Explicit maintenance of genetic diversity on genospaces

Robert E. Keller Wolfgang Banzhaf

Dortmund University
Computer Science Department
Systems Analysis Chair
D–44221 Dortmund

June 1994

Abstract

When evolving genotypes, i.e. structures, with an evolutionary algorithm (EA), e.g. *genetic programming* (GP), *genetic diversity*, i.e. structural diversity, of each generation is a necessary condition for the fast detection of a high-fitness individual and for a fast adaptation of the population to a changing environment. Thus, it is an important objective to maintain diversity during runtime of the EA.

Concepts and methods are introduced which propose extensions of EAs towards explicit maintenance of diversity by means of formally defined measures. For arbitrary *genospaces*, which are sets of genotypes, a *diversity measure* is proposed that is based on a *structural measure* which is defined by the minimal number of applications of *edit operations* needed to transform a genotype into another genotype.

The proposed measures, concepts and methods are general, i.e they are independent on the structure type of the actual genotypes. Only the edit operations depend on the actual structure type.

For the structure type of *node-labeled trees*, we propose edit operations, so that the above measures and methods can be used by GP paradigms operating on such structures.

Since the proposed measures are purely structure-based, they are orthogonal to fitness as a quality measure on genospaces.

key words:
diversity measure, edit operation, evolutionary algorithm, genetic diversity, genetic programming, node-labeled trees, structural measure, structure

1 Introduction

It is a well-known fact that diversity drops during runtime of an EA, especially if it is GA-based, since, due to the use of some *fitness-based* selection mechanism, clusters of individuals build up, whose members represent only a few different genotypes.

Often, the evolving population of individuals converges in local optima. If those *substructures* which could be recombined into a fitter individual do not exist in the actual generation, or if they exist in low-fitness individuals which become extinct before recombination, or if mutation, if used at all, is unlikely to create them, then this fitter individual will never be evolved or is unlikely to be evolved before termination of the EA.

Several concepts, for instance *demes* [references], have been introduced in order to maintain diversity. These concepts may be called **implicit** in that they do not *create* individuals dependent on the **genetic structure** of the actual generation. This means they ignore number and structure of different genotypes and the number of individuals representing a given genotype. Instead, they operate on *evolved* genotypes, e.g. allowing migration of evolved genotypes between demes.

We propose an *explicit* concept for maintaining diversity by creating individuals dependent on the actual generation's genetic structure. This concept is independent of the structure type of the genotypes, e.g. binary strings or trees.

On the other hand, we pay special attention to the application of this concept to **tree spaces**, i.e. genospaces containing *arbitrary node-labeled trees*, as they are used in the genetic programming paradigm of Koza [JoKo92] and associated paradigms.

First, we give an intuitive approach to genetic diversity. This leads to the formal definition of a structural measure on genospaces. By means of this measure, we define a diversity measure on genospaces. We then propose methods using these measures to maintain diversity on genospaces, focussing on the GP-relevant special case of tree spaces. An extended GP sceme is presented that incorporates the proposed methods.

2 Genetic Diversity

Genotypes are structures, in general built of substructures. A genotype is in a certain **structure class**, i.e. it is a binary string or a tree or a graph etc. We also call such a structure class **genospace**. We use "genotype" and "genospace" as synonyms for "structure" and "structure class", since we are dealing with these phenomena in the context of evolutionary algorithms.

The same genotype can be *represented* by one or more *individuals* of a population, i.e. one or more individuals exhibit the structure given by that genotype. Thus, individuals representing the same genotype cannot be distinguished by their structure. To tell them apart, one tags them with unique symbols like names or numbers. The individuals A, B, C, for instance, can represent the same binary string.

Let a population of individuals be evolved by some evolutionary algorithm by means of a fitness function. All individuals of this population come from the same genospace. We call an individual that may get evolved in some generation a **potential individual** and an individual that has been evolved and exists in the actual generation an **actual individual**. We call a genotype that is represented by a potential individual a **potential genotype**, dito **actual genotype**.

The notion behind the term *high genetic diversity* is that the actual genotypes "cover" the complete genospace, i.e. there are many different actual genotypes and they are "equally distributed" over the genospace. In this sense, diversity is maximal iff. the set of actual genotypes equals the genospace. Then, however, the evaluation of the fitness function would imply solving the underlying problem by enumeration instead of evolution. Mostly, of course, practically relevant genospaces are much too large to consider enumeration.

The above notion of "equally distributed" genotypes introduces the idea of *structural distance* between genotypes. Thus, in order to mathematically approach the idea of genetic diversity and its maintenance, we need a *structural measure* for genotypes.

3 Structural Measure

First we propose a mapping of an arbitrary genospace G onto a subset of $\mathbb{N}_0^n$, dependent on n edit operations used on the genotypes in G. A metric on $\mathbb{N}_0^n$ is introduced as a structural measure on G. We then apply the given results to the case of G being a tree space.

The spatial distance of two places A and B is the minimal number of unit-sized steps needed to get from A to B. Correspondingly, the **edit distance** of two structures $A, B \in G$ is commonly defined as the *minimal* number of applications of given *elementary edit operations* needed to transform A into B (see [VaHo94] for a discussion of distance measures for structured patterns and [references given by VaHo]).

On the space of fixed-length binary strings with bit flipping as edit operation, for instance, the Hamming distance, i.e. the minimal number of bit flippings needed to transform a string into another one, is an often used type of edit distance [reference].

Using edit operations to transform A into B corresponds to "walking" from A to B within G. Given a special genotype o_G in G as its **origin**, each other genotype that can be reached, starting at o_G, by applications of edit operations can be identified with the edit distance between itself and o_G.

Given G, *a set of edit operations* E_G *and an origin* o_G, we define G's **distance space** $D_G \subset \mathbb{N}_0$ as the space whose points are edit distances between o_G and any of G's genotypes. Thus, each genotype is identified with exactly one point in D_G, and, for each point q in D_G, there is at least one genotype g whose edit distance to o_G equals q. Let q be called g's **position**.

Consider, for instance, $G = \{0,1\}^4$, the space of binary strings with length four under the edit operation *flipping of an arbitrary bit*. Let 0000 be the origin. Obviously, $D_G = \{0,1,2,3,4\} \subset \mathbb{N}_0$ holds. For instance, 0110 is identified with $2 \in D_G$, and, for $4 \in D_G$, there is 1111, whose edit distance to 0000 equals 4.

The motivation behind the definition of a distance space D_G associated with a genospace G is to define the structural distance of arbitrary $g_1, g_2 \in G$ as the distance d of $q_1, q_2 \in D_G$. d will be given by a metric on D_G.

The term "geno*space*" motivates the question for the *dimensionality* of a given genospace G. The genospace of binary strings with length n, for instance, is represented by a n-dimensional hypercube. There is a bijection between the 2^n cube corners and the genotypes (see [JoKo92] for a discussion and visualization). This corresponds to the fact that bit flipping can be applied to n different bits.

"Walking" through a genospace motivates the question for the *direction* of the walk, i.e. towards or away from the origin. In terms of edit operations, this can be modeled by two inverse **modes** of the same edit operation, for instance, bit flipping from 0 to 1 and vice versa. The mode that corresponds to a walk *away from* the origin is called **positive**, the other one is called **negative**.

For the above example, the flipping from 0 to 1 is the positive mode of the bit-flipping operation.

In order to motivate consideration of the *dimensionality* of a distance space, take again $G = \{0,1\}^4$ as an example. Let exactly one bit flipping be applied to the origin 0000. This may lead to 0001, 0010, 0100, or 1000. Thus, these genotypes are identified with point 1 in D_G.

In general, when two different genotypes g_1, g_2 are identified with the same point $q \in D_G$, we call this a **collision** of g_1, g_2 in q.

Recall the above motivation behind distance spaces. The distance of the points in D_G corresponding to g_1, g_2 is zero, no matter what actual metric d on D_G will be considered, since these points are both q. On the other hand, a structural measure on G, defined on the basis of d, should give a non-zero distance for g_1, g_2, since they are different. Therefore, collisions are bad and their number should be kept minimal.

Collisions are due to the fact that G, in general, is high-dimensional, while the $D_G \subset \mathbb{N}_0$ is one-dimensional. For instance, the above sample genospace is four-dimensional, since bit flipping can be applied to four different bits of a string.

A distance space is one-dimensional, since, even if there are several edit operations, the numbers of the applications of the *different* edit operations needed to transform the origin into some genotype g are summed up to *one* number only, namely the edit distance between the origin and g.

One way to reduce the number of collisions for some given G and D_G is to increment the dimensionality of D_G. Therefore, we propose to sum up the applications of n different edit operations to n numbers. Let each application contribute $+1$ or -1 to the corresponding number, according to the mode of the applied edit operation. When identifying each edit operation with exactly one dimension of D_G and vice versa, this gives a **n-dimensional distance space** $D_G \subset \mathbb{N}_0^n$, $n \in \mathbb{N}$.

Take again $G = \{0,1\}^4$ as an example. Consider the edit operations *flipping bit* 1, .., *flipping bit* 4, fb_i in short. A genotype that is reached by $a \cdot fb_1, b \cdot fb_2, c \cdot fb_3, d \cdot fb_4$ is identified with (a,b,c,d) in D_G. This gives $D_G = \{(a,b,c,d) | a,b,c,d \in \{0,1\}\} \subset \mathbb{N}_0^4$. Since D_G is four-dimensional now, collisions are completely avoided.

The described identification of genotypes with points in D_G defines a *mapping* $\pi : G \to D_G$. π is surjective, because for each $q \in D_G$ there is a $p \in G$ with $\pi(p) = q$. π can be a *projection*, as seen in the example with $D_G \subset \mathbb{N}_0$, and it can be even a *bijection*, as seen in the example with $D_G \subset \mathbb{N}_0^4$.

For genotypes $i, j \in G$, we propose, as a **measure for structural difference on** G,

$$\delta(i,j) := d(\pi(i), \pi(j))$$

with d being a *metric* on a n-dimensional D_G, like the well-known

$$d(a,b) = \sqrt{(a_1 - b_1)^2 + .. + (a_n - b_n)^2}$$

.

A metric d must satisfy (1) $d(i,j) = d(j,i)$, (2) $d(i,j) \neq 0 \Leftrightarrow i \neq j$ and (3) the triangle inequality [reference]. Obviously, δ always satisfies (1) and (3). However, it satisfies (2) iff. π is injective, as is easily shown. In this case, (G, δ) is a *metric space* [reference], and the number of collisions is zero.

If π is not injective, only $\delta(i,j) \neq 0 \Rightarrow i \neq j$ holds. In this case, we call δ a **weak metric**.

Recall that we proposed a multidimensional distance space D_G in order to reduce the number of collisions. There is another way to do this besides incrementing the number of D_G's dimensions.

Since, for practical reasons, G is always finite, $D_G \subset \mathbb{N}_0^n$ is finite, too. Since $\mathbb{N}_0^n$ is infinite, it is always possible to reduce the number of collisions, even down to zero. This is done, for instance, by introducing weight factors which are multiplied with the number of applications of

some edit operation, dependent on the type of operation and/or on the substructure it operates on.

Recall, for instance, the collisions of 1000, 0100, 0010 and 0001 in position 1. From left to right, let the bits of each string be identified by 1,2,3,4. When multiplying the number of bit flippings applied to a bit i by i, this gives 1,2,3,4 as positions for the above genotypes. Thus, they do not collide any more.

However, a practical relevant genospace is large. When using weight factors, the positions will get extremely large. When implementing this concept, dependent on the hardware, string arithmetic may be needed. Fortunately, incrementing the dimension of a distance space D_G reduces the size of the weight factors needed to reduce collisions. Unfortunately, only in the case of D_G being merely two-dimensional, efficient algorithms are known which enable diversity maintenance in a straightforward way, as will be seen in the next section.

We now focus on a genospace G being a tree space whose trees have at most depth D, applying the general results given above to this special case. Let T, F denote the non-empty **function set** and **terminal set** as defined in [JoKo92]. In order to define an associated distance space D_G, edit operations and an origin o_G must be given first. In this context, we introduce an example.

Consider the terminal set $T = \{a, b, c, d\}$ and the function set $F = \{\pm, +, *, /\}$. Let $\pm$ denote sign inversion. The symbols used to denote the elements of these sets are used as node labels. We define the **label arity** of a label l as the arity of that function resp. terminal that is denoted by l.

In the context of defining edit operations and o_G, we introduce a **coding function** $c : \{T \cup F\} \rightarrow \mathbb{N}$. $c(l)$ is called the **label code** of label l. Let c be injective.

One can think of many different guidelines for the specification of c. For example, one can express the "complexities" of different terminals and functions by having the coding function assign different numerical values to the corresponding labels.

For instance, if the target language requires a terminal to possess a certain data type, e.g. short or array, low values are assigned to terminals with simple integer types, higher values are assigned to terminals with simple fractional types, still higher values are assigned to terminals with structured integer types and so on.

Since a terminal can be considered as a 0-arity function, the highest value assigned to a terminal is required to be lower than the lowest value assigned to a function. Low values are assigned to unary functions, higher values are assigned to binary functions and so on. Within an F subset of functions with the same arity, lower values are assigned to computationally cheap functions like +, higher values are assigned to expensive functions like *.

Note that, if functions with "contradicting" properties, like the unary but in general expensive factorial function $n!$, were included in F, the question for a proper label code had to be answered.

Following these sample guidelines, one can, for instance, define $c(a) = 1, c(b) = 2, .., c(*) = 7, c(/) = 8$ for the above T and F sets.

Since *unlabeled* nodes will be considered below, the definition of the above coding function is extended by $c(\text{“”}) = 0$, “” being the **empty label**.

We propose the following set E_G of edit operations on a tree space G.

- **node operation**

  append an unlabeled node to a node N (*positive* mode)

Let a node N that has been labeled with a label f (see below for *label operation*) offer edges $e_1, .., e_{arity(f)}$. Some e_i to which a node, i.e. a child of N, has not yet been appended is called **free**. When being applied, "append" selects the next free edge, starting with e_1. Appending to N is legal only, if N still offers a free edge. Especially, appending to a node labeled with a terminal label t is illegal, since $arity(t)$ equals zero.

delete an unlabeled terminal node N (*negative* mode)

"delete" removes N, thus freeing the edge that connected N to its parent.

- **label operation**

  increment the label of a node N (*positive* mode)

  Let N be labeled with label l. "increment" relabels N with a label m such that $c(m) = c(l) + 1$. The application of "increment" is legal only, if there is such a label m.

  decrement the label of a node N (*negative* mode)

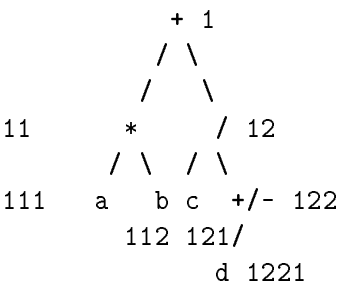
  Let N be labeled with label l. "decrement" relabels N with a label m such that $c(m) = c(l) - 1$. The application of "decrement" is legal only, if there is such a label m and if the label arity of m is not lower than the number of non-free edges of N.

For instance, let N be labeled with '/', and let it have no free edges. " decrement", applied to N, relabels it with '*'. The application of "increment" is illegal, since there is no label with a higher label code than that of '/'.

Corresponding to the above edit operations, let that tree with the smallest number of nodes and the smallest sum of label codes over all its nodes be the *origin* o_G of a tree space G. Since the coding function is injective, o_G computes as that unique tree that has exactly one node which is labeled by that terminal label with the smallest label code of all terminal labels.

Thus, for the above sample tree space G, o_G equals the tree with its node labeled 'a'.

Consider, for instance, the following tree i in the sample tree space.



o_G is transformed into i by this sequence of edit operations:

$(c(+)-c(a))\cdot increment, append(node11), c(*)\cdot increment, append(node111), c(a)\cdot increment,$
$append(node112), c(b)\cdot increment, append(node12), c(/)\cdot increment, append(node121),$
$c(c)\cdot increment, append(node122), c(\pm)\cdot increment, append(node1221), c(d)\cdot increment$

In general, for a given tree t, *exactly one* sequence transforming o_G into t is derived from a *depth-first traversal* of t (start at root node; recursively visit node, left subtree, right subtree). We call such a sequence **transformation sequence**. Interpreting a transformation sequence, knowing that it is derived from a depth-first traversal and knowing the label arity of each label,

gives exactly t. Thus, there is a bijection between a tree space and the set of transformation sequences derived from all trees of this space.

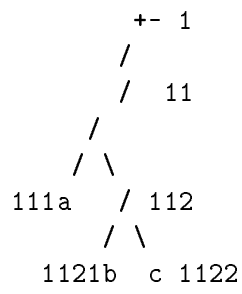
For each tree space, D_G is two-dimensional, since two edit operations have been proposed.

Since a node must be appended before its label can be incremented, the execution of the label operation "depends" on the execution of the node operation. This motivates the identification of the node operation resp. label operation with the x-dimension resp. y-dimension of D_G.

When transforming o_G into a tree t, the number x of positive node operations ("append") in the corresponding transformation sequence equals the number of t's nodes minus one, since o_G's node already is a node of t. The number y of positive label operations ("increment") equals the *sum of label codes* over the nodes of t minus the label code of o_G's label (cf. above transformation sequence, first "increment").

According to the above definition of a genotype's *position* in distance space, (x, y) is t's position in tree space. For instance, the position of the above tree i is $(7, 35)$.

Consider tree j in the sample tree space.



Its position is $5, 26$. According to the above definition of the measure δ the structural distance of the trees i, j equals approx. 9.22.

Recall the above argumentation that the genospace of binary strings with length n is n-dimensional. The same argumentation gives the dimension of a tree space as a function of the maximal tree depth D. As will be shown in the next section, a practically relevant tree space may have several million dimensions.

Recall the four-dimensional genospace of binary strings with length 4 under bit flipping. There, several genotypes collided in the one-dimensional distance space. Naturally, these collisions may also happen in the case of a tree space G and $D_G \subset \mathbb{N}_0^2$.

Consider, for instance, the trees $/ab$ and $/ba$ in the above sample tree space, which have the same positions. Another example for a collision is given by $\pm d$ and $+ab$.

Recall the above discussion about reducing collisions. Its results apply for computationally tractable tree spaces, since these spaces are finite. For instance, let the *node depth* of a node N in tree t be defined as the length of the shortest path between N and t's root node. The value of an introduced weight factor that is multiplied with the number of applications of "increment", applied to a node N, may depend on the node depth of N.

Note that label codes correspond to weight factors. Thus, proper modification of the coding function c reduces collisions. For the above sample tree space, redefine $c(\pm) = 50, c(+) = 60$, and $\pm d$ and $+ab$ do not collide any more (cf. above).

4 Diversity measure

For an arbitrary genospace G, by means of the above genospace-distance-space mapping π, we propose a measure for genetic diversity of the actual generation g. We then apply this measure to the special case of G being a tree space.

Recall that the basic notion of genetic diversity was that of "many *actual* genotypes being equally distributed over the complete genospace". During evolution, due to the use of some fitness-related selection method, more and more individuals may represent fewer and fewer genotypes. This results in "gaps" in genospace, i.e. areas of "relative emptiness". Following these notions, diversity obviously gets the lower the larger these areas become.

This implies, for instance, that certain structural properties are not found any more in the set of the actual genotypes. Imagine, as an example, that many actual genotypes of the above sample tree space have positions with low coordinate values. Knowing the underlying coding function gives the statement that there are only few actual individuals which represent genotypes with many nodes and expensive functions.

In a *two*-dimensional distance space $D_G \subset \mathbb{R}^2_{0+}$, the largest of the a.m. areas of "relative emptiness" may be approximated by the following rectangle R: let R be axis-parallel to the x-y-axes of $\mathbb{R}^2_{0+}$; let opposite corners of R be given as positions of actual genotypes; let there be no actual genotype whose position is in R; let R be the largest of all such rectangles by area.

In *algorithmic geometry*, considering the positions of all actual genotypes as a given point set in $\mathbb{R}^2_{0+}$, the described rectangle is known as the **largest-area reactangle (LER)**. There are several algorithms known to compute the LER of a given point set in $\mathbb{R}^2$. Especially, there is a very efficient algorithm that only takes $O(n \log^2 n)$ time and $O(n)$ memory [reference]. Without loss of generality, assume that this algorithm returns the bottom-left coordinates and the top-right coordinates of the LER.

Let $area(r)$ denote the area of a rectangle in D_G. D_G itself can be approximated by a minimum-area rectangle $\square_{D_G}, D_G \subset \square_{D_G} \subset \mathbb{R}^2_{0+}$. Let LER_g denote the LER in D_G given by the positions of all genotypes represented by individuals in generation g. We propose

$$\Delta(g) = 1 - \frac{area(LER_g)}{area(\square_{D_G})}$$

as a **diversity measure** on a genospace G. The value of $\Delta(g)$ is the genetic diversity of generation g.

The fewer collisions a given genospace-distance-space mapping $\pi : G \to D_G$ produces the better $\Delta(g)$ approximates the following interpretation. According to the above notion, in the case of high genetic diversity, there are many different genotypes which are well distributed over G. In this case, LER_g is small, and $\Delta(g)$ is close to 1. In the case of low genetic diversity, there are few different genotypes. Consequently, LER_g is large, and $\Delta(g)$ is close to 0.

In order to use the diversity measure on a tree space G, one must compute $area(\square_{D_G})$. This can be done dependent on the terminal and function sets T, F, the origin o_G, the coding function c and the maximal tree depth D.

Take, for instance, the sample tree space given in the previous section. Let D equal 20, what is a practically relevant depth. Since / is the function with largest arity and largest label code and d is the terminal with largest label code, the **maximal tree** t_{max}, which we define as a tree being most distant to o_G according to the structural measure δ, is a tree with depth D that consists of '/'-labeled internal nodes only, and d-labeled terminal nodes only.

The number of t_{max}'s nodes equals $arity^{D+1} - 1 = 2097151$, the label sum equals $arity^D \cdot c(/) + arity^D \cdot c(d) = 4194304$. According to the previous section, these numbers, decremented by one, give the position of t_{max} in D_G. In this sample case, it is easy to see that there is no tree whose position has a higher x- or y-coordinate. Therefore, $\square_{D_G}$ is given by $((0,0)(2097150, 4194303))$, what allows computation of $area(\square_{D_G})$.

In the case of more complex T, F and c it might prove hard to compute $area(\square_{D_G})$ exactly. For a crude approximation, assume all internal nodes to have maximal arity and to carry that internal-node label with maximal label code, assume all terminal nodes to carry that terminal-node label with maximal label code, and use the above formula.

5 Diversity restoring

By means of measures and concepts given in previous sections, we propose a method for restoring genetic diversity of a generation g in the context of a genospace whose distance space is two-dimensional.

Whenever, during runtime of an EA, f generations have been evolved, a **diversity checker** tests by means of diversity measure Δ, if diversity of the actual generation has dropped under a given **diversity threshold** $dt \in]0, 1]$. In this case, *diversity restoring* is invoked.

A set of all those *individuals* of the actual generation which represent the *same* genotype is called the **genetic cluster** of this genotype. Let $\Gamma(g)$ denote the set of all genetic clusters of generation g.

If, for instance, in generation g, each of the individuals i_1, i_2, i_3 represents the genotype i, then $\{i_1, i_2, i_3\}$ is the genetic cluster of i, and $\{i_1, i_2, i_3\}$ is in $\Gamma(g)$.

Diversity restoring must obey some conditions:

1. It must not delete *all* individuals of a genetic cluster, as the underlying genotype represents knowledge gained during evolution.

2. When the EA is a GA, - like GP, for instance - the population size must not be changed.

3. As the *ratios* of the sizes of genetic clusters represent knowledge gained during evolution, these ratios must not be altered significantly.

Because of 1. and because $\Delta(g)$ does *not* depend on the size of genetic clusters, diversity restoring can change diversity only by *creating* z individuals which represent *new* genotypes. These new genotypes are called **diversifiers**.

Thus, when running within a GA, diversity restoring must replace z individuals by diversifiers in order to comply with 2. Those individuals which may be replaced are called **reduncancies**. In order to have diversity restoring comply with 1., all *but one* individual of each genetic cluster are defined to be redundancies. In order to have diversity restoring comply with 3., the number of those redundancies that are actually going to be replaced by diversifiers is defined as

$$\rho(g) = \sum_{C \in \Gamma(g)} \lfloor r|C| \rfloor$$

$r \in]0,1[$ denotes a given *redundancy factor*. $\lfloor s \rfloor$ denotes the highest integer number less than or equal to s.

Since diversity of generation g depends on $area(LER_g)$ only ($area(\square_{D_G})$ is constant for a given genospace), the objective of diversity restoring is to try to reduce $area(LER_g)$, thus incrementing diversity. This can be done as follows.

```
for i:=1 to rho(g)
  begin
  1. compute LER_g
  2. create diversifier t whose position approximates LER_g's  center
  3. delete one redundancy
  end
```

The algorithm for step 1 is given in [reference]. We propose a basic idea for the implementation of step 2.

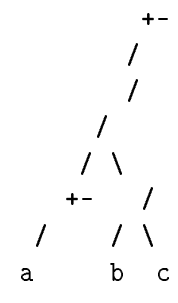
Let a rectangle R be given as $((x_1, y_1), (x_2, y_2))$, with the first tuple element being the left-bottom point and the second one being the top-right point.

The idea behind step 2 is to fill LER_g "best possible", i.e. such that that LER that fits into LER_g after filling has *minimal area*. Let LER_g be given by the positions $((x_j, y_j), (x_i, y_i))$ of two genotypes j, i. Then, LER_g's **center** computes as (*) $\mathbf{center(LER_g)} := (\mathbf{x_c}, \mathbf{y_c}) := (x_j + \frac{x_i - x_j}{2}, y_j + \frac{y_i - y_j}{2})$. Filling LER_g with a point in the center obviously satisfies the above minimal-area condition. Thus, a diversifier must be created whose genotype has $center(LER_g)$ as position.

Adding such nodes to j whose number and label codes add up to $\frac{x_i - x_j}{2}$ and $\frac{y_i - y_j}{2}$, corresponds to "walking" from j's position to LER_g's center (cf. above (*)). Thus, the diversifier in question can be created by adding such nodes to a copy of j.

In general, the center is not in $\mathbb{N}_0^2$, so that no diversifier can be created which has exactly the center as its position. In this case, m nodes $n_1, .., n_m$ must be added to j's copy such that $|n - x_c| + |\sum_i^m n_i|$ is minimal (error minimizing). Then, the position of the diversifier ist closest possible to the center.

As an example for the application of the above method, assume LER_g is given by $(5, 26), (7, 35)$, which are the positions of the sample trees j, i pictured in the previous section. Thus, LER_g's center equals $6, 30.5$. $(\frac{x_i - x_j}{2}, \frac{y_i - y_j}{2}) = (1, 4.5)$ proposes to add one node with label code five to j in order to create a diversifier whose position is close to LER_g's center. Adding one node labeled $\pm$ ($c(\pm) = 5$) gives, for instance, tree k



whose position is $(6, 31)$.

6 GA scheme with diversity restoring

The following scheme shows a standard GA and thus a standard GP with included diversity restoring.

1. creation of initial generation
2. fitness evaluation
→ 7.
3. genetic operations
4. f generations evolvedΓ
   no: → 3.
5. **diversity checking**
   - diversity restoring necessaryΓ
     no: → 2.
6. **diversity restoring**
→ 2.
7. termination criterion satisfiedΓ
   no: → 3.
8. end

Since the structure of an individual's genotype is a quality measure *orthogonal* to the individual's fitness, the fitness of a diversifier, which is an individual created on the basis of structure, may be low in comparison to that of an individual evolved on the basis of fitness. This is especially likely in the final phase of evolution, when diversity often lacks most. Therefore, a diversifier must be given a proper **artificial fitness** for exactly one generation, so that it can mate and reproduce, and thus may further enhance diversity.

For instance, if fitness-proportional selection is used, we propose that a diversifier is given the average fitness of the actual generation, iff. its fitness is below average fitness. In case its fitness is greater or equal average fitness, it will mate and reproduce, anyway.

7 Random seeding vs diversity restoring

As the creation of diversifiers is computationally more expensive as the creation of random individuals, it may be asked if diversity restoring can be reduced to random creation of "diversifiers".

We strongly expect that this is not the case, since random creation does not pay attention to the structural differences of the actual genotypes. It would, for instance, waste some of the "diversifiers", which are limited in number when running a GA, like GP, by placing them close to evolved genotypes instead of placing them, for example, "(almost) precisely in the middle of a large gap", as diversity restoring does. Even when running an EA with theoretically unlimited population size, the practical limits lead to the same conclusion.

The verification of the a.m. expectation is the main objective of further theoretical and empirical research.

8 Conclusions and prospects

Concepts and methods have been introduced which propose extensions of evolutionary algorithms towards explicit control of genetic diversity by means of structural and diversity mea-

sures on arbitrary genospaces. Formal definitions of such measures have been given, which are independent of the genospaces. The dependency on genospaces is limited to the definition of edit operations.

The proposed methods can be readily implemented as a **diversification module**. We are in the process of specifying such a module with a flexible interface for subsequent implementation in order to empirically verify the proposed measures, concepts and methods.

The analysis and application of concepts and measures like the introduced ones is a wide promising area in the field of evolutionary algorithms (*genometry* is proposed as a summarizing term), which has to be explored theoretically and empirically. This area is of practical relevance, because it allows the design of evolutionary paradigms which use an individual's structure as a *quality measure orthogonal to the individual's fitness*, thus potentially allowing a population the break-out of local optima.

The creation of diversifiers as an explicit way of restoring diversity, as it has been proposed in this paper, is only one use of this measure. *Selection* methods, for another instance, can be defined to be fitness- *and structure-based*. For instance, an individual's **quality** can be computed as a function of the individual's structure and fitness.

One could, as one more example, think of *recombination* of parents with certain structural properties in order to have evolution explore those subspaces of the genospace whose genotypes are recombinable by the parents.

Areas of further research are, for instance,

- connections between number of edit operations, weight factors and genospace/distance space dimensionality
- exploration of alternative edit operations and measures on tree space
- connections between problem domain and structural and diversity measures
- efficient algorithms for locating of "gaps" in distance spaces with more than two dimensions

Since binary strings are simple structures in comparison with trees, genospaces of binary strings are of special interest in the context of genometry [WoBa93].

References

[DiGe81] Gernert, Dieter; *Distance or similarity measures which respect the internal structure of the objects*, Methods of operations research, vol. 43; Oelgeschlager, Gunn & Hain, 1981

[JoDi84] Dixmier, Jacques; *General Topology*, Springer, New York, 1984

[JoKo92] Koza, John; *Genetic Programming*, MIT Press, 1992

[VaHo94] Honavar, Vasant; *Toward Learning Systems That Integrate Different Strategies and Representations*, Dept. of Computer Science, Iowa State University;

to appear in *Symbol Processors and Connectionist Networks for Artificial Intelligence and Cognitive Modelling: Steps toward Principled Integration*, Honavar, V.; Uhr, L.; (Ed.); Academic Press, New York, 1994

[WoBa93] Banzhaf, Wolfgang; *Genetic Programming for Pedestrians*, Int. Conference on Genetic Algorithms (ICGA-93), Champaign-Urbana, IL, 1993, S. Forrest (Ed.), Morgan Kaufmann Publishers, San Mateo, CA, 628