Progress properties and fitness bounds for geometric semantic search operators

Tomasz P. Pawlak¹ · Krzysztof Krawiec¹

Received: 19 December 2014 / Revised: 18 September 2015 / Published online: 22 October 2015
© The Author(s) 2015. This article is published with open access at Springerlink.com

Abstract Metrics are essential for geometric semantic genetic programming. On one hand, they structure the semantic space and govern the behavior of geometric search operators; on the other, they determine how fitness is calculated. The interactions between these two types of metrics are an important aspect that to date was largely neglected. In this paper, we investigate these interactions and analyze their consequences. We provide a systematic theoretical analysis of the properties of abstract geometric semantic search operators under Minkowski metrics of arbitrary order. For nine combinations of popular metrics (city-block, Euclidean, and Chebyshev) used in fitness functions and of search operators, we derive pessimistic bounds on fitness change. We also define three types of progress properties (weak, potential, and strong) and verify them for operators under those metrics. The analysis allows us to determine the combinations of metrics that are most attractive in terms of progress properties and deterioration bounds.

Keywords Geometric semantic genetic programming · Theory · Metric · Fitness landscape · Fitness bounds · Guarantees of progress

1 Introduction

Semantic genetic programming (SGP) is a relatively recent research thread in GP, where selected behavioral properties of programs are being exploited for the sake of making program synthesis more efficient and scalable. The properties in question are captured by means of formal objects termed *program semantics*, which describe

✉ Tomasz P. Pawlak
tpawlak@cs.put.poznan.pl

Krzysztof Krawiec
krawiec@cs.put.poznan.pl

¹ Institute of Computing Science, Poznan University of Technology, Poznan, Poland

the effects of interactions between program and data. By nature, program semantics is more detailed than the scalar fitness commonly used in GP, and in effect offers a more informative guidance for program synthesis. This virtue of SGP has brought substantial performance gains in numerous works [1, 3, 13, 19, 23, 27, 30, 31].

The prevailing part of SGP studies equate program semantics with the list of outputs produced by a program in response to training examples (fitness cases). This simple formalism, known as *sampling semantics* or simply *semantics*, provides a relatively detailed account on program execution at no additional computational cost. Previous studies have shown that, even though sampling semantics cannot convey information on all aspects of program execution, it can be leveraged to design efficient semantic-aware search operators [8, 13, 23, 26, 27, 30, 31], population initialization [2, 7] and selection techniques [6].

A particularly promising area in SGP is Geometric Semantic GP (GSGP), which focuses on the geometric properties of program semantics. GSGP is founded on the observation that fitness function in GP operates on program outputs, i.e., on program semantics, and thus endows the set of program semantics with a structure, turning it so into a *semantic space*. In particular, if the desired program outputs for fitness cases are known, they define a specific point in that space, called *target*. If the fitness function happens to be a metric, which is very often the case in GP, the resulting semantic space is a metric space. The geometric characteristics of the metric semantic space together with the target conveniently lend themselves to elegant derivation of properties and concepts of theoretical interest and practical relevance.

The arguably most promising results obtained in GSGP are the exact geometric search operators [19]. Crucially, the geometric semantic crossover and geometric semantic mutation *exactly* realize the properties expected from ideal search operators. This is achieved by expressing the desirable ‘mixing’ of the parents using the instructions of the programming language of consideration. In effect, those operators are *guaranteed* to produce offspring with specific geometric properties of semantics, and in turn ensure certain types of progress.

In this study, we define several types of new properties of progress and provide general formulations of proofs that hold for Minkowski metrics of arbitrary order. The main results of this work are the bounds on fitness deterioration for particular metrics. These contributions are not only interesting on their own, but bear material implications for the practice of GSGP.

This paper is structured as follows. In Sect. 2, we briefly formalize the basic concepts of program synthesis as practiced in GP. Section 3 reviews the most popular metrics and discusses how they shape the geometry of semantic spaces. Section 4 presents the relationship between program spaces and semantic spaces, connects them to fitness landscapes, and defines the geometric semantic search operators. Sections 5 and 6 gather the main contributions of this paper, presenting the derivations and proofs of theorems concerning the progress properties and fitness change bounds, respectively. Finally, after confronting these results with previous contributions in Sect. 7, in Sects. 8 and 9 we discuss the actual and potential consequences of the bounds, to conclude with closing remarks in Sect. 10.

2 Preliminaries: programs and semantics

In this section, we introduce the necessary toolbox of basic formalisms. We consider abstract search operators, and assume only that a program is a function $I \rightarrow O$, where I is the set of admissible inputs to a program and O is the set of outputs a program may produce. For instance, a typical Boolean function synthesis task involves $I = \{0, 1\}^l$ and $O = \{0, 1\}$, and for regression tasks $I = \mathbb{R}^l$ and $O = \mathbb{R}$, where l is the number of input variables. The symbol $\mathcal{P}$ denotes the set (space) of all programs of consideration that is searched by a GP algorithm.

A *fitness case* is a pair of a program input and the corresponding desired output, i.e., $(x, y) \subseteq I \times O$, which together define the desired behavior of a program applied to x . A list of fitness cases that together define program behavior for various inputs forms *training set* and will be denoted by T . We require T to contain one fitness case with a given input: $\forall (x_1, y_1) \in T : \neg \exists (x_2, y_2) \in T : x_1 = x_2$.

Given a list of fitness cases $T = ((x_1, y_1), (x_2, y_2), \dots, (x_n, y_n))$, the list t of desired outputs it defines will be referred to as *target*, i.e., $t = (y_1, y_2, \dots, y_n)$. Further on, we consider only programs that return numerical values (typically $O = \mathbb{R}$), in which case target is a *vector*. However, the derivations presented in this paper apply also to the Boolean domain by setting $I = \{0, 1\}^l$, $O = \{0, 1\}$.

Given a program $p \in \mathcal{P}$ and the list of fitness cases T , we define the *semantics* $s(p)$ of p as the list of outputs produced by p for the inputs listed in T , i.e., $s(p) = (p(x_1), p(x_2), \dots, p(x_n))$. A *semantic mapping* s maps the set of programs $\mathcal{P}$ onto a set of semantics $\mathcal{S}$, i.e., $s : \mathcal{P} \rightarrow \mathcal{S}$. The codomain $\mathcal{S}$ of s will be endowed with a structure in the next section, and will be by this token called *semantic space*. In analogy to the target above, $s(p)$ is a vector. Note that unless T enumerates all possible program inputs, $s(p)$ conveys only partial information about the behavior of p .

A *program synthesis/induction task* is a tuple $(\mathcal{P}, c_T)$, where $c_T : \mathcal{P} \rightarrow \{0, 1\}$ is a *correctness predicate* that verifies if a program behaves according to the fitness cases in T . Program synthesis task is thus inherently a *search problem*, with $\mathcal{P}$ being the search space and c_T testing if a goal state has been reached. In GP however, this problem is extended with a *fitness function* $f_T : \mathcal{P} \rightarrow \mathbb{R}$ that measures the quality of programs, and so becomes an *optimization problem*. In a simple case, f_T counts the number of fitness cases in T failed by a program, i.e.,

$$f_T(p) = |\{(x, y) \in T : p(x) \neq y\}|. \quad (1)$$

It will be convenient in the following to assume that f_T is minimized and attains the minimal value of zero for a correct program, i.e.,

$$f_T(p) = 0 \iff c_T(p). \quad (2)$$

In that case, the correctness predicate is not needed anymore and a programming task can be redefined as $(\mathcal{P}, f_T)$.

3 Geometry of semantic spaces

Program semantics captures the effects of computation, which are usually of primary interest in automatic program synthesis. Every terminating program has a corresponding semantics in the codomain $\mathcal{S}$ of the semantic mapping. Therefore, it is that codomain, and not the original space of programs, where the actual effects of computation should be analyzed.

The pivotal observation of GSGP is that in GP a fitness function structures the codomain of semantic mapping, turning it into a *semantic space*. In particular, the fitness functions predominantly used in GP are not any functions, but *metrics*. For instance, in symbolic regression one usually defines fitness as a root mean square error (RMSE) or a mean absolute error (MAE) calculated over the fitness cases $(x, y) \in T$. These two measures are equivalent (up to scaling) to the Euclidean and city-block metrics, respectively.

A fitness function in a program synthesis task $(\mathcal{P}, f_T)$ can be thus alternatively expressed as

$$f_T(p) = \|s(p), t\|, \quad (3)$$

where $\|\cdot, \cdot\|$ is a metric, $s(p)$ is the semantics of program p , and t is the target of T .

The fitness defined as above is ‘hooked’ at the target, i.e., its second argument is always t . However, if the task is solvable, i.e., there exists a program p such that $s(p) = t$, then $t \in \mathcal{S}$, i.e., t is just an element of the semantic space $\mathcal{S}$, and nothing precludes us from applying $\|\cdot, \cdot\|$ to *any pair of semantics in $\mathcal{S}$* . When detached from the target, the metric in (3) can be used as *semantic distance d* in $\mathcal{S}$. However, $\mathcal{S}$ can be alternatively structured using any metric, which we thoroughly discuss in this paper.

By being metrics, both d and f_T are non-negative and meet the three fundamental requirements, i.e., identity of indiscernibles ($\|s_1, s_2\| = 0 \iff s_1 = s_2$), symmetry ($\|s_1, s_2\| = \|s_2, s_1\|$) and triangle inequality ($\|s_1, s_2\| \leq \|s_1, s_3\| + \|s_3, s_2\|$).

The abovementioned examples of RMSE and MAE, and the corresponding Euclidean and city-block metrics can be conveniently captured by the umbrella of parameterized Minkowski metric

$$L_z(a, b) = \sqrt[z]{\sum_i |a_i - b_i|^z}.$$

The three most common representatives of L_z are $L_1 = \sum_i |a_i - b_i|$ (city-block metric), $L_2 = \sqrt{\sum_i (a_i - b_i)^2}$ (Euclidean metric), and $L_\infty = \max_i |a_i - b_i|$ (Chebyshev metric). Note that L_1 in a semantic space constrained to the unit hypercube ($\mathcal{S} = \{0, 1\}^n$) is equivalent to the Hamming distance, and thus suitable to handle the Boolean domain.

Below we define and illustrate several properties of these three metrics that are essential for our further conduct.

Definition 1 A *segment* between a and b is the set of all c that satisfy $\|a, b\| = \|a, c\| + \|c, b\|$, where $\|\cdot, \cdot\|$ is a metric and a, b, c are arbitrary objects.

Fig. 1 Segments in $\mathbb{R}^2$ under the L_1, L_2, L_∞ metrics

Fig. 2 Balls in $\mathbb{R}^2$ under the L_1, L_2, L_∞ metrics

Figure 1 presents exemplary segments under L_1, L_2 , and L_∞ in $\mathbb{R}^2$. Note that, though the segment under L_∞ in $\mathbb{R}^2$ happens to be a rectangle, it does not generalize to hyperrectangle in the higher-dimensional spaces.

Definition 2 A ball $B(a, r)$ of radius $r \in \mathbb{R}_{>0}$ with a center in a is the set of all objects b that satisfy $\|a, b\| \leq r$, where $\|\cdot, \cdot\|$ is a metric and a is an arbitrary objects. A *sphere* is the analogous set that satisfies $\|a, b\| = r$.

Figure 2 shows exemplary balls of a radius r centered in a under L_1, L_2 , and L_∞ metrics in $\mathbb{R}^2$.

Definition 3 A set X is *convex* iff the segments between all pairs of elements in X are contained in X . A *convex hull* $C(Y)$ of set Y is the intersection of all convex sets that contain Y .

In other words, a convex hull of Y is the smallest convex set that contains Y . In Fig. 3 we visualize exemplary convex hulls for the L_1, L_2 , and L_∞ metrics in $\mathbb{R}^2$.

4 Geometric semantic search operators

The consequences of a fitness function being a metric in semantic space (Eq. 3) are profound. When graphed against $\mathcal{S}$, f_T is a unimodal surface that attains the minimum value of zero at the target t , i.e., a *cone* with an apex in t . The further a

Fig. 3 Convex hulls in $\mathbb{R}^2$ under the L_1, L_2, L_∞ metrics

given point $s \in \mathcal{S}$ is from t , the greater f_T . This unimodal and smooth structure opens the door to designing efficient program synthesis approaches.

In GP, search is conducted in $\mathcal{P}$ by means of a k -ary search operator ($k \geq 1$), which is a function $o : \mathcal{P}^k \rightarrow \mathcal{P}$ (or a few such operators working in parallel). The key work on GSGP [19] showed that search operators exist that work with programs in $\mathcal{P}$ and at the same time have well-defined effects on the images of those programs in $\mathcal{S}$. In this paper we are interested in such operators, though not necessarily the particular ones proposed in [19]. We proceed with defining the properties of *abstract search operators* that are essential for achieving properties of progress considered in the next section.

Consistently with the convention used in GP, we refer to an 1-ary search operator as *mutation*, and to a k -ary, $k \geq 2$ search operator as *crossover*. Because these operators are usually stochastic in GP (formally, they are *random functions*), we distinguish the individual *acts* of operator's application from the operators as such. Also, we refer to the arguments of search operators as 'parents' regardless of operator's arity.

Definition 4 An application of a mutation operator $o : \mathcal{P} \rightarrow \mathcal{P}$ to a parent program $p \in \mathcal{P}$ is *r-geometric* iff the semantics $s(o(p))$ of the produced offspring $o(p)$ belongs to the ball of radius $r > 0$ centered in semantics $s(p)$ of its parent p . Formally:

$$s(o(p)) \in B(s(p), r)$$

A mutation operator is *r-geometric* iff all its applications are *r-geometric*.

Definition 5 An application of a $k \geq 2$ -ary crossover operator $o : \mathcal{P}^k \rightarrow \mathcal{P}$ to the parent programs $\{p_1, p_2, \dots, p_k\}$ is *geometric* iff the semantics $s(o(p_1, \dots, p_k))$ of the produced offspring $o(p_1, \dots, p_k)$ belongs to the convex hull of the semantics of its parents, i.e.,

$$s(o(p_1, \dots, p_k)) \in C(\{s(p_1), s(p_2), \dots, s(p_k)\})$$

A $k \geq 2$ -ary crossover is *geometric* iff all its applications are geometric.

Search operators define neighborhood, and fitness plotted against solutions is referred to as *fitness landscape* [33]. When using conventional GP search operators,

Fig. 4 Transformation of the solution space from $\mathcal{P}$ to $\mathcal{S}$ and its impact on fitness landscape. Note the not necessarily bijective nature of this transformation

Fig. 5 Hypothetical fitness landscapes under L_1 (left), L_2 (center), L_∞ (right) metrics. The concentric figures mark the level sets of fitness function

that landscape is usually very rugged (left part of Fig. 4), because the actions of operators may affect fitness in (almost) arbitrary way. However under geometric operators (right part of Fig. 4), fitness landscapes are smooth and conic, because the effects of operators are consistent with the distance to the target (i.e., obey the same geometry). Figure 5 shows the *fitness landscapes* under L_1 , L_2 , L_∞ metrics i.e., $f(p) = L_z(s(p), t)$. The level sets of fitness landscape are spheres under the corresponding metric and dimensionality of semantic space.

In the following, the references to $\mathcal{P}$, T , and $s(\cdot)$ are implicit. We allow an abuse of notation by using a symbol of a program in place of its semantics, e.g., $L_z(p_1, p_2)$ is a shorthand for $L_z(s(p_1), s(p_2))$.

5 Types of progress properties

In this section, we define several qualitative properties of search operators, which form the main novel contribution of this work.

Definition 6 The *image* $I(o, p_1, \dots, p_k)$ of a k -ary search operator $o : \mathcal{P}^k \rightarrow \mathcal{P}$ for the parents $p_1, \dots, p_k$ is the set of all offspring that may be returned by $o(p_1, \dots, p_k)$, i.e.,

$$I(o, p_1, \dots, p_k) = \{o(p_1, \dots, p_k)\},$$

where the expression on the right hand side of this formula is intended to denote the codomain of the random variable o for the parents $p_1, \dots, p_k$.

Definition 7 A k -ary search operator $o : \mathcal{P}^k \rightarrow \mathcal{P}$ is *weakly progressive* (WP) iff for any parents $p_1, \dots, p_k \in \mathcal{P}$ it holds

$$\max_{p \in I(o, p_1, \dots, p_k)} f(p) \leq \max_{p \in \{p_1, \dots, p_k\}} f(p)$$

In other words, o is weakly progressive if the worst offspring it may produce is not worse than the worst of the parents. Importantly, if this property holds for all operators involved in search, the worst fitness in the next generation population is not worse than the worst fitness in the current one. Weak progress precludes thus deterioration of fitness.

Definition 8 A k -ary search operator is *potentially progressive* (PP) iff for any parents $p_1, \dots, p_k \in \mathcal{P}$ it holds

$$\min_{p \in I(o, p_1, \dots, p_k)} f(p) \leq \min_{p \in \{p_1, \dots, p_k\}} f(p)$$

Thus, an operator that is potentially progressive may produce offspring that is not worse than all its parents.

Definition 9 A k -ary search operator is *strongly progressive* (SP) iff for any parents $p_1, \dots, p_k \in \mathcal{P}$ it holds

$$\max_{p \in I(o, p_1, \dots, p_k)} f(p) \leq \min_{p \in \{p_1, \dots, p_k\}} f(p)$$

Thus, a strongly progressive operator guarantees that all the offspring it produces are not worse than the best of the parents. Strong progress implies weak and potential progress.

We proceed with verifying these properties for the geometric search operators under the particular L_z metrics. To be as general as possible, we are not assuming that a search operator in question is geometric under the same metric as the metric in the fitness function. In doing so, we intend to address different combinations of those metrics and possibly come up with recommendations concerning their efficiency. The L_z metric used by the fitness function will be in the following denoted by L_{z_f} , while the semantic distance used to define operator's geometry by L_{z_d} .

We start with the properties of mutation operators.

Theorem 1 The r -geometric mutation under any semantic distance $d \equiv L_{z_d}$ and any fitness function $f \equiv L_{z_f}$ is *potentially progressive*.

Proof Consider a parent p and its fitness under L_{z_f} metric $f(p) = L_{z_f}(s(p), t)$. The L_{z_d} semantic distance between p and its offspring p' located in a ball of radius r centered in $s(p)$ amounts to $d(p, p') = L_{z_d}(p, p') \leq r$. From the Cauchy–Schwarz [29]

and Hölder's [28] inequalities, the distance in n -dimensional space between p and p' is bounded by

$$L_{z_d}(p, p') \leq L_{z_f}(p, p') \leq n^{\frac{1}{z_f} - \frac{1}{z_d}} L_{z_d}(p, p') \leq n^{\frac{1}{z_f} - \frac{1}{z_d}} r,$$

if $z_f \leq z_d$, or otherwise by

$$L_{z_f}(p, p') \leq L_{z_d}(p, p') \leq r.$$

The triangle inequality $f(p) \leq f(p') + L_{z_f}(p, p')$ bounds the offspring's fitness from bottom, and $f(p') \leq f(p) + L_{z_f}(p, p')$ bounds it from top. This may be equivalently rewritten as $f(p') = f(p) \pm L_{z_f}(p, p')$. This in turn is bounded by $f(p') = f(p) \pm n^{\frac{1}{z_f} - \frac{1}{z_d}} r$ if $z_f \leq z_d$, or $f(p') = f(p) \pm r$ otherwise. In conclusion, the offspring may be better than its parent and the r -geometric mutation is PP under all combinations of L_{z_f} and L_{z_d} metrics. The r -geometric mutation is not WP and SP, since the offspring may be also worse than the parent. $\square$

The following three theorems present the properties of crossover operators.

Theorem 2 *A geometric $k \geq 2$ -ary crossover operator under the semantic distance $d \equiv L_{z_d}$ and fitness $f \equiv L_{z_f}$ is weakly progressive if the L_{z_f} ball is a convex set under L_{z_d} .*

Proof Let $p_1, p_2, \dots, p_k$ be the parents. Assume without loss of generality that $f(p_1) = \max_{i=1..k} f(p_i)$. Let B be a set of all semantics of programs not worse than p_1 , i.e., B is an L_{z_f} -ball centered in t with a radius $r = f(p_1) = L_{z_f}(s(p_1), t)$. By definition, $\forall_{i=2..k} f(p_i) \leq r$ and $\{s(p_1), s(p_2), \dots, s(p_k)\} \subseteq B$. By the non-decreasing property of convex hull,¹ an L_{z_d} -convex hull $C(B)$ is a superset of L_{z_d} -convex hulls of all subsets of B , i.e., $C(\{s(p_1), s(p_2), \dots, s(p_k)\}) \subseteq C(B)$. If B is a convex set under L_{z_d} then $B = C(B)$ and $C(\{s(p_1), s(p_2), \dots, s(p_k)\}) \subseteq B$ holds. Otherwise $B \subset C(B)$, thus a pair of parents with semantics in B may exist such that an L_{z_d} -segment connecting them is not contained in B . In conclusion, if B is a convex set under L_{z_d} , the offspring $p' : s(p') \in C(\{s(p_1), s(p_2), \dots, s(p_k)\})$ is not worse than p_1 , i.e., $f(p') \leq f(p_1) = r$, and the geometric $k \geq 2$ -ary crossover operator in question is WP. $\square$

Theorem 3 *The geometric $k \geq 2$ -ary crossover operator under any semantic distance $d \equiv L_{z_d}$ and any fitness $f \equiv L_{z_f}$ is potentially progressive.*

Proof Let $p_1, p_2, \dots, p_k$ be the parents. Assume without loss of generality that $f(p_1) = \min_{i=1..k} f(p_i)$. Let B be the set of semantics of all programs not worse than the best of the parents, i.e., B is an L_{z_f} -ball centered in t with a radius $r = f(p_1) = L_{z_f}(s(p_1), t)$. By definition, $s(p_1) \in B$ and an L_{z_d} -convex hull of parents has a nonempty intersection with B , i.e., $s(p_1) \in C(\{s(p_1), s(p_2), \dots, s(p_k)\}) \cap B$. In conclusion, there exists an offspring $p' : s(p') \in C(\{s(p_1), s(p_2), \dots, s(p_k)\})$ that is

¹ $X \subseteq Y$ implies that $C(X) \subseteq C(Y)$.

not worse than the best of the parents, i.e., $f(p') \leq f(p_1) = r$ and the geometric $k \geq 2$ -ary crossover operator is PP. $\square$

Theorem 4 *A geometric $k \geq 2$ -ary crossover operator under a semantic distance $d \equiv L_{z_d}$ and fitness $f \equiv L_{z_f}$ is strongly progressive if the L_{z_f} -ball is a convex set under L_{z_d} and all parents have equal fitness.*

Proof Consult the proof of Theorem 2, replacing the constraint on parents' fitness from $f(p_1) = \max_{i=1..k} f(p_i)$ to $\forall_{i=2..k} f(p_i) = f(p_1)$, so that it requires all parents to have the same fitness. Then it comes that if a ball built on parents' semantics is a convex set under L_{z_d} , the offspring must be not worse than the worst (and simultaneously the best) parent. $\square$

6 Prerequisites for convexity and bounds of fitness change

Theorems 1 and 3 hold for any combination of the metric used by a fitness function and the metric under which a search operator is geometric. On the other hand, Theorems 2 and 4 are conditioned on the convexity of balls. It is thus interesting to ask: for which combinations of metrics' parameters, i.e., z_f and z_d , is a L_{z_f} -ball a convex set under L_{z_d} ?

In this section we provide an answer to this question. To aid the understanding of the following derivations, in Fig. 6 we present the L_{z_f} balls and their L_{z_d} convex hulls in two dimensions.

Lemma 1 *An L_{z_f} -ball is a convex set under L_1 for $z_f = \infty$.*

Proof Let the semantic distance $d \equiv L_1$. Given an L_{z_f} -ball $B_{z_f}(t, r)$, its L_1 -convex hull is the smallest hyperrectangle that encloses it. The diameter of the ball is $2r$, thus the convex hull's edge length is also $2r$ in each dimension, and every coordinate of a point of the convex hull differs from the coordinate of t by at most r . The distance of this point from t under any L_{z_f} metric in n dimensions is $\sqrt[n]{nr^{z_f}}$. In turn, if $z_f \neq \infty$, then $\sqrt[n]{nr^{z_f}} > r$ and the L_{z_f} -ball is not convex under L_1 . On the other hand if $z_f = \infty$, then $\lim_{z_f \rightarrow \infty} \sqrt[n]{nr^{z_f}} = r$, thus the L_∞ -ball is convex set under L_1 . $\square$

An immediate consequence of this lemma on an application of $k \geq 2$ -ary geometric crossover is formalized below.

Theorem 5 *For a $k \geq 2$ -ary L_1 -geometric crossover, the upper bound on L_{z_f} fitness of an offspring relative to the worst parent is $\sqrt[n]{nr^{z_f}}/r = \sqrt[n]{n}$, where n is dimensionality of semantic space.*

We proceed now with deriving an analogous result for L_2 .

Lemma 2 *An L_{z_f} -ball is a convex set under L_2 for $z_f \geq 1$.*

Proof For L_2 semantic distance, the convex hull is a convex polygon that snaps to the edges of L_{z_f} ball unless $z_f < 1$. Hence the L_{z_f} -ball and its L_2 -convex hull are the

Fig. 6 L_{z_d} convex hulls of L_{z_f} balls in two dimensions for $z_d, z_f \in \{1, 2, \infty\}$

same sets. For $L_{z_f} < 1$, the ball of radius r is not a convex set since the L_2 -convex hull of the $L_{z_f} < 1$ ball is equivalent to L_1 ball of radius r . Under $L_{z_f} < 1$, the point of the L_2 -convex hull that is most distant from t is located in the center of (any) face of the L_2 -convex hull, i.e., at the distance of $\sqrt[n]{n \times (\frac{r}{n})^{z_f}}$ from t , where n is the dimensionality of the space. $\square$

In analogy to L_1 , this lemma also results in an interesting relationship between the fitness of the worst parent and the worst possible offspring.

Theorem 6 For a $k \geq 2$ -ary L_2 -geometric crossover and a fitness function $f \equiv L_{z_f}$, $z_f < 1$, the upper bound on fitness of an offspring relative to the worst parent is $\sqrt[n]{n \times (\frac{r}{n})^{z_f}} / r = n^{\frac{1}{z_f} - 1}$, where n is dimensionality of semantic space.

Finally, we conclude with theorems for L_∞ .

Lemma 3 *An L_{z_f} -ball is a convex set under L_∞ for $z_f = 1$.*

Proof For L_∞ semantic distance, a convex hull takes the form of a L_1 -ball that encloses the entire L_{z_f} ball of radius r . The walls of L_∞ convex hull that are tangential to the L_{z_f} -ball are at the distance r from t under L_{z_f} . The edges of the L_∞ -convex hull have length of $2r$ under L_{z_f} , and the tangent points of the hull and the ball are in the middle of hull's edges. Thus, the L_{z_f} distance to the point of L_∞ -convex hull that is most distant from t (i.e., a vertex of the L_∞ -convex hull polygon) is $r \times 2^{1-\frac{1}{z_f}}$. For $z_f = 1$, this distance amounts to r , hence the L_1 -ball is a convex set under L_∞ . $\square$

Theorem 7 *For a $k \geq 2$ -ary L_∞ -geometric crossover and a fitness function $f \equiv L_{z_f}$, $z_f > 1$, the upper bound on fitness of an offspring relative to the worst parent is $r \times 2^{1-\frac{1}{z_f}}/r = 2^{1-\frac{1}{z_f}}$. For $z_f < 1$, the same bound holds as for L_2 semantic distance, i.e., $n^{\frac{1}{z_f}-1}$, where n is dimensionality of semantic space.*

Table 1 summarizes the results concerning the convexity of particular L_{z_f} -balls under different L_{z_d} metrics for z_d and $z_f = 1, 2$, and ∞ . Table 2 summarizes the maximum relative upper bounds of offspring's fitness with respect to the worst parent's under semantic distance $d \equiv L_{z_d}$ and fitness $f \equiv L_{z_f}$ for $z_f, z_d \in \{1, 2, \infty\}$.

7 Related work

This paper extends our previous work [25] on properties of progress by providing more general proofs that hold for more metrics and combinations of various metrics used in fitness function and by search operators.

Our results are consistent with the previous contributions by Moraglio [18], who showed that if both semantic distance and fitness function are the same metric and

Table 1 Convexity of L_{z_f} -balls under particular L_{z_d} metrics

$L_{z_d} \setminus L_{z_f}$	L_1	L_2	L_∞
L_1	×	×	✓
L_2	✓	✓	✓
L_∞	✓	×	×

Table 2 Maximum relative upper bound of offspring's fitness with respect to the fitness of the worst parent for geometric $k \geq 2$ -ary crossover for n -dimensional semantic space, semantic distance $d \equiv L_{z_d}$ and fitness function $f \equiv L_{z_f}$, $z_d, z_f \in \{1, 2, \infty\}$

$d \setminus f$	L_1	L_2	L_∞
L_1	n	$\sqrt{n}$	1
L_2	1	1	1
L_∞	1	$\sqrt{2}$	2

this metric satisfies Jensen's inequality [9] (e.g., L_2), the binary geometric crossover is weakly progressive. However, the proofs presented here allow different metrics in semantic distance and in fitness function. Also, none of these distance functions is required to fulfill Jensen's inequality, which enables use of a wider class of functions. Thirdly, the Theorems 2–4 hold for $k \geq 2$ -ary crossovers, not only to the binary ones. Thus, the above results are more general in certain respects. On the other hand, our derivations assume Minkowski metrics, which are nevertheless very common in GP.

Other related works concern fitness changes in effect of an application of a search operator. Moraglio showed in [18] that, given the same assumptions as above, a binary geometric crossover produces an offspring p' that has on average not worse fitness than the average fitness of its parents p_1, p_2 , i.e., $\mathbb{E}[f(p')] \leq \mathbb{E}[\frac{1}{2}(f(p_1) + f(p_2))]$. Krawiec [10] showed for binary geometric crossover under L_1 semantic distance and fitness function that, if an offspring is produced with uniform distribution in a convex hull of (a segment between) parents p_1, p_2 , the expected fitness of an offspring is the average of the parents' fitness, i.e., $\mathbb{E}[f(p')] = \mathbb{E}[\frac{1}{2}(f(p_1) + f(p_2))]$. With this study, we complement those results by providing the upper bounds on relative fitness change. Taken together, these properties should improve our understanding of GSGP and help designing better operators.

In another study [22], Moraglio and Sudholt transformed abstract convex evolutionary search (which GSGP is a form of) into so-called Pure Adaptive Search. Moreover, they derived the probability of improving the worst individual in population, under the assumption of using convex hull uniform recombination. The probability of finding the optimum by the convex search algorithm has been also determined in that work.

In a similar vein, Moraglio and Mambrini analyze how changes of fitness caused by geometric operators influence convergence speed of GSGP. In [20] they showed that a GSGP search equipped only with an L_2 -mutation reaches the optimum of a regression problem (with an error smaller than a given threshold ϵ) much faster than $(1 + 1)$ Evolutionary Strategy. Under the same assumptions, in [21] they derived the probabilities and bounds on time of reaching the optimum for a Boolean problem by GSGP running with different methods of geometric mutation. However, all these works assumed the same metric being used by semantic distance and fitness function.

Analogous studies have been also conducted outside GSGP. For instance, Durrett et al. [5] analyzed the probabilities of improvement and time bounds for solving majority and order problems using a simplified variant of GP. However, in absence of two distance metrics, one in fitness function and the other underlying a search operator, these and other non-GSGP works do not directly relate of this study.

8 Discussion

We partition the discussion of the presented results in two threads: one concerning the results on progress properties shown in Sect. 5, and the other concerning the bounds presented in Sect. 6.

Concerning the progress properties, let us start with an observation that all operators (mutations and crossovers) considered in this paper feature the PP property. This is not surprising, given the fact that the balls and segments that form the images of geometric operators always include the parents' semantics. The inequality used in the definition of the PP property is non-strict, so the presence of parents in operator's image guarantees that inequality to hold.

The knowledge on which combinations of z_f and z_d cause an L_{z_f} -ball to be convex under an L_{z_d} metric allows us, in connection with Theorems 2–4, to summarize in Table 3 how the weak, potential, and strong properties of progress hold for the geometric $k \geq 2$ -ary crossovers.

The probably most striking observation concerning Table 3 is that, contrary to the intuitions expressed in Sect. 3, using a search operator that obeys the same metric as the fitness function (i.e., $z_d = z_f$) is not always the best choice. In terms of the WP property, it is beneficial only for $z_d = z_f = 2$. For $z_d = z_f = 1$ and $z_d = z_f = \infty$, WP does not hold.

Among the remaining combinations of z_d and z_f , WP holds only in four cases. The case of $z_d = 1$ and $z_f = \infty$ has limited importance, because using L_∞ as a fitness function is particularly rare in GP (L_∞ compares programs on their worst outcomes across all fitness cases). The case of $z_d = \infty$ and $z_f = 1$ is more interesting: L_1 , equivalent to MAE, is commonly employed in GP. Given the WP property, it may be thus desirable to use an L_∞ search operator in problems that feature L_1 -based fitness function. Nevertheless, an L_∞ -geometric search operator has not been proposed yet, to the best of authors' knowledge.

The remaining two cases, $z_d = 2, z_f = 1$, and $z_d = 2, z_f = \infty$, evidence that the Euclidean distance is particularly useful for providing progress properties. L_2 -geometric operators can be said to be robust by being neutral about the choice of metric in fitness function.

While not questioning the soundness of these results, we should emphasize that it is in general unknown how the properties summarized in Table 3 affect the performance of *specific realizations* of geometric search operators. It is so because our considerations concerned only the images of stochastic search operators (see Definition 6), while abstracting from the *distributions* of the outcomes of operators' applications (or, more formally, the distributions of the corresponding random variables (see the remarks preceding Definition 4). And it is the distributions that determine operators' performance. In an extreme case, a badly designed WP operator may in fact on average perform worse than an operator that does not even have the WP property.

Table 3 Properties of progress of geometric $k \geq 2$ -ary crossovers for semantic distance $d \equiv L_{z_d}$ and fitness function $f \equiv L_{z_f}$, $z_d, z_f \in \{1, 2, \infty\}$

$d \setminus f$	L_1	L_2	L_∞
L_1	PP	PP	WP, PP
L_2	WP, PP	WP, PP	WP, PP
L_∞	WP, PP	PP	PP

The results on the performance bounds, summarized in Table 2, also abstract from distributions, but at least provide us with more quantitative insight. Recall that Table 2 presents the worst (pessimistic) relative change of offspring's fitness with respect to the worst parent under a given semantic distance and metric used by a *minimized* fitness function. Therefore, lower values of this indicator are more desirable. This summary gives clear hints to the users of GSGP systems. And the evidence in literature shows that the use of non-optimal combinations of semantic distance and fitness function metrics is common. For instance, [4, 32] use an L_1 -semantic distance and a (scaled) L_2 -fitness function, and [13] employ an L_2 -semantic distance and an L_1 -fitness function. With this paper, we hope to help the authors of future studies in GSGP, whether theoretical or more practice-oriented, to choose the better combinations.

A weakly progressive operator cannot by definition deteriorate fitness (cf. Definition 7). No wonder then that the WP operator-fitness combinations in Table 3 lead to relative fitness change of 1.0 in Table 2. In turn, any value smaller than 1.0 would indicate that a particular operator-fitness combination has the SP property.

Table 2 points also to configurations that are less efficient. The $z_d = \infty, z_f = 2$ and $z_d = \infty, z_f = \infty$ combinations are still not bad: although in the latter case the offspring can be even twice less fit than the parent, this ratio does not depend on the number of fitness cases. The $z_d = 1, z_f = 1$ and $z_d = 1, z_f = 2$ are much more dangerous in that respect: in the worst-case scenario, the offspring under $z_d = 1, z_f = 1$ can be n times worse than its parent. This is a strong argument against using L_1 -geometric search operators and, implicitly, a hint for preferring the L_2 metric for this role. One of practical implications of this work is thus the general recommendation for favoring L_2 -geometric operators.

We find this result important, because posing regression problems with L_1 as a fitness function is very common. Table 2 demonstrates that by 'defaulting' to L_1 -crossover operator one runs into a risk of producing offspring that are many times worse than their parents. Fortunately, L_1 -crossover is commonly avoided in practice, however not due to these properties we have shown here, but because its realization is cumbersome: it requires generation of a mixing program that returns different random values for particular fitness cases (more formally, its semantics needs to belong to a unit hypercube, see [19]). The L_2 binary crossover is, to the contrary, straightforward to implement by blending the parent programs using a constant random coefficient. The overall recommendation is thus to use of an L_2 -geometric crossover in regression problems.

Due to the discrete nature of Boolean domain, an L_{z_d} -convex hull of parents' semantics for $z_d > 1$ contains only semantics of those parents. This, by Definition 5, precludes an $L_{>1}$ -geometric crossover from producing an offspring that is semantically different from all parents and leads to stagnation. Therefore, in the Boolean domain one is essentially obliged to rely on L_1 -semantic distance. From Table 2 it comes that the highest upper bound of offspring-to-worst-parent fitness ratio is for an L_1 -fitness function, and the lowest for L_∞ . Thus, as odd as it may sound, with respect to fitness change bounds, L_∞ is the best choice for defining fitness function for Boolean problems. However, a GP configuration driven by

L_∞ -fitness is destined to perform bad, as such fitness provides no effective fitness gradient by returning 1 everywhere except from the optimum.

In integer domain, where semantic space features more than two values on each dimension, use of an L_{z_d} -crossover for $z_d > 1$ makes more sense. This is because L_{z_d} -convex hull of semantics of parents may contain other points than solely the parents' semantics. This gives us more flexibility in choice of fitness function, while still maintaining beneficial properties of crossover. For instance, an L_2 -crossover maintains 1.0 upper bound on offspring's fitness to the worst parent's one under L_{z_f} -fitness function for $z_f \in \{1, 2, \infty\}$.

9 Consequences: towards more efficient search drivers

The derivations presented in Sect. 5 and summarized in Tables 2 and 3 revolve around the consequences of using particular values of z in semantic distance and in fitness function. But, one may ask, do we have freedom in choosing them, given that a problem typically comes with a specific distance-based fitness function?

The idea of using metrics in abstraction from a problem may seem unnatural. The arguably conservative stance can be summarized as follows: because the programming task already comes with a built-in metric (Eq. 3), it is that metric that should be adopted to drive the search as well as to design search operators.

The results presented here clearly indicate that blindly sticking to problem-specific metric may be detrimental, because certain combinations of metrics enable dramatic deterioration of fitness. But even more importantly, the rationale for choosing a particular metrics is more profound and pertains to the answer to the fundamental question: what is the goal of a program synthesis task? The formulation at the end of Sect. 2 posed a GP programming task as a search problem; however, some programming tasks are *inherently* optimization problems. A prominent example is symbolic regression, where the goal is to find a program of satisfactory quality (low error), but not necessarily a perfect one. In such tasks, one cannot expect to always bring down program error to zero: such a program may not exist due to an insufficiently expressive instruction set or due to noisy fitness cases. It is natural then to adopt the metrics used in the original objective function as a fitness function as such choice maximizes the likelihood of finding a good solution with respect to this specific measure. Nevertheless, there are no reasons, other than the ones summarized in Table 2, to claim that also search operators should obey the same metric.

This logic does not apply though if the task in question is a search problem (cf. Sect. 2). In such tasks, program synthesis may be deemed successful only if a zero-error program is found (Eq. 2). The means to reach that goal are of secondary importance. For such tasks one is free to choose the combination of metrics that seems most desirable in terms of the characteristics presented in Tables 2 and 3.

These observations resonate with the hypotheses we posed in [12]. If a program synthesis task is a search problem, a conventional fitness function (like the one in Eq. 1) should be considered only as *one of the possible means* to drive the search process. There is no guarantee that this particular means is the best one, and

examples can be given where it is actually deceptive. Better *search drivers* may exist. Designing them may be facilitated by opening the ‘black box’ of evaluation function even wider than in SGP by, e.g., enabling inspection of entire *execution traces* of programs [14–16]. This is the main postulate of *behavioral programming* [12] which aims at better-informed program synthesis algorithms. Interestingly, the behavioral perspective proves effective also beyond GP, among others in coevolutionary algorithms [11, 17].

10 Conclusions and future work

In this paper, we provided a systematic theoretical analysis of the key properties of the abstract geometric semantic search operators under the most popular variants of Minkowski metric. In particular, we showed which combinations of metrics used by the fitness function and geometric search operators are most beneficial in terms of progress properties and deterioration bounds.

This work paves the way to further developments of both theoretical and practical nature. On the theoretical side, the knowledge on the relationships between the geometric entities discussed in this paper should facilitate deriving the actual *probabilities* of improvement and deterioration under a given distribution of operator’s outcomes. On the practical side, even though we worked here with abstract search, we hope to have contributed to design of better geometric search operator. The search operators proposed in [19] are particular realizations of the geometric semantic search paradigm. Though exact and elegant, they are not completely free from shortcomings, in particular from rapid code growth and limited generalization capability [26]. The design of alternative realizations of geometric semantic search operators may (or even should) take into account selected results presented here and shown in earlier work [18]. For instance, we demonstrated that the chances of producing a well-performing offspring for $k \geq 2$ -ary operators is maximized if the parents are located on a sphere (cf. Theorem 4). Our ongoing research aims at augmenting a geometric crossover operator with an appropriate mate selection mechanism [24].

Acknowledgments T. Pawlak acknowledges support from the Polish National Science Centre Grant No. DEC-2012/07/N/ST6/03066. K. Krawiec acknowledges support from the Polish National Science Centre Grant No. 2014/15/B/ST6/05205.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. L. Beadle, C. Johnson, Semantically driven crossover in genetic programming, in *Proceedings of the IEEE World Congress on Computational Intelligence*, ed. by J. Wang (IEEE Computational Intelligence Society, IEEE Press, Hong Kong, 2008), pp. 111–116. doi:10.1109/CEC.2008.4630784, <http://results.ref.ac.uk/Submissions/Output/1423275>

2. L. Beadle, C.G. Johnson, Semantic analysis of program initialisation in genetic programming. *Genet. Program. Evolv. Mach.* **10**(3), 307–337 (2009). doi:[10.1007/s10710-009-9082-5](https://doi.org/10.1007/s10710-009-9082-5). <http://www.springerlink.com/content/yn5p4572316tr487>
3. L. Beadle, C.G. Johnson, Semantically driven mutation in genetic programming. in *2009 IEEE Congress on Evolutionary Computation*, ed. by A. Tyrrell, (IEEE Computational Intelligence Society, IEEE Press, Trondheim, Norway, 2009), pp. 1336–1342. doi:[10.1109/CEC.2009.4983099](https://doi.org/10.1109/CEC.2009.4983099)
4. M. Castelli, L. Vanneschi, S. Silva, Prediction of high performance concrete strength using genetic programming with geometric semantic genetic operators. *Exp. Syst. Appl.* **40**(17), 6856–6862 (2013). doi:[10.1016/j.eswa.2013.06.037](https://doi.org/10.1016/j.eswa.2013.06.037). <http://www.sciencedirect.com/science/article/pii/S0957417413004326>
5. G. Durrett, F. Neumann, U.M. O'Reilly, Computational complexity analysis of simple genetic programming on two problems modeling isolated program semantics. in *Foundations of Genetic Algorithms*, ed. by H.G. Beyer, W.B. Langdon (ACM, Schwarzenberg, Austria, 2011), pp. 69–80. doi:[10.1145/1967654.1967661](https://doi.org/10.1145/1967654.1967661)
6. E. Galvan-Lopez, B. Cody-Kenny, L. Trujillo, A. Kattan, Using semantics in the selection mechanism in genetic programming: a simple method for promoting semantic diversity. in *2013 IEEE Conference on Evolutionary Computation*, ed. by L.G. de la Fraga (Cancun, Mexico, 2013) vol. 1, pp. 2972–2979. doi:[10.1109/CEC.2013.6557931](https://doi.org/10.1109/CEC.2013.6557931)
7. D. Jackson, Phenotypic diversity in initial genetic programming populations. in *Proceedings of the 13th European Conference on Genetic Programming, EuroGP 2010, LNCS*, ed. by A.I. Esparcia-Alcazar, A. Ekart, S. Silva, S. Dignum, A.S. Uyar (Springer, Istanbul, 2010), vol. 6021, pp. 98–109. doi:[10.1007/978-3-642-12148-7_9](https://doi.org/10.1007/978-3-642-12148-7_9)
8. D. Jackson, Promoting phenotypic diversity in genetic programming. in *PPSN 2010 11th International Conference on Parallel Problem Solving From Nature, Lecture Notes in Computer Science*, ed. by R. Schaefer, C. Cotta, J. Kolodziej, G. Rudolph (Springer, Krakow, Poland, 2010), vol. 6239, pp. 472–481. doi:[10.1007/978-3-642-15871-1_48](https://doi.org/10.1007/978-3-642-15871-1_48)
9. J. Jensen, Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Math.* **30**(1), 175–193 (1906). doi:[10.1007/BF02418571](https://doi.org/10.1007/BF02418571)
10. K. Krawiec, Medial crossovers for genetic programming. in *Proceedings of the 15th European Conference on Genetic Programming, EuroGP 2012, LNCS*, ed. by A. Moraglio, S. Silva, K. Krawiec, P. Machado, C. Cotta (Springer Verlag, Malaga, Spain, 2012), vol. 7244, pp. 61–72. doi:[10.1007/978-3-642-29139-5_6](https://doi.org/10.1007/978-3-642-29139-5_6)
11. K. Krawiec, P. Liskowski, Automatic derivation of search objectives for test-based genetic programming. in *18th European Conference on Genetic Programming, LNCS*, ed. by P. Machado, M.I. Heywood, J. McDermott, M. Castelli, P. Garcia-Sanchez, P. Burelli, S. Risi, K. Sim (Springer, Copenhagen, 2015), vol. 9025, pp. 53–65. doi:[10.1007/978-3-319-16501-1_5](https://doi.org/10.1007/978-3-319-16501-1_5)
12. K. Krawiec, U.M. O'Reilly, Behavioral programming: a broader and more detailed take on semantic GP. in *GECCO '14: Proceedings of the 2014 conference on Genetic and evolutionary computation*, ed. by C. Igel, D.V. Arnold, C. Gagne, E. Popovici, A. Auger, J. Bacardit, D. Brockhoff, S. Cagnoni, K. Deb, B. Doerr, J. Foster, T. Glasmachers, E. Hart, M.I. Heywood, H. Iba, C. Jacob, T. Jansen, Y. Jin, M. Kessentini, J.D. Knowles, W.B. Langdon, P. Larranaga, S. Luke, G. Luque, J.A.W. McCall, M.A. Montes de Oca, A. Motsinger-Reif, Y.S. Ong, M. Palmer, K.E. Parsopoulos, G. Raidl, S. Risi, G. Ruhe, T. Schaul, T. Schmickl, B. Sendhoff, K.O. Stanley, T. Stuetzle, D. Thierens, J. Togelius, C. Witt, C. Zarges (ACM, Vancouver, BC, Canada, 2014), pp. 935–942. doi:[10.1145/2576768.2598288](https://doi.org/10.1145/2576768.2598288). Best paper
13. K. Krawiec, T. Pawlak, Locally geometric semantic crossover: a study on the roles of semantics and homology in recombination operators. *Genet. Program. Evol. Mach.* **14**(1), 31–63 (2013). doi:[10.1007/s10710-012-9172-7](https://doi.org/10.1007/s10710-012-9172-7)
14. K. Krawiec, A. Solar-Lezama, Improving genetic programming with behavioral consistency measure. in *13th International Conference on Parallel Problem Solving from Nature, Lecture Notes in Computer Science*, ed. by T. Bartz-Beielstein, J. Branke, B. Filipic, J. Smith (Springer, Ljubljana, Slovenia, 2014), vol. 8672, pp. 434–443. doi:[10.1007/978-3-319-10762-2_43](https://doi.org/10.1007/978-3-319-10762-2_43)
15. K. Krawiec, J. Swan, Guiding evolutionary learning by searching for regularities in behavioral trajectories: a case for representation agnosticism. in *How Should Intelligence Be Abstracted in AI Research: MDPs, Symbolic Representations, Artificial Neural Networks, or ...*, no. FS-13-02 in 2013 AAAI Fall Symposium Series, ed. by S. Risi, J. Lehman, J. Clune (AAAI Press, Arlington, Virginia, USA, 2013), pp. 41–46. URL <http://www.aaai.org/ocs/index.php/FSS/FSS13/paper/view/7590>
16. K. Krawiec, J. Swan, Pattern-guided genetic programming, in *Proceedings of the 15th international conference on Genetic and evolutionary computation conference, GECCO '13*. ACM, Amsterdam, The Netherlands (2013)

17. P. Liskowski, K. Krawiec, Discovery of implicit objectives by compression of interaction matrix in test-based problems. in *Parallel Problem Solving from Nature – PPSN XIII, Lecture Notes in Computer Science*, ed. by T. Bartz-Beielstein, J. Branke, B. Filipič, J. Smith (Springer, 2014), vol. 8672, pp. 611–620. doi:[10.1007/978-3-319-10762-2_60](https://doi.org/10.1007/978-3-319-10762-2_60)
18. A. Moraglio, Abstract convex evolutionary search. in *Foundations of Genetic Algorithms*, ed. by H.G. Beyer, W.B. Langdon (ACM, Schwarzenberg, Austria, 2011), pp. 151–162. doi:[10.1145/1967654.1967668](https://doi.org/10.1145/1967654.1967668)
19. A. Moraglio, K. Krawiec, C.G. Johnson, Geometric semantic genetic programming. in *Parallel Problem Solving from Nature, PPSN XII (part 1), Lecture Notes in Computer Science*, ed. by C.A. Coello Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, M. Pavone (Springer, Taormina, Italy), vol. 7491, pp. 21–31. doi:[10.1007/978-3-642-32937-1_3](https://doi.org/10.1007/978-3-642-32937-1_3)
20. A. Moraglio, A. Mambrini, Runtime analysis of mutation-based geometric semantic genetic programming for basis functions regression. in *GECCO '13: Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*, ed. by C. Blum, E. Alba, A. Auger, J. Bacardit, J. Bongard, J. Branke, N. Bredeche, D. Brockhoff, F. Chicano, A. Dorin, R. Doursat, A. Ekart, T. Friedrich, M. Giacobini, M. Harman, H. Iba, C. Igel, T. Jansen, T. Kovacs, T. Kowaliw, M. Lopez-Ibanez, J.A. Lozano, G. Luque, J. McCall, A. Moraglio, A. Motsinger-Reif, F. Neumann, G. Ochoa, G. Olague, Y.S. Ong, M.E. Palmer, G.L. Pappa, K.E. Parsopoulos, T. Schmickl, S.L. Smith, C. Solnon, T. Stuetzle, E.G. Talbi, D. Tauritz, L. Vanneschi (ACM, Amsterdam, The Netherlands, 2013), pp. 989–996. doi:[10.1145/2463372.2463492](https://doi.org/10.1145/2463372.2463492)
21. A. Moraglio, A. Mambrini, L. Manzoni, Runtime analysis of mutation-based geometric semantic genetic programming on boolean functions. in *Foundations of Genetic Algorithms*, ed. by F. Neumann, K. De Jong (ACM, Adelaide, Australia, 2013), pp. 119–132. doi:[10.1145/2460239.2460251](https://doi.org/10.1145/2460239.2460251). http://www.cs.bham.ac.uk/~axm322/pdf/gsgp_foga13.pdf
22. A. Moraglio, D. Sudholt, Runtime analysis of convex evolutionary search. in *GECCO*, ed. by T. Soule, J.H. Moore (ACM, 2012), pp. 649–656. <http://dblp.uni-trier.de/db/conf/gecco/gecco2012.html#MoraglioS12>
23. T. Pawlak, Combining semantically-effective and geometric crossover operators for genetic programming, in *13th International Conference on Parallel Problem Solving from Nature, Lecture Notes in Computer Science*, ed. by T. Bartz-Beielstein, J. Branke, B. Filipic, J. Smith (Springer, Ljubljana, Slovenia, 2014), vol. 8672, pp. 454–464. doi:[10.1007/978-3-319-10762-2_45](https://doi.org/10.1007/978-3-319-10762-2_45)
24. T.P. Pawlak, Competent algorithms for geometric semantic genetic programming. Ph.D. thesis, Poznan University of Technology, Poznań, Poland (2015). <http://www.cs.put.poznan.pl/pawlak/link/?PhD>
25. T.P. Pawlak, K. Krawiec, Guarantees of progress for geometric semantic genetic programming. in *Semantic Methods in Genetic Programming. Ljubljana*, ed. by C. Johnson, K. Krawiec, A. Moraglio, M. O'Neill (Slovenia, 2014). <http://www.cs.put.poznan.pl/kkrawiec/smgp2014/uploads/Site/Pawlak.pdf>. Workshop at Parallel Problem Solving from Nature 2014 conference
26. T.P. Pawlak, B. Wieloch, K. Krawiec, Review and comparative analysis of geometric semantic crossovers. *Genet. Program. Evol. Mach.* doi:[10.1007/s10710-014-9239-8](https://doi.org/10.1007/s10710-014-9239-8)
27. T.P. Pawlak, B. Wieloch, K. Krawiec, Semantic backpropagation for designing search operators in genetic programming. *IEEE Trans. Evolut. Comput.* **19**(3), 326–340 (2015). doi:[10.1109/TEVC.2014.2321259](https://doi.org/10.1109/TEVC.2014.2321259)
28. L.J. Rogers, An extension of a certain theorem in inequalities. *Messenger Math.* **17**, 145–150 (1888)
29. J.M. Steele, *The Cauchy-Schwarz Master Class: An Introduction to the Art of Mathematical Inequalities* (Cambridge University Press, New York, NY, USA, 2004)
30. N.Q. Uy, N.X. Hoai, M. O'Neill, R.I. McKay, E. Galvan-Lopez, Semantically-based crossover in genetic programming: application to real-valued symbolic regression. *Genet. Program. Evol. Mach.* **12**(2), 91–119 (2011). doi:[10.1007/s10710-010-9121-2](https://doi.org/10.1007/s10710-010-9121-2)
31. N.Q. Uy, N.X. Hoai, M. O'Neill, R.I. McKay, D.N. Phong, On the roles of semantic locality of crossover in genetic programming. *Inf. Sci.* **235**, 195–213 (2013). doi:[10.1016/j.ins.2013.02.008](https://doi.org/10.1016/j.ins.2013.02.008). <http://www.sciencedirect.com/science/article/pii/S0020025513001175>
32. L. Vanneschi, M. Castelli, L. Manzoni, S. Silva, A new implementation of geometric semantic GP and its application to problems in pharmacokinetics. in *Proceedings of the 16th European Conference on Genetic Programming, EuroGP 2013, LNCS*, ed. by K. Krawiec, A. Moraglio, T. Hu, A.S. Uyar, B. Hu (Springer Verlag, Vienna, Austria, 2013), vol. 7831, pp. 205–216. doi:[10.1007/978-3-642-37207-0_18](https://doi.org/10.1007/978-3-642-37207-0_18)
33. S. Wright, The roles of mutation, inbreeding, crossbreeding and selection in evolution. *Proc. Sixth Int. Congr. Genet.* **1**, 356–366 (1932)