A Comparative Analysis of Dynamic Locality and Redundancy in Grammatical Evolution

Eric Medvet

Department of Engineering and Architecture, University of Trieste, Trieste, Italy
emedvet@units.it

Abstract. The most salient feature of Grammatical Evolution (GE) is a procedure which maps genotypes to phenotypes using the grammar production rules; however, the search effectiveness of GE may be affected by low locality and high redundancy, which can prevent GE to comply with the basic principle that offspring should inherit some traits from their parents. Indeed, many studies previously investigated the locality and redundancy of GE as originally proposed in [31]. In this paper, we extend those results by considering redundancy and locality during the evolution, rather than statically, hence trying to understand if and how they are influenced by the selective pressure determined by the fitness. Moreover, we consider not only the original GE formulation, but three other variants proposed later (BGE, πGE, and SGE). We experimentally find that there is an interaction between locality/redundancy and other evolution-related measures, namely diversity and growth of individual size. In particular, the combined action of the crossover operator and the genotype-phenotype mapper makes SGE less redundant at the beginning of the evolution, but with very high redundancy after some generations, due to the low phenotype diversity.

Keywords: Genetic Programming, Diversity, Genotype-phenotype mapping, Genetic operators

1 Introduction

Grammatical Evolution (GE) [31, 19] is a form of grammar-based [14] Genetic Programming (GP) [9] which can evolve programs in an arbitrary language, provided that a context-free grammar (CFG) describing that language is available. The salient and distinguishing feature of GE is that each candidate solution, i.e., each individual, is represented as a bit string (*genotype*) which is transformed into the actual program by means of a mapping procedure. The mapping procedure consumes the genotype in order to select a specific sequence of production rules of the grammar to be used: the outcome (*phenotype*) is a tree whose leaves are non-terminal symbols of the grammar which constitute the program.

GE has been successfully used for approaching a wide range of applications, where the evolved artifacts spanned from actual computer programs, e.g., for

computing the similarity among strings [1], to rules expressed according to a specific language, e.g., for market trading [23], or geometrical shapes complying with predefined constraints [17]. In all cases, GE allowed practitioners and researchers to exploit Evolutionary Algorithms (EAs) without requiring them to adapt some EA aspects to their specific application. In other words, given any grammar and an appropriate fitness function defining a problem, GE allows to (try to) solve that problem.

Despite its widespread adoption, however, GE has been criticized for being, among the many different EAs, a variant which suffers from two issues: low locality [29] and high redundancy [32]. In general, *locality* is the property of an evolutionary algorithm according to which small modifications to an individual correspond to small deviations in its behavior: if this property is not or poorly reflected by an EA variant, that variant is said to have low locality. In GE, locality declines in two ways which reflect the working principle of GE: (a) the degree to which small modifications in the genotype correspond to small modifications in the phenotype, and (b) the degree to which small modifications in the phenotype correspond to small modifications in the fitness. While the latter is mainly determined by the problem itself, i.e., by the language and the fitness function, the former depends on the genotype-phenotype mapping procedure and, in the context of the evolution, on the genetic operators which cause the modifications of the genotype.

Other than locality, which applies to every EA where an individual is assessed by means of a fitness function, redundancy is specific to those EAs where individual representation is someway decoupled, as in GE. In brief, *redundancy* in GE is the property according to which many different genotypes result in the same phenotype. As for the locality of GE mapping procedure, redundancy in GE is not related to the problem being solved, but to the mapping procedure, which in essence can be viewed as a function operating between two spaces (genotype and phenotype) whose injectivity determines the redundancy. Redundancy in GE may be harmful because, at least, it slows down the search and, in case it is not uniform, makes some regions of the search space very unlikely to be reached [32].

Low locality and high redundancy of GE are well known in the EA research community: the fact that they are issues or, rather, features of GE is itself subject of an ample debate. In this paper, we intend to advance the understanding of locality and redundancy in GE by means of two contributions. First, we perform a *comparative* analysis in which contenders are four among the most relevant and recent variants in the genotype-phenotype mapping procedure of GE: the standard original version [31], the Breadth-first variant [4], Position-independent Grammatical Evolution (πGE) [22], and Structured Grammatical Evolution (SGE) [11]. Second, we study locality and redundancy in a *dynamic* context, that is, we observe if and how the locality and redundancy of a given GE variant change during the evolution, as a result of the combined action of several factors, the foremost being—besides the mapping procedure—the operators, the genotype/phenotype size, and the selective pressure. The latter is clearly related to the fitness function and hence this kind of analysis cannot be done abstractly,

i.e., by reasoning on one single problem and extending the findings to the general case. To this end, we experimented with different problems, including some benchmark problems following the guidelines for experimental assessment of GP approaches [13, 35], and a problem which we defined ad hoc in order to facilitate our analysis.

Our experiments show that there is a strong interaction between redundancy/locality and other factors, mainly individual size and population diversity. This interaction manifests after the initial phase of the evolution has ended and hence cannot be observed by performing a static experimentation, i.e., one in which the fitness function does not play any role.

2 Related work

As sketched in the previous section, since the birth of GE in [31], many studies have been carried out about this technique. We here focus on two kinds of works: (a) those explicitly targeting locality/redundancy in GE and (b) those studying other aspects of GE that might impact on locality/redundancy.

The seminal study about locality in GE has been conducted by Rothlauf and Oetzel [29]. The authors focus on the GE genotype-phenotype mapping procedure and start from the long-established principle [3] that if similar solutions do not similarly solve the problem, the optimization performed by an EA tends to resemble a random search. In their study, they define locality in GE as the degree to which neighboring genotypes are also neighboring phenotypes, where two genotypes (phenotypes) are neighbors if their distance is minimal. The authors use the Hamming distance for genotypes and the tree edit distance for phenotypes. In our work, we used the edit distance in the former case, because, in the dynamic scenario, the length of the genotypes may change as a result of operators application, and the Hamming distance is not defined for sequences with different length; moreover, we also explored the use of edit distance (besides tree edit distance) for the phenotypes (see Section 4). In [29], the locality is assessed experimentally by applying one-bit flip mutations to a number of randomly generated genotypes mapped to phenotypes according to a simple grammar. As a side effect, [29] also measures the percentage of cases in which such distance is zero, a figure which represents an estimate of the redundancy. The result is twofold: in 90% of cases a mutation does not change the phenotype and, in the remaining 10% of cases, a significant fraction corresponds to mutations causing heavy changes in the phenotype.

A more recent study [33] by the same research group focused on the locality of search operators and considered both GE and GP. In this case, the key motivation of the study is to assess experimentally to which degree the search operators produce an offspring which is similar to its parent(s). In particular, mutation should guarantee that the distance between the offspring and its parent is bounded, whereas crossover should guarantee that the distance of the offspring from its farthest parent is lower than the distance between the two parents—this principle being introduced previously [10] and later denoted as "locality of

search operators" [27] or "geometry of search operators" [15]. As in [29], the locality is assessed experimentally by measuring distances resulting from operator application during a random search—i.e., without applying the selective pressure given by the fitness, as we do in our present work. The result of the cited paper is that both GE and GP standard operators suffers from low locality: however, in GE locality is slightly greater.

A different point of view on how GE mutation and crossover affect phenotypes is given by [2]. The authors aim at investigating the destructive (or constructive) nature of those operators and, in particular, to verify if it is related to the position in which the operator actually applies: in facts, due to the mapping procedure of GE, modifications to the leftmost bits of a genotype are expected to cause changes close to the root of the tree in the phenotype. Differently than aforementioned works, in [2] the focus is not on locality as defined before, but rather on whether an operator application results in an increased (constructive) or decreased (destructive) fitness: however, the authors do not study if and how that figure varies during the evolution, as we do in the present study with locality. The result is that genotype modifications which occur at leftmost positions are in general more destructive on the phenotype—as expected—but, at the same time, may result in the largest increments in the fitness.

The locality of different genotype representations, here intended as the way in which integer are codified by bits slices (codons) in the genotype, is studied in [6, 7]. The authors compare experimentally the binary and grey code representations and their combined effect on locality with 4 different mutation operators. They find that the standard combination is not improved by other options, neither in locality nor in effectiveness: moreover, this result induces the authors to wonder if locality is indeed beneficial to GE performance.

Since the locality issue in GE has been often attributed to the genotype-phenotype procedure, it is not surprising that improvements to that procedure have been proposed which try to tackle the issue. This trend has also been studied recently in [34], where the authors noted how GE, which was born as a purely grammar-agnostic approach, slowly incorporated information from the grammar into crossover, mutation, and individual initialisation, blurring the distinction between genotype and phenotype.

A first comparative analysis of different genotype-phenotype mappings has been conducted in [4], including the standard GE mapping, its breadth-first variant, and πGE. The experimental campaign showed an advantage of the latter, but the authors themselves acknowledged that more research was needed to fully comprehend the reasons of their findings.

Recently, and later than the previously cited works, a novel variant of GE, called Structural GE (SGE), has been introduced [11] (see Section 3.4). SGE mapping procedure has been designed with high locality and low redundancy as first-class goals and its experimental evaluation on a set of benchmark problems showed that it outperformed the standard GE. The good performance of SGE has been later analysed in [12], where several experiments are described, including some concerning locality and redundancy. The authors find that SGE has a better

locality than GE and argue that, together with lower redundancy, this is among the key factors for the superiority of their approach. In this present study, we included SGE in our comparative analysis.

Reduncancy in GE has been the focus of fewer works than locality—more works, instead, considered redundancy in other EAs or in general [28]. In a very recent study [32], redundancy in GE has been characterized beyond its simple quantification. The author analyzed the entire solution space resulting from two simple grammars and a fixed genotype length. She concludes that the standard GE mapping procedure is strongly non-uniformly redundant, i.e., some phenotypes are mapped from many genotypes whereas other phenotypes are mapped by few genotypes.

3 GE variants

We compared four variants of GE in our analysis: standard GE, breadth-first GE, πGE, and SGE. Three of them differ only in the mapping procedure, whereas the fourth and most recent (SGE) also bases on different versions of the genetic operators. We did not include in the analysis few other GE variants we were aware of (such as, e.g. [30, 8]), because they introduced minor changes over standard GE or did not result in relevant improvements. We present the 4 variants in the following sections, detailing in particular the genotype-phenotype mapping procedure.

We recall that, for all GE variants, the mapping procedure maps a genotype g into a phenotype p using the production rules of a CFG $\mathcal{G} = (N, T, s_0, R)$, where N is the non-empty set of non-terminal symbols, T is the non-empty set of terminal symbols, $s_0 \in N$ is the starting symbol (or axiom), and R is the set of production rules. The phenotype p can be either viewed (a) as a tree whose leaves are elements of T that, visited from left to right, are a string of the language $\mathcal{L}(\mathcal{G})$ defined by $\mathcal{G}$; or, more simply, as (b) a string of the language $\mathcal{L}(\mathcal{G})$.

3.1 Standard GE

In standard GE [31], the genotype is a variable-length bit string: it can be read as a sequence of integers by grouping n consecutive bits, each integer being called *codon*. The value of n is a parameter of the mapping procedure and should be large enough so that 2^n is greater or equal than the maximum number of options in the rules of the grammar. Conventionally, it is set to $n = 8$, but in some applications (e.g., [1]) it has been set to the lowest value which met the above criterion.

The mapping procedure of GE works as follows. Initially, the phenotype is set to $p = s_0$, i.e., to the grammar starting symbol, the counter i is set to 0, and the counter w is set to 0. Then, the following steps are iterated. (1) Expand the leftmost non-terminal s in p by using the j-th option (zero-based indexing) in the production rule r_s for s in $\mathcal{G}$. The value of j is set to the remainder of the division between the value g_i of the i-th codon (zero-based indexing) and the

number $|r_s|$ of options in r_s, i.e., $j = g_i \mod |r_s|$. (2) Increment i; if i exceeds the number of codons, i.e., if $i > \frac{|g|}{n}$, then set i to 0 and increment w—the latter operation is called wrapping and w represents the number of wraps performed during the mapping. (3) If w exceeds a predefined threshold n_w, then abort the mapping, i.e., produce a *null phenotype* and set its fitness to the worst possible value. (4) If p contains at least one non-terminal to be expanded, return to step 1, otherwise end.

With wrapping the genotype may be reused: each codon may hence concur in defining the expansion of different non-terminals—a feature which has been called polymorphism [18]. Wrapping has been included in order to make GE work with any CFG, including recursive grammars which, in turn, correspond to languages containing non-finite strings. On the other hand, the upper bound n_w to the number of wrapping operations is imposed to avoid an endless mapping. Clearly, the choice of the size $|g|$ of the genotype plays a crucial role in this respect and cannot be determined easily. In some application, such as in [1], $|g|$ is chosen such that one or more target phenotypes can be generated without resorting to wrapping.

It is straightforward to relate the role of the rule option choice driven by a module operation ($j = g_i \mod |r_s|$) with the redundancy. For instance, in a genotype for which the mapping does not result in any wrapping, all the values of a codon for which the remainder of the division remains the same correspond to the same phenotype. Moreover, if the genotype g is such that only the first k codons are used in the mapping, then any genotypes g' that differs from g in codons whose index is larger than k will correspond to the same phenotype of g.

Evolution in GE is commonly driven by GA, but other approaches have been proposed [24], such as Particle Swarm Optimization (PSA) [21] and Differential Evolution (DE) [16]. In this work, we consider a classical GA configuration in which the genetic operators are the bit flip mutation (i.e., each bit in the phenotype may be flipped according to a predefined probability p_{mut}) and the one-point crossover. Among the many options for the two operators (mutation and crossover), we chose the two mentioned because they can result in genotype modifications of varying impact, which is functional to our analysis of locality. Concerning crossover, it has been shown that one-point crossover is not significantly worse than other options [20, 5].

3.2 Breadth-first GE

Breadth-first GE (we here denote it with BGE) has been introduced in [4] and only slightly differs from standard GE. In the iterative mapping procedure (at step 1), instead of choosing the leftmost non-terminal in p, the least deep (closest to the tree root) non-terminal is chosen.

Despite experiments in [4] show that BGE is not significantly better or worse than GE, we include it in our analysis because its underlying mapping mechanism might suggest that codon positions have a different role in determining the tree: while in standard GE first codons concur in building the "left" portion of the tree,

up to the leaves, in BGE first codons concur in building the "higher" portion of the tree (i.e., the root and its immediate descendant). This difference could in turn result in a different impact on locality.

All the other approach components of BGE—namely the operators—are the same of standard GE. The same considerations concerning redundancy in standard GE apply also to BGE.

3.3 πGE

A significant modification has been proposed to the standard GE mapping procedure by O. Neill et al. in [22], named Position-independent GE (πGE). The salient feature of πGE is that it decouples the choice of the non-terminal to be expanded from the choice of the specific expansion (i.e., option in the rule). The main idea behind this design choice is, according to authors, to break the codon dependency chain of the mapping in order to allow the arising, in the genotype, of short codon sequences which should act as building blocks, as subtrees do in GP. Despite the fact that the cited work did not provide any strong evidence of the actual arising of such building blocks, the experimental campaign showed that πGE indeed delivered better results than GE in the majority of the problems analysed in [22]. That result was further confirmed later in [4].

Differently than in GE, in πGE each codon consists of a pair $g_i^{\text{nont}}, g_i^{\text{rule}}$ integers, each of n bits—conventionally, n is set to 8. In the mapping procedure (at step 1), the non-terminal of p to be expanded is not chosen statically (i.e., the leftmost in GE or the closest to the root in BGE) but is instead chosen as follows: let be n_s the number of non-terminals in p, then the j^{nont}-th one is chosen, where $j^{\text{nont}} = g_i^{\text{nont}} \mod n_s$. Then, the rule option to be used is determined, as in standard GE, with $j^{\text{rule}} = g_i^{\text{rule}} \mod |r_s|$. In other words, in πGE it is the genotype itself which encodes the positions of the phenotype which are to be expanded and those positions evolve independently from corresponding contents.

As for BGE, operators and implications on redundancy do not change in πGE w.r.t. standard GE.

3.4 SGE

The most recent GE variant which we consider in our comparison is Structural GE (SGE), which has been first described in [11] and then more deeply analyzed in [12]. The key idea behind SGE is to structure the mapping such that each codon corresponds to exactly one expansion of a given non-terminal. The aim of this idea is, according to the authors, to increase the locality, since "the modification of a single gene [codon] does not affect the derivation options of other non-terminals". Moreover, SGE introduce two other relevant variations to the standard GE: (i) it redefines the genotype structure which results in a remarkable reduction in redundancy and (ii) it applies only to non-recursive grammars which results in the guarantee of mapping any genotype to a non-null phenotype (differently from GE, BGE, and πGE). The latter point may appear as a strong limitation in the approach applicability, since for most problems of

practical interest the grammar is recursive. However, the authors of [11] state that the limitation should not severely affect SGE practicality and recall that in many other EAs there is an upper bound to the size of an individual (e.g., in GP). They also provide the sketch of a procedure for transforming any possibly recursive grammar G into a non-recursive grammar G' by imposing a maximum tree depth: it is yet not fully clear how that parameter should be set in problems where the tree depth of an acceptable solution is not known in advance and, from another point of view, if and how its value affects SGE effectiveness in searching for a solution.

Differently than in previously presented variants, in SGE the genotype g is a fixed-length integer string. The string is itself composed of $|N|$ substrings, i.e., one substring g_s for each non-terminal $s \in N$ of the grammar G. The length of (i.e., number of codons in) each substring g_s is determined by the maximum number of expansions which can be applied to the corresponding non-terminal s according to the non-recursive grammar G'. Moreover, the domain of each codon is set to $\{0, \dots, |r_s| - 1\}$, where r_s is the production rule for s. It follows that the redundancy in SGE should be reduced w.r.t. GE, BGE, and πGE, since the modulo is not needed: the only redundancy which remains is the one related to the possibility that some codons are not used in the mapping because of the small size of the phenotype.

The mapping procedure of SGE works as follows. Initially, the phenotype is set to $p = s_0$, and a counter i_s for each non-terminal $s \in N$ is set to 0. Then, the following steps are iterated. (1) Expand the leftmost non-terminal s in p by using the g_{s,i_s}-th option (zero-based indexing) of the rule r_s, g_{s,i_s} being the value of the i_s-th codon (zero-based indexing) in g_s. (2) Increment i_s. (3) If p contains at least one non-terminal to be expanded, return to step 1, otherwise end. It can be noted that, in step 1, there is no need to use the modulo on the codon value, since its domain is $\{0, \dots, |r_s| - 1\}$.

The operators in SGE are peculiar to that approach. The mutation is a probabilistic (according to a parameter p_{mut}) codon mutation which is guaranteed to set a random new codon value which stays in the proper domain. The crossover works as follows: given two genotypes g^1 and g^2, a subset $N' \subseteq N$ of non-terminals is randomly chosen; then, for each $s \in N'$, the two corresponding genotype substrings g^1_s, g^2_s are swapped.

4 Experiments and results

Our aim is to study if and how locality and redundancy vary during the evolution across different GE variants; as a secondary goal, we are interested in studying the tendency to generate null phenotypes. We here define the *indexes* by means of which we measured these concepts; coping with a dynamic scenario, we cast definitions considering measures to be collected at each generation. We denote with $d_g(i_1, i_2)$ and $d_p(i_1, i_2)$ the genotype and phenotype distance among two individuals i_1, i_2, respectively.

- *Invalidity.* The percentage of not-*valid* individuals being generated, i.e., those in which the phenotype is null.
- *Redundancy.* The percentage of valid individuals i being generated in which the genotype of i is different from both parent genotypes and the phenotype of i is equal to at least one of the two parent phenotypes.
- *Locality.* The correlation between the genotype distance from the closest parent $\min(d_g(i, i_1), d_g(i, i_2))$ and the phenotype distance from the closest parent $\min(d_p(i, i_1), d_p(i, i_2))$, considering valid individuals for which both distances are not zero—we used Pearson correlation. The rationale for this index is to generalize the high level definition of locality as a measure of the degree to which small modifications in the genotype correspond to small modification in the genotype. From another point of view, our definition extends the one introduced in [26] (and later used in [12]) which, basically, corresponds to the mean of $\min(d_p(i, i_1), d_p(i, i_2))$, measured separately for mutation and crossover and called Mutation Innovation (MI) and Crossover Innovation (CI), respectively.

All of the 4 indexes can be measured w.r.t. a specific operator or globally: we present the results for invalidity globally, and for each operator for locality and redundancy.

We considered 4 problems: 3 are established benchmark problems (Harmonic regression, Polynomial regression, and Santa-Fe trail—see, e.g., [12] for the corresponding grammars) whereas the last (Text) is a problem that we designed specifically for this analysis.

- *Harmonic.* This is a symbolic regression problem in which the goal is to approximate the function $f(x) = \sum_i^x \frac{1}{i}$ in the x values $\{1, \ldots, 50\}$. The fitness is given by the sum of absolute errors, to be minimized.
- *Polynomial.* This is another symbolic regression problem in which the goal is to approximate the function $f(x) = x^4 + x^3 + x^2 + x$ in the x values $\{-1, -0.9, \ldots, 0.9, 1\}$. The fitness is given by the sum of absolute errors, to be minimized.
- *Santa-Fe.* This is a classic path-finding problem which consists in finding a program able to guide an artificial ant to reach 89 statically placed food items in a 32×32 grid given a maximum number of steps. The fitness is given by the number of missed food items, to be minimized.
- *Text.* The goal of this problem is to generate a string which matches a statically defined target, which was `Hello world!` in our experimentation. The fitness is given by the edit distance between the string encoded by the individual and the target string. The rationale for including this problem is to (try to) neutralize the influence of the phenotype to fitness mapping: in facts, the phenotype distance between individuals corresponds (when using the edit distance for phenotypes) by definition to their fitness distance. Figure 1 shows the grammar for this problem.

For each problem and each GE variant, we performed 30 different runs, with the settings shown in Table 1—for SGE, we experimented with two values for

```
     <text> ::= <sentence> ␣ <text> | <sentence>
 <sentence> ::= <Word> ␣ <sentence> | <word> ␣ <sentence> | <word> <punct>
     <word> ::= <letter> <word> | <letter>
     <Word> ::= <Letter> <word>
   <letter> ::= <vowel> | <consonant>
    <vowel> ::= a | e | i | o | u
<consonant> ::= b | c | d | ... | z
   <Letter> ::= <Vowel> | <Consonant>
    <Vowel> ::= A | E | I | O | U
<Consonant> ::= B | C | D | ... | Z
    <punct> ::= ! | ? | .
```

Fig. 1. The CFG for the Text problem.

the max tree depth. In each run, after each generation, we measured the 3 indexes defined above. For the locality, we used as d_g the edit distance (for GE, BGE, and πGE) and the Hamming distance (for SGE) and as d_p both the edit distance (between phenotypes viewed as strings of $\mathcal{L}(\mathcal{G})$) and the tree edit distance (between phenotypes viewed as trees and computed using the method of [25]) for all the variants. For brevity, we here present results only for the tree edit distance, but in essence they do not change when considering the edit distance, which is also much faster to compute.

Table 1. Settings for the experimental analysis.

	GE, BGE, πGE	SGE
Population	500	500
Pop. initialization	Random	Random
Generations	50	50
Crossover rate	0.8	0.8
Crossover operator	One-point	SGE crossover
Mutation rate	0.2	0.2
Mutation operator	Bit flip with $p_{\text{mut}} = 0.01$	SGE mutation with $p_{\text{mut}} = 0.02$
Selection	Tournament with size 5	Tournament with size 5
Replacement	$m + n$ strategy with $m = n$	$m + n$ strategy with $m = n$
Initial genotype size	1024	n.a.
Max wraps	5	n.a.
Max tree depth	n.a.	$6, 10$

4.1 Results and discussion

Figure 2 presents the results in form of several plots. There is one column of plots for each problem and one row of plots for each index. In each plot, the x axis represents the generation, the y axis represents the index: there is one curve for each GE variant. The curves are obtained by averaging the measures collected at each generation across the 30 runs. The figure is the result of the evaluation of an overall $\approx 14\,700\,000$ operator applications.

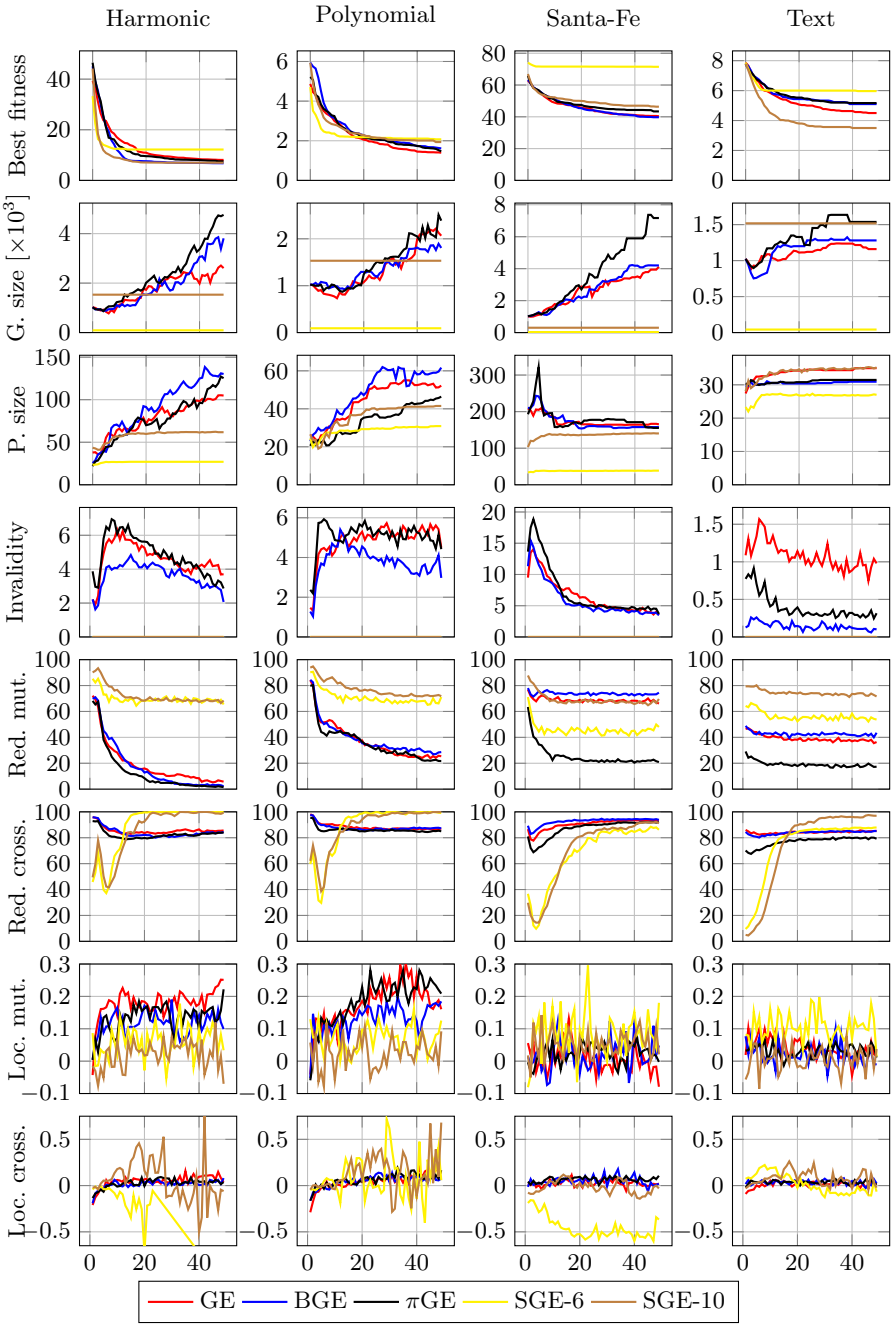

Fig. 2. Results: one plot row per index, one plot column per problem, one curve per GE variant.

Context: fitness and individual size. In order to provide some context for the investigation about invalidity, locality, and redundancy, the first 3 plot rows of Figure 2 show the fitness, genotype size, and phenotype size of the best individual. The plots about best fitness show that (in our configuration, which, we remark, has been chosen to facilitate our analysis rather than for obtaining good fitness values) GE, BGE, and πGE exhibit similar behavior. Concerning SGE, there is a difference between the two values of maximum depth: SGE-10 is significantly better ($p \leq 0.001$ with the Mann-Whitney test) than all the other variants on the Text problem, whereas SGE-6 is significantly worse than all the others on the Santa-Fe problem. This difference suggests that the choice of the maximum depth may be crucial.

The plots about the genotype size of the best individual show that for GE, BGE, and πGE, the index increases during the evolution, roughly linearly, for all but the Text problem—for SGE, the genotype size is fixed by design. For the Text problem, the genotype size tends to remain constant after some tens of generations and is, in general, smaller than for the other problems at the end of the evolution. Concerning phenotype size, in GE, BGE, and πGE it increases for the two symbolic regression problems, whereas it stabilizes after few generations for the other two problems: this can be explained by the fact that, for Santa-Fe and Text, large phenotype sizes may negatively affect the fitness (more instructions, longer text), whereas more complex expressions may not result in less accurate regression than simple expressions. Differently, with SGE the phenotype early reaches a stable size which is, for all but the Text problem, lower than for the other GE variants, being always lower for SGE-6 than for SGE-10. We analyzed the runs in order to better understand this behavior and we found that SGE suffers (in our experimentation) of a low phenotype diversity, measuread ratio between the number of unique phenotypes and the population size. A detailed analysis about diversity in different variants of GE is beyond the scope of this paper, but we speculate that low diversity likely limits the performance of SGE: indeed, it can be seen in Figure 2 that the fitness curves for SGE become and remain flat after few generations in all the problems for which other variants evolve a larger phenotype. More in general, we think that diversity, redundancy, and locality interact.

Invalidity. The 4th row of plots in Figure 2 shows the invalidity, i.e., the tendency to generate null phenotypes. We recall that, by design, it never happens in SGE that a genotype is not mapped to a valid phenotype. Concerning the other three mappers, it can be seen that for three on four problems (all but Text), invalidity tends to set around 5%, slowly decreasing during the evolution: at the beginning of the evolution, for Santa-Fe, GE, BGE, and πGE exhibit a much larger invalidity ($\approx 15\%$). In the Text problem, the figure is much lower ($\leq 1.5\%$) and the differences among the three mappers are sharper, BGE being the one with the lowest values. We conjecture that there is some dependency between the invalidity and the complexity of the problem grammar: in particular, in the

Text grammar, the production rules with the largest number of options are those including only non-terminals, differently from the other problems.

Redundancy. The bottommost 4 rows of plots in the figure show the most salient results, i.e., redundancy and locality indexes measured separately for the two operators (mutation and crossover). The redundancy for mutation is very high at the beginning of the evolution for all but the Text problem—between 60% and 90%: this result is somewhat consistent with the findings of [29] which, however, are limited to a static scenario (our generation zero). In regression problems and for GE, BGE, and πGE, redundancy decreases during the evolution, reaching very low ($\leq 10\%$) or low ($\leq 30\%$) values—however, no significant differences are noticeable among the three variants. Notably, πGE exhibits a lower redundancy both in Santa-Fe and Text, where GE and BGE behave similarly. Mutation redundancy curves related to SGE show instead a diverse shape: the decreasing is much less important in regression problems and absolute values are in general higher. Moreover, SGE-6 exhibits in general a lower mutation redundancy than SGE-10, which is indeed consistent with the fact that the genotype is much smaller for the former—there are hence fewer codons which do not take part in the mapping.

We think that the findings about mutation redundancy can be explained mainly in terms of the phenotype size. When, at the beginning of the evolution, the phenotypes are small, the redundancy is high for all the GE variants because a large part of the genotype is not expressed (i.e., does not play any role in the mapping procedure); as far as the phenotypes grow, the mutation redundancy settles down to lower values. Since in SGE the phenotype size early reaches and maintains rather small values, redundancy cannot be strongly reduced. From another point of view, we think that mutation redundancy is only marginally related to the modulo and is instead more determined by unexpressed codons—a condition which occurs in all variants.

Concerning crossover, it can be seen that there are two clearly different phenomena. For GE, BGE, and πGE, redundancy tends to remain around 80%–90% in all cases (with πGE being always slightly less redundant). We think that this behavior is motivated by how one-point crossover works: the rightmost portions of the two parents are swapped in the offspring; if, however, in the mapping procedure the new genotype portion is never used, the child phenotype is equal to the first parent phenotype. For SGE, instead, the redundancy is initially always lower than for the other GE variants (initial values are consistent with those found in [12]), but quickly increases reaching very high values, close to 100% in almost all cases: moreover, no significant differences exist between SGE-6 and SGE-10. We justify the low initial values by the different working principle of SGE crossover and SGE mapping. With respect to the subsequent very high redundancy values, we think they are a consequence of the very low phenotype diversity in the population after some tens of generations. That is, no matter how good the operator or the mapping procedure are in avoiding redundancy by design if they work on identical or almost identical parents.

Locality. Finally, the last two rows of plots in Figure 2 show the locality for the two operators. Looking at these plots, no clear conclusions can be drawn concerning locality. However, a different behavior of GE, BGE, and πGE, on one side, and SGE, on the other, can be observed.

Concerning GE, BGE, and πGE, the locality is always rather low, with a correlation which, in the best case, scores 0.3. Moreover, no significant trend can be observed during the evolution: only for the Polynomial problem some general tendency to increase locality may be observed during the first half of the evolution.

For SGE, the locality greatly varies during the evolution, sometimes being greater than for the other three variants, and often being negative—i.e., the smaller d_g, the larger d_p—which may appear surprising. We analyzed the collected measures in details and found that the values for d_p and d_g were much smaller for SGE than for the other variants, due to the low diversity in the population (shorter distances), and, in general, assumed few different values, hence causing a very unstable trend of the curve. We motivate this finding as follows: if the population does not provide the crossover with parents which are sufficiently different, the operator fails in generating a new individual which is somewhat midway between the two parents. From a more general point of view, we think that despite the fact that in principle SGE should exhibit a greater locality, the actual presence of such property may be frustrated in actual conditions by other factors, in particular by low diversity.

5 Concluding remarks and future work

We considered 4 different variants of GE, including the most recent SGE, and performed an experimental campaign aimed at understanding to which degree locality and redundancy properties are exhibited by those variants during the evolution, rather than statically. To this end, we considered 4 problems, including 3 benchmark problems commonly used in GP and GE approaches assessment, and measured redundancy (by means of an established definition) and locality (for which we propose a definition which generalizes previous definitions) after each generation in 30 independent runs for each problem/variant.

According to our experimental results, previous findings about redundancy and locality in a static scenario are partially confirmed. However, we also found that, considering the dynamic scenario occurring during the evolution, there is a strong interaction between individual size and redundancy, which may positively affect the latter in variable size GE variants (GE, BGE, πGE). Moreover, locality appears to be influenced by the diversity in the population, which may hamper, in particular, the by-design ability of SGE mapping procedure and operators to favor an high locality.

Acknowledgements

The author is grateful to Alberto Bartoli and Fabio Daolio for their insightful comments.

References

1. Bartoli, A., De Lorenzo, A., Medvet, E., Tarlao, F.: Syntactical similarity learning by means of grammatical evolution. In: International Conference on Parallel Problem Solving from Nature. pp. 260–269. Springer International Publishing (2016)
2. Castle, T., Johnson, C.G.: Positional effect of crossover and mutation in grammatical evolution. In: European Conference on Genetic Programming. pp. 26–37. Springer (2010)
3. Doran, J.E., Michie, D.: Experiments with the graph traverser program. In: Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences. vol. 294, pp. 235–259. The Royal Society (1966)
4. Fagan, D., O'Neill, M., Galván-López, E., Brabazon, A., McGarraghy, S.: An analysis of genotype-phenotype maps in grammatical evolution. In: European Conference on Genetic Programming. pp. 62–73. Springer (2010)
5. Harper, R., Blair, A.: A structure preserving crossover in grammatical evolution. In: 2005 IEEE Congress on Evolutionary Computation. vol. 3, pp. 2537–2544. IEEE (2005)
6. Hugosson, J., Hemberg, E., Brabazon, A., O'Neill, M.: An investigation of the mutation operator using different representations in grammatical evolution. In: Proc. 2nd International Symposium Advances in Artificial Intelligence and Applications. vol. 2, pp. 409–419 (2007)
7. Hugosson, J., Hemberg, E., Brabazon, A., O'Neill, M.: Genotype representations in grammatical evolution. Applied Soft Computing 10(1), 36–43 (2010)
8. Keijzer, M., O'Neill, M., Ryan, C., Cattolico, M.: Grammatical evolution rules: The mod and the bucket rule. In: European Conference on Genetic Programming. pp. 123–130. Springer (2002)
9. Koza, J.R.: Genetic programming: on the programming of computers by means of natural selection, vol. 1. MIT press (1992)
10. Liepins, G.E., Vose, M.D.: Representational issues in genetic optimization. Journal of Experimental & Theoretical Artificial Intelligence 2(2), 101–115 (1990)
11. Lourenço, N., Pereira, F.B., Costa, E.: Sge: a structured representation for grammatical evolution. In: International Conference on Artificial Evolution (Evolution Artificielle). pp. 136–148. Springer (2015)
12. Lourenço, N., Pereira, F.B., Costa, E.: Unveiling the properties of structured grammatical evolution. Genetic Programming and Evolvable Machines pp. 251–289 (2016)
13. McDermott, J., White, D.R., Luke, S., Manzoni, L., Castelli, M., Vanneschi, L., Jaskowski, W., Krawiec, K., Harper, R., De Jong, K., et al.: Genetic programming needs better benchmarks. In: Proceedings of the 14th annual conference on Genetic and evolutionary computation. pp. 791–798. ACM (2012)
14. Mckay, R.I., Hoai, N.X., Whigham, P.A., Shan, Y., O'Neill, M.: Grammar-based genetic programming: a survey. Genetic Programming and Evolvable Machines 11(3-4), 365–396 (2010)
15. Moraglio, A., Poli, R.: Topological interpretation of crossover. In: Genetic and Evolutionary Computation Conference. pp. 1377–1388. Springer (2004)
16. O'Neill, M., Brabazon, A.: Grammatical differential evolution. In: IC-AI. pp. 231–236 (2006)
17. O'Neill, M., McDermott, J., Swafford, J.M., Byrne, J., Hemberg, E., Brabazon, A., Shotton, E., McNally, C., Hemberg, M.: Evolutionary design using grammatical evolution and shape grammars: Designing a shelter. International Journal of Design Engineering 3(1), 4–24 (2010)

18. O'Neill, M., Ryan, C.: Automatic generation of programs with grammatical evolution. Proceedings of AICS pp. 72–78 (1999)
19. O'Neill, M., Ryan, C.: Grammatical evolution. IEEE Transactions on Evolutionary Computation 5(4), 349–358 (2001)
20. O'Neill, M., Ryan, C., Keijzer, M., Cattolico, M.: Crossover in grammatical evolution. Genetic programming and evolvable machines 4(1), 67–93 (2003)
21. O'Neill, M., Brabazon, A.: Grammatical swarm: The generation of programs by social programming. Natural Computing 5(4), 443–462 (2006)
22. O'Neill, M., Brabazon, A., Nicolau, M., Mc Garraghy, S., Keenan, P.: πgrammatical evolution. In: Genetic and Evolutionary Computation Conference. pp. 617–629. Springer (2004)
23. O'Neill, M., Brabazon, A., Ryan, C., Collins, J.: Evolving market index trading rules using grammatical evolution. In: Workshops on Applications of Evolutionary Computation. pp. 343–352. Springer (2001)
24. O'Sullivan, J., Ryan, C.: An investigation into the use of different search strategies with grammatical evolution. In: European Conference on Genetic Programming. pp. 268–277. Springer (2002)
25. Pawlik, M., Augsten, N.: Efficient computation of the tree edit distance. ACM Transactions on Database Systems (TODS) 40(1), 3 (2015)
26. Raidl, G.R., Gottlieb, J.: Empirical analysis of locality, heritability and heuristic bias in evolutionary algorithms: A case study for the multidimensional knapsack problem. Evolutionary Computation 13(4), 441–475 (2005)
27. Rothlauf, F.: Design of modern heuristics: principles and application. Springer Science & Business Media (2011)
28. Rothlauf, F., Goldberg, D.E.: Redundant representations in evolutionary computation. Evolutionary Computation 11(4), 381–415 (2003)
29. Rothlauf, F., Oetzel, M.: On the locality of grammatical evolution. In: European Conference on Genetic Programming. pp. 320–330. Springer (2006)
30. Ryan, C., Azad, A., Sheahan, A., O'Neill, M.: No coercion and no prohibition, a position independent encoding scheme for evolutionary algorithms–the chorus system. In: European Conference on Genetic Programming. pp. 131–141. Springer (2002)
31. Ryan, C., Collins, J., O'Neill, M.: Grammatical evolution: Evolving programs for an arbitrary language. In: European Conference on Genetic Programming. pp. 83–96. Springer (1998)
32. Thorhauer, A.: On the non-uniform redundancy in grammatical evolution. In: International Conference on Parallel Problem Solving from Nature. pp. 292–302. Springer (2016)
33. Thorhauer, A., Rothlauf, F.: On the locality of standard search operators in grammatical evolution. In: International Conference on Parallel Problem Solving from Nature. pp. 465–475. Springer (2014)
34. Whigham, P.A., Dick, G., Maclaurin, J., Owen, C.A.: Examining the best of both worlds of grammatical evolution. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation. pp. 1111–1118. ACM (2015)
35. White, D.R., Mcdermott, J., Castelli, M., Manzoni, L., Goldman, B.W., Kronberger, G., Jaśkowski, W., O'Reilly, U.M., Luke, S.: Better gp benchmarks: community survey results and proposals. Genetic Programming and Evolvable Machines 14(1), 3–29 (2013)