Examining the Landscape of Semantic Similarity based Mutation

Nguyen Quang Uy
Natural Computing Research & Application Group
University College Dublin
Belfield, Dublin 4, Ireland
quanguyhn@gmail.com

Nguyen Xuan Hoai
Faculty of Information Technology
Hanoi University
Hanoi, Vietnam
nxhoai@gmail.com

Michael O'Neill
Natural Computing Research & Application Group
University College Dublin
Belfield, Dublin 4, Ireland
m.oneil@ucd.ie

ABSTRACT

This paper examines how the semantic locality of a search operator affects the fitness landscape of Genetic Programming (GP). We compare the fitness landscapes of GP search when standard subtree mutation and a recently proposed semantic-based mutation, Semantic Similarity-based Mutation (SSM) [34], are used. The comparison is based on two well-studied fitness landscape measures, namely, the auto-correlation function and information content. The experiments were conducted on a family of symbolic regression problems with increasing degrees of difficulty. The results show that SSM helps to significantly smooth out the fitness landscape of GP compared to standard subtree mutation. This gives an explanation for the better performance of SSM over standard subtree mutation operator.

Categories and Subject Descriptors

I.2.8 [**Problem Solving, Control Methods, and Search**]: Heuristic methods

General Terms

Algorithms

Keywords

Genetic Programming, Semantics, Fitness Landscape, Mutation

1. INTRODUCTION

Genetic Programming (GP) [31, 25] is an evolutionary paradigm for automatically finding solutions for a problem. Since its introduction, GP has been applied to a wide range of fields [31], and routinely exhibits human-competitive performance [24]. So far, GP researchers have largely focused on syntactic aspects of GP representation, and a great deal of research has specifically focused on grammar-based approaches [27]. However, from a programmer's perspective, maintaining syntactic correctness is just a part of program construction: programs must be correct not only syntactically, but also semantically, and this has been recognised as a significant open issue in the field [30]. Thus incorporating semantic awareness in the GP evolutionary process could potentially improve performance. Recently, several researchers have taken an interest in incorporating semantic information into GP, leading to a sharp increase in related publications (e.g., [13, 15, 16, 19, 3, 28, 33, 35, 4]).

In a recent work, Uy et al. proposed a Semantic Similarity based Mutation (SSM) with the aim to improve semantic locality [34]. The basic idea of SSM is keeping a semantically small change in mutation by replacing a subtree with a semantic similar subtree. The experiments in [34] shows that SSM helps to significantly improve GP performance in comparison with standard mutation and another mutation for promoting diversity. However, the reason why the performance of SSM is superior to other mutations is still unclear.

Fitness landscape have been used as a tool to understand a complex process [17], and in this paper we attempt to use fitness landscapes to shed light on the improvement in performance of SSM versus standard mutation.

The remainder of the paper is organised as follows. In the next section, we give a review of related work on semantic used in GP and a brief review of fitness landscape in Evolutionary Computation (EC). Section 3 presents a way to measure semantics in GP, describes SSM and two ways to characterise fitness landscapes. The experimental settings are detailed in Section 4. The results of the experiments are presented and discussed in section 5, and finally Section 6 concludes the paper and highlights some potential future work.

2. RELATED WORK

This section briefly reviews the relevant literature for this study. Firstly, a summary of ways in which semantics have been employed in GP is provided, and this is followed by an overview on the use of fitness landscapes in EC.

2.1 Semantics in Genetic Programming

Semantics is a broad concept that has been studied in a number of different fields, including Natural Language [1], Psychology [5] and Computer Science [29]. While the objective of using semantics varies from field to field, in GP, semantic information has generally been used to provide additional guidance to the evolutionary search. The way semantics are exploited in GP depends on the problem domain

.

(Boolean or real-valued), individual representation (Grammar-, Tree- or Graph-based), and the search algorithm components (fitness measure, genetic operators,...). There have been three main approaches for representing, extracting, and using semantics to guide the evolutionary process of GP:

1. grammar-based [41, 4, 7]
2. formal methods [13, 15, 16, 21, 19]
3. GP s-tree representation [3, 28, 33, 35, 26]

In the first way, attribute grammars are the most popular form. They extend context-free grammars, providing context sensitivity through a finite set of attributes [23]. GP individuals expressed in the form of attribute grammar derivation trees can incorporate semantic information, which can be used to eliminate bad (i.e., less fit) individuals from the population [7] or to prevent the generation of semantically invalid individuals [41, 4]. The attributes used to present semantics are generally problem-dependent, and it is not always obvious how to determine the attributes for a given problem.

Johnson, in his recent work, has advocated formal methods as a means to incorporate semantic information into the GP process [13, 15, 16]. In Johnson's work, semantic information extracted by formal methods, such as abstract interpretation or model checking, is used to quantify the fitness of individuals on some problems for which traditional sample-point-based fitness measure are unavailable or misleading [13, 15, 16]. Subsequently, Katz and Peled [19] also used model checking to define fitness in a GP system for the mutual exclusion problem.

In other work, Keijzer [21] used interval analysis to check whether an individual is defined over the whole range of input values – if an individual is undefined anywhere, that individual can be assigned minimal fitness or simply eliminated from the population. This allowed Keijzer to avoid discontinuities arising from protected operators, improving the evolvability of the system.

The advantage of formal methods lies in their rigorous mathematical foundations, potentially helping GP to evolve computer programs. However they are high in complexity and difficult to implement, possibly explaining the limited number of related publications since the advocacy of Johnson [14]. Their main application to date has been in evolving control strategies.

Methods for extracting semantics from expression trees depend on the problem domain. The finite inputs of Boolean domains mean that semantics can be accurately estimated in a variety of ways. Beadle and Johnson [3] investigated the direct use of semantic information to guide GP crossover. They checked the semantic equivalence of the offspring with their parents by transforming them to Reduced Ordered Binary Decision Diagrams (ROBDDs). This information is used to determine which individuals are copied to the next generation. If the offspring are semantically equivalent to their parents, they are discarded and the crossover is restarted. The authors argued that this results in increased semantic diversity in the evolving population, and a consequent improvement in GP performance.

By contrast, in [28], semantic is extracted from a Boolean expression tree by enumerating all its possible inputs. The semantics of two aspects were considered: the semantics of subtrees and the semantics of context (the remainder of an individual after removing a subtree). Special attention was paid to fixed-semantic subtrees: subtrees where the semantics of the tree does not change when this subtree is replaced by another subtree. The authors showed that there may be many such fixed semantic subtrees when the tree size increases during the evolutionary process; thus it becomes increasingly difficult to change the semantics of trees with crossover and mutation, reducing the semantic diversity [28].

While, most of previous research on semantics in GP were focused on combinatorial and boolean problems [4, 3, 28, 19], research on real-valued domains [33, 35, 26] is much more recent. Krawiec and Lichocki [26] based the semantics of individuals on fitness cases, using it to guide crossover (*Approximating Geometric Crossover* - AGC). AGC turned out no better than standard crossover (SC) on real-valued problems, and only slightly better on Boolean.

Uy et al. [33] proposed Semantics Aware Crossover (SAC), another crossover operator promoting semantic diversity, based on checking semantic equivalence of subtrees. It showed limited improvement on some real-value problems; it was subsequently extended to Semantic Similarity based Crossover (SSC) [35], which turned out to perform better than both standard crossover and SAC [35]. The idea of SSC was then extended to mutation leading to a counterpart semantic mutation: Semantic Similarity based Mutation (SSM) [34]. The experimental results in [34] again confirmed the superior performance of SSM to standard mutation (SM).

2.2 Fitness Landscapes in Evolutionary Computation

A fitness landscape is a way of describing the search space of a problem in evolutionary algorithms. The concept of fitness landscape was first proposed in [42] to study the evolutionary process in biology. Since then, it has widely been used to model the problem difficulties in evolutionary algorithms (EAs) [32, 8]. A Fitness landscape uses metaphors from nature such as peaks, hills, valleys, ridges, basins, watersheds etc. to characterise the search space of a problem that EAs might encounter. A fitness landscape with many local peaks surrounded by deep valleys is called rugged. For the problems with this fitness landscape, it is more difficult to find solutions (the highest peaks), since the algorithms can be trapped in any local peak. General knowledge is the more rugged fitness landscape, the more difficult the problem is. If all genotypes have the same fitness values, on the other hand, a fitness landscape is said to be flat. For this fitness landscape, using algorithms can not exploit knowledge (e.g., fitness gradients) from the search space.

In practice, the visualisation of the whole search space of a problem is problematic. Therefore, a number of methods that attempt to describe the structure of fitness landscapes have been proposed [40, 32, 38]. In fact, before describing a fitness landscape, its primary components must be defined [17]. The first component is the representation, i.e., how we encode the problem on a genotype structure to represent all potential solutions of the problem. The second component is the operator that transforms a candidate solution from one point to another point in the search space. The third component is comprised of a function (the fitness function) which allows us to assign a measure of quality (fitness) to each candidate solution, a fitness space, and a partial order of solutions over the fitness space.

The structure of fitness landscapes influences the ability of an evolutionary algorithm to perform an efficient search. There are several characteristics associated with the landscape that define its structure. These characteristics include number, type, magnitude, and the sizes of the optima, and of their basins of attraction. Researchers have investigated the different aspects of the structure of fitness landscapes, such as landscape deceptiveness [9, 10], modality [2] and ruggedness [20, 40], to understand the nature of evolutionary search under different conditions.

Some authors suggest the use of an operator that minimises distance traveled in the search space when studying fitness landscape [12, 36, 37]. In this paper, we follow the work in [22, 17, 18, 39] to define the metric in terms of the operators. It is reasonable since the main objective of this research is to compare the characteristics of the fitness landscape with different operators.

3. METHODS

This section presents the methods used in the study. A way to measure semantics in GP is presented followed by a detailed description of Semantic Similarity based Mutation (SSM). Finally, we discuss two ways that are used to characterise fitness landscapes in the experiments.

3.1 Measuring Semantics

Although, an exact definition of semantics is non-trivial, in the field of GP, semantics of an individual program is often understood as the behavior of that program with respect to a set of input values. In this paper, we follow the previous work in [34] in using sampling semantics to measure semantics of any subtree. Formally, the *Sampling Semantics* (SS) of a (sub)tree is defined as follows:

Let F be a function expressed by a (sub)tree T on a domain D. Let P be a sequence of points sampled from domain D, $P = (p_1, p_2, ..., p_N)$. Then, the *Sampling Semantics* of T on P in domain D is the corresponding sequence $S = (s_1, s_2, ..., s_N)$ where $s_i = F(p_i), i = 1, 2, ..., N$.

The optimal choice of N and P depends on the problem; we follow the approach of [34] in setting the number of points for evaluating the semantics equal to the number of fitness cases (20 points – Section 4) and in choosing the sequence of points P uniformly randomly from the problem domain.

Based on SS, we define a *Sampling Semantic Distance* (SSD) between two subtrees. It differs from that in [34] in using the mean absolute difference in SS values, rather than the sum of absolute differences. Let $U = (u_1, u_2, ..., u_N)$ and $V = (v_1, v_2, ..., v_N)$ represent the SSs of two subtrees, S_1 and S_2; then the SSD between S_1 and S_2 is defined in equation 1:

$$\mathrm{SSD}(S_1, S_2) = \frac{\sum_{i=1}^{N} |u_i - v_i|}{N} \quad (1)$$

We follow [35] in defining a semantic relationship, *Semantic Similarity* (SSi), on the basis that the replacing of subtrees in mutation is most likely to be beneficial if they are not semantically identical, but also not too different. Two subtrees are semantically similar if their SSD lies within a positive interval. The formal definition of SSi between subtrees S_1 and S_2 is given in the following equation:

Algorithm 1: Semantic Similarity based Mutation

```
select a Parent P;
Count=0;
while Count<Max_Trial do
    choose a random mutation point Subtree_1 in P;
    generate at random a subtree Subtree_2;
    generate a number of random points (Q) on the
    problem domain;
    calculate the SSD between Subtree_1 and Subtree_2
    on Q
    if Subtree_1 is similar to Subtree_2 then
        replace Subtree_1 by Subtree_2;
        add the children to the new population;
        return true;
    else
        Count=Count+1;
Choose a random mutation point Subtree_1 in P;
generate at random a subtree Subtree_2;
replace Subtree_1 by Subtree_2;
return true;
```

$$SSi(St_1, St_2) = \begin{array}{l} \textbf{if } \alpha < \mathrm{SSD}(St_1, St_2) < \beta \\ \quad \textbf{then true} \\ \quad \textbf{else false} \end{array}$$

where α and β are two predefined constants, the *lower* and *upper* bounds for semantics sensitivity. In general, the best values for these semantic sensitivity bounds have been found to be problem dependent. In this work we set $\alpha = 10^{-3}$ and $\beta = 0.4$, which have been found to provide good performance in the case of SSM.

3.2 Semantic Similarity based Mutation

The mutation for improving semantic locality is inspired from the crossover for improving semantic locality, SSC. This mutation is called *Semantic Similarity-based Mutation* (SSM). SSM is implemented in a similar manner to SSC, but adapted to constrain the semantics in mutation rather than in crossover. In SSM, a parent is selected for mutation in the normal manner. A mutation point is randomly chosen in the parent and a new subtree is stochastically generated. Then the semantic equivalence is checked to determine if these two subtrees (replaced and replacing subtrees in the mutation operation) are semantically similar. If they are similar, we perform mutation by simply replacing the subtree at the mutation point with the new *semantically similar* subtree. If they are not semantically similar, SSM uses multiple trials to find a semantically similar pair, only reverting to random selection after passing a bound on the number of trials. Algorithm 1 outlines the operation of SSM in detail.

To characterise the fitness landscape using these two different mutation operators, we use two well known techniques. The first is the autocorrelation function, and the second is the information content.

3.3 Autocorrelation Function

Correlation analysis is a set of techniques for characterising the problem difficulty by measuring the correlation be-

tween fitness of neighbouring points [8]. There are three main techniques of correlation analysis. Of these the autocorrelation metric of fitness landscape has been used in biological evolution, computational evolution, and will be used in this paper.

Using an autocorrelation function to study fitness landscapes was first proposed in [40]. For a given fitness landscape with f as the fitness function, a starting point s_0 is randomly selected. Using a mutation operator to create a neighbouring point s_1 of s_0. Repeat this process N times to get a random walk of N steps $F = \{f(s_i)\}_{i=0}^{N}$. Then the autocorrelation function of this random walk is defined as follows:

$$\rho(h) = \frac{R(h)}{s_f^2} \tag{2}$$

where h is the distance between two points in the random walk. s_f^2 is variance of the sequence and calculated as follows:

$$s_f^2 = \frac{\sum_{i=0}^{N}(f(s_i) - m_F)^2}{N+1} \tag{3}$$

and $R(h)$ is the autocovariance function of sequence F. For each h, R(h) is estimated by:

$$R(h) = \frac{\sum_{i=0}^{N-h}(f(s_i) - m_F).(f(s_{i+h}) - m_F)}{N} \tag{4}$$

where $m_F = \frac{1}{N+1}\sum_{i=0}^{N}(f(s_i))$, is the mean of fitness sequence F. The autocorrelation function indicates the correlation between points that are separated by a distance h. A smooth landscape is highly correlated as the fitness difference between a point with its neighouring is small and the autocorrelation function is greater. Conversely, if a fitness landscape is rugged, the fitness difference is high, and the autocorrelation function is smaller.

3.4 Information Content

A method for characterising fitness landscapes based on the concept of information content was first proposed in [38]. Similar to the autocorrelation function, a random walk of $N - steps$, $F = \{f(s_i)\}_{i=0}^{N}$ is first obtained. A sequence driven from F, $S(\epsilon) = s_1, s_2, ..., s_N$, to represent this random walk for each ϵ, is generated, where

$$s_i = \Psi(i, \epsilon) \tag{5}$$

and

$$\Psi(i,\epsilon) = \begin{cases} -1 & \text{if } f_i - f_{i-1} < -\epsilon; \\ 0 & \text{if } |f_i - f_{i-1}| \leq \epsilon; \\ 1 & \text{if } f_i - f_{i-1} > \epsilon; \end{cases} \tag{6}$$

The parameter ϵ is a real-number from the range [0..L], where L is the maximal pair-wise difference in the sequence F. ϵ determines the accuracy of calculation of $S(\epsilon)$. If ϵ=0, the function $\Psi(i,\epsilon)$ = will be very sensitive, and insensitive when ϵ=L.

After that, four measures of entropy and amount of fitness change during the random walk are calculated as follows:

1. *Information Content* ($H(\epsilon)$): indicates the ruggedness of landscape
2. *Partial information content* ($M(\epsilon)$): indicates the modality of landscape.
3. *Information stability* ($\epsilon*$): indicates magnitude of optima in landscape.
4. *Density-basin information* ($h(\epsilon)$): characterises the structure of landscape around optima.

The information content characterises the ruggedness of the fitness landscape. It is defined as follows:

$$H(\epsilon) = \sum_{p \neq q} P_{[pq]} log_6 P_{[pq]} \tag{7}$$

where the probability $P_{[pq]}$ are the frequencies of 6 possible blocks qp, $p \neq q$, of elements from $S(\epsilon)$. They are defined as

$$P_{[pq]} = \frac{N_{[pq]}}{N} \tag{8}$$

where $N_{[pq]}$ is the number of occurrences of pq in $S(\epsilon)$.

The partial information content is defined as

$$M(\epsilon) = \frac{\mu}{N} \tag{9}$$

where N is the length of frequency $S(\epsilon)$. μ is calculated by calling a recursive function of three integer arguments and called by $\Phi_S(1,0,0)$, with $\Phi_S(i,j,k)$ is defined as follows:

$$\Phi_S(i,j,k) = \begin{cases} k & \text{if i>N} \\ \Phi_S(i+1,i,k+1) & \text{if } i = 0 \text{ and } s_i \neq 0; \\ \Phi_S(i+1,i,k+1) & \text{if } j > 0 \text{ and } s_i \neq 0 \\ & \text{and } s_i \neq s_j; \\ \Phi_S(i+1,j,k) & otherwise; \end{cases} \tag{10}$$

When $M(\epsilon)$=0 it is an indication that there is no slope in the random walk and when $M(\epsilon)$=1, it is an indication that there is a maximal of multi-modality in the random walk. Moreover, for a given partial information content, $M(\epsilon)$, the number of optimal (Optimal No. in Table 3) in the random walk can be calculated as $\lfloor \frac{N.M(\epsilon)}{2} \rfloor$.

The last measurement, density-basin information, $h(\epsilon)$, can be calculated as

$$h(\epsilon) = \sum_{p \subseteq \{-1,0,1\}} P_{[pp]} log_3 P_{[pp]} \tag{11}$$

where pp is one of three sub-blocks, 00, -1-1, 11. A high value of $h(\epsilon)$ presents the high number of peaks in a small area of the landscape. The low value means that the optima is isolated.

In general, higher values of information content, partial information content, number of optimal in the landscape, and density-basin information define a more rugged landscape which is more difficult to search. Smaller values define a smoother landscape which is easier for a search method.

4. EXPERIMENTAL SETTINGS

To examine the fitness landscape of GP using semantic based mutation, we use a class of problem where difficulty can be controlled and increased. The aim is to investigate if the difficulty of these problems is (partly) caused by the

nature of the fitness landscape. The problem class is the binomial-3 problem. The binomial-3 problem consists of approximating the function $f(x) = (x+1)^3$. Using the terminal set of {x, Ephemeral Random Constant (ERC)}, it has been shown [6, 11] that the difficulty of this problem is increased with changes to the ERCs. In other words, if ERC ≥ 1, the difficulty of binomial-3 is increased by increasing ERC. In this experiment we use three ranges of ERC that have been used in [6, 11], they are [-1, 1], [-10, 10] and [-100, 100]. Binomial-3 with these ranges of ERC will be referred to as Bin1, Bin10 and Bin100, respectively.

The GP parameter settings are similar to ones in [34, 6]. Fitness cases include 20 points of step 0.05 in interval [0,1). Raw fitness is the sum of absolute error on all fitness cases. The function set comprises +, -, *, / (protected version), sin, cos, exp, log (protected version) and the terminal set consists of X and ERC. The lower semantic sensitivity used for SSM is 10^{-3} and the upper semantic sensitivity is set to 0.4. The *Max_Trial* (see Algorithm 1) of SSM is set at 20. These values of the parameters for SSM have been shown as good values for the performance of SSM.

We divided our experiments into two sets. The first set was to investigate the performance of Semantic Similarity based Mutation (SSM) and standard mutation (SM) in solving these problems. The question we aim to address is whether semantic based mutation help to improve the performance of GP in solving these problems, especially when the difficulty of the problems were increased. For this experiment, we used a population size of 500 and run 50 generations. A run is successful if any individuals hits (<0.01) on all fitness cases. The mutation rate was set at 1.0 and crossover rate was set at 0, and 100 runs were performed.

The second set of experiments was to examine the fitness landscape of the above problems with SSM and SM. To investigate the fitness landscape, random walks of 10000 steps are created using standard mutation (SM) and Semantic Similarity-based Mutation (SSM) for each problem. All fitness values of individuals encountered during the random walk were recorded. For each problem, 200 random walks were conducted, making a total of 2,000,000 fitness evaluations for each experiment.

5. RESULTS AND DISCUSSION

This section presents the results of the experiments and draws some discussion based on that.

5.1 The Comparison of Performance

To compare the performance of these mutation, we use two classical metrics, namely mean best fitness and the number of successful runs. The results of the number of successful runs out of 100 runs of these mutations (No. Suc row) and the best fitness found, averaged over all 100 runs of each GP system (Mean Best row) are presented in Table 1.

Table 1 shows some interesting results. Firstly, it can be observed that the difficulty of these problem is increased from Bin1 to Bin100. It is reflected by the results of both number of successful runs and mean best fitness. The number of successful runs was decreased from Bin1 to Bin100 while the mean best fitness was increased (It means that the rate of minimising of the best fitness was decreased from Bin1 to Bin100). The results are consistent with the results in [6, 11].

Secondly, the table shows that semantic based mutation, SSM, helps to improve the performance of GP in solving these problems, even when the problems become harder (Bin10 and Bin100). It can be seen that while SM can not find any solution, SSM can still find some solutons and the mean best fitness of SSM was also smaller than the one found by SM. We also tested the statistical significance of the results of mean best fitness in Table 1 using a Wilcoxon signed-rank test with a confidence level of 95% and the results of statistical test show that all enhancement of SSM over SM are significant.

Table 1: The Comparison of the Performance

Metrics	Mutations	Bin1	Bin10	Bin100
No. Suc	SM	0	0	0
	SSM	12	3	1
Mean Best	SM	0.64	1.40	4.28
	SSM	0.28	0.88	2.40

Table 2: Autocorrelation Analysis (greater one is better).

Distance (h)	Mutations	Bin1	Bin10	Bin100
1	SM	0.664	0.662	0.668
	SSM	0.739	0.740	0.732
2	SM	0.562	0.566	0.580
	SSM	0.673	0.676	0.673
4	SM	0.431	0.439	0.442
	SSM	0.581	0.588	0.586
8	SM	0.287	0.299	0.324
	SSM	0.464	0.473	0.472
16	SM	0.161	0.170	0.192
	SSM	0.348	0.340	0.340
32	SM	0.073	0.077	0.091
	SSM	0.228	0.208	0.206

5.2 The Comparison of Fitness Landscape

To characterise the fitness landscape of these problems, the autocorrelation and information content of SM and SSM were measured and the results are shown in Table 2 and Table 3.

Table 2 shows that improving semantic locality of mutation helps to smooth out the fitness landscape of a problem. From the table it is clear that the value of the autocorrelation function of SSM is always greater than that of SM. This means that the fitness value of the individuals in the population of SSM has stronger correlation than SM and hence the fitness landscape of the population that are constructed by SSM is smoother than the one of SM. We also statistically tested the difference between autocorrelation values of SM and SSM using the Wilcoxon rank test and the results show that all the difference are significant with a confidence level of 95%. In comparison between each function groups, the table shows very little difference between Bin1 to Bin100. In fact it can be seen that autocorrelation of Bin100 is often slightly greater than Bin1, however, the statistical test results show that this difference is not significant.

The results presented in Table 3 are consistent with Table 2 confirming the ability to smooth out fitness landscape

Table 3: Information Content Analysis of Binomial-3 (smaller one is better).

Epsilon(ϵ)	Problems	Mutations	H(ϵ)	h(ϵ)	M(ϵ)	Optima No.
1	Bin1	SM	0.771	0.681	0.423	2116
		SSM	0.769	0.680	0.392	1963
	Bin10	SM	0.761	0.696	0.410	2047
		SSM	0.755	0.695	0.380	1902
	Bin100	SM	0.742	0.689	0.380	1900
		SSM	0.731	0.687	0.360	1801
2	Bin1	SM	0.767	0.665	0.369	1848
		SSM	0.749	0.656	0.331	1656
	Bin10	SM	0.751	0.669	0.363	1815
		SSM	0.733	0.657	0.327	1639
	Bin100	SM	0.724	0.658	0.343	1715
		SSM	0.706	0.645	0.319	1595
4	Bin1	SM	0.715	0.609	0.300	1502
		SSM	0.669	0.574	0.256	1280
	Bin10	SM	0.703	0.614	0.303	1516
		SSM	0.663	0.658	0.263	1314
	Bin100	SM	0.675	0.604	0.295	1473
		SSM	0.642	0.575	0.265	1328
8	Bin1	SM	0.607	0.509	0.224	1121
		SSM	0.533	0.453	0.181	906
	Bin10	SM	0.610	0.528	0.238	1190
		SSM	0.546	0.480	0.197	986
	Bin100	SM	0.592	0.530	0.240	1214
		SSM	0.540	0.490	0.211	1056
16	Bin1	SM	0.461	0.394	0.154	774
		SSM	0.383	0.342	0.124	620
	Bin10	SM	0.490	0.436	0.179	897
		SSM	0.418	0.386	0.145	727
	Bin100	SM	0.492	0.458	0.196	982
		SSM	0.430	0.416	0.168	842
32	Bin1	SM	0.326	0.305	0.111	557
		SSM	0.280	0.271	0.093	468
	Bin10	SM	0.374	0.361	0.139	697
		SSM	0.324	0.323	0.116	583
	Bin100	SM	0.393	0.401	0.165	826
		SSM	0.351	0.369	0.145	728

by improving the semantic locality of mutation. The values that characterise information content of SSM are always smaller than those for SM, meaning that SSM helps not only to reduce the ruggedness of the fitness landscape but also to decrease the number of local optima in landscape, and decrease the magnitude of this local optima. We also statistically tested the difference between SM and SSM in Table 3 using the Wilcoxon rank test with a confidence level of 95%. The results of this test show most of the differences are significant with some exceptions when $\epsilon = 1$.

Comparing between functions, the results in this table are slightly different from the results in Table 2. Table 3 shows that Bin1 is more rugged than Bin10 and Bin100 with the small values of ϵ (1, 2, 4) but Bin10 and Bin100 are more rugged when ϵ is increased (16, 32). Therefore, it is difficult to conclude that the difficulty of Bin10 and Bin100 are caused by the changing fitness landscape. We believe that the increasing difficulty of Bin10 and Bin100 versus Bin1 is because of the conflict between context and content as they have been shown in [6].

6. CONCLUSIONS AND FUTURE WORK

In this paper we study the fitness landscape of both standard mutation and a recently proposed semantic based mutation, Semantic Similarity based Mutation (SSM). The landscape was characterised using two common methods, the autocorrelation function and information content. The experiments were conducted on a family of binomial-3 problems with increasing difficulty. The results show that improving semantic locality helps to significantly smooth the fitness landscape of these problems, which should make search with the semantic based mutation much easier. These results help to explain the improvement of the SSM operator versus standard mutation.

Future work includes investigating fitness landscapes of other semantic based operators such as Semantic Similarity based Crossover using the techniques in this paper to further understand the relationship between semantic locality and fitness landscape.

Acknowledgment

This paper was funded under a Postgraduate Scholarship from the Irish Research Council for Science Engineering and Technology (IRCSET).

7. REFERENCES

[1] C. Alan. *Meaning and Language: An introduction to Semantics and Pragmatics*. Oxford Textbooks in Linguistics, Cambridge, UK, 2004.

[2] L. Altenberg. The evolution of evolvability in genetic programming. In K. E. Kinnear, Jr., editor, *Advances in Genetic Programming*, chapter 3, pages 47–74. MIT Press, 1994.

[3] L. Beadle and C. Johnson. Semantically driven crossover in genetic programming. In *Proceedings of the IEEE World Congress on Computational Intelligence*, pages 111–116. IEEE Press, 2008.

[4] R. Cleary and M. O'Neill. An attribute grammar decoder for the 01 multi-constrained knapsack problem. In *Proceedings of the Evolutionary Computation in Combinatorial Optimization*, pages 34–45. Springer Verlag, April 2005.

[5] A. M. Collins and M. R. Quillian. Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behavior,*, 8:240–247, 1969.

[6] J. M. Daida, R. R. Bertram, S. A. Stanhope, J. C. Khoo, S. A. Chaudhary, O. A. Chaudhri, and J. A. Polito II. What makes a problem GP-hard? analysis of a tunably difficult problem in genetic programming. *Genetic Programming and Evolvable Machines*, 2(2):165–191, June 2001.

[7] M. de la Cruz Echeanda, A. O. de la Puente, and M. Alfonseca. Attribute grammar evolution. In *Proceedings of the IWINAC 2005*, pages 182–191. Springer Verlag Berlin Heidelberg, 2005.

[8] K. Deb. Fitness landscape. In *Handbook of Evolutionary Computation*. Oxford University Press, 1997.

[9] D. E. Goldberg. Simple genetic algorithms and the minimal, deceptive problem. In *Genetic Algorithms and Simulated Annealing*, pages 74–88. Morgan Kaufmann, 1987.

[10] D. E. Goldberg. Genetic algorithms and Walsh functions: Part II, deception and its analysis. *Complex Systems*, 3(2):153–171, 1989.

[11] S. Gustafson, A. Ekart, E. Burke, and G. Kendall. Problem difficulty and code growth in genetic programming. *Genetic Programming and Evolvable Machines*, 5(3):271–290, Sept. 2004.

[12] N. X. Hoai. *A Flexible Presentation for Genetic Programming: Lessons from Natural Language Processing*. PhD thesis, University of New South Wales, 2004.

[13] C. Johnson. Deriving genetic programming fitness properties by static analysis. In *Proceedings of the 4th European Conference on Genetic Programming (EuroGP 2002)*, pages 299–308. Springer, 2002.

[14] C. Johnson. Genetic programming with guaranteed constraints. In *Recent Advances in Soft Computing*, pages 134–140. The Nottingham Trent University, 2002.

[15] C. Johnson. What can automatic programming learn from theoretical computer science. In *Proceedings of the UK Workshop on Computational Intelligence*. University of Birmingham, 2002.

[16] C. Johnson. Genetic programming with fitness based on model checking. In *Proceedings of the 10th European Conference on Genetic Programming (EuroGP 2002)*, pages 114–124. Springer, 2007.

[17] T. Jones. *Evolutionary Algorithms, FitnessLandscapes, and Search*. PhD thesis, University of New Mexico, 1995.

[18] T. Jones. One operator, one landscape, June 02 1995.

[19] G. Katz and D. Peled. Model checking-based genetic programming with an application to mutual exclusion. *Tools and Algorithms for the Construction and Analysis of Systems*, 4963:141–156, 2008.

[20] S. A. Kauffman and S. Levin. Towards a general theory of adaptive walks on rugged landscapes. *J. Theor. Biol.*, 128:11–45, 1987.

[21] M. Keijzer. Improving symbolic regression with interval arithmetic and linear scaling. In *Proceedings of EuroGP'2003*, pages 70–82. Springer-Verlag, April

2003.

[22] K. E. Kinnear. Fitness landscapes and difficulty in genetic programming. In *Proceedings of the 1994 IEEE World Conference on Computational Intelligence*, volume 1, pages 142–147, Orlando, Florida, USA, 27-29 June 1994. IEEE Press.

[23] D. Knuth. Semantics of context-free languages. *Mathematical Systems Theory*, 2:95, 1968.

[24] J. Koza. Human-competitive results produced by genetic programming. *Genetic Programming and Evolvable Machines*, 11(3-4):251–284–, 2010.

[25] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, Cambridge, Massachusetts, 1992.

[26] K. Krawiec and P. Lichocki. Approximating geometric crossover in semantic space. In F. Rothlauf, editor, *Genetic and Evolutionary Computation Conference, GECCO 2009, Proceedings, Montreal, Québec, Canada, July 8-12, 2009*, pages 987–994. ACM, 2009.

[27] R. McKay, X. Nguyen, P. Whigham, Y. Shan, and M. O'Neill. Grammar-based genetic programming - a survey. *Genetic Programming and Evolvable Machines*, 11(3-4):365–396, 2010.

[28] N. McPhee, B. Ohs, and T. Hutchison. Semantic building blocks in genetic programming. In *Proceedings of 11th European Conference on Genetic Programming*, pages 134–145. Springer, 2008.

[29] H. R. Nielson and F. Nielson. *Semantics with Applications: An Appetizer*. Springer, Springer-Verlag, London, UK, 2007.

[30] M. O'Neill, L. Vanneschi, S. Gustafson, and W. Banzhaf. Open issues in genetic proramming. *Genetic Programming and Evolvable Machines*, 11(3-4):339–363, 2010.

[31] R. Poli, W. B. Langdon, and N. F. McPhee. *A Field Guide to Genetic Programming*. Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk`, 2008. (With contributions by J. R. Koza).

[32] C. Reeves and J. Rowe. *Genetic Algorithms: Principles and Perspectives*. Kluwer Academic Pulisher, 2003.

[33] N. Q. Uy, N. X. Hoai, and M. O'Neill. Semantic aware crossover for genetic programming: the case for real-valued function regression. In *Proceedings of EuroGP 2009*, pages 292–302. Springer, April 2009.

[34] N. Q. Uy, N. X. Hoai, and M. O'Neill. Semantics based mutation in genetic programming: The case for real-valued symbolic regression. In R. Matousek and L. Nolle, editors, *15th International Conference on Soft Computing, Mendel'09*, pages 73–91, Brno, Czech Republic, June 24-26 2009.

[35] N. Q. Uy, M. O'Neill, N. X. Hoai, B. McKay, and E. G. Lopez. Semantic similarity based crossover in GP: The case for real-valued function regression. In P. Collet, editor, *Evolution Artificielle, 9th International Conference*, Lecture Notes in Computer Science, pages 13–24, October 2009.

[36] L. Vanneschi, M. Tomassini, P. Collard, and M. Clergue. Fitness distance correlation in genetic programming: A constructive counterexample. In R. Sarker, R. Reynolds, H. Abbass, K. C. Tan, B. McKay, D. Essam, and T. Gedeon, editors, *Proceedings of the 2003 Congress on Evolutionary Computation CEC2003*, pages 289–296, Canberra, 2003. IEEE Press.

[37] L. Vanneschi, M. Tomassini, P. Collard, and M. Clergue. Fitness distance correlation in structural mutation genetic programming. In C. Ryan, T. Soule, M. Keijzer, E. Tsang, R. Poli, and E. Costa, editors, *Genetic Programming, Proceedings of EuroGP'2003*, volume 2610 of *LNCS*, pages 455–464, Essex, 14-16 Apr. 2003. Springer-Verlag.

[38] V. K. Vassilev, T. C. Fogarty, and J. F. Miller. Information characteristics and the structure of landscapes. *Evolutionary Computation*, 8(1):31–60, Spring 2000.

[39] V. K. Vassilev, J. F. Miller, and T. C. Fogarty. Digital circuit evolution and fitness landscapes. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, editors, *Proceedings of the Congress on Evolutionary Computation*, volume 2, pages 1299–1306, Mayflower Hotel, Washington D.C., USA, 6-9 July 1999. IEEE Press.

[40] E. D. Weinberger. Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics*, 63:325–336, 1990.

[41] M. L. Wong and K. S. Leung. An induction system that learns programs in different programming languages using genetic programming and logic grammars. In *Proceedings of the 7th IEEE International Conference on Tools with Artificial Intelligence*, 1995.

[42] S. Wright. The roles of mutation, inbreeding, crossbreeding, and selection in evolution. In *Proceedings of the Sixth Congress on Genetics*, volume 1, page 365, 1932.