understanding of the article. Here the terms, Fitness Landscape, Fitness Clouds, Unitation Functions that are used in the work, Metropolis-Hastings Sampling and the NSC, are explained.

2.1 Fitness Landscape

The fitness landscape can be defined as a search space (S), a metric and scalar fitness function defined on the

elements of S. If we assume that the goal is to maximize fitness, we may assume that the best global solutions are the 'peaks' in the search space. We can also define local optima in the following way: first we assume a non-negative-real valued, scalar fitness function f(s) over a binary string s of the length l, where

$$f(s)\epsilon\Re \geq 0 \ . \tag{1}$$

Generally, let us presume that 'f' has to be maximized. The local optimum in the discrete S search space is a point or an area, whose fitness function value is larger than of the all of its closest neighbours. A Region is considered to be a Neutral Network having the same fitness. In other words, a set of points in proximity of the nearest-neighbour points of equal fitness are considered as a single optimum. For a move operator in search space we use bit mutation. Under the term 'nearest neighbour' we consider a measure that shows the distance between points s_1 and s_2 where s_1 and s_2 represent two binary strings. In this work, the Hamming distance was used (i.e. the number of bit positions by which two binary strings differ) [7].

2.2 Fitness Clouds

Let $\gamma = (\gamma_1, \gamma_2, ..., \gamma_n)$ represent the entire search space of a GA problem and $V(\gamma)$ be a set of all the neighbours of the $\gamma\epsilon\Gamma$ individual, which we obtained by the application of a standard bit-flip mutation. The choice of neighbours is a result of a tournament selection method with k=10 being the selection parameter. An individual with the highest fitness value is picked up as a neighbour. We can define the following set of points at the bi-dimensional plane:

$$P = \{(f(\gamma), f(\nu)), \forall\gamma\epsilon\Gamma, \forall\nu\epsilon V(\gamma)\} \ . \tag{2}$$

The P diagram is a scatterplot fitness of the values of all the individuals that belong to the search space versus their neighbours' fitness. A fitness cloud implicitly gives an insight into a genotype against the phenotype mapping [19].

Metropolis-Hastings Sampling Generally, the search space is too big to consider all the individuals. Therefore we use samples, and since all the points are not equally important, we want to sample the space by using a distribution that puts more weight on individuals with a higher fitness value. In order to achieve that, we use Metropolis-Hastings sampling, which is an extension of the Metropolis sampling towards non-symmetric stationary probability distributions [19].

2.3 Negative Slope Coefficient

The Negative Slope Coefficient is an algebraic measure for problem difficulty. It can be calculated in the following way: the fitness cloud C is divided into a certain number of segments $C_1, ..., C_m$ which are such that $\left(f_a, f_a^{'}\right)\epsilon C_j$ and $\left(f_b, f_b^{'}\right)\epsilon C_k$, where j ¡ k implies that $f_a < f_b$. An average fitness is calculated as:

$$\overline{f_i} = \frac{1}{C_i} \sum_{(f,f')\epsilon C_i} f \ . \tag{3}$$

and

$$\overline{f_i^{'}} = \frac{1}{C_i} \sum_{(f,f')\epsilon C_i} f^{'} \ . \tag{4}$$

The points $\left(\overline{f_i}, \overline{f_i^{'}}\right)$ can be viewed as polyline peaks, which successfully represent a 'skeleton' of the fitness cloud. For each of these segments a slope can be defined,

$$\overline{S_i} = \left(f_{i+1}^{'} - f_i^{'}\right) / \left(f_{i+1} - f_i\right) \ . \tag{5}$$

Eventually, an NSC is defined as:

$$nsc = \sum_{i=1}^{m-1} min\left(0, S_i\right) \ . \tag{6}$$

If NSC = 0 then the problem is easy, and if the NSC < 0 then the problem is difficult and the NSC value shows to which extent it is difficult [15].

2.4 Unitation and Functions of Unitation

In this work we use Onemax, Onemix and Trap unitation functions [10] [16].

Definition 1 *Let s be a bit string of the length l. The unitation $u(s)$ of s is a function defined as:*

$$u(s) = u(s_i...s_l) = s_1 + ... + s_n = \sum_{i=1}^{l} s_i \ . \tag{7}$$

In other words, unitation represents the number of units in the bit string.

2.4.1 Onemax Function.

Onemax functions are generalizations of the unitation u(s), of a bit string s:

$$f(s) = du(s) \ . \tag{8}$$

where d in a general case is 1.

2.4.2 Trap Function.

Deb and Goldberg have defined the trap function as follows:

$$f(s) = \begin{cases} \frac{a}{z}(z - u(s)) & \text{if u (s)} \leq \text{z} \\ \frac{b}{l-z}(u(s) - z) & \text{otherwise .} \end{cases} \quad (9)$$

Where 'a' represents a local optimum, 'b' is a global optimum and 'z' is a slope-change location. It can be demonstrated that if the following relation is valid, then the trap function is completely deceptive.

$$\frac{a}{b} = r \geq \frac{2 - \frac{1}{l-z}}{2 - \frac{1}{z}} . \quad (10)$$

2.4.3 Onemix Function.

This function is a mixture of the Onemax problem and a Zeromax problem. Like these functions, it is a function of unitation, u, which represents a number of 1's in a string. Our new function becomes an Onemax function when the unitation values are higher than l/2. If the unitation values are lower, it is Onemax when u is odd; otherwise it is a scaled version of Zeromax. Onemix is formally defined as:

$$f(s) = \begin{cases} (1+a)\left(\frac{l}{2} - u(s)\right) + \frac{l}{2} & \text{if g (s)} \\ u(s) & \text{otherwise .} \end{cases} \quad (11)$$

where g(s) is equal to 1 when u(s) is even and u(s)$<$1/2. Value 'a' represents a constant that is higher than 0 [16].

3 Experimental Parameters

Binary strings of length l = 10 were used in the experiments, by Metropolis-Hastings sampling 100 individuals that constituted the first generation were picked up and a standard bit-flip mutation was used with a p_m mutation coefficient for obtaining neighbours. For each individual, 10 neighbours were generated by the mutation operator and the one with the highest value of fitness was picked.

In the Onemax function example, all the experiments prove that the NSC works properly, which indicates that the Onemax is an easy problem to solve for a GA, which is in accordance with the performance measure.

In order to present flaws of the NSC as a problem difficulty measure we will conduct a number of experiments with different parameters for the Trap Function. The parameters for the Trap function are; local optimum 'a' equals 10, global optimum 'b' equals 11 and slope-change location 'k' equals 9. The parameter predefined (marked predef. in table) value set at yes marks that in this starting population each possible fitness value appears at least once. The parameters are chosen to be, as much as possible, in accordance with the previous research mentioned in the work [20]. The division into segments in the first 4 experiments is made by setting the segment width to the constant value d, where d = 1, and in the other 5 experiments we used size driven bisection where 10 is the minimal number of points that may belong to a segment and 10% presents the minimal difference between the leftmost and the rightmost points contained in a segment. All results in tables represents the lowest values of NSC we calculated. Mean value is in more than half of experiments of value 0. Our experiments show that the NSC is a good measure for the Trap function only when the mutation coefficient is relatively high, what can be seen in experiments 3, 4 and 5. Also, we see that NSC depends on members of first population (as displayed in experiment 3, table 2.). For the Onemix function, the NSC correctly presumes that the problem is difficult, but it shows incorrectly that the Onemix problem is more difficult than the Trap problem, which does not correspond to the results achieved by the performance measure [20]. The parameters used in the experiments and the NSC results obtained are presented in tables below.

Table 1: NSC experimental results for Onemax function.

onemax function	pop_size	p_m	NSC
experiment 1	11	0.05	0
experiment 2	11	0.10	0
experiment 3	100	0.05	0
experiment 4	100	0.10	0

Table 2: NSC experimental for Trap function results for small population size and predefined individual values.

trap function	pop_size	p_m	predef.	NSC
experiment 1	11	0.05	yes	0
experiment 2	11	0.05	no	0
experiment 3	100	0.10	yes	-2
experiment 4	100	0.10	no	0

Table 3: NSC experimental results for Trap function.

trap function	pop_size	p_m	NSC
experiment 1	100	0.05	0
experiment 2	100	0.10	0
experiment 3	100	0.20	-0.79
experiment 4	100	0.30	-1.29
experiment 5	100	0.40	-0.025

Table 4: NSC experimental results for Onemix function.

onemix function	pop_size	p_m	NSC
experiment 1	11	0.05	0
experiment 2	11	0.10	-2
experiment 3	100	0.05	-4.335
experiment 4	100	0.10	-3.667
experiment 5	100	0.20	-0.596
experiment 6	100	0.30	-0.079
experiment 7	100	0.40	0

4 The New Negative Slope Coefficient Measure

In order to show in an experiment whether a function is difficult for a GA or not, we propose a modification of the Negative Slope Coefficient, which we call new NSC. This measure is based on the relation between the number of units with value 1 on the x-line of the diagram and the fitness value for the individuals at the y-line of the diagram.

The experiments are based on the assumption that the maximum fitness value should be obtained. If there are no changes in the direction of the line and if the slope coefficient is positive, then the problem is easy: however if the slope coefficient is negative, then the problem is difficult. If there are changes to the slope segments of the line, then we calculate the slope for each of these segments. Each segment of the line S_1,...,S_m is defined minimally by two points with $V_1(x_1, y_1)$ and $V_2(x_2, y_2)$ coordinates. Then the slope of each segment S_i is defined by the formula:

$$a = \frac{y_2 - y_1}{x_2 - x_1} \ . \tag{12}$$

Values x_1 and x_2 represent two neighbour values of unitation and values y_1 and y_2 two values of fitness or fitness offsprings.

The total slope for the fitness cloud with the number of units at the x-line and the individual fitness on the y-line is:

$$I_f = \sum_{i=1}^{m} a_i \ . \tag{13}$$

and for the fitness cloud with the number of units at the x-line and the neighbour fitness at the y-line:

$$\overline{I_f'} = \sum_{i=1}^{m} a_i \ . \tag{14}$$

A neighbour is chosen in the same way as in NSC measure where it represents an offspring with the maximum fitness value within 10 iterations. Finally, the new NSC amounts:

$$nsc = \left| \frac{I_f}{I_f'} \right| \ . \tag{15}$$

The bigger the result is, the problem is harder for GA to solve. The pseudo-code for the new NSC measure is:

```
function new_nsc (array_unitation){
 f[]=fo[]= compute(array_unitation);
 i1= segments(array_unitation,f[]);
 i2= segments(array_unitation,fo[]);
 nsc=abs(i1/i2);
}
function segments(x,y){
 foreach value of x,y{
  array_x=slope change value(x);
  array_y=slope change value(y);
 }
 for i=2 to upper_bound(x){
  y=array_y(i)-array_y(i-1);
  x=array_x(i)-array_x(i-1);
  sum+=y/x;
 }return sum;
}
```

Now we repeat the same experiments for the new NSC measures. When new measure can not be calculated exactly, then words 'easy' and 'difficult' represent indication of problem hardness. In tables and figures below are represented results that we obtained. Other figures are not displayed in this paper due to the lack of space.

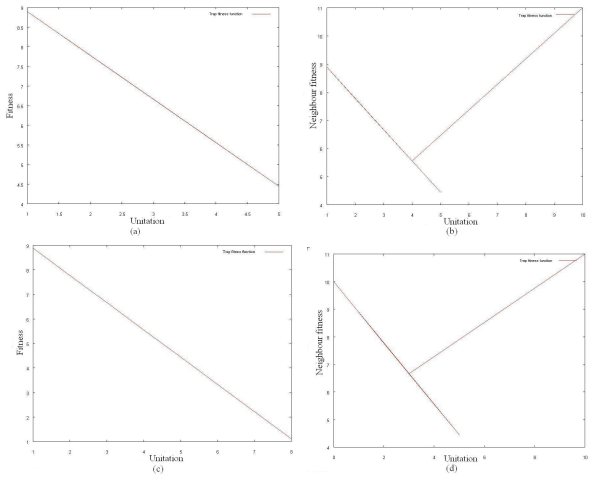

Figure 1: (a) graph unitation/fitness for Trap function experiment 2, (b) graph unitation/fitness neighbour for Trap function experiment 2, (c) graph unitation/fitness for Trap function experiment 4, (d) graph unitation/fitness neighbour for Trap function experiment 4.

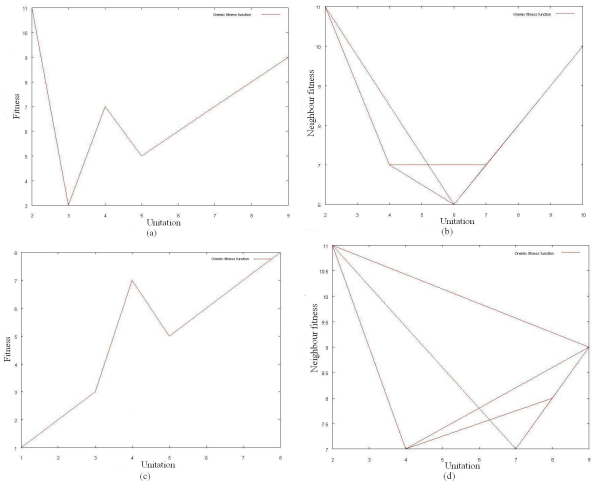

Figure 2: (a) graph unitation/fitness for Onemix function experiment 1, (b) graph unitation/fitness neighbour for Onemix function experiment 1, (c) graph unitation/fitness for Onemix function experiment 2, (d) graph unitation/fitness neighbour for Onemix function experiment 2.

Table 5: Experimental results for new NSC measure for Onemax function.

onemax function	pop_size	p_m	new NSC
experiment 1	100	0.05	easy
experiment 2	100	0.10	easy
experiment 3	100	0.20	easy
experiment 4	100	0.30	easy

Table 6: Experimental results for new NSC measure for Trap function.

trap function	pop_size	p_m	new NSC
experiment 1	100	0.05	difficult
experiment 2	100	0.10	5.55
experiment 3	100	0.20	5.45
experiment 4	100	0.30	2.26

Table 7: Experimental results for new NSC measure for Onemix function.

onemax function	pop_size	p_m	new NSC
experiment 1	100	0.05	1.81
experiment 2	100	0.10	2.80
experiment 3	100	0.20	2.39
experiment 4	100	0.30	1.73

5 Conclusions and Future Work

Experiments on original NSC have shown that it is reliable difficulty measure only in accordance with relatively high mutation factor and individuals in first generation. Also, size driven bisection lacks formality for now, so its parameters must be selected in arbitrary way. Experiments on new NSC have shown that it also depends on individuals in first generation and on mutation rate. In other side the new measure is, in worst case, a reliable indicator of problem difficulty and in best case rather precise measure of problem hardness (difficult or easy). One more advantage of the new NSC is that there is no need to separate the points in segments via arbitrary segment size or size driven bisection (or any other way). We believe that the new NSC has some interesting features, but like original NSC it will always be dependant of mutation rates and individuals in first generation. Further experiments, on larger number of problem functions should be conducted to reach the final decision is it possible in such a way precisely calculate difficulty of a problem. First step must be to find completely reliable indicator of problem difficulty on larger number of different functions.

References:

[1] Y. Borenstein, R. Poli: Information Landscapes and Problem Hardness. Genetic And Evolutionary Computation Conference, Proceedings of the 2005 conference on Genetic and evolutionary

computation, Washington DC, 2005, pp. 1425–1431

[2] K. Deb, D. E. Goldberg: Analyzing deception in trap functions. In: D. Whitley, (ed) Foundations of Genetic Algorithms 2, Morgan Kaufmann, San Franscisco, 1993, pp. 93-108

[3] S. Forrest, M. Mitchell: What makes a problem hard for a genetic algorithm? some anomalous results and their explanation. Machine Learning, vol. 13, Springer, 1993, pp. 285–319

[4] M. Golub: Genetski algoritam: dio I, in Croatian. Faculty of Electrical Engineering and Computing, University of Zagreb, 2004

[5] R. L. Haupt, S. E. Haupt: Practical Genetic Algorithms. John Wiley & Sons, 2004

[6] J. Horn, D. E. Goldberg: Genetic Algorithm Difficulty and the Modality of Fitness Landscapes. In: L. D. Whitley, and M. D. Vose, (eds), Proceedings of the Third Workshop on Foundations of Genetic Algorithms, Morgan Kaufmann, San Francisco, California, 1995, pp. 243–270

[7] T. Jones: Evolutionary algorithms, Fitness Landscapes and Search. Ph. D. thesis, The University of New Mexico, Albuquerque, New Mexico, 1995

[8] T. Jones, S. Forrest: Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In: L. J. Eshelman, (ed) Proceedings of the Sixth International Conference on Genetic Algorithms, Morgan Kaufmann, San Franscisco, 1995, pp. 184–192

[9] M. Madras: Lectures on Monte Carlo Methods. American Mathematical Society, Providence, Rhode Island, 2002

[10] O. J. Mengshoel, D. E. Goldberg, D. C. Wilkins: Deceptive and Other Functions of Unitation as Bayesian Networks. Symposium on Genetic Algorithms, Madison, Wisconsin, 1998

[11] B. L. Miller, D. E. Goldberg: Genetic Algorithms, Tournament Selection and the Efects of Noise. Technical Report, University of Illinois at Urbana-Champaign, 1995

[12] M. Mitchell: An Introduction to Genetic Algorithms. The MIT Press, Cambridge, 1999

[13] M. Mitchell, S. Forrest, J. Holland: The royal road for genetic algorithms: fitness landscapes and ga performance. In: F. J. Varela, P. Bourgine (eds) Toward a Practice of Autonomous Systems, Proceedings of the First European Conference on Artificial Life, The MIT Press, Cambridge, Massachusetts, 1992, pp. 245–254.

[14] S. Picek: Decepcijski problemi, in Croatian. Faculty of Electrical Engineering and Computing, University of Zagreb, 2008

[15] R. Poli, L. Vanneschi: Fitness-Proportional Negative Slope Coefficient as a Hardness Measure for Genetic Algorithms. Genetic And Evolutionary Computation Conference, Proceedings of the 9th annual conference on Genetic and evolutionary computation, London, England, 2007, pp. 1345–1342

[16] R. Poli, A. H. Wright, N. F. McPhee, W.B. Langdon: Emergent Behaviour, Population-based Search and Low-pass Filtering. In: 2006 IEEE World Congress on Computational Intelligence, 2006 IEEE Congress on Evolutionary Computation, Vancouver, Canada, 2006, pp. 395–402

[17] M. Tomassini, L. Vanneschi, P. Collard, M. Clergue: A study of fitness distance correlation as a difficulty measure in genetic programming. Evolutionary Computation,13 (2), MIT Press, Cambridge, 2005, pp. 213–239

[18] L. Vanneschi: Theory and Practice for Efficient Genetic Programming. Ph. D. thesis, Faculty of Science, University of Lausanne, Switzerland, 2004

[19] L. Vanneschi, M. Clergue, P. Collard, M. Tomassini, S. Vrel: Fitness clouds and problem hardness in genetic programming. In: K. D. et al.(ed) Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'04, LNCS vol. 3103, Springer, Berlin, Heidelberg, New York, 2004, pp. 690–701

[20] L. Vanneschi, M. Tomassini, P. Collard, S. Vrel: Negative slope coefficient. A measure to characterize genetic programming. In: P. Collet, M. Tomassini, M. Ebner, S. Gustafson, A. Ekrt (eds) Proceedings of the 9th European Conference on Genetic Programming, vol. 3905 of Lecture Notes in Computer Science, Springer, Budapest, Hungary, 2006 pp. 178–189

[21] T. Weise: Global Optimization Algorithms - Theory and Application, 2008