Towards an Understanding of Locality in Genetic Programming

Edgar Galván-López
Natural Computing Research
& Applications Group
University College Dublin, Ireland
edgar.galvan@ucd.ie

James McDermott
Natural Computing Research
& Applications Group
University College Dublin, Ireland
jamesmichaelmcdermott@gmail.com

Michael O'Neill
Natural Computing Research
& Applications Group
University College Dublin, Ireland
m.oneill@ucd.ie

Anthony Brabazon
Natural Computing Research
& Applications Group
University College Dublin, Ireland
anthony.brabazon@ucd.ie

ABSTRACT

Locality – how well neighbouring genotypes correspond to neighbouring phenotypes – has been defined as a key element affecting how Evolutionary Computation systems explore and exploit the search space. Locality has been studied empirically using the typical Genetic Algorithm (GA) representation (i.e., bitstrings), and it has been argued that locality plays an important role in EC performance. To our knowledge, there are few explicit studies of locality using the typical Genetic Programming (GP) representation (i.e., tree-like structures). The aim of this paper is to address this important research gap. We extend the genotype-phenotype definition of locality to GP by studying the relationship between genotypes and fitness. We consider a mutation-based GP system applied to two problems which are highly difficult to solve by GP (a multimodal deceptive landscape and a highly neutral landscape). To analyse in detail the locality in these instances, we adopt three popular mutation operators. We analyse the operators' genotypic step sizes in terms of three distance measures taken from the specialised literature and in terms of corresponding fitness values. We also analyse the frequencies of different sizes of fitness change.

Categories and Subject Descriptors

I.2 [**ARTIFICIAL INTELLIGENCE**]: Automatic Programming; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Algorithms

Keywords

Locality, Problem Hardness, Fitness Landscape, Difficulty, Genetic Programming, Neutrality

1. INTRODUCTION

The concept of a fitness landscape [23] has dominated the way geneticists think about biological evolution and has been adopted within the Evolutionary Computation (EC) community. In simple terms, a fitness landscape can be seen as a plot where each point on the horizontal axis represents all the genes in an individual corresponding to that point. The fitness of that individual is plotted as the height against the vertical axis. Thus, a fitness landscape is a representation of a search space which may contain peaks, valleys, hills and plateaus.

How an algorithm explores and exploits such a landscape is a key element during evolutionary search. Rothlauf [17, 19] has described and analysed the importance of locality in performing an effective evolutionary search of landscapes. Locality refers to how well neighbouring genotypes correspond to neighbouring phenotypes. This research distinguished two forms of locality, low and high. A representation has high locality if all neighbouring genotypes correspond to neighbouring phenotypes. On the other hand, a representation has low locality if many neighbouring genotypes do not correspond to neighbouring phenotypes. It is demonstrated that a representation of high locality is necessary for efficient evolutionary search. In Section 2 we further explain the concept of locality.

In his original studies, Rothlauf used bitstrings to conduct his experiments [17] (and more recently he further explored the idea of locality using grammatical evolution at the chromosome level [19]). To our knowledge, no studies on locality exist using the typical Genetic Programming (GP) [11, 16] representation (i.e., tree-like structures). For this purpose we will extend the definition of locality to GP, and due to the lack of distinction between genotype and phenotype, we will study the locality of the genotype-fitness mapping. The goal of this paper then is to shed some light on the type of locality present in GP. We use three different mutation operators, three different genotypic distance measures, and two problems with significantly different landscape features: the artificial ant problem (a multimodal deceptive landscape) [13] and the even-3-parity problem (a highly neutral landscape) [3].

The motivation for studying locality is as an indicator of problem difficulty. The principle of strong causality states that for successful search, a small change in genotype should result in a small change in fitness [1].

This paper is organised as follows. In the next section, locality in EC is summarised. In Section 3, we describe three well-defined distance metrics for tree-like structures. In Section 4, we describe how we study the locality of the genotype-fitness mapping in GP. In Section 5, we present and discuss our findings. Finally, in Section 6 we draw some conclusions.

2. LOCALITY

Understanding of how well neighbouring genotypes correspond to neighbouring phenotypes is a key element in understanding evolutionary search [17, 19]. In the abstract sense, a mapping has *locality* if neighbourhood is preserved under that mapping[1]. In EC this generally refers to the mapping from genotype to phenotype. This is worthy of study because if neighbourhood is not preserved, then the algorithm's attempts to exploit the information provided by an individual's fitness will be misled when the individual's neighbours turn out to be very different.

Rothlauf [17] is perhaps the authoritative work on the topic of locality in EC. It and other work on locality [19, 18] has shed new light on several problems. The definition of locality used in [17] assumes that a distance measure exists on both genotype and phenotype spaces, that for each there is a minimum distance, and that neighbourhood can be defined in terms of minimum distance. In standard GP, there are no phenotypes distinct from genotypes. It is common therefore to study the locality of the mapping from genotype to fitness [12], and we take this approach here.

It can be stated that there are two types of locality: low and high locality. A representation is said to have the property of high locality if all neighbouring genotypes correspond to neighbouring phenotypes. On the other hand, a representation has low locality if some neighbouring genotypes do not correspond to neighbouring phenotypes. Rothlauf claims that a representation that has high locality will be more efficient at evolutionary search. If a representation has high locality then any search operator has the same effects in both the genotype and phenotype space. It is clear then that the difficulty of the problem remains unchanged compared to an encoding in which no genotype-phenotype map is required.

This, however, changes when a representation has low locality. To explain how low locality affects evolution, Rothlauf considered three different categories[2]. These are:

- *easy*, in which fitness increases as the global optimum approaches,
- *difficult*, for which there is no correlation between fitness and distance and,
- *misleading*, in which fitness tends to increase with the distance from the global optimum.

If a given problem lies in the first category (i.e., easy), a low-locality representation will change this situation by making it more difficult and now, the problem will lie in the second category. This is due to low locality randomising the search. This can be explained by the fact that representations with low locality lead to uncorrelated fitness landscapes, so it is difficult for heuristics to extract information.

If a problem lies in the second category, a low-locality representation does not change the difficulty of the problem. There are representations that can convert a problem from difficult (class two) to easy (class one). However, such representations are rare.

Finally, if the problem lies in the third category, a representation with low locality will transform it so that the problem will lie in the second category. That is, the problem is less difficult because the search has become more random. As can be seen, this is a mirror image of a problem lying in the first category and using a representation that has low locality.

In his original studies, Rothlauf mentioned the existence of high and low locality. Contrary to other measures of complexity (e.g., fitness distance correlation [9] where Jones set thresholds to categorise problems' difficulty), Rothlauf focused his attention on correspondence of genotypic and phenotypic neighbourhoods. In this work, we are going to adopt the notion of high locality corresponding to a single unit of fitness change (i.e., fitness distance = 1[3]). When distance > 1, low locality is present (in Section 5 we further discuss this).

In this work, we are interested in seeing what kind of locality is present in GP by using three different mutation operators. For this purpose, it is necessary to define a genotypic distance measure.

We will present three distance measures in the following section and highlight the results they give in Section 5.

3. TREE DISTANCE MEASURES FOR GP

Several tree distance measures have been used for Genetic Programming (e.g., [20, 15, 21, 22]). No single distance measure can be regarded as universally most appropriate. Each measure emphasises different aspects of similarity and leads to different results. In this section, we therefore review and give motivation for three contrasting measures:

Edit Distance is integer-valued and reflective of a very intuitive notion of the distance between trees, based on the number of individual edits required to transform one into the other.

Tree Alignment Distance is a good general-purpose measure which reflects the fact that the roots of syn-

[1]The term *locality* has also been used in an unrelated context, to refer to the quasi-geographical distribution of an EC population[5].

[2]These categories were taken from the work presented in [9].

[3]Notice that in this work we are using problems of discrete values and so, it is reasonable to adopt this notion of high neutrality. We also make a distinction with neutral mutation, where the fitness distance = 0.

tactic trees tend to be more important than their lower levels.

Normalised Compression Distance using a preorder serialisation of trees is entirely unbiased and uninformed by any prior ideas of similarity of trees, which may count both as an advantage and a disadvantage.

3.1 Edit Distance

O'Reilly [15] proposed an approach, called edit distance, with the main goal of having a metric that specifies the degree of dissimilarity between two individuals in the form of tree-like structures. The idea of edit distance is to calculate the minimum cost (number of moves) that is required to transform a given tree to a target tree step by step. For this purpose, the author defined the use of three types of edits: (a) Substitution: changing a node into another, (b) Insertion: adding a node within the tree and (c) Deletion: removing a node from the tree. This distance is notable because it is closely aligned with a mutation operator defined in the same paper. This property is desirable because of the principle stated by [9] in the context of fitness-distance correlation that each operator induces its own landscape.

3.2 Tree Alignment Distance

The tree distance proposed by Vanneschi and colleagues [21, 22] is based on that proposed by [8]. Formally, the distance between trees T_1 and T_2 with roots R_1 and R_2, respectively, is defined as follows:

$$dist(T_1, T_2, k) = d(R_1, R_2) + k \sum_{i=1}^{m} dist(child_i(R_1), child_i(R_2), \frac{k}{2})$$

where: $d(R_1, R_2) = (|c(R_1) - c(R_2)|)^z$ and; $child_i(Y)$ is the i^{th} of the m possible children of a node Y, if $i < m$, or the empty tree otherwise. Note that c evaluated on the root of an empty tree is 0 by convention. The parameter k is used to give different weights to nodes belonging to different levels in the tree and $z \in \mathbb{N}$ is a parameter of the distance. The depth-weighting is well-motivated, in that GP trees' roots tend to be more important than their lowest levels. Code for this calculation is available in [6].

This distance is notable because, for $k = 1$ (i.e. without depth-weighting) and for a particular function/terminal set, the distance is *coherent with* a specially-constructed pair of mutation operators. This means that the distance between a pair of trees and the number of mutations required to transform one into the other are linearly related. This idea is closely related to the work of [15]. However, operator-distance coherence can not hold for typical operators, such as subtree crossover and subtree mutation, which can transform any individual into any other.

3.3 Normalised Compression Distance

The so-called "universal similarity metric" is a theoretical measure of similarity between any two data structures (for example strings), defined in terms of *Kolmogorov complexity* [14]. This is defined as the length of the shortest program which creates the given string. Informally, two strings are very similar if the Kolmogorov complexity of their concatenation is close to the complexity of just one of them. This idea was made practical by [2]: they approximated the (uncomputable) Kolmogorov complexity of a string by the length of its compressed version, as calculated by off-the-shelf compression software. The "normalised compression distance" or NCD is defined as follows:

$$d(x, y) = \frac{C(xy) - \min(C(x), C(y))}{\max(C(x), C(y))}$$

where x and y are two strings, xy is their concatenation, and the function C gives the length of the compressed version of its argument.

The NCD has been used as a distance measure for trees in fields other than EC [2] and has been used for linear structures within EC [7], and GP has been used to approximate Kolmogorov complexity [4]. However, to the authors' knowledge the NCD has not yet been used to measure distance between GP trees. It can be applied to trees simply by encoding them as strings in preorder format. For fixed node arities, this encoding is injective, i.e. distinct trees will give distinct preorder strings. It has the advantage that a single mutation in a large tree will have a smaller effect on distance than a single mutation in a small tree. On the other hand, mutations near the root have the same weight as deeper ones. It shares this disadvantage with, for example, tree edit distance. Such distances are general-purpose, in that they are intended for general trees, not only the syntactic trees as used in standard GP.

Table 1: Parameters used to create our sampling.

Selection	Tournament (size 7)
Initial Population	Ramped half and half (depth 1 to 6)
Population size	500
Generations	50
Runs	50
Operators	Crossover
	Subtree Mutation
Crossover rate	70%
Mutation rate	30%

4. EXPERIMENTAL SETUP

We are interested in determining how locality affects GP search, and as such we need to consider whether neighbouring genotypes have similar fitness values. In this study we define genotypic neighbourhood via mutation operators: two individuals are neighbours if one is transformed into the other by a single mutation event. We therefore focus on the sizes of the fitness changes induced by the three mutation operators. At this point some questions arise: What kind of relationship should we expect between the genotypic step size of an operator and the corresponding change in fitness? What will be the proportions of individuals of higher, lower, and unchanged fitness after being affected by operators? To answer these questions we will focus our attention on how different types of mutation affects individuals, in terms of both genotypic structure and fitness.

For our analysis, we have used two problems. The first, the Artificial Ant Problem [11, pp. 147–155], consists of finding a program that can successfully navigate an artificial ant along a path of 89 pellets of food on a 32 x 32 toroidal grid. When the ant encounters a food pellet, its (raw) fitness increases by one, to a maximum of 89. The terminal set used for this problem is $T = \{Move, Right, Left\}$ and the function set is $F = \{IfFoodAhead, Prog2, Prog3\}$ (see [11] for a full description of them). This problem has been shown to be difficult for GP for its characteristics (e.g., multimodal-deceptive features)[13, Chapter 9]. These features are believed to be common in many real-world applications. The problem is in itself challenging for many reasons. The ant must eat all the food pellets (normally in 600 steps) along a track that has single, double and triple gaps along it. Moreover, the food trail is twisted.

The second problem is the Boolean even-3-parity where the goal is to evolve a function that returns true if an even number of the inputs evaluate to true, and false otherwise. This type of function is difficult for GP if no bias favorable to their induction is added in any part of the algorithm (e.g. the function set). The terminal set used for this problem is the number of inputs (three) and the function set is defined as $F = \{NOT, OR, AND\}$. The maximum fitness for this problem is 8 (2^3).

For our studies we have considered the use of three different mutation operators: (a) subtree mutation replaces a randomly selected subtree with another randomly created subtree [11]; (b) one-point mutation replaces a node (leaf or internal) in the individual subject to a probability (i.e., more than one can be changed or none); and (c) structural mutation, which is composed of inflate mutation and deflate mutation. The former consists of inserting a terminal node beneath a function whose arity a is lower than the maximum arity defined in the function set and replacing the function by another of arity $a + 1$; the latter consists of deleting a terminal beneath a function whose arity is at least 1 and replacing that function by another of arity $a - 1$ [22]

To have sufficient statistical data, we created 1,250,000 individuals for each of the three mutation operators described previously (in total 3,750,000 individuals). These samplings were created using traditional GP runs which used all three mutation operators, to avoid bias (see Table 1).

For each data point in the sample data, we created an offspring via mutation. In the following section we present and describe the results on locality using these mutations on the artificial ant problem and on the even-3-parity problem.

5. RESULTS AND DISCUSSION

Let us start by analysing the occurrences of individuals which are fitter, less fit and equally-fit after applying the three mutations (i.e., subtree, structural and one-point) explained in Section 4. For the artificial ant problem, it has been shown in [13, Chapter 9], that the number of individuals with very low fitness (e.g., fitness < 15) regardless of their length is much larger compared with individuals of very high fitness (e.g., fitness > 80). This explains why the highest peak in occurrences is within this region (fitness < 15, see top of Figure 1). There are also peaks in the range $[50 - 70]$ and at $79 - 80$ and 85. These peaks occur in the original distribution and thus tell us about the distribution of fitness values in the search space (and the method of sampling), rather than about the behaviour of the operators.

There is, however, one element that is quite interesting in the range (fitness < 15): most of the mutations that take place have a neutral effect [10] and, in fact, there is a good number of such mutations within the range of $[50-70]$ also[4].

This situation is similar on the even-3-parity problem (see bottom of Figure 1), but on the range of highly fit individuals (fitness > 5), where a great number of mutations are fitness-neutral, in particular for subtree and one-point mutation. The lack of occurrences in the region of $[0-2]$ happens because the sampling process creates very few individuals whose fitness lies in this range.

The locality is high if the corresponding phenotypes are similar to each other. Figure 2 shows the frequency of each possible fitness distance between individuals, for the two problems. Let us first focus our attention on the even-3-parity problem (8 fitness cases and 9 possible fitness values, including 0). As can be seen from Figure 2 (right) there is a good number of neutral mutations (i.e., fitness distance $= 0$). Regarding to the locality of the operators, we can see that structural mutation presents the highest locality among the three operators used in this work. Subtree and one-point mutation both present low locality (i.e., fitness distance > 1). This last finding is quite interesting due to the data shows that subtree mutation has a "higher" locality compared to one-point mutation. This in itself is revealing because one could imagine the opposite given the nature of the operators. That is, subtree mutation could change an entire subtree (and in fact the whole tree), whereas one-point mutation changes one node by another. This trend continues throughout the rest of the fitness distance (i.e., fitness distance ≥ 2).

For the artificial ant problem and for clarity purposes, we have plotted the first 9 fitness values (see left of Figure 2). Again, as in the previous problem, we can see that a high number of changes translates into neutral mutations (fitness distance $= 0$). Focusing our attention on the locality of the operators for this problem, we observe a similar story compared to the even-3-parity problem. Structural mutation presents the highest locality among the three operators. The remaining two operators, subtree and one-point mutation, show a very similar locality.

So far we have seen the overall locality present on three different mutation operators on two problems: the artificial ant problem and the even-3-parity problem. However, we can not say how the step size (both in terms of distance and fitness difference between parent and offspring) occurred. We will now analyse separately the cases where mutation is beneficial, detrimental, and fitness-neutral. In each case we will examine the genotypic step-size of the mutation which gave rise to the given effect on fitness. For clarity and due to space limitations, we will further examine the locality properties of these mutation operations on the even-3-parity problem only.

Let us focus our attention on the case where the resulting offspring has higher fitness than the parent. As can be seen in Figure 3, for high values of original fitness, only relatively small genotypic distances between parent and offspring result in increased fitness. This is true regardless of the mutation operator used (i.e., subtree, structural, onepoint). The opposite can be observed for low original fitness values (i.e., fitness < 4): fitness can be improved by mutations of relatively large genotypic distance. This is very interesting because it suggests that the closer the individual is to the global optimum, smaller changes are required at the genotypic level. In other words, a small genotypic step-size is needed to achieve this. This is consistent throughout the

[4]It is worth mentioning that this finding corresponds to that reported for grammatical evolution in [19], where Rothlauf mentioned *"... in about 90% of cases a mutation of a genotype does not change the corresponding phenotype"*

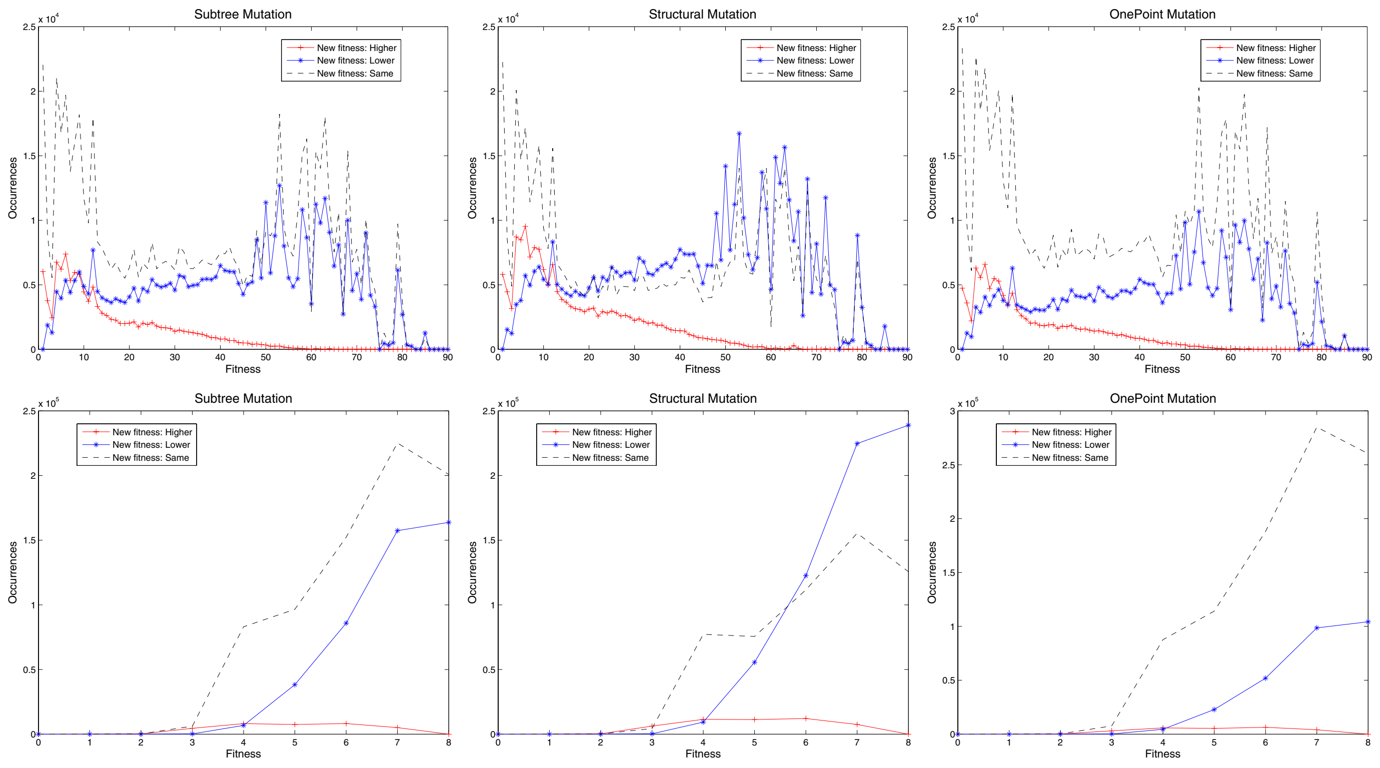

Figure 1: Results on the artificial ant problem (top) and on the even-3-parity problem (bottom). Occurrences of individuals after applying subtree (left), structural (middle) and one-point mutation (right).

three distances used in this work, where tree-alignment and normalised compression distance are the most illustrative.

Now, we turn our attention on detrimental mutations (i.e., where resulting offspring's fitness are lower compared to their parent's fitness). Figure 4 illustrates this scenario, where we see a similar effect of what happens in the presence of beneficial mutations (i.e., where resulting offspring's fitness are higher compared to their parent's fitness). That is, the distance between parent-offspring at the genotypic level is lower when it is closer to the global optimum. This supports our previous findings (i.e., beneficial mutations).

Finally, let us discuss the effects when we do not see any effect at the fitness level. This is shown in Figure 5. As it has been shown in [3], the even-3-parity problem has a highly neutral landscape. By definition of neutrality, we know that we will not see any change at fitness level (see left of Figure 5), but we do not know what are the distances that one may find on this scenario. As we can see we have a symmetrical image where the highest distance peak is at fitness 4 (regardless of the distance metrics and mutation operators used). This suggest that the closer the individual is to the half of the global optimum the bigger the changes are at the genotypic level.

6. CONCLUSIONS

When analysing the relative effects of mutations of large and small genotypic step-size starting at high- and low-fitness individuals, we observed that when the original fitness is high, mutations of large genotypic distance tend to be quite detrimental. When the original fitness is low, large genotypic jumps can lead to fitness improvements. This is an instance of the balance between exploration and exploitation: when the population is relatively poor, exploration is beneficial, but when the population has begun to converge on high-quality areas of the search space, smaller, exploitative mutations are better. Strategies which vary mutation rates to achieve a varying balance between exploration and exploitation have been used in the past; our results on the genotypic step-sizes of the three mutations operators suggest that varying the mutation operator itself could have the same effect.

The most important result we have obtained is that it is structural mutation which gives rise to the highest locality among the operators used in this paper. This is not surprusing given the nature of the operator. That is, the result of applying this operator on a tree results on similar tree that is smaller o larger by one node. It must be emphasised that structural mutation was originally developed for problems such as the royal tree problem, where every node type has a distinct arity and an individual's semantics are defined purely by genotypic structure, and so structural mutation functioned as a "minimal" mutation [21, 22]. In other problems, such as the two we have studied, structural mutation is not the minimal mutation. The other two operators, one-point and subtree mutation, give rise to relatively low locality, where subtree mutation, surprisingly shows a higher locality compared to one-point mutation. This encorage us to explore the possibility of increasing the locality of these two operators.

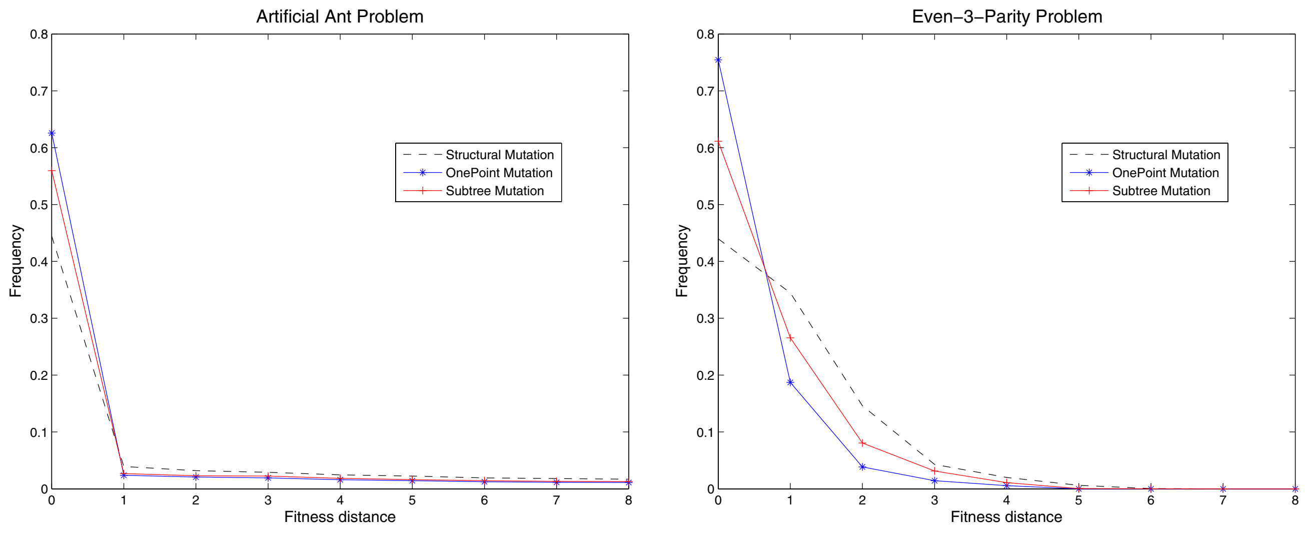

Figure 2: Distribution of fitness distance vs. frequency. Results on the artificial ant problem (left) and on the even-3-parity problem (right) using structural, one-point and subtree mutation. Notice that for clarity purposes we are taken the first 9 fitness values for the artificial ant problem (left).

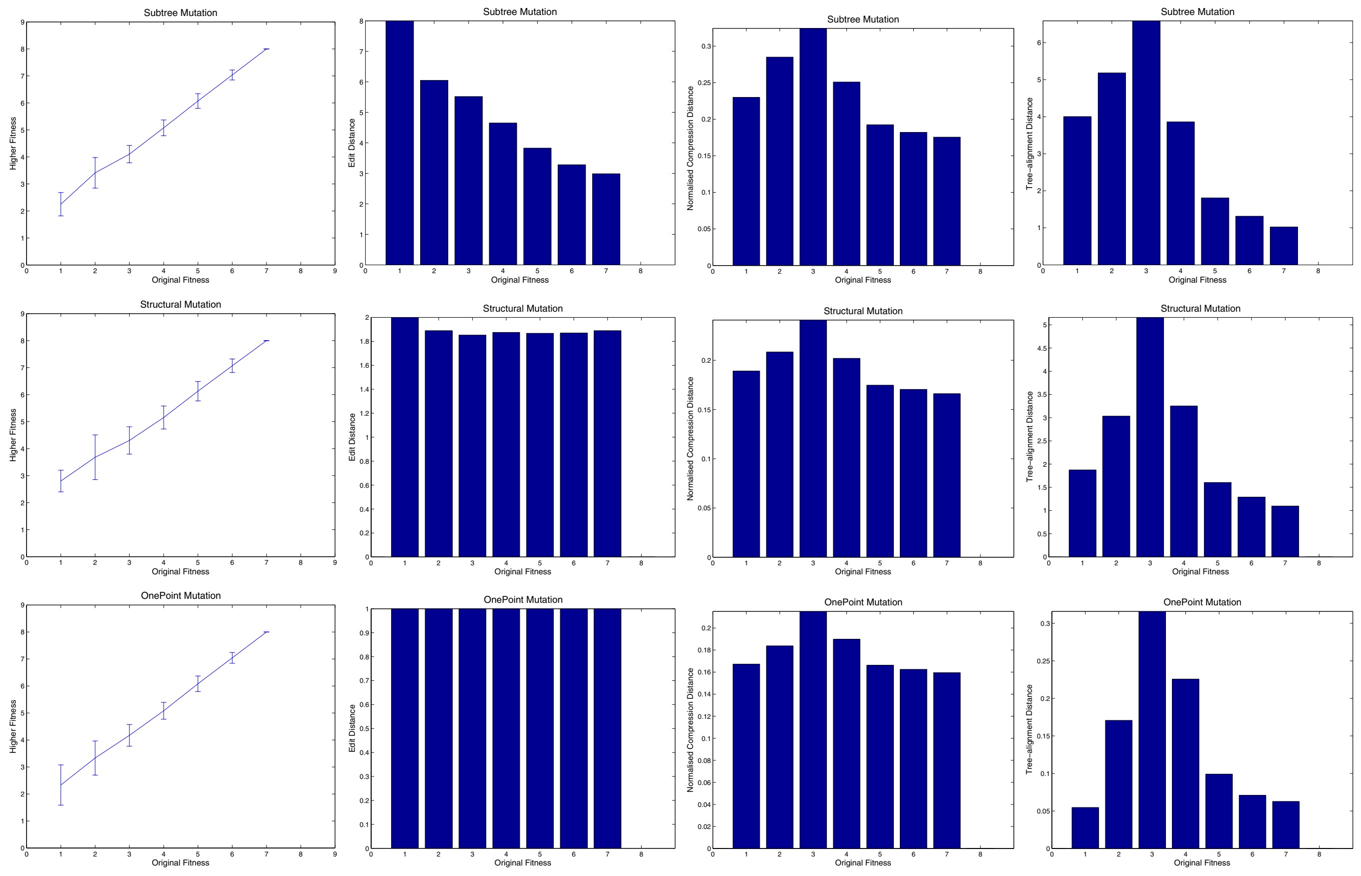

Figure 3: Results after applying subtree (top), structural (middle) and one-point mutation (bottom) using the even-3-parity problem. Original fitness versus higher fitness (first column) and distance metrics using edit (second column), normalised compression (third column) and tree-alignment distance (fourth column).

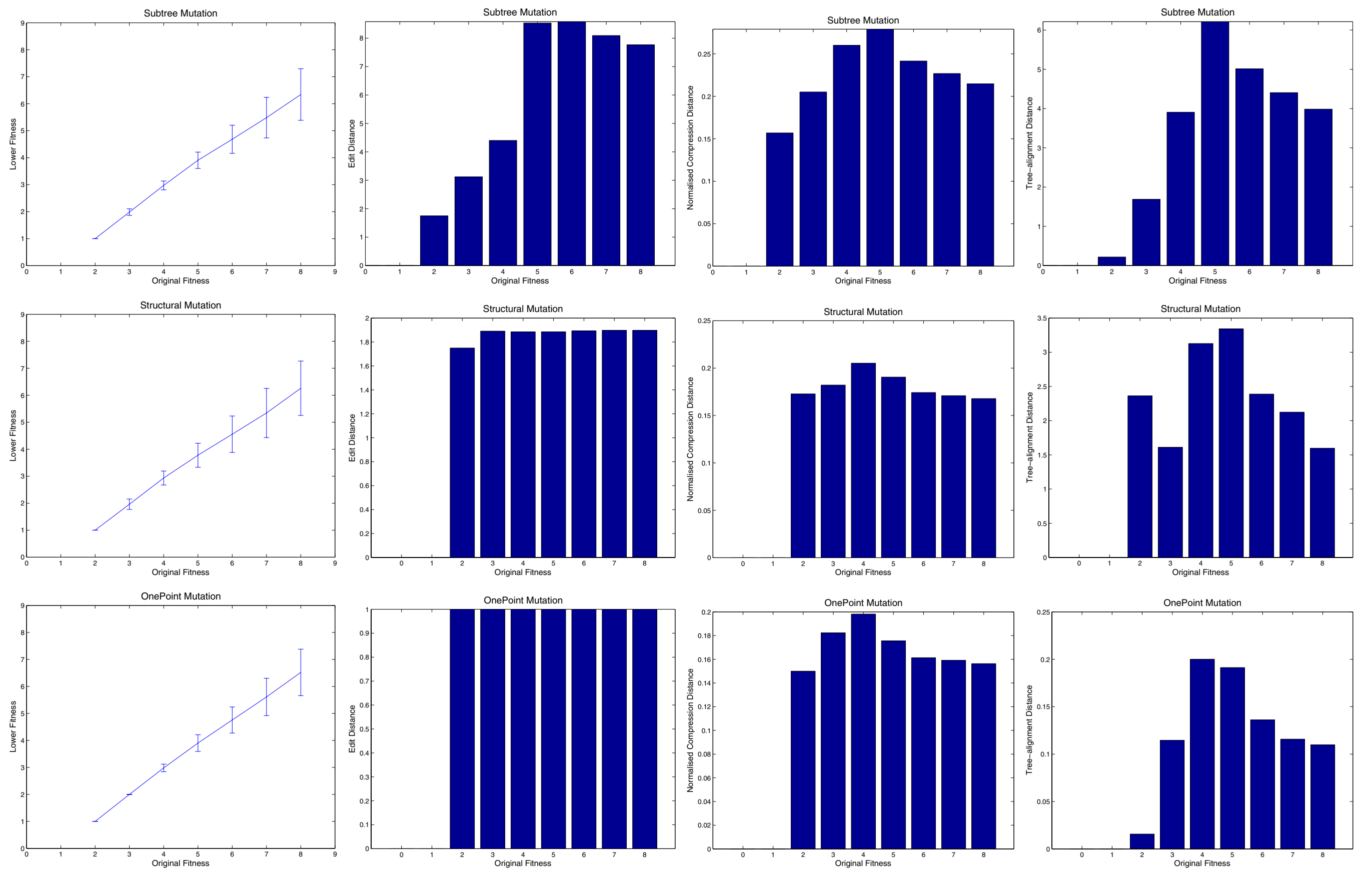

Figure 4: Results after applying subtree (top), structural (middle) and one-point mutation (bottom) using the even-3-parity problem. Original fitness versus lower fitness (first column) and distance metrics using edit (second column), normalised compression (third column) and tree-alignment distance (fourth column).

Acknowledgments

This research is based upon works supported by Science Foundation Ireland under Grant No. 08/IN.1/I1868 and by the Irish Research Council for Science, Engineering and Technology under the Empower scheme.

7. REFERENCES

[1] H. Beyer and H. Schwefel. Evolution strategies–A comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.

[2] R. Cilibrasi and P. M. B. Vitanyi. Clustering by compression. *IEEE Transactions on Information theory*, 51(4):1523–1545, 2005.

[3] M. Collins. Finding needles in haystacks is harder with neutrality. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 1613–1618, New York, NY, USA, 2005. ACM.

[4] I. De Falco, A. Iazzetta, E. Tarantino, A. Della Cioppa, and G. Trautteur. A Kolmogorov complexity based genetic programming tool for string compression. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, 2000.

[5] P. D'haeseleer and J. Bluming. Effects of locality in individual and population evolution. In K. E. Kinnear, editor, *Advances in Genetic Programming*, pages 177–198. MIT Press, 1994.

[6] E. Galván-López. *An Analysis of the Effects of Neutrality on Problem Hardness for Evolutionary Algorithms*. PhD thesis, School of Computer Science and Electronic Engineering, University of Essex, United Kingdom, 2009.

[7] F. J. Gomez. Sustaining diversity using behavioral information distance. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 113–120, Montréal, Canada, 2009. ACM.

[8] T. Jiang, L. Wang, and K. Zhang. Alignment of trees – an alternative to tree edit. *Theoretical Computer Science*, 143(1):137–148, 1995.

[9] T. Jones. *Evolutionary Algorithms, Fitness Landscapes and Search*. PhD thesis, University of New Mexico, Albuquerque, 1995.

[10] M. Kimura. *The Neutral Theory of Molecular Evolution*. Cambridge University Press, Cambridge, UK, 1983.

[11] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural*

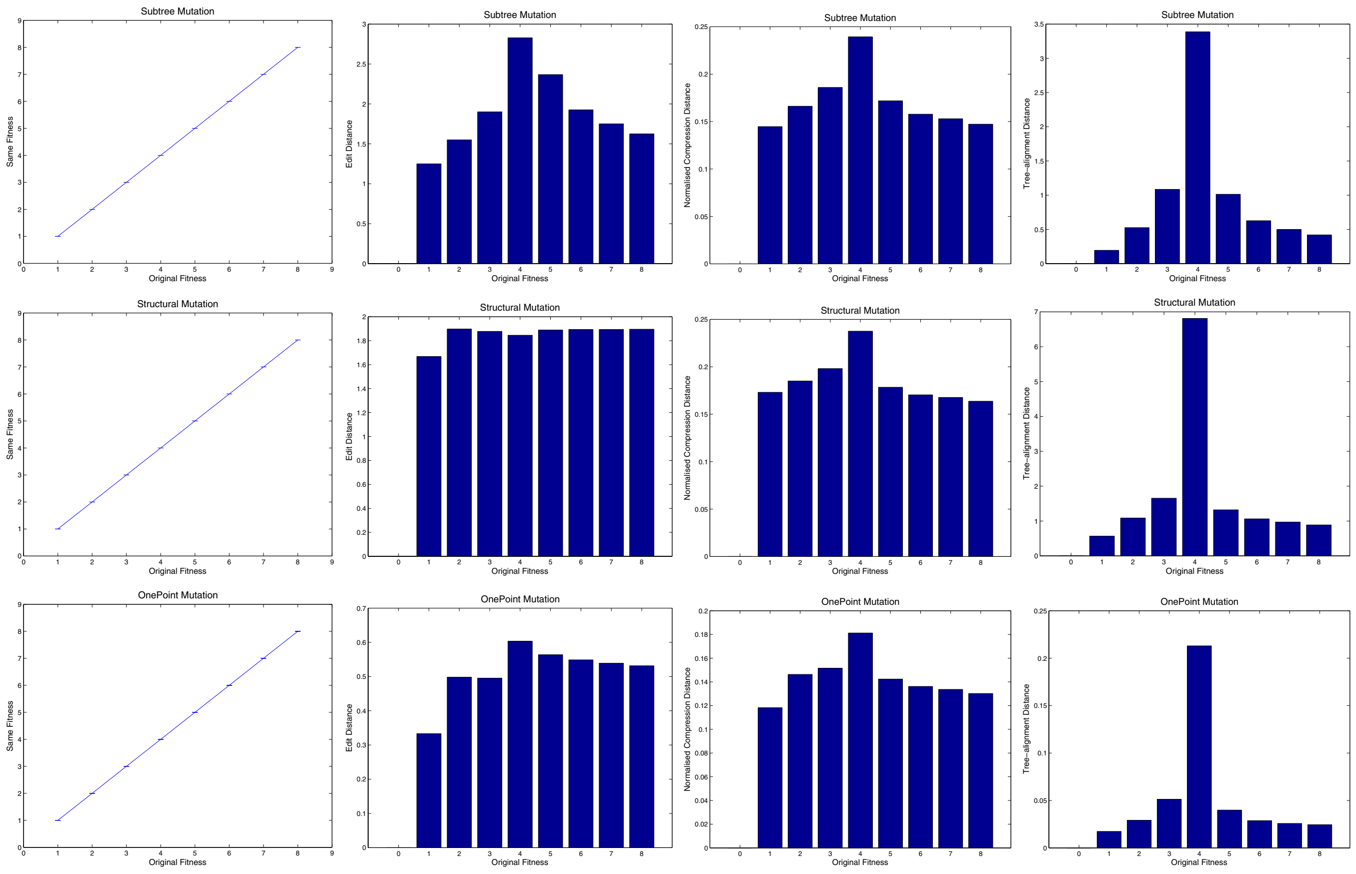

Figure 5: Results after applying subtree (top), structural (middle) and one-point mutation (bottom) using the even-3-parity problem. Original fitness versus unchanged fitness (first column) and distance metrics using edit (second column), normalised compression (third column) and tree-alignment distance (fourth column).

Selection. The MIT Press, Cambridge, Massachusetts, 1992.

[12] W. Langdon and R. Poli. Why ants are hard. In J. R. Koza, editor, *Proceedings of the Third Annual Conference on Genetic Programming*, pages 193–201. Morgan Kaufmann, Madison, USA, 1998.

[13] W. B. Langdon and R. Poli. *Foundations of Genetic Programming.* Springer, Berlin, 2002.

[14] M. Li and P. Vitanyi. *An introduction to Kolmogorov complexity and its applications.* Springer Verlag, 1997.

[15] U.-M. O'Reilly. Using a distance metric on genetic programs to understand genetic operators. In *IEEE International Conference on Systems, Man, and Cybernetics: Computational Cybernetics and Simulation*, volume 5, 1997.

[16] 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).

[17] F. Rothlauf. *Representations for Genetic and Evolutionary Algorithms.* Physica-Verlag, 2nd edition, 2006.

[18] F. Rothlauf. On the bias and performance of the edge-set encoding. *IEEE transactions on evolutionary computation*, 13(3):486–499, June 2009.

[19] F. Rothlauf and M. Oetzel. On the Locality of Grammatical Evolution. In P. Collet, M. Tomassini, M. Ebner, S. Gustafson, and A. Ekart, editors, *EuroGP*, volume 3905 of *Lecture Notes in Computer Science*, pages 320–330. Springer, 2006.

[20] V. Slavov and N. I. Nikolaev. Fitness Landscapes and Inductive Genetic Programming. In G. D. Smith, N. C. Steele, and R. F. Albrecht, editors, *Artificial Neural Nets and Genetic Algorithms: Proceedings of the International Conference, ICANNGA97*, University of East Anglia, Norwich, UK, 1997. Springer-Verlag.

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

[22] L. Vanneschi. *Theory and Practice for Efficient Genetic Programming.* PhD thesis, Faculty of Science, University of Lausanne, Switzerland, 2004.

[23] S. Wright. The Roles of Mutation, Inbreeding, Crossbreeding and Selection in Evolution. In D. F. Jones, editor, *Proceedings of the Sixth International Congress on Genetics*, volume 1, pages 356–366, 1932.