The influence of population size in geometric semantic GP

Mauro Castelli [a,*], Luca Manzoni [c], Sara Silva [b], Leonardo Vanneschi [a], Aleš Popovič [d]

[a] NOVA IMS, Universidade Nova de Lisboa, 1070-312 Lisboa, Portugal
[b] LabMAg, FCUL, Universidade de Lisboa, 1749-016 Lisboa, Portugal
[c] Dipartimento di Informatica Sistemistica e Comunicazione (DISCo), University of Milano Bicocca, 20126 Milan, Italy
[d] University of Ljubljana, Faculty of Economics, Kardeljeva ploscad 17, 1000 Ljubljana, Slovenia

ARTICLE INFO

Accepted 31 May 2016

Keywords:
Genetic programming
Semantics
Population size

ABSTRACT

In this work, we study the influence of the population size on the learning ability of Geometric Semantic Genetic Programming for the task of symbolic regression. A large set of experiments, considering different population size values on different regression problems, has been performed. Results show that, on real-life problems, having small populations results in a better training fitness with respect to the use of large populations after the same number of fitness evaluations. However, performance on the test instances varies among the different problems: in datasets with a high number of features, models obtained with large populations present a better performance on unseen data, while in datasets characterized by a relative small number of variables a better generalization ability is achieved by using small population size values. When synthetic problems are taken into account, large population size values represent the best option for achieving good quality solutions on both training and test instances.

1. Introduction

As reported in several studies (see for instance [8,4,5,16,37,39]) the performance of Genetic Programming (GP) [25] is strongly dependent on the value of a set of parameters. Among those parameters, one that has a deep impact on GP's functioning is the size of the population, i.e. the number of candidate solutions that are evolved. In particular, the size of the population is involved in several phenomena that characterize GP. For instance, population size and population diversity are related to premature convergence [45,7] and it was also hypothesized that the population size is related to the occurrence of bloat [34]. Furthermore, while existing studies suggest that bloat and overfitting are unrelated phenomena [43], other studies hint the existence of a relation between these phenomena [32]. Under this perspective, an incorrect choice of the population size may be one of the reasons for the overfitting of training data. For all these reasons, an accurate choice of the value of this parameter is often crucial. For instance, a small population may result in premature convergence or in poor performance of GP. On the other hand, a large population may cause a slowdown of the algorithm due to the high number of fitness evaluations that are needed.

The study of the parameters that characterize GP is an important hot topic, in particular when geometric semantic operators, defined by Moraglio and coauthors in 2013 [28], are used to explore the search space [44]. The definition of these operators, in fact, has opened a new research line in the GP community, and a lot of theoretical studies have appeared [29,31]. Besides being grounded in a strong body of theory, the use of these genetic operators has produced substantially better results, compared to standard GP, on a number of problems, both benchmarks [42] and real-world applications [12,13].

The objective of this paper is to study the role of population size on the learning process of GP when geometric semantic operators are used. In particular, we want to investigate the role of the population size in achieving good quality models, both on training and unseen data. This study has been performed considering several test problems and different population size values, including GP with only one candidate solution in the population.

The paper is organized as follows: Section 2 presents previous works related to the importance of population size in evolutionary computation, pointing out some interesting findings; Section 3 reports the definition of the geometric semantic operators presented in [28]; Section 4 presents the experimental settings and the obtained results, discussing the effect of different population size values on the learning process. In particular, an analysis of the quality of the obtained models and their ability to generalize on unseen data is proposed. Finally, Section 5 concludes the paper and provides hints for future research directions.

[*] Corresponding author.
E-mail addresses: mcastelli@novaims.unl.pt (M. Castelli), luca.manzoni@disco.unimib.it (L. Manzoni), sara@fc.ul.pt (S. Silva), lvanneschi@novaims.unl.pt (L. Vanneschi), apopovic@novaims.unl.pt (A. Popovič).

2. Population size: previous and related work

The study of the effect of the population size in evolutionary algorithms has been investigated in several works so far. In this section a brief literature review is presented, in order to frame our work in the context of the existing studies. The first studies that have appeared concern Genetic Algorithms (GAs): in [18] the notion of genetic drift was introduced and a study on the relation between genetic drift and population size has been reported. Genetic drift was defined as an effect based on stochastic property of the algorithm. Consider having population of single digit binary strings, the first half of them is “l”, and the second one is “0”. If we will choose strings by chance for the creation of the new generation, we can expect getting equal quantity of different strings. In real, as generations passed, we will observe increase of heterogeneity, which can finally lead to the disappearance of the definite type strings from population. This phenomenon of loss of strings and their parts was called allele loss. In [18], the author observed that increasing the population size we can reduce not only allele loss, but also genetic drift significantly. In the same work, the author also showed that, in large populations, fitness improves more slowly in the initial phase of the evolution but the overall result is better than the one achieved with a small population. In [22], the results of experiments to determine the optimum population size and mutation rate were presented for a simple real genetic algorithm defined to solve problems in the electromagnetic domain. A general domain-independent theoretical analysis of the optimal population size is provided in [20]. In [41] a deeper look on the problem of the population size and limit of generations was taken. The research involved investigation of GA's behavior on different types of functions (i.e., functions with many optima, functions with limits in search space and NP-hard functions). Results have shown that a population size increase improves the performance of GAs and affects the results more than changing the number of generations.

Other studies have investigated the relation between population size and selection or crossover. For instance, the authors of [15] investigated the relation between the population size and the selection strategy. They showed that the number of individuals needed to find a solution is much larger for proportionate selection strategy then for the truncation selection. The GA algorithm they used did not take into consideration mutation rate and it used only one-point crossover. In the same way, the authors of [17] investigated joint influence of crossover type and population size on the performance of GAs. They used an analysis of strings disruption by crossover and its interaction with population sizing. The authors have shown that in small populations more disruptive crossover (uniform and n-point crossover operators) is more effective, and the use of less disruptive crossover (1- and 2-point crossover) in large populations leads to better performance. The possibility of using a GA to control the parameters of another GA has also been investigated. In [21] a meta-GA is applied with interesting results to control the parameters of another GA, including the population size. Several studies have tried to adapt the size of the population during the search process. In [1], the authors pointed out how the size of the population can be critical in many GA applications. They proposed an adaptive method for maintaining variable population size, which grows and shrinks together according to some characteristics of the search. Experimental results indicated some advantages of the proposed method. In the same vein, the work reported in [38] presents an algorithm which adjusts the population size with respect to the probability of selection error. More recently, the authors of [36] performed an experimental study about the importance of population size for the algorithm's performance. Results have shown that tuning the population size is not an easy task and the impact of that choice on the algorithm's performance is significant.

Regarding GP, some of the most relevant results are due to the work of Poli and collaborators. Based on their previous theoretical results obtained on GAs [33], they have been able to establish a schema theory that is able to bind the population size of GP to the code growth and the convergence rate. Using this theory, as better specified in [35], they discovered that, in standard GP, there is a direct relationship between the growth in the size of the individuals in the population and number of individuals in the population itself. In synthetic terms, in large populations individuals tend to grow more rapidly than in small populations. As a direct consequence, if we plot the evolution of fitness against the number of fitness evaluations (as we will do also in the experimental part of this paper), populations of large size are generally penalized. More in particular, adding individuals to a population is beneficial for the effectiveness of GP until a certain threshold, which depends on the particular problem that GP is trying to solve. When this threshold is reached, further increasing the population size produces a large computational effort, that is not anymore compensated by the obtained gain in terms of fitness. Interestingly, this theoretical study (strongly corroborated by experimental evidence in [35]) was only done for fitness on the training set. To the best of our knowledge, no studies have appeared so far aiming at studying the relationship between the population size and the generalization ability of GP. We speculate that the reason for this lack of study is due to an idea, widely diffused among researchers in machine learning, including GP, according to which larger individuals are likely to overfit. As such, given that large populations tend to generate large individuals, large populations have implicitly been considered as being affected by overfitting. Our work is very different from the work of Poli and collaborators. In fact, we do not consider standard GP, but geometric semantic GP (GSGP). The theoretical results of Poli and collaborators cannot for sure be applied to GSGP for the simple reason that, as it will be clear later in this paper, in GSGP the growth rate of the individuals in the population is fixed and independent from the population size. This is the first, and possibly most important, motivation for our work: for the first time, we want to understand the impact of population size on the performance of GSGP. Furthermore, we think that such a study cannot overlook the generalization ability of the studied system. For this reason, great attention is dedicated in this work to results obtained by GSGP on testing data, unseen at training time.

A contribution of a slightly different nature has been proposed in [45], where the authors studied a GP variable population size for dynamic optimization problems. Another contribution is described in [19], where the authors proposed a method for reducing the size of populations at a linear rate. This was achieved by removing a fixed number of individuals at each generation. This technique was called plague and it has been shown to have some positive effects on GP performance. A refinement of this work has been proposed in [40], where an extension of the plague technique, aimed at varying the population size in an intelligent way during the execution of each GP run, was presented. In that model, add and suppression of individuals are operated dynamically on the basis of the behavior of the GP system: population size is decreased while the algorithm is progressing (i.e. fitness is improving) and it is increased when the algorithm reaches the stagnation phase.

3. Geometric semantic operators

Even though the term semantics can have several different interpretations, it is a common trend in the GP community (and this is what we do also here) to identify the semantics of a solution with the vector of its output values on the training data [44]. Under this perspective, a GP individual can be identified with a point (its semantics) in a multidimensional space that we call semantic space. The term Geometric Semantic Genetic Programming (GSGP) indicates a recently introduced variant of GP in which traditional crossover and mutation operators are replaced by the so-called geometric semantic operators, which exploit semantic awareness and induce precise geometric properties on the semantic space.

Geometric semantic operators, introduced by Moraglio et al. [28], are becoming more and more popular in the GP community [44] because of their property of inducing a unimodal fitness landscape on any problem consisting in matching sets of input data onto known targets (like for instance supervised learning problems such as regression and classification). To have an intuition of this property (whose proof can be found in [28]), let us first consider a GA problem in which the unique global optimum is known and the fitness of each individual (to be minimized) corresponds to its distance to the global optimum (our reasoning holds for any employed distance). In this problem, if we use, for instance, ball mutation[1] [26] (i.e. a variation operator that slightly perturbs some of the coordinates of a solution), then any possible individual different from the global optimum has at least one fitter neighbor (individual resulting from its mutation). So, there are no local optima. In other words, the fitness landscape is unimodal, and consequently the problem is characterized by a good evolvability.

Now, let us consider the typical GP problem of finding a function that maps sets of input data into known target values (as we said, regression and classification are particular cases). The fitness of an individual for this problem is typically a distance between its predicted output values and the target ones (error measure). Geometric semantic operators simply define transformations on the syntax of the individuals that correspond to geometric crossover and ball mutation in the semantic space, thus allowing us to map the considered GP problem into the previously discussed GA problem. In particular,[2] *geometric semantic crossover* generates the expression $T_{XO} = (T_1 \cdot T_R) + ((1 - T_R) \cdot T_2)$ as the unique offspring of parents $T_1, T_2 \colon \mathbb{R}^n \to \mathbb{R}$, where T_R is a random real function whose output values range in the interval $[0, 1]$. Analogously, *geometric semantic mutation* returns the expression $T_M = T + ms \cdot (T_{R1} - T_{R2})$ as the result of the mutation of an individual $T \colon \mathbb{R}^n \to \mathbb{R}$, where T_{R1} and T_{R2} are random real functions with codomain in $[0, 1]$ and ms is a parameter called mutation step.

As Moraglio et al. point out, these operators create much larger offspring than their parents and the fast growth of the individuals in the population rapidly makes fitness evaluation unbearably slow, making the system unusable. In [42,10], a possible workaround to this problem was proposed, consisting in an implementation of Moraglio's operators that makes them not only usable in practice, but also very efficient. Basically, this implementation is based on the idea that, besides storing the initial trees, at every generation it is enough to maintain in memory, for each individual, its semantics and a reference to its parents. As shown in [42], the computational cost of evolving a population of n individuals for g generations is $O(ng)$, while the cost of evaluating a new, unseen, instance is $O(g)$.

[1] Similar considerations hold for many types of crossover, including various kinds of geometric crossover [26].

[2] Here we report the definition of the geometric semantic operators as given by Moraglio et al. for real functions' domains, since these are the operators we will use in the experimental phase. For applications that consider other types of data, the reader is referred to [28].

4. Experimental study

4.1. Test problems and experimental settings

For the experimental study presented in this section, we have decided to consider eight different test problems: six of them are complex real-life problems, while two of them are well-known theoretical synthetic functions. All these problems have been widely used as benchmarks for GP and a discussion of all of them can be found in [27]. The objective of three of the real-life problems taken into account is the prediction of different pharmacokinetic parameters of potentially new drugs: human oral bioavailability (%F), median lethal dose (LD50) and protein plasma binding level (PPB). For a discussion of these problems the reader is referred to [27,46]. We have also employed three other real-life problems related to the prediction of more "physical" properties: the airfoil self-noise (ASN), the concrete compressive strength (CCS), and the concrete slump test (CST). The remaining benchmarks are two synthetic functions. The first one, called Keijzer-6, has been defined and used for the first time in [24], the second one, called Vladislavleva-14, has been introduced in [47] and successively both have been used in several other studies on GP.

A short description of the employed datasets is reported in Table 1, where, for each test problem, we reported the number of variables (features) and the number of instances.

Regarding the experimental settings, we performed 100 runs of GSGP using the implementation described in [10], for each considered population size value. In particular, the population sizes that we have tested were 1, 2, 5, 10, 20, 50, 100, 200, 500 and 1000 individuals. For the six studied real-life problems, in each independent run we used a different random partition of the data into training and test set: 70% of the data, chosen at random with uniform distribution, has been used as training set, while the remaining 30% has been used as test set. This random split of the dataset has not been used for the considered synthetic benchmarks, where the training and test sets specified in [27] have instead been considered. The experimental study we present consists in two different parts: in the first part, geometric semantic mutation was the only genetic operator used, while in the second part a combination of crossover and mutation has been considered. For the latter part, crossover probability was fixed to 0.9 and the mutation probability to 0.5. The analysis of geometric semantic mutation as the only genetic operator used to evolve the population has been taken into account considering that previous studies [30,9] have demonstrated that, in GSGP, mutation plays a major role for achieving good quality solutions. When a population consisting of a single individual has been considered, it is important to distinguish two cases: when mutation was the only genetic operator used in the search process, it was applied with

Table 1
Characteristics of the test problems considered in this work. For each problem, we report the type (real world or synthetic problem), the number of features and the number of instances of the dataset and a bibliographic reference where it is possible to find a description of the problem.

Problem	Type	# Variables	# Instances	Reference
Airfoil Self-Noise (ASN)	Real world	5	1502	[6]
Bioavailability (%F)	Real world	241	359	[2]
Concrete Compressive Strength (CCS)	Real world	8	1029	[12]
Protein Plasma Binding Level (PPB)	Real world	626	131	[3]
Concrete Slump Test (CST)	Real world	9	102	[23]
Drug Toxicity (TOX)	Real world	626	234	[3]
Keijzer-6	Synthetic	1	170	[24]
Vladislavleva-14	Synthetic	6	6024	[47]

probability 1; when both crossover and mutation have been considered, the crossover has been replaced by the reproduction of the single individual in the population. This produces the same result, on the semantics of the individuals, as performing a geometric semantic crossover between two copies of the same individual *T*. Regarding the geometric semantic mutation, a mutation step *ms* equal to 1 has been used. The effect of crossover and mutation operators in GSGP has been investigated in [9,11]. In particular, the former work shows that mutation is used by GSGP for exploring the search space, while crossover is used for exploiting and further improving good quality solutions. The latter work provides guidelines to set crossover and mutation rates and, following those guidelines as well as considering previous work [46] where the same test problems have been used, we came out with the crossover and mutation rates previously outlined. Compared to the usual mutation rates in traditional evolutionary algorithms, it may be observed that in GSGP mutation rate is an unconventionally high value. This is coherent with respect to previous studies that have experimentally demonstrated the importance of mutation in GSGP [30,9]. The selection phase was performed using tournament selection with a tournament size equal to 4. The maximum initial depth and the depth of the random trees used by the geometric semantic operators were fixed to 6, while the initialization was done using a ramped-half-and-half method. The functional symbols used were the binary arithmetic operators, including division, protected as in [25]. Besides a number of variables equal to the number of features in the dataset, random constants between -100 and 100 were also allowed in the terminal symbols set. In order to perform a fair comparison between the different GSGP configurations (i.e., configurations with different population sizes), we considered as termination criterion the number of fitness evaluations performed by the system. More in particular, the search process terminates after 10^5 fitness evaluations. Elitism (i.e. unchanged copy of the best individual in the next population) was used, and fitness has been calculated as the root mean square error (RMSE) between calculated and target values. In all the plots shown in the continuation, we consider the median calculated over the 100 independent runs. We preferred the median with respect to the average because it is known to be more robust to outliers. All the used parameters are summarized in Table 2.

4.2. *Experimental results*

As reported in Section 1, the objective of this study is to understand whether the population size influences the performance of GSGP, considering both the quality of the final model and its ability to generalize on unseen data. We begin the analysis of the obtained results by discussing the quality of the model produced at the end of the GP evolution on the training set when only mutation has been used. The results are reported in Fig. 1. For the ASN dataset (Fig. 1(a)), populations from 1 and up to 100 individuals produce comparable results, while bigger populations with 200, 500 and 1000 individuals result in worse quality solutions. In particular, the worsening in terms of fitness follows the increase of the population size. The best performer on this problem is the population with only 1 individual.

Table 2
Values of the parameters used in the experimental study. When the value of the population size was equal to 1, only mutation was used.

Parameter	Value
Population size	1, 2, 5, 10, 20, 50, 100, 200, 500, and 1000
Fitness evaluations	10^5
Tournament size	4
Crossover rate	0.9
Mutation rate	0.5
Functional symbols	$+, -, *, //$
Terminal symbols	Number of features of the dataset and random constants in $[-100; 100]$

Considering the %F problem (Fig. 1(b)), it is possible to see that the best performance is achieved considering the population with only one individual. Also, it is interesting to remark that, with the other considered population size values, GSGP results are better as the population size decreases. In other words, for the %F dataset, in order to achieve the best model on the training set, the best option is to use a small population.

When the CCS problem (Fig. 1(c)) is taken into account, it is possible to notice a similar behavior with respect to the previous test problems. In detail, populations that consist of 1, 2, 5, 10 and 20 individuals produce comparable good quality solutions at the end of the search. Then, increasing the population size from 50 and up to 1000 individuals will result in an increasing (i.e., worsening) of the quality of the individuals returned at the end of the evolutionary process. Finally, also for this problem, the best numerical fitness values have been achieved with a population size equal to 1.

For the PPB dataset (Fig. 1(d)), with population size values up to 50, we have obtained comparable results at the end of the search process. Nonetheless, also in this case, a GP practitioner would probably prefer small populations: in fact, small populations converge to a good quality solution in a smaller number of fitness evaluations compared to the ones needed by larger populations. Considering population size values from 100 and up to 1000, it is possible to see a serious degradation in terms of fitness, with the largest population that produces, also in this test problem, the worst performance.

Taking into account the CST dataset (Fig. 1(e)), it is possible to draw similar considerations with respect to the previous datasets: better training fitness values are achieved with smaller populations and, while population size values up to 100 produce comparable results, it is important to underline that smaller populations (with 1, 2, 5 and 10 individuals) are able to converge towards good quality solutions in a smaller number of fitness evaluations with respect to bigger population size values.

Considering the LD50 dataset (Fig. 1(f)), a behavior similar to the one already observed for the %F dataset is visible: GSGP produces better results as the population size decreases. Nevertheless, analyzing the performance achieved by GSGP on the LD50 dataset, it is possible to see a different characteristic: when large populations are considered (from 50 and up to 1000 individuals), the search process begins with lower fitness values with respect to the ones observable with the other population size values. This fact can be explained considering the difficulty of the problem: GSGP requires a large number of fitness evaluations to obtain good quality solutions on this problem (this fact was already well-known, as it was discussed in [46]). The fact that there is a low fitness value at the beginning of the run is simply related to the large number of individuals in the population: considering the random initialization process, there is a high probability of producing a better quality solution than the one generated, for instance, with a population of 5 or 10 individuals. Anyway, considering the fitness values at the end of the search process, the same considerations as for the %F dataset still hold. The fitness values related to the two synthetic problems (Keijzer-6 and Vladislavleva-14) are reported in Fig. 1(g) and (h) respectively. When the Keijzer-6 problem is taken into account, the behavior that can be observed is different from the one obtained with the other benchmark problems. In particular, the performance of GSGP increases as the population size increases. The same behavior can be observed for Vladislavleva-14 (Fig. 1(h)), where better results are

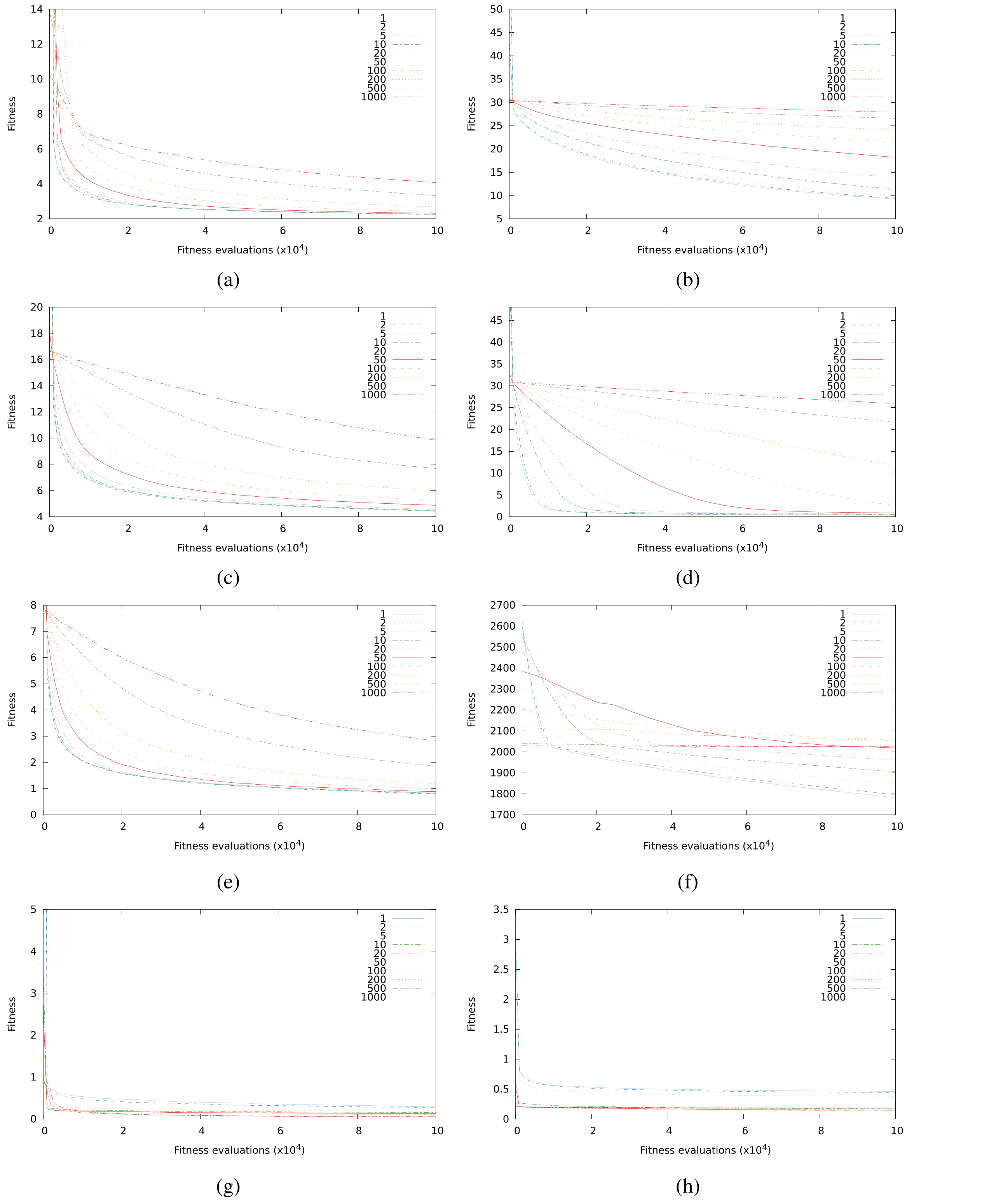

Fig. 1. Training fitness for the considered benchmark problems. Only geometric semantic mutation has been used in these experiments. (a) ASN, (b) %F, (c) CCS, (d) PPB, (e) CST, (f) LD50, (g) Keijzer-6, and (h) Vladislavleva-14. Medians over 100 independent runs are reported.

achieved by using larger population size values. The behavior of the two benchmark problems can be explained by focusing on their characteristics: these are benchmark problems that have been designed to be unlikely solvable by GP (as the reader can see from the original papers where they have been introduced). Hence, for this kind of benchmarks, recombination operators are basically useless, while it is much more easy to achieve better fitness values simply by randomly generating a large number of individuals. A further corroboration of this analysis is given by the fitness of the best solutions produced by GP during the considered amount of fitness evaluations: basically, solutions returned by GP have the same quality as the ones (randomly) produced at the beginning of the search.

To conclude the analysis of the performance achieved on the

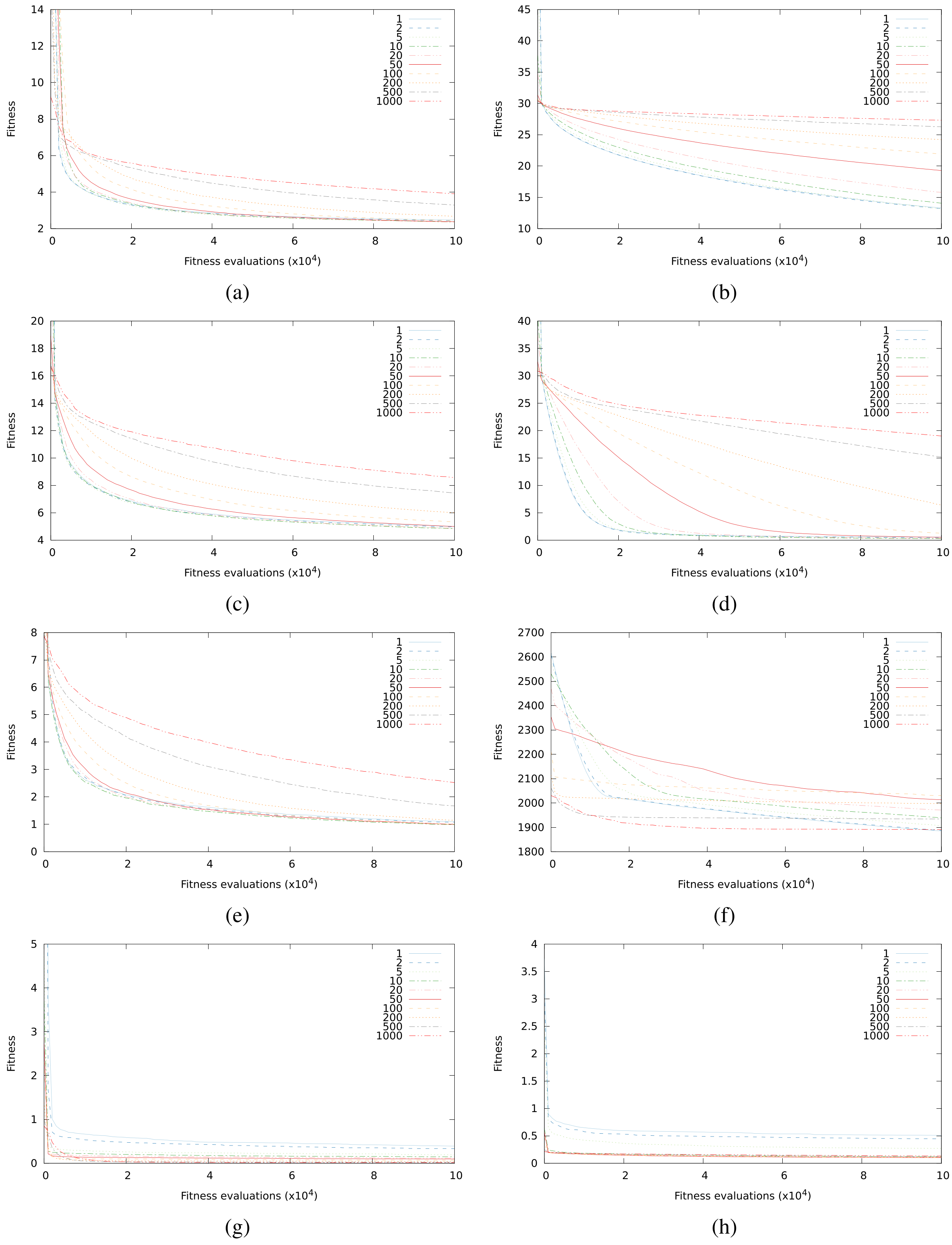

Fig. 2. Training fitness for the considered benchmark problems. Contrarily to Fig. 1, both geometric semantic crossover and mutation have been used here. (a) ASN, (b) %F, (c) CCS, (d) PPB, (e) CST, (f) LD50, (g) Keijzer-6, and (h) Vladislavleva-14. Medians over 100 independent runs are reported.

training instances, we now present the results obtained considering both the geometric semantic operators. These results are reported in Fig. 2. We observe that on all the considered test problems and for all the considered population size values, the same qualitative results obtained considering only the mutation operator have been obtained again. However, an important difference emerges: in two of the considered problems, the numerical fitness values obtained at the end of the search process are larger (i.e., worse) than the ones obtained using only mutation. This result corroborates previous findings and confirms that, in several cases, GSGP finds the best solutions when only geometric semantic mutation is used [46]. Interestingly, the relation between the population size and the quality of the solutions found by GSGP on the training set does not seem to depend on the genetic operators that are considered. In other terms, using only mutation or considering both the genetic operators results in a comparable

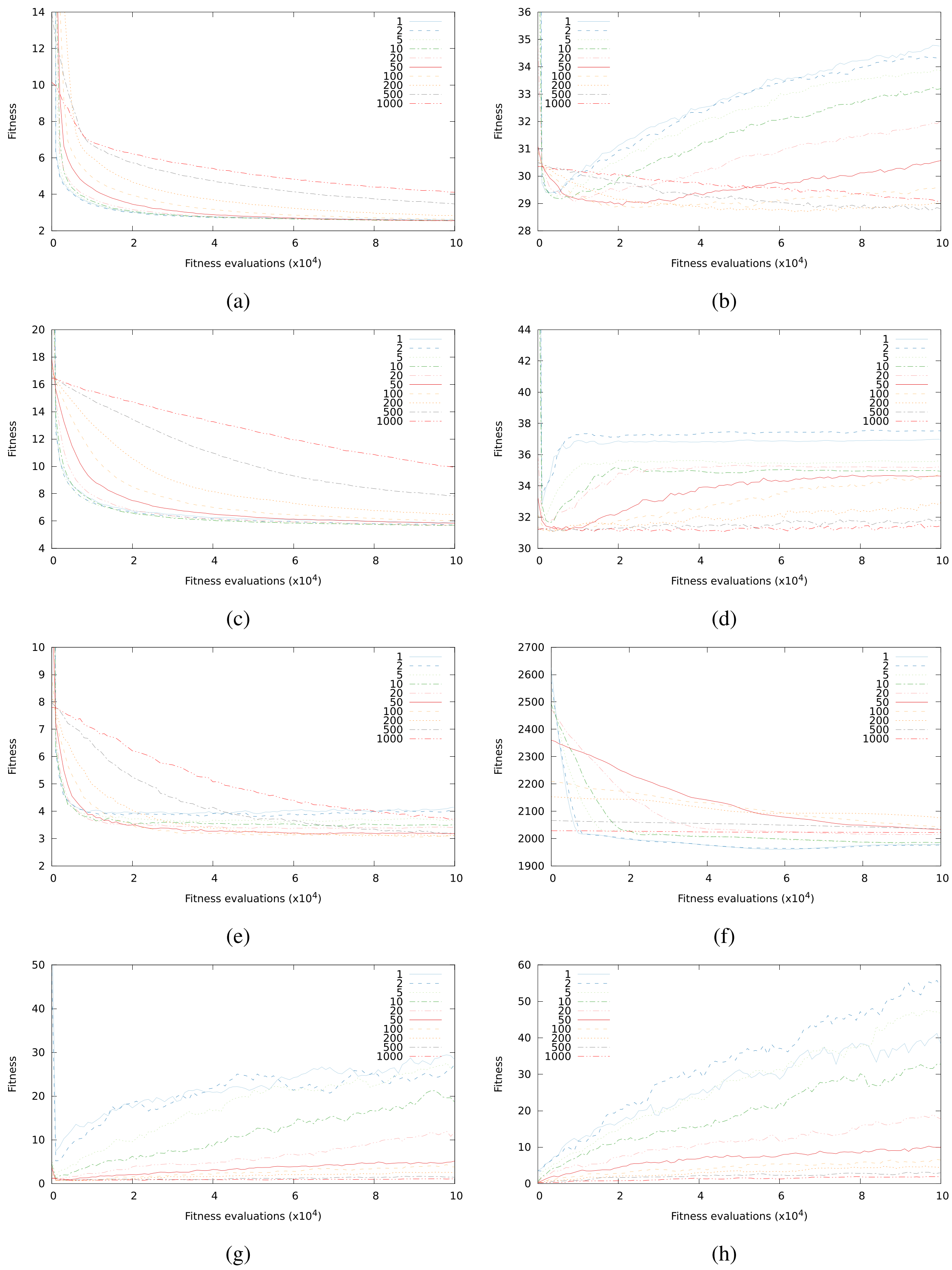

Fig. 3. Test fitness for the considered benchmark problems. Only geometric semantic mutation has been used in these experiments. (a) ASN, (b) %F, (c) CCS, (d) PPB, (e) CST, (f) LD50, (g) Keijzer-6, and (h) Vladislavleva-14. Medians over 100 independent runs are reported.

behavior of the training fitness achieved with the different studied population size values.

To summarize the results we have obtained on the training set, for the studied real-life applications small populations always outperform large ones. On the other hand, for the two synthetic problems, better results have been found using larger populations. As previously explained, this is mainly related to the characteristics of these problems.

The result obtained on the six real-life problems is interesting because it allows GP practitioners to save a lot of computational effort by relying on a small number of individuals. Anyway, it is fundamental to also study the generalization ability of the obtained models, in order to understand whether a relation between the performance on unseen data and the population size exists. To accomplish this objective, we perform an analysis like the one considered for the training instances, but this time we report the results obtained on the test set. Again, we first present the results we have obtained using only mutation. These results are reported in Fig. 3. Considering the ASN dataset (Fig. 3(a)) and the CCS problems (Fig. 3(c)), it is possible to see that they present the same behavior observed on the training instances. In detail, also on unseen instances, better results are achieved by using small population size values. Moreover, while population size values from 1 and up to 100 produce comparable fitness values at the end of the search, the speed of convergence is greater for the smallest populations.

Different results can be observed taking into account the %F (Fig. 3(b)) and the PPB (Fig. 3(d)) problems. In this case, better results are obtained as the population size increases. Hence, on unseen instances, GSGP produces results that are completely different from the ones achieved on the training instances on the same datasets: while better training fitness values can be obtained with small populations, in order to have a better generalization ability large populations should be considered.

For the CST problem (Fig. 3(e)), better results are obtained by considering values of population size greater than or equal to 50. The exception is represented by populations that consists of 1000 individuals. In this case, the test fitness is worse than the ones achieved with other population size values (greater than 2). This fact is understandable by taking a closer look at the training fitness (Fig. 1(e)): for this problem a population size of 1000 returns solutions with a poorer quality with respect to the other population size values taken into account. A possible improvement of the fitness (both on training and test instances) can be achieved for this dataset by incrementing the number of fitness evaluations. To conclude the analysis of this dataset, it is important to underline that, while better results have been achieved with larger populations, populations with only 1 or 2 individuals only present a negligible amount of overfitting compared to the one observed in Fig. 3(b) and (d).

When the LD50 dataset (Fig. 3(f)) is taken into account, GSGP shows a behavior that is similar to the one that we have observed on the training instances: the best results are achieved with a population that consists of just one individual. Then, except for population size values of 500 and 1000, the generalization ability decreases as the population size increases. While this result seems to contradict the results obtained on the previous datasets, there is an explanation for this behavior: the LD50 is a particularly hard problem and GSGP needs a lot of fitness evaluations for finding good quality solutions for it [46]. This can also be seen considering that, at the end of the evolution, the training fitness achieved with populations of 500 and 1000 individuals is comparable to the one obtained after the initialization of the population. Hence, for this problem, it is difficult to draw a conclusion about the relation between generalization ability and population size. A larger number of fitness evaluations is needed for getting a more clear view on this benchmark problem. Anyway, given the number of variables and the number of instances that characterize the dataset, performing a larger number of fitness evaluations would result in an unbearable slowness of the search process.

To conclude the analysis, let us discuss the results obtained on the test set considering the Keijzer-6 (Fig. 3(g)) and the Vladislavleva-14 benchmarks (Fig. 3(g)). For these datasets, better results are achieved as the population size increases. This confirms the trend observed on the same problem when training fitness has been studied (Fig. 1(g) and (h)). Anyway, it is worth pointing out that the learning task for these datasets is different from the other considered benchmarks: while for the six real-life datasets the performance on the test instances is assessed considering an interpolation problem, in the synthetic problems performance on unseen data is evaluated considering an extrapolation task.

To conclude the analysis of the performance achieved on the test set, we now present the results obtained using both the geometric semantic operators. Results are reported in Fig. 4. Also in this case, on all the considered problems and for all the considered population size values, GSGP produces results that are qualitatively analogous to the ones obtained using only mutation. Also on test data, the relationship between population size and performance of GSGP does not seem to be influenced by the fact of using both genetic operators or mutation only.

After concluding the discussion of the obtained experimental results, it is possible to draw some general considerations: the analysis we have performed seems to suggest that, in real-life datasets, small populations are able to produce fitter models but, on the other hand, the behavior on unseen instances is different. In detail, while for some problems these models do not show the same (good) generalization ability obtained considering large populations, on other problems they still outperform solutions produced by using large population size values. This finding seems to be related to the number of independent variables that characterizes each problem. In particular, datasets characterized by a “small” number of variables seem to be able to produce good results on both training and test instances by considering a limited number of individuals in the population. On the other hand, datasets characterized by the presence of a larger number of independent variables are not able to produce on the test instances the same good performance achieved on the training set by using small populations. In this last case, solutions obtained by taking into account small population size values overfit the training data and they are not able to generalize over unseen instances. Differently, larger values of population size are able to produce solutions with a better generalization ability on datasets characterized by a large number of variables.

The role of the number of variables on the observed performance can be interpreted as follows: in a dataset with a large number of variables, the target model will generally contain a large number of the independent variables. Anyway, there exist several models that can be learned by GSGP that are able to produce good performance on the training instances by using just a small number of independent features [14]. Hence, a population with a larger number of individuals has a greater chance to produce, during the initialization process, candidate solutions characterized by different features that, when combined by means of semantic operators, will better approximate the ideal target (with respect to both training and fitness instances). These individuals may have a worse fitness on the training set with respect to the ones obtained with smaller populations (given to the number of different *generations* they are allowed to evolve) but, contrary to the latter, they are able to avoid to overfit the training instances and they represent a better choice when generalization ability is taken into account.

Regarding the synthetic benchmarks, it is clear that larger populations represent a better choice. Anyway, as previously explained, this is mainly due to the characteristics of the datasets that are created to make useless the use of the genetic operators. Moreover, training and test instances are drawn differently with respect to the real-life benchmark problems.

To conclude this section, we discuss the statistical significance of the results presented so far. A set of tests has been performed on the median errors. As a first step, the Kolmogorov–Smirnov test has shown that the data are not normally distributed and hence a rank-based statistic has been used. Successively, the Wilcoxon rank-sum test has been used under the alternative hypothesis that the second set of samples have a lower median (i.e., better fitness) than the first set using a significance level of $\alpha = 0.05$. Results of the statistical analysis are not reported in the paper (a lot of tables

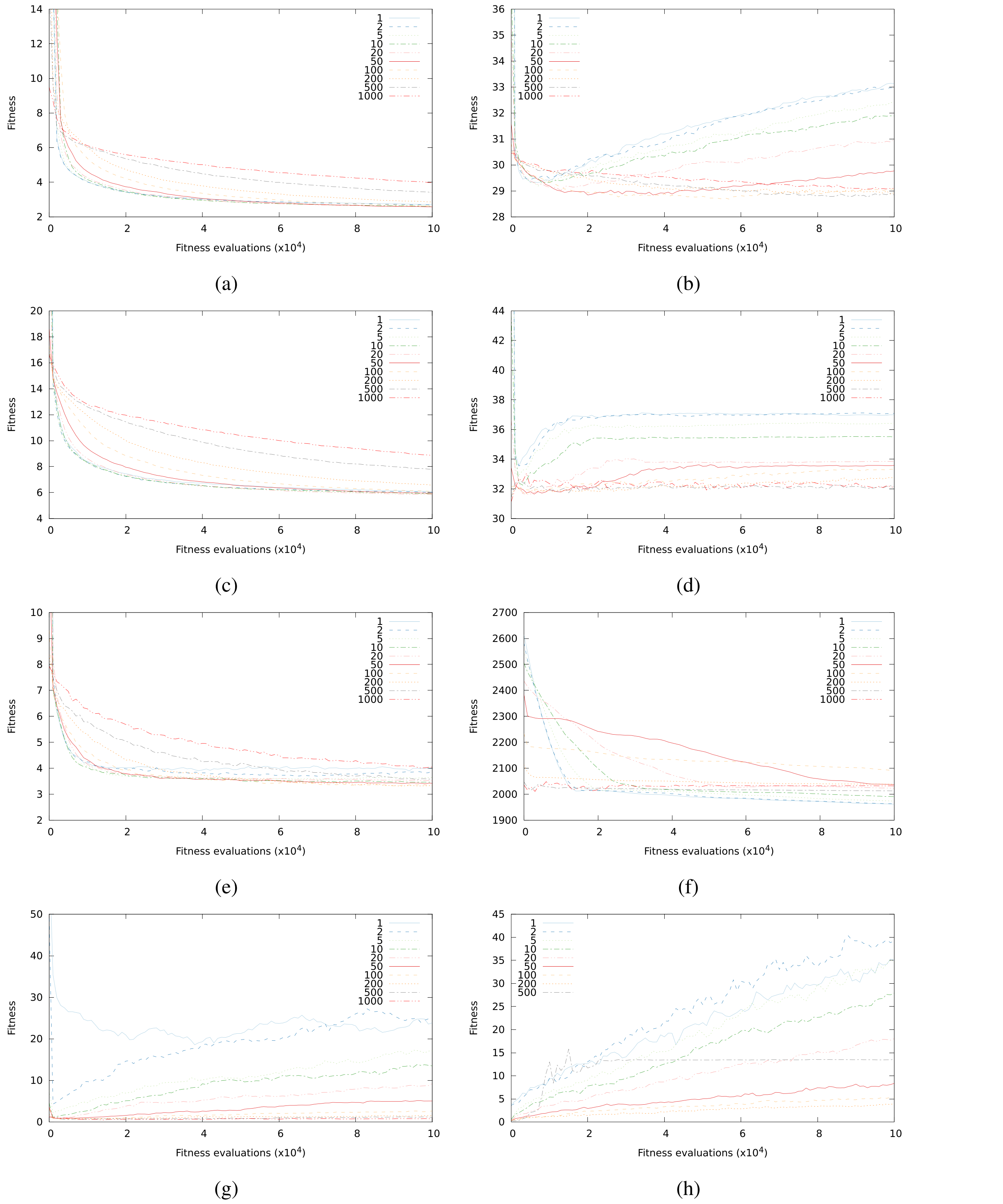

Fig. 4. Test fitness for the considered benchmark problems. Contrarily to Fig. 3, both geometric semantic crossover and mutation have been used here. (a) ASN, (b) %F, (c) CCS, (d) PPB, (e) CST, (f) LD50, (g) Keijzer-6, and (h) Vladislavleva-14. Medians over 100 independent runs are reported.

would have been necessary), but it is possible to summarize them. The *p*-values returned by the statistical test confirm the qualitative analysis previously discussed. In particular, on the first five datasets (ASN, %F, CCS, PPB and CST) small populations produce training fitness values that are statistically better than the ones achieved with large populations (both considering only mutation or crossover and mutation). On the test set, for the ASN and CCS datasets, statistically better fitness values can be obtained by using small populations (up to 50 individuals), while on the CST dataset the optimal population size value is 100. %F and PPB produce statistically better results (among the considered population size values) on the test instances when large population size values are considered. For %F the ideal population size value is 500, while for the PPB dataset a population with 1000 individuals produces statistically better test fitness values. On the LD50 dataset, smaller populations produce the best fitness values on both training and test instances. For this dataset populations with 1 and 2 individuals perform statistically better than the other considered population sizes. Finally, on synthetic datasets large populations (500 and 1000 individuals) produce results that are statistically better on both training and test instances with respect to the ones achieved with the other considered population size values.

Generally speaking, the statistical tests suggest that also very small variations of the population size have an impact on the search process. For instance, in the considered problems, passing from a population of 2 individuals to a population of 5 individuals or from a population of size 5 to a population of size 10 causes a difference in terms of fitness values that is statistically significant. This fact highlights the importance of correctly determining the population size before executing GSGP for solving a given problem.

5. Conclusions

Several studies have discussed the importance of a correct choice of the parameters that characterize evolutionary algorithms and, more in particular, genetic programming (GP). These studies have shown that the performance of GP is strongly dependent on the values of some parameters. Hence, considering the difficulty that characterize the parameter tuning phase, a plethora of contributions has appeared trying to analyze the impact of the different parameters. With the recent definition of geometric semantic genetic programming (GSGP), it is necessary to reconsider previous findings. In fact, GSGP uses genetic operators that have different characteristics with respect to the traditional syntax-based operators used by standard GP. Under this perspective, in this study we have investigated the role of population size on the performance of GSGP. A set of experiments has been performed, comparing the performance achieved by GSGP with different population sizes, including populations consisting of only one individual. The results we have presented should allow GSGP users to better understand the role of population size: in synthesis, on real-life problems, small populations result in better performance on the training set, but performance on the test instances varies among the different problems: in datasets with a high number of features, models obtained with large populations present a better performance on unseen data, while in datasets characterized by a relative small number of variables a better generalization ability is achieved by using small population size values. When synthetic problems are taken into account, large population size values represent the best option for achieving good quality solutions on both training and test instances. Interestingly, these results hold, and are qualitatively identical, both in case geometric semantic crossover and mutation, or only mutation, are used.

Several activities in this research track are planned for the future. First of all, we are trying to deeply investigate how the number of features impacts on the generalization ability of the models returned by GSGP. Secondly, the development of a GSGP system with variable size population, inspired by the few ones proposed for standard GP so far, is planned. According to our expectations, this system should be able to choose, in accordance with the characteristics of the problem at hand, the ideal population size to guarantee both a good optimization performance and a good generalization ability. Last but not least, we feel that there is a big gap between the amount of theory that has studied the influence of the population size on standard GP and the substantial lack of analogous theoretical results for GSGP. If, on the one hand, this deficit of theoretical results is justified by the extremely young age of GSGP, on the other hand, it is also true that GSGP is becoming more and more popular, and, given the extremely promising results it has allowed to obtain so far, it is possible to expect it to become a standard method in the future. As such, a solid body of theory is needed, to justify and strengthen the approach. A theoretical study of the influence of the population size on the GSGP performance is one of the most ambitious and stimulating tracks of our planned future research.

References

[1] J. Arabas, Z. Michalewicz, J. Mulawka, Gavaps—a genetic algorithm with varying population size, in: 1994 Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, vol. 1, 1994, pp. 73–78.
[2] F. Archetti, S. Lanzeni, E. Messina, L. Vanneschi, Genetic programming for human oral bioavailability of drugs, in: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, GECCO '06, ACM, New York, NY, USA, 2006, pp. 255–262.
[3] F. Archetti, S. Lanzeni, E. Messina, L. Vanneschi, Genetic programming and other machine learning approaches to predict median oral lethal dose (ld50) and plasma protein binding levels, in: Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, Lecture Notes in Computer Science, vol. 4447, Springer, Berlin, Heidelberg, 2007, pp. 11–23.
[4] T. Bäck, Self-adaptation in genetic algorithms, in: Proceedings of the First European Conference on Artificial Life, MIT Press, Cambridge, Massachusetts, USA, 1992, pp. 263–271.
[5] T. Bäck, Optimal mutation rates in genetic search, in: Proceedings of the 5th International Conference on Genetic Algorithms, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993, pp. 2–8.
[6] T. Brooks, D. Pope, A. Marcolini, Airfoil Self-Noise and Prediction, Technical Report, NASA RP-1218, 1989.
[7] E. Burke, S. Gustafson, G. Kendall, N. Krasnogor, Advanced population diversity measures in genetic programming, in: Parallel Problem Solving from Nature - PPSN VII, Lecture Notes in Computer Science, vol. 2439, Springer, Berlin, Heidelberg, 2002, pp. 341–350.
[8] M. Castelli, L. Manzoni, L. Vanneschi, Parameter tuning of evolutionary reactions systems, in: GECCO '12: Proceedings of the Fourteenth International Conference on Genetic and Evolutionary Computation Conference, ACM, Philadelphia, Pennsylvania, USA, 2012, pp. 727–734.
[9] M. Castelli, L. Manzoni, L. Vanneschi, S. Silva, A. Popovič, Self-tuning geometric semantic genetic programming, Genet. Program. Evol. Mach. (2015) 1–20.
[10] M. Castelli, S. Silva, L. Vanneschi, A C++ framework for geometric semantic genetic programming, Genet. Program. Evol. Mach. 16 (1) (2014) 73–81.
[11] M. Castelli, L. Vanneschi, A. Popovič, Parameter evaluation of geometric semantic genetic programming in pharmacokinetics, Int. J. BioInspir. Comput. 8 (1) (2016), http://dx.doi.org/10.1504/IJBIC.2016.074634.
[12] M. Castelli, L. Vanneschi, S. Silva, Prediction of high performance concrete strength using genetic programming with geometric semantic genetic operators, Expert Syst. Appl. 40 (17) (2013) 6856–6862.
[13] M. Castelli, L. Vanneschi, S. Silva, Prediction of the unified parkinson's disease rating scale assessment using a genetic programming system with geometric semantic genetic operators, Expert Syst. Appl. 41 (10) (2014) 4608–4616.
[14] M. Castelli, L. Vanneschi, S. Silva, Semantic search-based genetic programming and the effect of intron deletion, IEEE Trans. Cybern. 44 (1) (2014) 103–113.
[15] D. Cvetkovic, D.S. Augustin, H. Muhlenbein, The Optimal Population Size for Uniform Crossover and Truncation Selection, Technical Report GMD-AS-TR-94-11, German National Research Center for Computer Science (GMD), 1994.
[16] L. Davis, Adapting operator probabilities in genetic algorithms, in: Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1989, pp. 61–69.
[17] K. De Jong, W. Spears, An analysis of the interacting roles of population size and crossover in genetic algorithms, in: H.P. Schwefel, R. Manner (Eds.), Parallel Problem Solving from Nature, Lecture Notes in Computer Science, vol. 496, Springer, Berlin, Heidelberg, 1991, pp. 38–47.
[18] K.A. De Jong, An analysis of the behavior of a class of genetic adaptive systems (Ph.D. thesis), Ann Arbor, MI, USA, 1975.
[19] F. Fernandez, M. Tomassini, L. Vanneschi, Saving computational effort in genetic programming by means of plagues, in: The 2003 Congress on Evolutionary Computation, 2003, CEC '03, vol. 3, 2003, pp. 2042–2049.
[20] D.E. Goldberg, Sizing populations for serial and parallel genetic algorithms, in: Proceedings of the 3rd International Conference on Genetic Algorithms, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1989, pp. 70–79.
[21] J. Grefenstette, Optimization of control parameters for genetic algorithms, IEEE Trans. Syst. Man Cybern. 16 (1) (1986) 122–128.
[22] R. Haupt, Optimum population size and mutation rate for a simple real genetic algorithm that optimizes array factors, in: Antennas and Propagation Society International Symposium, 2000, IEEE, Salt Lake City, UT, USA, vol. 2, 2000, pp. 1034–1037.
[23] Y. I-Cheng, Simulation of concrete slump using neural networks, Constr. Mater. 162 (1) (2009) 11–18.
[24] M. Keijzer, Improving symbolic regression with interval arithmetic and linear scaling, in: Proceedings of the 6th European Conference on Genetic Programming, EuroGP'03, Springer-Verlag, Berlin, Heidelberg, 2003, pp. 70–82.

[25] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, MA, USA, 1992.
[26] K. Krawiec, P. Lichocki, Approximating geometric crossover in semantic space, in: GECCO '09: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, ACM, Montreal, 2009, pp. 987–994.
[27] J. McDermott, D.R. White, S. Luke, L. Manzoni, M. Castelli, L. Vanneschi, W. Jaskowski, K. Krawiec, R. Harper, K. De Jong, U.M. O'Reilly, Genetic programming needs better benchmarks, in: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, GECCO '12, ACM, New York, NY, USA, 2012, pp. 791–798.
[28] 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, vol. 7491, Springer, Berlin, Heidelberg, 2012, pp. 21–31.
[29] A. Moraglio, A. Mambrini, Runtime analysis of mutation-based geometric semantic genetic programming for basis functions regression, in: 2013 Genetic and Evolutionary Computation Conference, GECCO '13, Amsterdam, The Netherlands, July 6–10, 2013, pp. 989–996.
[30] A. Moraglio, A. Mambrini, Runtime analysis of mutation-based geometric semantic genetic programming for basis functions regression, in: Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation, GECCO '13, ACM, New York, NY, USA, 2013, pp. 989–996.
[31] A. Moraglio, J. Togelius, S. Silva, Geometric differential evolution for combinatorial and programs spaces, Evol. Comput. 21 (4) (2013) 591–624.
[32] T.R. Naik, V.K. Dabhi, Improving generalization ability of genetic programming: comparative study, J. Bioinf. Intell. Control 2 (4) (2013) 243–252 (2013-12-01T00:00:00).
[33] R. Poli, Recursive conditional schema theorem, convergence and population sizing in genetic algorithms, in: Proceedings of the Foundations of Genetic Algorithms Workshop, FOGA 6, MorganKaufman, 2000, pp. 143–163.
[34] R. Poli, N.F. McPhee, L. Vanneschi, The impact of population size on code growth in GP: analysis and empirical validation. in: M. Keijzer, G. Antoniol, C.B. Congdon, K. Deb, B. Doerr, N. Hansen, J.H. Holmes, G.S. Hornby, D. Howard, J. Kennedy, S. Kumar, F.G. Lobo, J.F. Miller, J. Moore, F. Neumann, M. Pelikan, J. Pollack, K. Sastry, K. Stanley, A. Stoica, E.G. Talbi, I. Wegener (Eds.), GECCO '08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, ACM, Atlanta, GA, USA, 2008, pp. 1275–1282.
[35] R. Poli, N.F. Mcphee, L. Vanneschi, L., The impact of population size on code growth in gp: analysis and empirical validation, in: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, GECCO 2008, 2008, p. 1275.
[36] A. Simoes, E. Costa, The influence of population and memory sizes on the evolutionary algorithm's performance for dynamic environments, in: Applications of Evolutionary Computing, Lecture Notes in Computer Science, vol. 5484, Springer, Berlin, Heidelberg, 2009, pp. 705–714.
[37] J. Smith, T. Fogarty, Self adaptation of mutation rates in a steady state genetic algorithm, in: 1996 Proceedings of IEEE International Conference on Evolutionary Computation, 1996, pp. 318–323.
[38] R. Smith, E. Smuda, Adaptively resizing populations: algorithm, analysis, and first results, Complex Syst. 9 (1) (1995) 57–72.
[39] C.R. Stephens, I.G. Olmedo, J.M. Vargas, H. Waelbroeck, Self-adaptation in evolving systems, Artif. Life 4 (2) (1998) 183–201.
[40] M. Tomassini, L. Vanneschi, J. Cuendet, F. Fernandez, A new technique for dynamic size populations in genetic programming, in: 2004 Congress on Evolutionary Computation, CEC2004, vol. 1, 2004, pp. 486–493.
[41] Y.R. Tsoy, The influence of population size and search time limit on genetic algorithm, in: 2003 Proceedings of the 7th Korea-Russia International Symposium on Science and Technology, KORUS 2003, vol. 3, 2003, pp. 181–187.
[42] L. Vanneschi, M. Castelli, L. Manzoni, S. Silva, A new implementation of geometric semantic GP and its application to problems in pharmacokinetics, in: K. Krawiec, A. Moraglio, T. Hu, A.S. Uyar, B. Hu (Eds.), Proceedings of the 16th European Conference on Genetic Programming, EuroGP 2013, LNCS, vol. 7831, Springer Verlag, Vienna, Austria, 2013, pp. 205–216.
[43] L. Vanneschi, M. Castelli, S. Silva, Measuring bloat, overfitting and functional complexity in genetic programming, in: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, GECCO '10, ACM, New York, NY, USA, 2010, pp. 877–884.
[44] L. Vanneschi, M. Castelli, S. Silva, A survey of semantic methods in genetic programming, Genet. Program. Evol. Mach. 15 (2) (2014) 195–214.
[45] L. Vanneschi, G. Cuccu, Variable size population for dynamic optimization with genetic programming, in: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, GECCO '09, ACM, New York, NY, USA, 2009, pp. 1895–1896.
[46] L. Vanneschi, S. Silva, M. Castelli, L. Manzoni, Geometric semantic genetic programming for real life applications, in: Genetic Programming Theory and Practice, Springer, Ann Arbor, 2013.
[47] E. Vladislavleva, G. Smits, D. Den Hertog, Order of nonlinearity as a complexity measure for models generated by symbolic regression via pareto genetic programming, IEEE Trans. Evol. Comput. 13 (2) (2009) 333–349.