Causality of Hierarchical Variable Length Representations

Christian Igel, Institut für Neuroinformatik, Ruhr-Universität Bochum, Germany

Abstract

In this paper the strong causality of program tree representations is considered. A quantitative, probabilistic causality measure is used in contrast to statistical fitness landscape analysis methods. Although it fails to rank different problems according to their difficulty, it is helpful to choose the right coding for a given task. The investigation utilizes a metric on the search space called tree edit distance. Different ways to define such a measure are discussed.

1. Introduction

Algorithms in which problem solving is formulated as a search in the space of programs (i.e. expressions of a context free formal language) represented by trees are successfully applied to a growing field of tasks. However, there are few theoretical investigations concerning the properties of this coding, which is mainly used in genetic programming, but also within the scope of stochastic hillclimbing and simulated annealing.

O'Reilly [1] proposed an analysis of the variable length representation based on a *tree edit distance*. Such an analysis may provide deeper insight into the dynamics of the algorithms and can help to answer the question, whether a tree representation is suitable for a given task and how the algorithms, especially the operators[1], can be improved.

This paper investigates the *strong causality* of program tree representations w.r.t. the tree edit distance. The next section introduces a quantitative, probabilistic causality measure, proposed by Sendhoff et al. in [2], to fitness landscape analysis and briefly compares it to statistical methods. In Section 3 the application to standard genetic algorithms that use a binary fixed length representation is described. Section 4 discusses different distance measures between program trees. Section 5 reports on experiments which analyze the strong causality of variable length representations and the ability to approximate the tree edit distance by a simple string edit distance.

[1] The function of a search algorithm that generates new points in the search space to be tested for fitness is called operator throughout this paper.

2. Fitness Landscapes and Causality

Fitness landscape analysis is a promising approach to understand and improve the behavior of optimization algorithms, see e.g. [3, 4, 5]. A distance measure on the search space and the distribution of fitness values over the search space constitute a fitness landscape or adaptive surface. The application of an operator corresponds to a step on the landscape.

It is a common point of view that *ruggedness* of the landscape is one aspect which makes it difficult for search algorithms to find an optimal solution for a given problem. On the other hand, plateaus, i.e. regions of the search space where neighboring points have the same fitness values and therefore provide no "gradient" information, cause problems to the search process.

On a rugged surface each point of the search space contains relatively little information about the fitness of its neighboring points [3, 4, 5, 6]. If the fitness of a single point carries no information about the fitness of the surrounding points at all, hillclimbing e.g. is not more powerful than random search.

In order to exploit the knowledge regarding the fitness of the neighbors an operator is necessary which is related to the used distance measure d on the search space. The step size measured by d has to be small enough (e.g. one-point mutation and the Hamming distance in genetic algorithms) or, even better, can be controlled (e.g. mutation in evolution strategies [7]). The difficulties that arise if the distance measure is not related to the operator are discussed in [8]. A general way to define a suitable distance measure induced by a given operator is presented in [2].

Additional problems occur if the search algorithm is population based and sexual or even panmictic operators are used, e.g. crossover: A distance measure between two points in the search space that reflects the probability that one point in the search spaces is transformed into the other by a single application of such an operator depends on the composition of the actual population. Therefore the distance between two points varies due to the dynamics of the search process.

In the context of evolutionary computation, the principle of strong causality states that small changes in the search space lead to small changes in the fitness space [9, 10, 2, 11]. Summarizing, a search algorithm using a representation that obeys the principle of strong causality is theoretically able to

- perform a local search and gradually approach the optimum solution in small steps,
- control the exploration/exploitation tradeoff by changing the parameters of its operators,
- and self-adapt parameters that control the evolutionary process.

A smooth[2] fitness landscape implies strong causality. In most search algorithms there are steps where the strong causality is violated. In order to characterize a representation by its causality a quantitative measure is needed. The idea of such a measure was developed by Sendhoff et al. [12, 2], who introduced a probabilistic measure for the strong causality of genotype-phenotype mappings in the framework of evolutionary computation.

Let $d : I \times I \rightarrow \mathbb{R}_0^+$ be a distance measure on the search space I and $\Phi : I \rightarrow \mathbb{R}_0^+$ a fitness function. The question is, how much information $i \in I$ carries about its *neighboring* points. A simple way to define the set $N_{r,d}(i)$ of neighbors of i is

$$N_{r,d}(i) =_{def} \{j | j \in I \wedge d(j,i) \leq r\}$$

with $r \in \mathbb{R}^+$. For instance, r could be the maximum distance that should be moved in the search space by one step of the algorithm. Let $i_1 \in I$ and $i_2, i_3 \in N_{r,d}(i_1)$ be uniformly distributed random variables. The degree of local strong causality of a representation with respect to d and Φ can be regarded as the conditional probability $P(A|B)$ where the Boolean random variables A and B are defined as

$$\begin{aligned} A &=_{def} \quad ||\Phi(i_1) - \Phi(i_2)|| < ||\Phi(i_1) - \Phi(i_3)|| \\ B &=_{def} \quad d(i_1, i_2) < d(i_1, i_3) \quad . \end{aligned}$$

In other words, $P(A|B)$ is the probability that locally the property "The larger the distance, the larger the fitness difference" holds. Condition A can be weakened to

$$A' \quad =_{def} \quad ||\Phi(i_1) - \Phi(i_2)|| \leq ||\Phi(i_1) - \Phi(i_3)|| \quad .$$

Differences between $P(A|B)$ and $P(A'|B)$ reflect the frequency and size of plateaus.

The proposed measure has to be compared with existing landscape analysis approaches, which extract statistical information about the structure of the adaptive surface. Weinberger [5] and Manderick et al. [4] investigated the autocorrelation function of a landscape, which expresses how correlated the fitnesses of points separated by a certain distance are. Jones and Forrest proposed the fitness distance correlation (FDC) to analyze a fitness landscape. The FDC represents the correlation between fitness and distance from the nearest optimum in the search space [3]. The probabilistic causality measure is preferred in this paper because of the following reasons:

- The probabilistic causality measure is invariant under arbitrary monotonic transformations of the fitness function. The autocorrelation function lacks this property, see [13] for an example.
- The meaningfulness of the correlation coefficients depends on statistical assumptions concerning the distribution of the fitness and distance values.
- The FDC can only be computed if the global maxima are known, i.e. it can not be used to characterize unsolved problems.
- Techniques based on adaptive (and therefore non stationary) walks require a criterion that indicates whether a local maximum is reached. It is not possible to find such a criterion that can be calculated efficiently for some of the problems discussed in this paper.

It is possible to express the basic idea behind the probabilistic definition of strong causality by a correlation coefficient

$$\rho(D, F) = \frac{\mathrm{cov}(D, F)}{\sigma(D)\sigma(F)} \quad ,$$

where the random variables D and F represent the distance between two points in the search space and their absolute fitness difference, respectively. The standard derivations of the variables are denoted by $\sigma(D)$ and $\sigma(F)$. The function $\mathrm{cov}(D, F)$ is the covariance between D and F.

Correlation coefficients are not measured on a ratio scale, so e.g. strictly speaking it is not possible to compute the mean of two coefficients directly or conclude from a twice as high correlation coefficient that the measured dependency is twice as high. For a better comparison, a correlation coefficient $r \neq 1$ should be transformed to a value Z measured on a ratio scale,

$$Z = \frac{1}{2} \ln \left(\frac{1+r}{1-r} \right) \quad ,$$

see e.g. [14].

[2] Here *smooth* is used as the opposite of rugged, not in its strict mathematical sense.

In this paper, the probabilistic causality measure is applied to fixed length binary representations and tree representations. The investigated landscapes are not expected to be *isotropic*, except the *NK*-landscapes. Broadly speaking, isotropy means that the samples used to calculate the characteristic measures are "representative" of the entire landscape (see [13] for a detailed definition). However, the results should provide a heuristic that describes important properties of the landscapes and can be used to judge different representations and operators.

3. Causality and Prediction of Difficulty

NK-landscapes [6] are a family of landscapes with tunable ruggedness. The parameter $N \in \mathbb{N}$ denotes the dimension of the search space and $K \in \mathbb{N}$ $(0 \leq K \leq N-1)$ the degree of *epistatic interaction*. Increasing K increases the ruggedness of the landscape. Because a smooth landscape implies strong causality the quantitative causality measures introduced in the previous section should reflect the degree of epistasis. The results of a corresponding experiment are shown in Table 1. Three points in the search space were randomly created at a time and the Hamming distances and the fitness differences between them were calculated. This was done 10^5 times for each test problem. Based on these samples, the causality measures were estimated. The values of $P(A|B)$ and $P(A'|B)$ for each problem were approximately equal. The three measures rank the landscapes correctly.

Table 1. Results for *NK*-landscapes, *N*=20

K	$P(A\|B)$	Z
0	0.545	0.1433
1	0.532	0.1022
2	0.528	0.0634

K	$P(A\|B)$	Z
3	0.519	0.0489
4	0.510	0.0304
5	0.505	0.0045

Ruggedness of the adaptive surface is one aspect that makes a problem hard for a search process. Can the causality measure be used to predict the "GA difficulty" of a problem? Of course, one "global" value measuring just one property of the representation can only be a weak predictor, especially in the case of non isotropic landscapes.

Parameter optimization using a GA is taken as an example. Finding the minimum of a high dimensional sphere model[3] is easier if the representation uses Gray-code instead of standard binary-code. As shown in Table 2, the quantitative causality measures predict this behavior. This shows how the proposed heuristic can be helpful to choose the right representation: Higher degrees of strong causality should be preferred.

The step function adds plateaus to the sphere model. As regions in the search space with equal fitness values appear, $P(A|B)$ and $P(A'|B)$ differ significantly. The comparison of codings with different values for $P(A|B)$ and $P(A'|B)$ is difficile. In these examples, the values of $P(A'|B)$ and the coefficient Z rank the problems correctly according to their difficulty.

Table 2. Results for GA parameter optimization problems

function	coding	$P(A\|B)$	$P(A'\|B)$	Z
sphere	binary	0.502	0.502	0.0014
sphere	Gray	0.531	0.531	0.0815
step	Gray	0.511	0.544	0.0809

The causality measure works well in the case of the sphere model, but there are counterexamples where the results are misleading, e.g. the coding of the counting ones problem violates the principle of strong causality though the problem can be regarded as very "GA easy".

4. Tree Representation and Fitness Landscapes

In the framework of genetic algorithms the program tree representation was introduced by Cramer [15]. Koza [16] improved and popularized this search technique and termed it genetic programming (GP). O'Reilly and Oppacher [17] have isolated the hierarchical variable length representation from the evolution/population-based approach and use it with simulated annealing (SA) and stochastic iterated hill-climbing (SIHC). The main operator in GP is one-point crossover, which swaps subtrees between two individuals of the population to produce two new offsprings. Standard mutation can be realized by crossover with a randomly created tree. So basically the same operators can be used in GP, SA and SIHC (see [18]).

Kinnear [19] examined the fitness landscapes on which GP operates by comparing the autocorrelation functions of random walks and analyzing adaptive walks. The "unusually low" autocorrelations appeared not to be an indicator of the difficulty of the test problems.

Obviously program trees do not fulfill the principle of strong causality in general: Changing just a single node can lead to drastic changes in the program behavior. On the other hand it seems to be intuitive that on average a larger number of changes leads to a larger

[3] The test functions in this section are described in detail in [7].

variation in the fitness space. This property highly depends on the chosen function and terminal set and the fitness function.

For landscape analysis a distance or neighborhood measure on the search space is needed. In [1] the importance of such a measure in GP is outlined and reasons for the use of a tree edit distance are given. The considered program representations are special forms of rooted, ordered, and labeled trees (for shortness simply referred to as trees in this paper). The problem of measuring the distance between trees is not new: there are applications in vision, pattern recognition, molecular biology, program compilation and natural language processing (see [20] for references). The edit distance between two trees is defined by calculating the minimum cost sequence of elemental edit operations that transforms one tree into the other. These elemental operations on trees are:

Deletion A node n is removed. The children of n become children of the former parent of n.

Insertion A single node n is added. If n is inserted as a child of a node m then a consecutive subsequence of children of m may become the children of n.

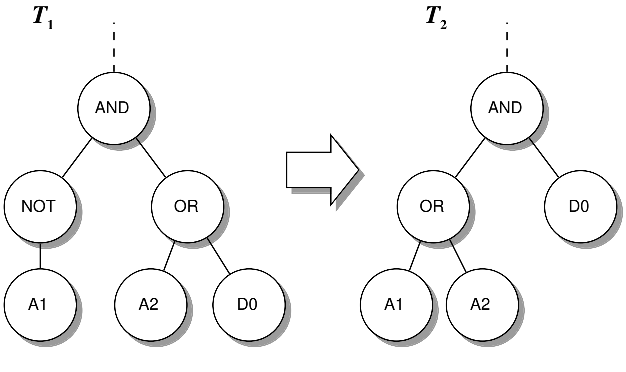
Replacement The label of a node is changed.

Every single operation is assigned a cost. Throughout this paper a cost of one for all elemental operations is assumed.

It has to be stressed that there are different ways to define and calculate an edit distance between two trees T_1 and T_2. Selkow [21, 22] presented an algorithm to calculate an edit distance where the operations of inserting and deleting may only be applied at the leaves. Let $|\cdot|$ denote the number of nodes in a tree, then the edit distance proposed by Selkow can be computed in time complexity $\mathcal{O}(|T_1| \cdot |T_2| \cdot \min(\text{depth}(T_1), \text{depth}(T_2)))$. Tai [23] introduced an unrestricted edit distance. Zhang and Shasha [24] found a dynamic programming algorithm to compute this distance in time $\mathcal{O}(|T_1| \cdot |T_2| \cdot \min(\text{depth}(T_1), |T_1|) \cdot \min(|T_2|, \text{depth}(T_2)))$. Zhang [20] presented a new metric, called constrained editing distance, which can be computed in $\mathcal{O}(|T_1| \cdot |T_2|)$: "The intuitive idea behind this definition is that two separate subtrees of T_1 should be mapped to two subtrees of T_2 and vice versa." In his paper the functions d_S and d_T denote the tree edit distances introduced by Selkow and Tai, respectively.

Program trees have the special property that every label is assigned a fixed arity. The exact number of children of a node is determined by the arity of its label, so every program tree is uniquely defined by its preorder traversal. This property induces another distance measure which can be computed by a fast and simple dynamic programming algorithm. Let $GP_{FS,TS,\alpha}$ be the set of all legal program trees with function set FS, terminal set TS and arity function α. Then the function $d_\mathrm{L} : GP_{FS,TS,\alpha} \times GP_{FS,TS,\alpha} \to \mathbb{R}_0^+$ defined as the *Levenshtein distance* between the preorder traversals of its arguments is a distance measure on $GP_{FS,TS,\alpha}$. The Levenshtein distance is a common distance measure between strings (see e.g. [25]). It is defined as the minimum cost for a sequence of insertion, deletion and substitution operations that transform one string into the other. It can be computed in time complexity $\mathcal{O}(|T_1| \cdot |T_2|)$. It is not "unnatural" to view genetic programs as strings: as their preorder traversals, as integer strings [26] or binary strings [27].

All proposed measurement functions are metrics on $GP_{FS,TS,\alpha}$. It holds $d_\mathrm{L}(T_1, T_2) \leq d_\mathrm{T}(T_1, T_2) \leq d_\mathrm{S}(T_1, T_2)$ for all T_1, $T_2 \in GP_{FS,TS,\alpha}$. An example, where d_T, d_S and d_L have different values is given in Fig. 1.: Deletion of OR, deletion of NOT and insertion of OR is a minimum transformation sequence that leads to a tree edit distance $d_\mathrm{T}(T_1, T_2) = 3$. If deletion and insertion is restricted to the leaves, two more elemental operations are necessary, i.e. $d_\mathrm{T}(T_1, T_2) = 5$. The Levenshtein distance between the preorder traversals of the trees is two. A minimum string edit sequence is deletion of OR and substituting NOT by OR. Hence $d_\mathrm{L}(T_1, T_2) = 2$.



T_1: AND NOT A1 OR A2 D0
T_2: AND OR A1 A2 D0

Figure 1. Example of two program (sub)trees with corresponding preorder traversals

Insertion, deletion or substitution of a single node is not expected to yield a legal program tree. Let d be one of the tree distance measures proposed in this section and $n > 1$ an integer. If two program trees $T_1, T_2 \in GP_{FS,TS,\alpha}$ have edit distance $d(T_1, T_2) \geq 2$ there exists in general *no* sequence of valid program trees $T_1', T_2', \ldots, T_n' \in GP_{FS,TS,\alpha}$ with $T_1 = T_1'$, $T_2 = T_n'$ and $d(T_m', T_{m+1}') = 1$ for all positive integers $m < n$. Because of this, several landscape analysis methods can not be used directly with the proposed edit distances. The fact, that the intermediate edit operations of the distance calculation do not result in correct program trees is no problem for the distance measure in general: "The end result is all we need" [1].

The tree edit distance was proposed by O'Reilly to be a powerful tool to understand and analyze search algorithms which use a variable length representation. As stated in the Section 2 a distance measure on the search space should be related to the used search operators. O'Reilly [17, 18] introduced such an operator called HVL-mutate which was designed to correspond to the edit distance as close as possible. In the next section the degree of strong causality of variable length representations w.r.t. the unconstrained tree edit distance d_{T} is analyzed. In addition, d_{T} and d_{L} are compared.

5. Experimental Results

Three standard GP problems were used to judge the strong causality of the program tree coding, all introduced and described in detail in [16]. The first task was to evolve a 6-multiplexer: find a composition of primitive Boolean functions, namely $\{\mathrm{not}, \mathrm{and}, \mathrm{or}, \mathrm{if}\}$, which calculates a known Boolean function with six arguments. The fitness of an individual is the number of correct answers to the 64 possible inputs. The second task was a simple symbolic regression problem: find a function that fits a given fourth order polynomial. The sets $TS = \{x, 0, 1, 2, \ldots, 9\}$ and $FS = \{+, -, \times\}$ were used and the fitness space was the integer interval $[0, 1000]$. The fitness was based on the differences between the function values of the evolved and the target function at ten prespecified sample points. The third test case was the artificial ant problem: evolve a move rule for an ant trying to find all food lying on an irregular trail. The "Santa Fe Trail" consisting of 89 food items, $TS = \{\mathrm{left}, \mathrm{right}, \mathrm{move}\}$, $FS = \{\text{if-food-ahead}, \mathrm{prog2}, \mathrm{prog3}\}$ and a maximum of 600 moves were used. Experiments (e.g. in [16]) show that the ant problem is more difficult for GP and SIHC than the symbol regression problem, which is more difficult than the multiplexer problem.

A sample of $2 \cdot 10^5$ crossover operations per test problem was used to approximate the proposed causality measures: Three program trees with a maximum depth of six were randomly created and crossover was performed between these trees. Then the fitness differences and tree edit distances d_{T} and d_{L} between parents and offsprings were computed. This scenario is not quite appropriate for population based search algorithms like GP where crossover is performed between individuals of a population formed under the influence of selection. However, the goal was to judge the representation not a certain search technique. The neighborhood parameter r was set to infinity.

The results are shown in Table 3. It is striking that

Table 3. Results for GP test problems

task	$P(A\|B)$	$P(A'\|B)$	Z	SM
6-Mux	0.41	0.77	0.0831	40 %
Symbreg	0.64	0.73	0.4521	11 %
Ant	0.41	0.81	0.1439	44 %

no measure is correlated to the "GP difficulty" and that the discrepancies between $P(A|B)$ and $P(A'|B)$ are high. These differences stem from the redundancies of the codings and the small fitness spaces. In the three test problems the search space is mapped on only 65, 1001 and 90 different fitness values respectively. A high difference between $P(A|B)$ and $P(A'|B)$ coincides with a high rate of "silent mutations", i.e. steps in the search space without change in the fitness space. The percentage of silent mutations (SM) is smaller in the symbol regression problem, because the function set does not lead to as much redundant code as in the case of the other tasks. The influences of the size of the fitness space and the redundancies of the coding make the results deceptive.

The values for both $P(A|B)$ and $P(A'|B)$ in the case of the symbol regression problem are surprisingly high compared to other codings. Simple parameter optimization problems with a fixed length Gray code encoding have $P(A'|B) < 0.6$, see Section 3 and [2, 12]. The high degree of strong causality in the case of the symbol regression problem shows that HVL-mutate seems to be at least as able as point mutation in GA parameter optimization problems to perform a local search.

Although the depth of the program trees was restricted, it took *drastically* more time to compute d_{T} than d_{L}. The string edit distance d_{L} coincided with the "real" tree edit distance d_{T} in more than 96% of all computations and the rare differences were very small.

6. Conclusion

The probabilistic causality measure reflects the ruggedness of the adaptive surface and is suitable to judge different codings or operators for a given problem: Higher degrees of strong causality should be preferred. It fails, like other fitness landscape analysis approaches, to indicate the difficulty of GP problems. The degree of strong causality in the case of the symbol regression problem can be regarded as comparatively high, encouraging the usage of the coding.

The tree edit distance was used to analyze the fitness landscape of program tree representations. It can be replaced or approximated by the Levenshtein distance between their preorder traversals to save computation time.

References

[1] U.-M. O'Reilly, "Using a distance metric on genetic programs to understand genetic operators," in *Late Breaking Papers at the Genetic Programming 1997 Conference*, J. R. Koza, Ed., Stanford University, 1997.

[2] B. Sendhoff, M. Kreutz, and W. von Seelen, "A condition for the genotype-phenotype mapping: Causality," in *Proceedings of the Seventh International Conference on Genetic Algorithms*, T. Bäck, Ed. 1997, Morgan Kauffmann Press.

[3] T. Jones and S. Forrest, "Fitness distance correlation as a measure of problem difficulty for genetic algorithms," in *Proceedings of the Sixth International Conference on Genetic Algorithms*. 1995, Morgan Kauffmann Press.

[4] B. Manderick, M. d. Weger, and P. Spiessens, "The genetic algorithm and the structure of the fitness landscape," in *Proceedings of the Fourth International Conference on Genetic Algorithms*, R. Belew and L. Booker, Eds. 1991, Morgan Kaufmann Press.

[5] E. Weinberger, "Correlated and uncorrelated fitness landscapes and how to tell the difference," *Biological Cybernetics*, vol. 63, 1990.

[6] S. A. Kauffman, *The Origins of Order*, Oxford University Press, 1993.

[7] T. Bäck, *Evolutionary Algorithms in Theory and Practice*, Oxford University Press, 1996.

[8] L. Altenberg, "Fitness distance correlation analysis: An instructive counterexample," in *Proceedings of the Seventh International Conference on Genetic Algorithms*, T. Bäck, Ed. 1997, Morgan Kauffmann Press.

[9] I. Rechenberg, *Evolutionsstrategie '94*, Werkstatt Bionik und Evolutionstechnik. Frommann-Holzboog, 1994.

[10] J. P. Rosca and D. H. Ballard, "Causality in genetic programming," in *Proceedings of the Fifth International Conference on Genetic Algorithms*. 1995, Morgan Kaufmann Press.

[11] B. Sendhoff and M. Kreutz, "Analysis of possible genome-dependence of mutation rates in genetic algorithms," in *Evolutionary Computing*, T. C. Fogarty, Ed. 1997, Springer Verlag.

[12] B. Sendhoff, M. Kreutz, and W. von Seelen, "Causality and the analysis of local search in evolutionary algorithms," Tech. Rep. 97–16, Institut für Neuroinformatik, Ruhr-Universität Bochum, 44780 Bochum, Germany, 1997.

[13] C. A. Macken and P. F. Stadler, "Evolution on fitness landscapes," in *Lectures in Complex Systems*, L. Nadel and D. L. Stein, Eds., Santa Fe Institute Studies in the Science of Complexity. Addison-Wesley, 1993.

[14] J. Bortz, *Statistik*, Springer Verlag, 1993.

[15] N. Cramer, "A representation for the adaptive generation of simple sequential programs," in *Proceedings of an International Conferece on Genetic Algorithms and their Application*, J. J. Grefenstette, Ed., 1985.

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

[17] U.-M. O'Reilly and F. Oppacher, "Program search with a hierarchical variable length representation: Genetic programming, simulated annealing and hill climbing," in *Parallel Problem Solving from Nature – PPSN III*, Y. Davidor, H.-P. Schwefel, and R. Männer, Eds. 1994, Springer Verlag.

[18] U.-M. O'Reilly and F. Oppacher, "Hybridized crossover-based search techniques for program discovery," Working Paper 95–02–007, Santa Fe Institute, 1995.

[19] K. Kinnear, "Fitness landscapes and difficulty in genetic programming," in *Proceedings of the IEEE World Congress on Computational Intelligence*, J. D. Schaffer, H. P. Schwefel, and H. Kitano, Eds. 1994, Morgan Kaufmann Publishers.

[20] K. Zhang, "Algorithms for the constrained editing distance between ordered labeled trees and related problems," *Pattern Recognition*, vol. 28, no. 3, 1995.

[21] A. S. Noetzel and S. M. Selkow, *An Analysis of the General Tree-Editing Problem*, chapter 8, In Sankhoff and Kruskal [25], 1983.

[22] S. M. Selkow, "The tree-to-tree editing problem," *Information Processing Letters*, vol. 6, no. 6, 1977.

[23] K.-C. Tai, "The tree-to-tree correction problem," *Journal of the ACM*, vol. 26, no. 3, 1979.

[24] K. Zhang and D. Shasha, "Simple fast algorithms for the edit distance between trees and related problems," *SIAM Journal of Computing*, vol. 18, no. 6, 1989.

[25] D. Sankhoff and J. B. Kruskal, Eds., *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, Addison-Wesley Publishing Company, Inc., 1983.

[26] N. R. Patterson and M. Livesey, "Distinguishing genotype and phenotype in genetic programming," in *Late Breaking Papers at the Genetic Programming 1996 Conference*, J. R. Koza, Ed., 1996.

[27] W. Banzhaf, "Genotype-phenotype-mapping and neutral variation – a case study in genetic programming," in *Parallel Problem Solving from Nature – PPSN III*, Y. Davidor, H.-P. Schwefel, and R. Männer, Eds. 1994, Springer Verlag.