Self-tuning Geometric Semantic GP

Mauro Castelli*[1], Luca Manzoni†[2], Sara Silva‡[3], and Leonardo Vanneschi§[4]

[1]*ISEGI, Universidade Nova de Lisboa, 1070-312 Lisboa, Portugal*
[2]*Dipartimento di Informatica, Sistemistica e Comunicazione, Università degli Studi di Milano-Bicocca, Viale Sarca 336, 20126 Milano, Italy*
[3]*LabMAg, FCUL, Universidade de Lisboa, 1749-016 Lisboa, Portugal*
[4]*ISEGI, Universidade Nova de Lisboa, 1070-312 Lisboa, Portugal*

Abstract

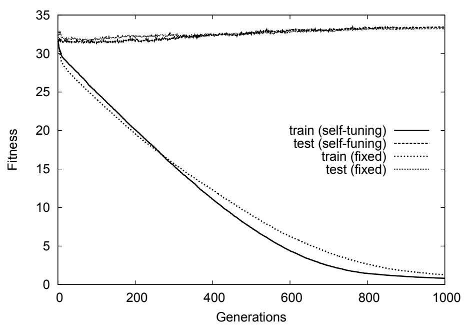
In Geometric Semantic GP (GSGP), similarly to normal GP, parameter tuning is necessary to attain good performances. Here we introduce a method for self-tuning GSGP that not only saves the user the tuning task, but it also outperforms traditional hand-tuned GSGP.

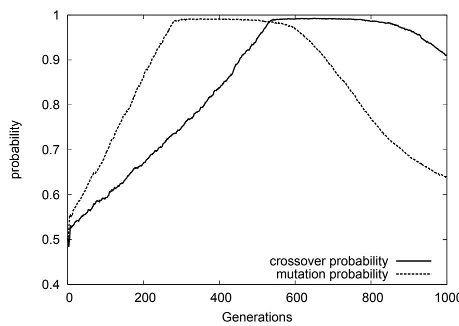
Introduction and basic notions. Geometric Semantic Genetic Programming (GP) is a recent new variation of GP inspired by the work on geometric operators [1] and it has successfully been compared to standard GP [2]. However, GSGP still has the time-consuming task of parameter tuning. Here we propose a method for self-tuning the crossover and mutation probabilities.

Self-tuning algorithm. The self-tuning algorithm is based on the idea that each individual T in a population has its own crossover probability $p_c(T)$ and its mutation probability $p_m(T)$ (initially a random number between 0 and 1). During the crossover phase a pair of individuals T_1 and T_2 generates an individual by crossover with probability $p = \frac{1}{2}\left(p_c(T_1) + p_c(T_2)\right)$ and the resulting individual T_c will have crossover probability $p_c(T_c) = p + r$, where r is a small *positive* random number (at most 0.01) if the fitness of T_c is better than the fitness of both parents, and a small *negative* random number (at least -0.01) otherwise. Similarly, an individual T is mutated with probability $p_m(T)$, and its offspring T_m will have mutation probability $p_m(T_m) = p_m(T) + r$, where r is a small positive number if the fitness of T_m is better than the fitness of T and a small negative random number otherwise. Therefore, self-tuning actually increases the crossover (resp., mutation) probability when the crossover (resp., mutation) produces better offspring and it decreases the probability otherwise.

*mcastelli@isegi.unl.pt
†luca.manzoni@disco.unimib.it
‡sara@fc.ul.pt
§lvanneschi@isegi.unl.pt

Experimental results. We have used the %F, %PPB, and LD50 symbolic regression datasets in the experiments (see [2] for a description). We have compared GSGP with self-tuning against GSGP with mutation probability 0.5 and crossover probability 0.9. In both cases the population size was set to 100, the selection method used was tournament selection with tournament size 4, the population was initialized with the ramped-half-and-half method with a maximum depth of 6, and the mutation step was set to 1. The functional symbols were $+$, $-$, $\times$, and protected division; we allowed random constants in $[-100, 100]$ as terminals. We performed 100 independent runs each consisting of 1000 generations. The dataset was split with 70% of the instances used as training instances and the remaining used as test instances. Here we present the results on the %PPB dataset, showing the fitness (calculated as the root mean square error between target and predicted values) on the train and test sets on the left, while the evolution of the median of the average crossover and mutation probabilities generation by generation is shown on the right.





In all the problems the median fitness obtained by self-tuning GSGP is lower (i.e., better) than the one obtained by normal GSGP. Moreover, in 2 of the problems, the difference in terms of training fitness between the considered systems is statistically significant. Considering the variation of the crossover and mutation probabilities, the self-tuning method seems to "adapt" the two parameters to the current state of the search.

Further remarks. We have introduced a self-tuning algorithm for GSGP that, by changing the crossover and mutation parameters in an adaptive way, is able to produce better results with respect to hand-tuned GSGP and to avoid the time-consuming parameter tuning task. We plan to further investigate and improve the effectiveness of this self-tuning algorithm.

References

[1] A. Moraglio, K. Krawiec, and C. G. Johnson. Geometric semantic genetic programming. In *Parallel Problem Solving from Nature - PPSN XII*, volume 7491 of *LNCS*, pages 21–31. Springer, 2012.

[2] L. Vanneschi, S. Silva, M. Castelli, and L. Manzoni. Geometric semantic genetic programming for real life applications. In *Genetic Programming Theory and Practice XI*, pages 191–209. Springer, 2014.