Approximating Geometric Crossover in Semantic Space

Krzysztof Krawiec and Paweł Lichocki
Institute of Computing Science
Poznan University of Technology
Piotrowo 2, 60965 Poznań, Poland
kkrawiec@cs.put.poznan.pl,
lichocki@man.poznan.pl

ABSTRACT

We propose a crossover operator that works with genetic programming trees and is approximately geometric crossover in the semantic space. By defining semantic as program's evaluation profile with respect to a set of fitness cases and constraining to a specific class of metric-based fitness functions, we cause the fitness landscape in the semantic space to have perfect fitness-distance correlation. The proposed approximately geometric semantic crossover exploits this property of the semantic fitness landscape by an appropriate sampling. We demonstrate also how the proposed method may be conveniently combined with hill climbing. We discuss the properties of the methods, and describe an extensive computational experiment concerning logical function synthesis and symbolic regression.

Categories and Subject Descriptors: I.2.8 [Problem Solving, Control Methods, and Search]: Heuristic methods

General Terms: Algorithms

Keywords: Geometric Crossover, Program Semantics, Global Convexity, Fitness-distance Correlation

1. INTRODUCTION

Geometric crossover, also known as *topological crossover*[9], under metric d is a binary genetic operator with the property that each offspring lies in the segment between its parents in the metric space induced by d in the population of individuals. Geometric crossover has been intensively studied for genetic programming (GP) as well as for vector-based solution representations. In [9], Moraglio and Poli introduced a representation-independent geometric generalization of crossover and mutation for binary strings and real vectors. The usefulness of crossover for the all-pairs shortest path problem using EA has been recently proven theoretically [3]. Other related concepts, like distance preserving crossover and respectful crossover have been also proposed and studied [8].

GECCO'09, July 8–12, 2009, Montréal Québec, Canada.

The lure of geometric crossover, though not always explicitly expressed in the publication, comes from at least two factors. Firstly, geometric crossover does what crossover is supposed to do, i.e., produces an offspring that preserves some common properties of its parents. Secondly, the offspring produced by geometric crossover is likely to benefit from fitness-distance correlation (also known as *big valley*) [15]. Fitness-distance correlation (*fdc*) is the property of the fitness landscape that may be shortly characterized as significant correlation of the value of the fitness function f and the distance from the global optimum. Technically, this correlation is strongly positive for minimized fitness function, and strongly negative for maximized fitness function; in following we assume f is minimized. Though formally *fdc* is a *measure* of the fitness landscape, in following we refer to it as a property of the problem.

Fitness landscapes with *fdc* are often (though not necessarily) unimodal and approximately convex, hence yet another name of this property, *global convexity*. The word 'global' is supposed to point out that the convexity does not have to hold locally, but at a sufficiently large scale, the landscape appears convex.

Given a problem with strictly convex fitness landscape, geometric crossover is guaranteed to produce an offspring with fitness that is not worse than the worse of its parents. Moreover, the offspring is likely to outperform both parents if they happen to occupy the opposite slopes of the same ridge of the fitness landscape. Of course, such strict convexity is not common among real-world problems, and, for globally convex problems, these observations have to be stochastically relaxed. Nevertheless, for such problems the probability of producing a well-performing offspring using geometric crossover is higher than in case of problems with low *fdc*.

It should be emphasized that global convexity is not a property of the problem alone, but depends also on the metric applied to the solution space. In theory, given any enumerable solution space, it is possible to reorder the solutions (and, thus, define a metric) in such a way that the fitness landscape spanned over them becomes convex. Similarly, the choice of the metric is also critical for the geometricity of the crossover operator.

The dependence of the convexity of the fitness landscape both on the space as well as on its metric is of crucial importance for the method presented in this paper. In particular, we demonstrate how to benefit from the global convexity of the space of semantics of GP programs.

2. CONVEX FITNESS LANDSCAPES OF PROGRAM SEMANTICS

One of major difficulties that hinders our understanding of GP is the gap that spans between the *genotype* (the symbolic encoding) of the evolved program and its *phenotype* (what the program is actually doing). The domain-specific knowledge embedded in GP function set, terminals, data types, etc., make the genotype-phenotype mapping extremely complex and difficult to generalize across different application areas. This issue has been raised multiple times in past literature on the topic, with Rothlauf's work on locality of genotype-phenotype mapping being the prominent example [12, 13]. No wonder that recently we can observe a quest for tools that could help us to close this gap.

One of such tools is *program semantics*. Though there is no firm consensus around the formalization of this term, we may roughly define it as a concise and minimal (irreducible) description of what the program is doing, expressed in simpler terms than the original GP tree (for the sake of this study, we identify GP program with GP tree). As GP programs usually process some input data, semantics is often defined with respect to them. In this spirit, Beadle and Johnson [2] used reduced ordered binary decision diagrams (ROBDDs) to describe semantics of GP programs and to define a semantically-driven crossover for GP. Their crossover computes ROBDDs for the parents and the offspring candidate, and discards offspring with the ROBDD equivalent to any of its parents. The representation of semantics of GP solutions as binary decision diagrams has been also used by Yanagiya[17] for performance purposes.

In this paper, we rely on the definition of program semantics inspired by Poli's and Page's work on sub-machine code GP [11] and McPhee *et al.*'s work on semantic building blocks [7]. This definition assumes that the fitness function is based on a finite vector of fitness cases $\mathbf{c}$. Given such a set, we define the semantics $\mathbf{s}$ of a program p as the vector of the values returned by p for consecutive elements of $\mathbf{c}$:

$$\mathbf{s}_{\mathbf{c}}(p) \equiv [p(c_1),\, p(c_2),\, \ldots,\, p(c_m)] \tag{1}$$

In following, we omit the index $\mathbf{c}$ for clarity. In other words, $\mathbf{s}(p)$ is defined here as an evaluation profile of p. If we agree to identify the semantics of a program with its phenotype, $\mathbf{s}()$ becomes the genotype-phenotype mapping.

We focus here on a subclass of problems for which the ideal semantic $\mathbf{s}^*$ is known, i.e., one knows in advance what is the desired response (output, individual's behavior) for all fitness cases. Examples of such problems are quite common in practice and include symbolic regression (c_i's are the independent coordinates of approximation points and $\mathbf{s}^*$ is the vector of the desired values of the dependent variable), and synthesis of logical functions like parity or multiplexer (c_i's usually enumerate all combinations of independent variables and $\mathbf{s}^*$ is the vector of desired responses of the evolved function). In this class of problems, the [minimized] fitness function f is usually defined as a metric that measures the divergence of $\mathbf{s}$ with respect to $\mathbf{s}^*$. For instance, symbolic regression typically involves square error (Euclidean metric) while logical problems rely on Hamming distance.

Obviously, metric-based fitness functions are unimodal by definition and have fitness-distance correlation equal to 1, because such fitness *is* a distance in the semantic space. Any linear combination of a pair of semantics is guaranteed to be not worse than the worse of them. Unfortunately, there is no obvious way of exploiting this property, as we do not control the semantics of a GP program directly: $\mathbf{s}(p)$ is only an image of what p computes for $\mathbf{c}$. Because the genotype-phenotype mapping is typically a very complex, many-to-one function, the inverse of the genotype-phenotype mapping does not exist and the convexity of the fitness function in the semantic space cannot be directly exploited in the genotype space. Nevertheless, in the following section we propose a method that does it in an indirect way.

3. SEMANTIC CROSSOVER

Past work on geometric crossover in the genotype space of GP trees demonstrates that it is not easy to design a crossover operator that is geometric under a metric in the genotype space [10]. Usually, one needs both an appropriately designed crossover operator *and* a specialized metric to guarantee geometricity (for instance, homologous crossover [5] requires specialized normalized structural Hamming distance to become geometric[10]).

Given the complexity of the genotype-phenotype mapping in GP, the prospects of designing a crossover operator that works in the genotype space *and* behaves geometrically in the corresponding semantic space are even more gloomy. Though in theory it is possible, the required assumptions would be very strong and render it impractical. Therefore, rather than *guaranteeing* the geometric behavior, our operator tries to *approximate* it by analysing the offspring *after* it has been bred. In other words, despite the poor locality of the genotype-phenotype mapping in GP [13], it tries to behave like a geometric crossover in the space of semantics. Technically, it makes an attempt to produce offspring that has semantics as close as possible to the linear combination of its parent's semantics. We refer to this operator as *approximately geometric semantic crossover* (SX for short).

SX employs a form of *brood selection* to search for a good approximation of geometric offspring [14, 1]. Given a pair of parents (x, y), SX applies n times a binary crossover operator CX to (x, y) to create a pool of candidates C. Technically, CX, referred to as *base* [crossover] *operator* in following, can be any crossover operating in the genotype space, like the common tree-swapping operator. Next, for both parents and for each candidate $z \in C$, the algorithm calculates their semantics $\mathbf{s}(x)$, $\mathbf{s}(y)$, $\mathbf{s}(z)$. Finally, it appoints as the offspring the candidate that minimizes the following expression:

$$\arg\min_{z \in C} d(\mathbf{s}(z), \mathbf{s}(x)) + d(\mathbf{s}(z), \mathbf{s}(y)) \tag{2}$$

where d is a *metric* in the semantic space, i.e., a non-negative symmetric function that observes the triangle inequality and such that $d(x, x) \equiv 0$. Thus, the algorithm chooses the candidate that is semantically most similar to both parents.

Due to triangle inequality observed by d, the lower bound of (2) is $d(\mathbf{s}(x), \mathbf{s}(y))$. When (2) reaches this bound, the offspring's semantic is geometric with respect to the semantics of its parents. Let us refer to offspring with that property as *semantically geometric offspring* and denote it by $\overline{z}(x, y)$. For a pair of parents, there may exist many such offspring.

The geometric interpretation of $\mathbf{s}(\overline{z}(x, y))$ obviously depends on the metric d. For the Euclidean distance, $\mathbf{s}(\overline{z}(x, y))$ belongs to the section $[\mathbf{s}(x); \mathbf{s}(y)]$. For the space of Boolean-valued semantics and under Hamming distance, $\mathbf{s}(\overline{z}(x, y))$ is

any bitstring with the property that, for each bit, its value is the same as the value of the corresponding bit in the semantics of at least one of its parents (x, y) [9].

Note that there may be many geometric offspring for a given pair of parents (x, y). Also, for some metrics d (e.g., the Hamming distance), there may exist more than one semantic $\mathbf{s}(z)$ that globally minimizes (2). However, these observations do not affect the following reasoning.

With geometric offspring, one could exploit the global convexity in the most efficient manner. Unfortunately, producing $\overline{z}$ is non-trivial. There are at least three factors that influence the likelihood of minimizing (2).

1. The first of them and the most critical one are the particular parents (x, y). If the parents do not contain the appropriate genetic material (GP code fragments), breeding $\overline{z}(x, y))$ becomes impossible, no matter how sophisticated is the base crossover operator CX.

2. Secondly, if the parents' genomes contain the required genetic material, then the chance of minimizing (2) depends mostly on the particular base crossover operator CX. It may be expected that some types of GP crossovers are more likely than others to produce an offspring that has semantics $\mathbf{s}(\overline{z}(x, y))$; some operators will be unable to do so. However, we hypothesize that this depends also on the semantics of particular GP functions and terminals. Therefore, we rely here on the standard tree-swapping crossover and postpone the consideration of other base crossovers.

3. Finally, the third factor that determines the chance of minimizing (2) is the pool size n. Most of popular GP crossover operators are indeterministic and yield different outcomes when applied multiple times to the same parent pair (x, y). The set of all offspring that may be produced by CX for a pair of parents (x, y) is sometimes referred to as *image* Im [9]. Now, if $\overline{z} \in Im(CX(x, y))$, then the probability of our algorithm producing $\overline{z}$ is greater than zero and grows with the candidate pool size n.

Given these three factors, it becomes clear that the overall chance of producing $\overline{z}$ by the proposed algorithm is rather low. However, it may be easily proved that the expected distance of the offspring semantic from the semantic of $\overline{z}$ does not increase with n:

$$E_{x,y}[d(\mathbf{s}(SX_n(x, y)),\, \mathbf{s}(\overline{z}(x, y)))] \\ \leq E_{x,y}[d(\mathbf{s}(SX_{n+1}(x, y)),\, \mathbf{s}(\overline{z}(x, y)))] \tag{3}$$

where SX_n denotes our approximately geometric semantic crossover working with the candidate pool of size n. The proof is trivial: SX chooses the candidate that minimizes (2), and that minimum cannot get greater with larger pool.

This implies that, leaving aside the two other factors discussed earlier, n stochastically controls the 'geometricity' of the produced offspring. This an important observation, as the more the semantics of the actual offspring resembles $\mathbf{s}(\overline{z})$, the more chance for benefiting from fitness-distance correlation.

The obvious conclusion is that we should use large n. Unfortunately, this comes at a price of extra calls of CX and calculating semantics, with the latter being usually computationally more expensive. Calculating the semantics requires applying the candidate to all fitness cases in $\mathbf{c}$, so this action is computationally almost as expensive as evaluating the individual.

This inconvenience, however, opens a possibility of an easy extension of the SX procedure: with $\mathbf{s}(z)$ already calculated, the fitness $f(z) = f^* - d(\mathbf{s}(z), \mathbf{s}^*)$ comes at almost no cost (the fitness is maximized and refers to the ideal fitness f^*). Thus, as $\mathbf{s}(z)$'s are computed anyway, we calculate also $f(z)$ for all candidates and find the most fit of them:

$$z_{best} = \arg\max_{z \in C} f(z) \tag{4}$$

Next, if $f(z_{best}) > \max(f(x),\, f(y))$, we return z_{best}. Otherwise, the algorithm returns the candidate indicated by (2). In short, if a candidate outperforms both parents, we appoint it as offspring, sacrifying semantic geometricity for the sake of fitness. This may be also interpreted as switching to hill climbing (HC) if it brings profit; that is why we term this method SHCX.

Finally, it seems worthy to discuss a possible weakness of both SX and SHCX. For a given constant value l, all candidates z for which $d(\mathbf{s}(z), \mathbf{s}(x)) + d(\mathbf{s}(z), \mathbf{s}(y)) = l$ are *equally* penalized by the term in formula (2), no matter how far they are from the closer of the parents. In particular, a candidate being a semantic clone of one of its parents (e.g., $\mathbf{s}(z) = \mathbf{s}(x)$) will be considered as good as any other perfectly semantically geometric offspring $\overline{z}(x, y)$. This warns us that by promoting geometricity, SX may also unwillingly promote excessive semantic similarity between the parents and its offspring. This phenomenon is inconvenient, as it decreases the chances of exploiting the convexity of the semantic fitness landscape.

To counteract this risk, we come up with an *equidistance-promoting* extension of SX, referred to as SX+ in following. It consists in extending Formula (2) by a penalty term that promotes balanced distance from both parents:

$$\arg\min_{z \in C} d(\mathbf{s}(z), \mathbf{s}(x)) + d(\mathbf{s}(z), \mathbf{s}(y)) \\ +|d(\mathbf{s}(z), \mathbf{s}(x)) - d(\mathbf{s}(z), \mathbf{s}(y))| \tag{5}$$

We introduce also the same extension to SHCX, leading to the method called henceforth SHCX+.

It should not come as a surprise that, in technical implementation of all aforementioned methods it is worth to cache offspring semantics as well as fitness values. Thanks to caching the former, the parents' semantics is already known when applying the crossover in the subsequent generation. Thanks to the latter, the evolutionary algorithm does not require an extra evaluation stage. Of course, these observations hold provided that the offspring does not undergo subsequent genetic manipulations, like mutation. Under this assumption and assuming that the costs of other stages of evolutionary algorithm are negligible, our algorithm is approximately $(n - 1)$ times slower than standard GP.

4. THE EXPERIMENT

The objective of the experiment is to verify the usefulness of considering the semantics in the family of SX crossover operators introduced in Section 3. In particular, we want to test if they have any positive impact on the convergence and success rate of evolutionary run.

There are two features of our operators that may potentially contribute to their performance when compared to conventional GP crossover operators: the tendency to

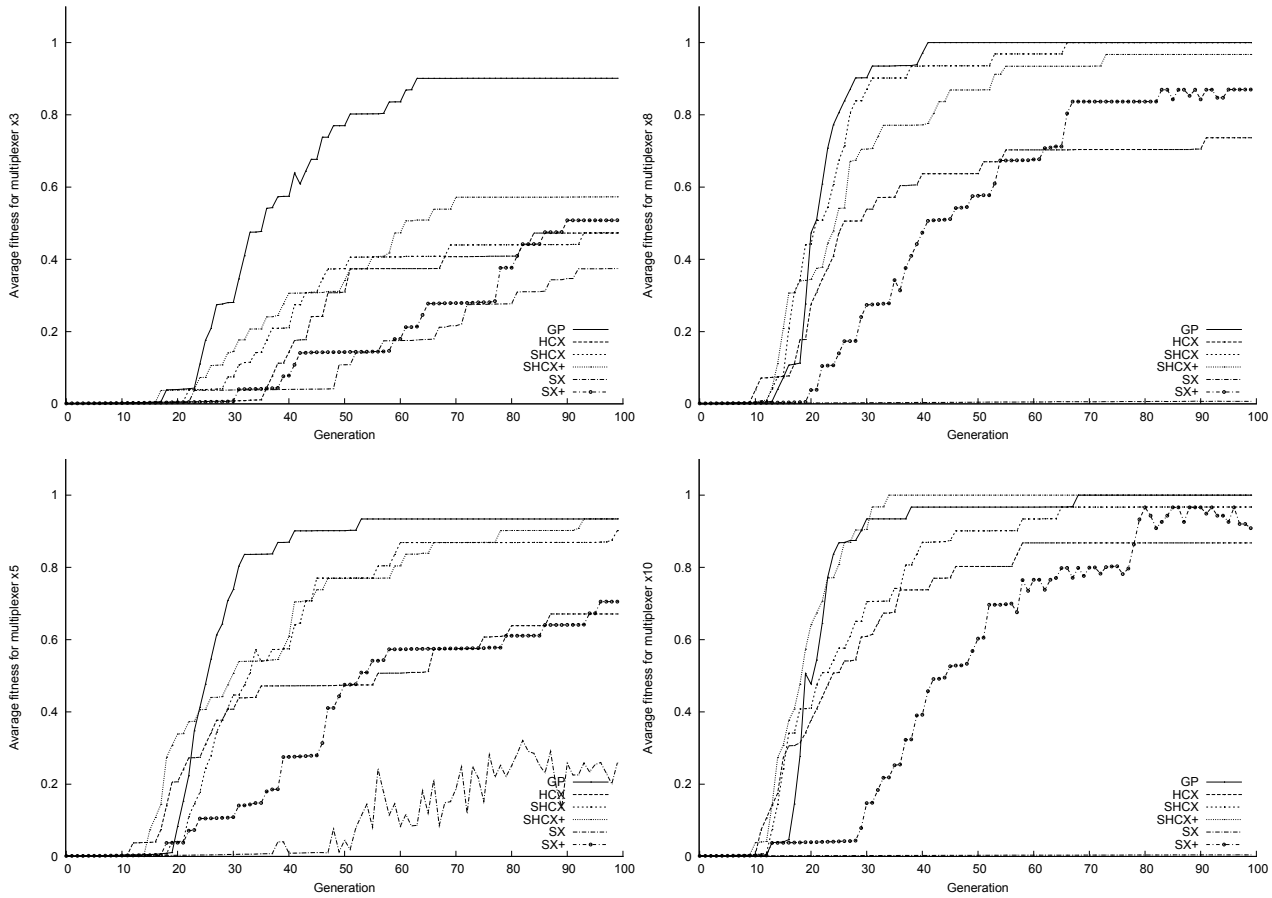

Figure 1: Mean fitness graphs for mux11, for $n = 3, 5, 8,$ and 10.

select the most semantically geometric candidate, and, in the case of the hill climbing-enabled operators (SHCX and SHCX+), incorporation of hill climbing, i.e., choosing the best offspring if it outperforms the better of its parents, no matter what its semantics. To isolate the contribution of the former feature, for the control experiment we design a comparable yet semantic-less crossover operator called hill-climbing crossover (HCX). Similarly to SX, HCX builds a pool of candidates of size n and returns the fittest candidate if its fitness outperforms the better of the parents. Otherwise however, it returns a randomly chosen candidate. Thus, other things being equal, HCX has no idea about semantics, so any difference in behaviors may be attributed only to SX's ability to take the semantics into account. Assuming the same pool size n, computational overheads caused by SX and HCX are almost the same, the only difference being the – practically negligible – cost of comparing the semantics using the metrics d.

The baseline for SX and HCX is the canonical GP equipped with subtree-swapping crossover. To compensate for the presence of pools of size n in SX and HCX, we run GP with n-times larger population (10240 individuals compared to 1024 in the remaining runs).

To sum up, for each problem and each considered pool size $n = 3, 5, 8,$ and 10, six methods will be confronted in this experiment:

- Two control methods:
  - GP: standard Koza-I style genetic programming (GP) using population size 10240,
  - HCX: hill-climbing semantic-less crossover (HCX) using population size 1024,
- Four methods of interest (all using population size 1024):
  - SX: the basic approximately geometric semantic crossover (see formula (2)),
  - SHCX: approximately geometric semantic crossover switching to hill-climbing if better offspring is found (cf. formula (4)),
  - SX+: SX extended by promotion of equidistance (formula (5)),
  - SHCX+: SHCX + promotion of equidistance.

To verify the generality of the semantic approach, we test it on two qualitatively distinct benchmark problems: logic function synthesis (11-bit multiplexer, *mux11*) and symbolic regression (sextic polynomial, *sextic*). For the former, we substitute Hamming distance for d; for the latter, d is defined as Euclidean distance. Another important difference between these two problems is that, for the former problem,

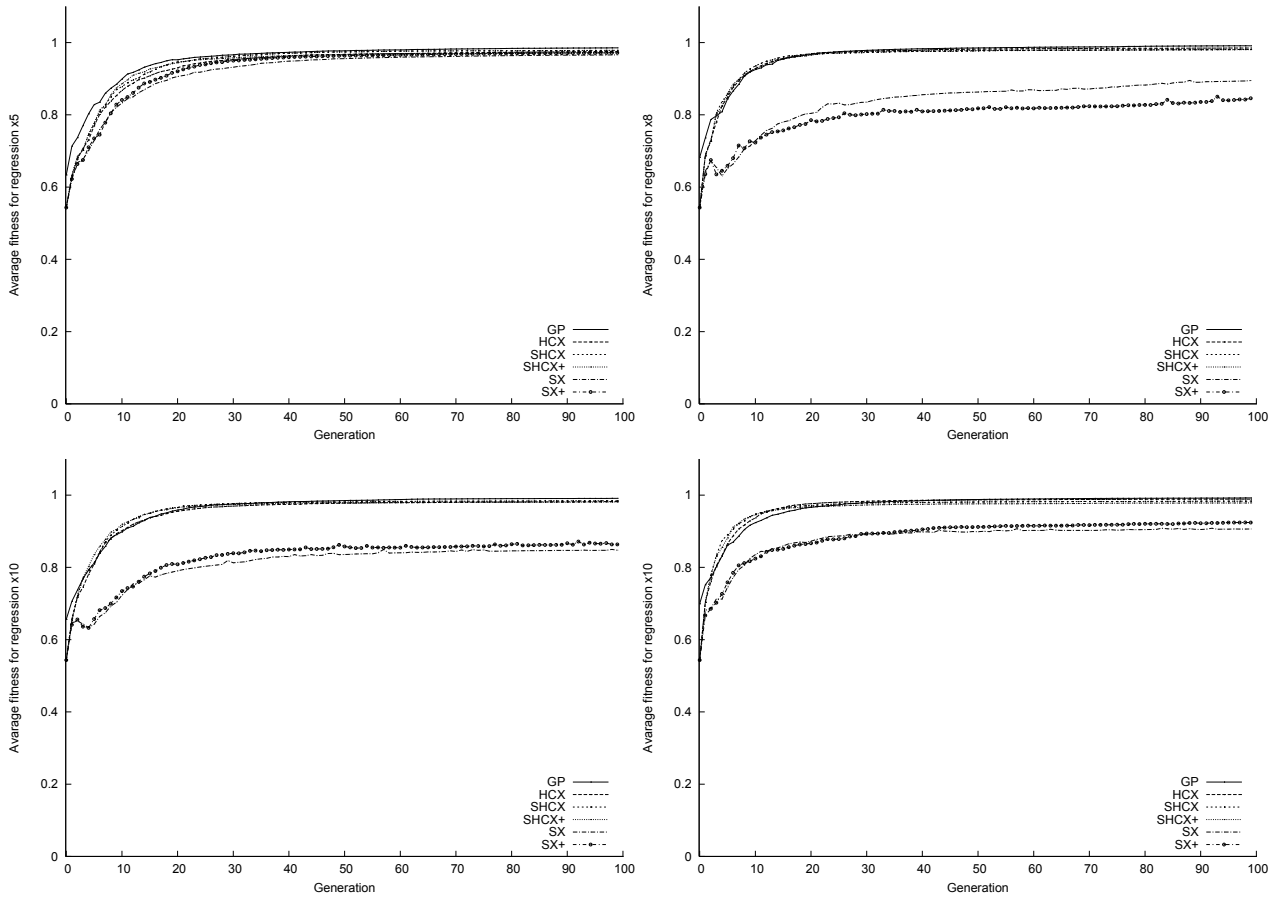

Figure 2: Mean fitness graphs for sextic, for $n = 3, 5, 8,$ and 10.

the fitness cases c_i enumerate all possible combinations of independent (input) variables ($2^{11} = 2048$ in this case), while for the latter c_i's are only a limited sample of the infinite domain of the approximated function; in our experiment, we used 20 fitness cases for symbolic regression.

The software testbed has been implemented with help of ECJ [6]. For both problems, we use the default Koza-I-style settings [4] as implemented in ECJ. In particular, our operators employ the standard subtree-swapping crossover as the base operator CX. For each considered settings and method, we run evolution 30 times and present the aggregated results in the following tables and graphs.

Figure 1 presents the fitness graphs averaged over 30 runs for the mux11 problem. Figure 2 depicts the analogous graphs for the sextic problem. Table 1 lists the mean best-of-run fitness for all methods (means ± confidence intervals).

The most evident observation concerning the presented data is the inconclusive result for the sextic problem. For most values of n, the methods converge here on average in a very similar way, the only exception being the SX and SX+ methods for $n = 10$. Our approximately geometric semantic crossover not only did not outperform the control methods on this problem; it did not behave any different from them in a significant and systematic way. In our opinion, the most likely explanation of this phenomenon is the relative simplicity of the sextic problem, which caused all the algorithms to converge very quickly and did not leave much space for improvement for our geometric crossovers. Another plausible reason is the probably very poor locality of the genotype-phenotype mapping of symbolic regression problems – such low locality does not give chance our base crossover operator CX to generate an offspring with semantic that is at least *slightly* geometric with respect to its parents.

Things look very different for the mux11 problem (Fig. 1): the particular crossover methods exhibit here significantly different dynamics. Purely semantic methods (SX, SX+) perform the worse, and only when combined with the hill climbing (SHCX, SHCX+) are they able to compete with the reference runs (GP, HCX), sometimes temporarily outperforming them. SHCX and SHCX+ are also usually better than the bare hill-climbing crossover (HCX), indicating that the contribution of the semantic geometricity is significant. Finally, we may observe also the anticipated dependency on n: the greater its value, the more competitive the semantic crossovers when compared to GP and HCX.

For the methods that combine geometric crossover with hill climbing (SHCX and SHCX+), Figures 4 and 5 present the share of offspring produced by the 'semantic principle', i.e., the frequency of cases when the crossover outputs the most geometric offspring. The remaining share, i.e., one minus the number observed in the graph, is the frequency of cases when the operator found an offspring that outperforms the better of the parents. Not surprisingly, the share geometric offspring is quite moderate in the beginning of the

Table 1: Best-of-run fitness for all methods averaged over 30 runs with .95 confidence intervals.

		Best-of-run fitness					
Problem	n	GP	HCX	SX	SHCX	SX+	SHCX+
sextic	3	0.9852±0.0048	0.9754±0.0063	0.9658±0.0113	0.9785±0.0057	0.9716±0.0102	0.9730±0.0090
	5	0.9910±0.0026	0.9821±0.0051	0.8474±0.0741	0.9845±0.0037	0.8636±0.0754	0.9810±0.0053
	8	0.9911±0.0028	0.9849±0.0036	0.8949±0.0457	0.9810±0.0039	0.8457±0.0644	0.9805±0.0058
	10	0.9922±0.0025	0.9889±0.0023	0.9064±0.0337	0.9833±0.0050	0.9240±0.0276	0.9783±0.0053
mux11	3	0.9010±0.1081	0.4727±0.1795	0.3744±0.1732	0.4736±0.1792	0.5082±0.1790	0.5730±0.1777
	5	0.9339±0.0900	0.6710±0.1693	0.2589±0.1489	0.9018±0.1072	0.7050±0.1639	0.9344±0.0894
	8	1.0000±0.0000	0.7367±0.1589	0.0069±0.0005	1.0000±0.0000	0.8699±0.1206	0.9672±0.0643
	10	1.0000±0.0000	0.8677±0.1228	0.0043±0.0003	0.9672±0.0643	0.9086±0.0874	1.0000±0.0000

run, when the mean fitness in population is low and producing a better-performing offspring is relatively likely. In this phase of evolution, hill climbing dominates the behavior of SHCX and SHCX+. As the evolution progresses and the overall fitness increases, geometric offspring becomes more frequent and account for up to over 90% of the crossover applications. This tendency is very well observable for mux11 (Fig. 4); for the sextic problem (Fig. 5), this process is disturbed in the very beginning of the evolution. Explanation of this phenomenon require an extra study.

However, the results presented in Figs. 4 and 5 do not tell us *how geometric* are the actual offspring produced by SX and SX+. To monitor this feature, in Fig. 6 we present a graph of *geometricity*, defined as (cf. Formula (2)):

$$\min_{z \in C} \frac{d(\mathbf{s}(z), \mathbf{s}(x)) + d(\mathbf{s}(z), \mathbf{s}(y))}{d(\mathbf{s}(x), \mathbf{s}(y)) + 1} \tag{6}$$

and averaged over all calls of semantic crossover in a given generation (+1 term in denominator saves us from division by zero for semantically equivalent parents). The graph proves that both crossover operators are able to maintain good geometricity over the entire timespan of evolutionary run. This is remarkable, knowing that with time the solutions and their semantics become more and more similar. Also, the version of the operator equipped with the promotion of equidistant offspring (Formula (5)) produces offspring that is more geometric on average.

It should be emphasized that in the above experiment we fairly compensated for the presence of pools of size n in SX and HCX, rewarding GP with n times greater population. If we assume that the unit cost of all crossover operators is the same, things change dramatically. Figure 3 presents the results of the considered methods confronted with GP running with the same population size (1024 individuals), for $n = 3$. The graphs present the performance of the best-so-far individuals. This time, GP is inferior with respect to all the other methods, whether they involve hill climbing (like SHCX+) or not (like SX). For larger values of n, the difference in favour of semantic-based operators is even more significant. This clearly demonstrates that semantics and convexity of the semantic space are worth considering.

5. CONCLUSION AND FUTURE WORK

In this study we have shown that the perfect fitness-distance correlation of the semantic fitness landscape of GP problems with the fitness function based on fitness cases may be exploited by a geometric semantic crossover operator SX that tries to approximate the geometric behavior in the space of semantics. Evolutionary runs that use SX perform not worse than the conventional GP in terms of effort. SX is representation-independent and applicable to a broad set of real-world applications of GP.

In the fair competition that equaled the computational effort of all methods, the proposed method in its current shape does not outperform the reference approach. However, we envision some promising ways of attaining this goal. The most obvious of them would consist in using partial (incomplete) semantics vectors that would be based only on a *subset* of fitness cases from **c**. Attaining perfect geometricity in the semantic space is unrealistic in GP, so it should not do much harm if we minimize expression (2) only in an approximate way. This could allow us to reduce the computational cost of SX and possibly find a break-even size of the candidate pool.

The weak point of the presented methodology is that it is extremely oriented towards the phenotype space, to the extent that it actually *ignores* the individuals' genotypes. This was intentional, as we wanted to investigate the pure effect of exploiting semantic geometricity only. In this light, referring to fitness-distance correlation, though methodologically correct, is slightly inappropriate, as fdc has been designed to investigate the relations between the genotype space and the phenotype space (and, indirectly, their relations to fitness). Thus, in a longer perspective it would be desirable to make the semantic crossover more 'aware' of the genotypes, by, for instance, exploiting the convexity of the semantic space to choose the most promising *loci* for crossover. Our current research aims at this objective. Apart from that, we would like to investigate more thoroughly what is the effect of the modification of selection pressure (with respect to standard GP), which the semantic crossover inevitably involves. For that purpose, we plan to employ the methodology proposed in [16].

Acknowledgment

This research has been supported by grant N N519 3505 33 and grant DS-91-471.

6. REFERENCES

[1] L. Altenberg. Emergent phenomena in genetic programming. In A. V. Sebald and L. J. Fogel, editors, *Evolutionary Programming — Proceedings of the Third Annual Conference*, pages 233–241, San Diego, CA, USA, 24-26 Feb. 1994. World Scientific Publishing.

[2] L. Beadle and C. Johnson. Semantically driven crossover in genetic programming. In J. Wang, editor,

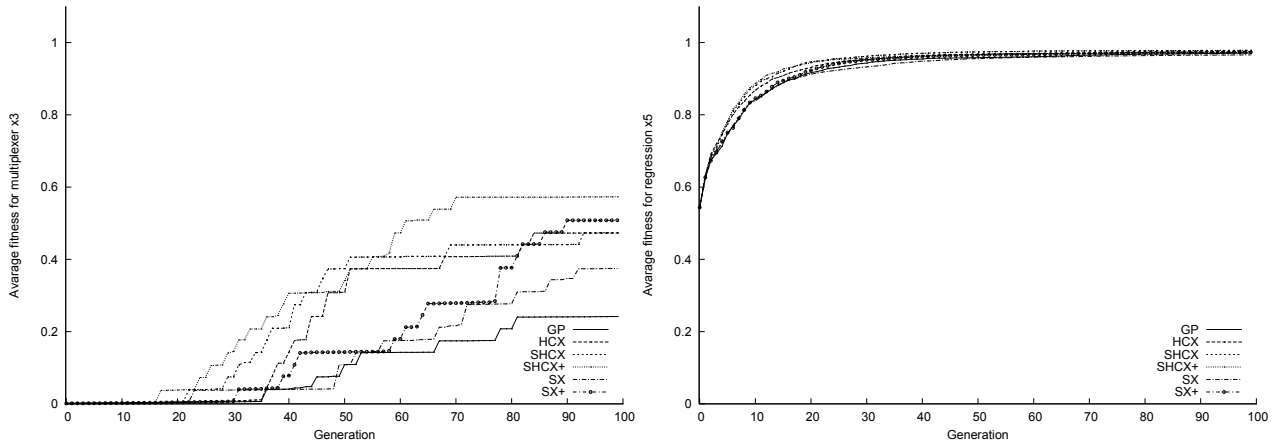

Figure 3: Mean best-so-far fitness graphs for mux11 (left) and sextic (right) and for $n = 3$ without compensation for computational effort (GP runs with the same population size (1024) as the other methods).

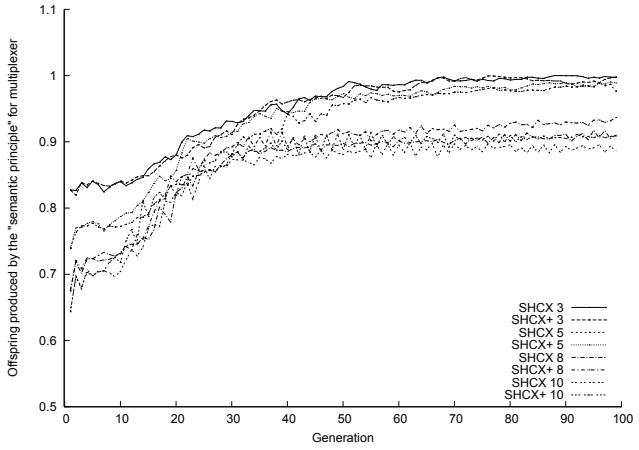

Figure 4: Average share of offspring produced using the semantic principle for the mux11 problem.

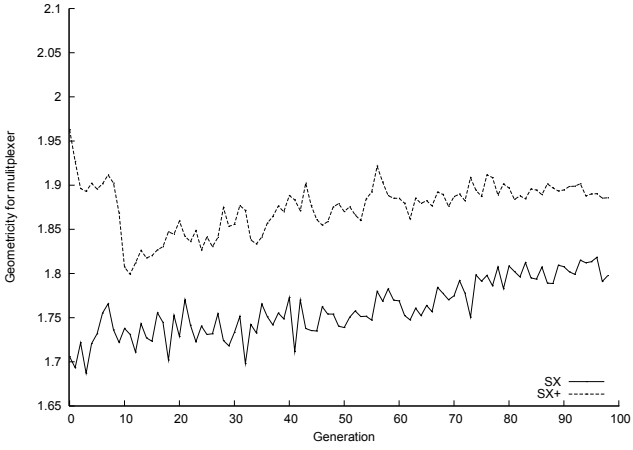

Figure 6: Average semantic geometricity of the offspring produced by SX and SX+ for the mux11 problem.

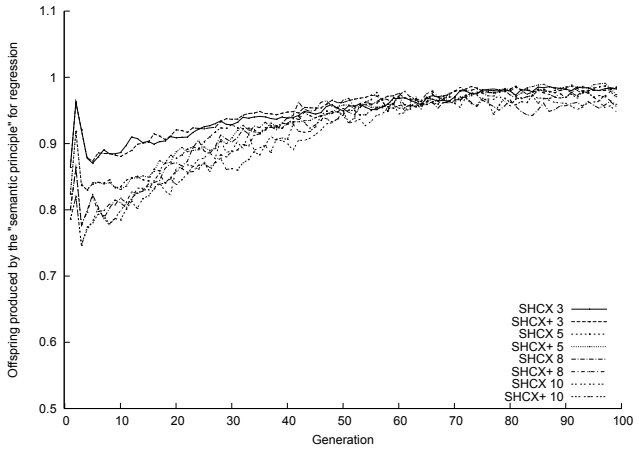

Figure 5: Average share of offspring produced using the semantic principle for the sextic problem.

Proceedings of the IEEE World Congress on Computational Intelligence, pages 111–116, Hong Kong, 1-6 June 2008. IEEE Computational Intelligence Society, IEEE Press.

[3] B. Doerr, E. Happ, and C. Klein. Crossover can provably be useful in evolutionary computation. In *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 539–546, New York, NY, USA, 2008. ACM.

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

[5] W. B. Langdon and R. Poli. *Foundations of Genetic Programming*. Springer-Verlag, 2002.

[6] S. Luke. ECJ evolutionary computation system, 2002. (http://cs.gmu.edu/ eclab/projects/ecj/).

[7] N. F. McPhee, B. Ohs, and T. Hutchison. Semantic building blocks in genetic programming. In M. O'Neill, L. Vanneschi, S. Gustafson, A. I. E. Alcázar, I. D.

Falco, A. D. Cioppa, and E. Tarantino, editors, *Genetic Programming*, volume 4971 of *LNCS*, pages 134–145. Springer, 2008.

[8] P. Merz. A comparison of memetic recombination operators for the traveling salesman problem. In *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 472–479. Morgan Kaufmann Publishers.

[9] A. Moraglio and R. Poli. Topological interpretation of crossover. In K. Deb, R. Poli, W. Banzhaf, H.-G. Beyer, E. Burke, P. Darwen, D. Dasgupta, D. Floreano, J. Foster, M. Harman, O. Holland, P. L. Lanzi, L. Spector, A. Tettamanzi, D. Thierens, and A. Tyrrell, editors, *Genetic and Evolutionary Computation – GECCO-2004, Part I*, volume 3102 of *Lecture Notes in Computer Science*, pages 1377–1388, Seattle, WA, USA, 26-30 June 2004. Springer-Verlag.

[10] A. Moraglio and R. Poli. Geometric landscape of homologous crossover for syntactic trees. In *Proceedings of the 2005 IEEE Congress on Evolutionary Computation (CEC-2005)*, volume 1, pages 427–434, Edinburgh, 2-4 Sept. 2005. IEEE.

[11] R. Poli and J. Page. Solving high-order boolean parity problems with smooth uniform crossover, sub-machine code GP and demes. *Genetic Programming and Evolvable Machines*, 1(1/2):37–56, Apr. 2000.

[12] F. Rothlauf. On the locality of representations. Technical report, University of Mannheim, Department of Information Systems 1, 2003.

[13] F. Rothlauf and M. Oetzel. On the locality of grammatical evolution. Working Paper 11/2005, Department of Business Administration and Information Systems, University of Mannheim, D-68131 Mannheim, Germany, Dec. 2005.

[14] W. A. Tackett and A. Carmi. The unique implications of brood selection for genetic programming. In *Proceedings of the 1994 IEEE World Congress on Computational Intelligence*, Orlando, Florida, USA, 27-29 June 1994. IEEE Press.

[15] L. Vanneschi and M. Tomassini. Pros and cons of fitness distance correlation in genetic programming. In A. M. Barry, editor, *GECCO 2003: Proceedings of the Bird of a Feather Workshops, Genetic and Evolutionary Computation Conference*, pages 284–287, Chigaco, 11 July 2003. AAAI.

[16] H. Xie, M. Zhang, and P. Andreae. An analysis of constructive crossover and selection pressure in genetic programming. In D. Thierens, H.-G. Beyer, J. Bongard, J. Branke, J. A. Clark, D. Cliff, C. B. Congdon, K. Deb, B. Doerr, T. Kovacs, S. Kumar, J. F. Miller, J. Moore, F. Neumann, M. Pelikan, R. Poli, K. Sastry, K. O. Stanley, T. Stutzle, R. A. Watson, and I. Wegener, editors, *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, volume 2, pages 1739–1748, London, 7-11 July 2007. ACM Press.

[17] M. Yangiya. Efficient genetic programming based on binary decision diagrams. In *1995 IEEE Conference on Evolutionary Computation*, volume 1, pages 234–239, Perth, Australia, 29 Nov. - 1 Dec. 1995. IEEE Press.