Multi-Population Genetic Programming with Adaptively Weighted Building Blocks for Symbolic Regression

Zhixing Huang, Jinghui Zhong, Weili Liu*
School of Computer Science and Engineering
South China University of Technology
jinghuizhong@gmail.com

Zhou Wu
School of Automation
Chongqing University

ABSTRACT

Genetic programming(GP) is a powerful tool to solve Symbolic Regression that requires finding mathematic formula to fit the given observed data. However, existing GPs construct solutions based on building blocks (i.e., the terminal and function set) defined by users in an ad-hoc manner. The search efficacy of GP could be degraded significantly when the size of the building blocks increases. To solve the above problem, this paper proposes a multi-population GP framework with adaptively weighted building blocks. The key idea is to divide the whole population into multiple sub-populations with building blocks with different weights. During the evolution, the weights of building blocks in the sub-populations are adaptively adjusted so that important building blocks can have larger weights and higher selection probabilities to construct solutions. The proposed framework is tested on a set of benchmark problems, and the experimental results have demonstrated the efficacy of the proposed method.

CCS CONCEPTS

• **Computing methodologies → multi population mechanism**;

KEYWORDS

Building Block, Genetic programming, Symbolic Regression

ACM Reference Format:
Zhixing Huang, Jinghui Zhong, Weili Liu and Zhou Wu. 2018. Multi-Population Genetic Programming with Adaptively Weighted Building Blocks for Symbolic Regression. In *GECCO '18 Companion: Genetic and Evolutionary Computation Conference Companion, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, Article 4, 2 pages. https://doi.org/10.1145/3205651.3205673

1 INTRODUCTION

Symbolic Rrgression (SR) which requires finding mathematic formula to fit the given observed data is an active research topic that has a range of applications such as time series prediction, data mining and knowledge discover [3, 4]. Genetic Programming(*GP*) is the most common approach to solve SR problem [1]. So far, there are various enhanced GP variants for SR, such as the Geometric Semantic Genetic Programming [2] and the Self-Learning Genetic Expression Programming [5, 6].

One fundamental operation of GP to solve SR problem is defining the terminal set and the function set for solution construction. To ensure finding high quality mathematic formula, the size of terminal and function set should be set large enough so that the target mathematic formula is included in the search space defined by the terminal and function set. However, the search space will increase exponentially with the size of terminal and function set. How to define the properly terminal and function set to balance the search efficiency and solution quality is still a challenging task in the GP community.

To solve this problem, this paper proposes a multi-population mechanism with adaptively weighted building blocks. In the proposed mechanism, the population is divided into a number of sub-populations. Each sub-population is assigned with building blocks with different weights and focused on using the small set of building blocks with higher weights to construct solutions. In this way, the search space of each sub-population becomes much smaller than the original search space. Besides, the weights of building blocks in each sub-population are adaptively adjusted during the evolution so that important building blocks can gradually be emphasized to improve the search efficiency. The proposed framework is integrated with a recently published GP variant named SL-GEP [6] to form a multi-population gene expression programming (MP-GEP). Experiments on benchmark problems have shown that the proposed MP-GEP is able to provide very promising performance.

2 SR PROBLEM DEFINITION

In an SRP, a mathematical formula Γ is constructed to fit a given set of measurement data which consists of input variables and the corresponding output responses. The objective is to construct a formula Γ^* using functions (e.g., "+","×","sin") and terminals (e.g., variables and constants) defined in advance, to minimize the fitting error:

$$\Gamma^* = \arg\min_{\Gamma} f(\Gamma) \tag{1}$$

where $f(\Gamma)$ is commonly defined as the root-mean-square-error, i.e,(RMSE):

$$f(\Gamma) = \sqrt{\frac{\sum_{i=1}^{M}(\Gamma(x_i) - y_i)^2}{M}} \tag{2}$$

where x_i is the ith input data, y_i is the true output of the ith input data, and M is the number of samples in the training data set.

2.1 Proposed Method

In the proposed search framework, the entire population is divided into M sub-populations. The weight vector of building blocks in

*Jinghui Zhong is the corresponding author

GECCO '18 Companion, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5764-7/18/07…$15.00
https://doi.org/10.1145/3205651.3205673

each sub-population is denoted as **P**= $\{p_1, p_2, ...\}$, where p_i represents the weight (or selection probability) of the ith symbol (the symbol can be either a function or a terminal). Each sub-population is associated with a distinct **P**. During the evolution, the **P**s are updated by considering the frequency of building blocks in the current sub-population, i.e.,$p_s^{g+1} = \tau p_s^g + (1-\tau)q_s$, where p_s^{g+1} is the weight of symbol s in the next generation, q_s is the frequency of symbol s in the current sub-population, and τ is the update rate. Usually, $\tau \in [0, 1]$ is set to a relatively large value (e.g., 0.99), so that the p_s can be updated gradually.

To improve the search efficiency, a migration operation is introduced in the proposed framework to share search information between sub-populations. The migration is performed in a fixed period. In the migration, the best individual of every sub-population will be replaced by the global best individual. Besides, for each sub-population, a random individual is selected and will replace a random individual in another random sub-population if the former is better than the latter. To ensure population diversity, when the similarity between the **P**s of sub-populations is larger than a threshold θ, the migration between these two sub-populations is cancelled.

In this study, SL-GEP[6] is selected as the base algorithm of the proposed framework. We choose SL-GEP as the base algorithm for it has been shown to perform well on SR problems compared with several well-known GP variants. The reproduction operator of SL-GEP is modified to make use of **P**.

3 EXPERIMENTAL RESULTS

In the experiment studies, five benchmark SR problems as shown in Table 1 are utilized to test the effectiveness of the proposed MP-GEP. The parameters of MP-GEP are set as follows: the size of population is 512, the number of ADFs in the chromosome is 2, the length of each chromosome is 35, θ is 0.8, and τ is 0.99. The population is divided into 4 sub-populations. The set of building blocks for solution construction contains one input variable (i.e., x) and seventeen mathematic operations, i.e., +, −, ∗, /, max, min, *power*, *mod*, sin, cos, exp, ln, *sgn*, $f(x) = 10 * x$, $f(x) = x + 15$, $f(x) = 20 * x$, $f(x) = 30 + x$. The maximum generations is 4000 and the migration interval is 100 generations. We compare the proposed MP-GEP with the SL-GEP to demonstrate its effectiveness. Each algorithm will be performed 200 independent runs on each test problem. When an algorithm achieves a fitting error smaller than $1e-4$, a successful hit is obtained. We used the success rate (Suc) and the RMSE obtained by each algorithm for comparison. The comparison results in Table 1 show that the proposed MP-GEP performs much better or at lease competitive on all problems in terms of Suc and RMSE. Further, we conduct experiment to investigate the scalability of the proposed method. We vary the function set to validate the degrading speed as the number of redundant building blocks increase. Specifically, we add three new redundant functions (i.e., $f(x) = x * 30$, $f(x) = x + 45$ and $f(x) = x + 60$) to the original seventeen functions to form a function pool. Fig.1 illustrates the Suc of MP-GEP degrades much slower than the SL-GEP on f_3, which demonstrates the better scalability of the proposed method.

Table 1: Comparison results of SL-GEP ond MP-GEP.

Problem	SL-GEP		MP-GEP	
	Suc	RMSE	Suc	RMSE
$f_1 = x^5 - 2x^3 + x$	0.55	0.006	0.88	0.002+
$f_2 = x^6 + x^5 + x^4 + x^3 + x^2 + x$	0.91	0.007	0.97	0.005≈
$f_3 = \sin(x^2)\cos(x) - 1$	0.33	0.005	0.62	0.004+
$f_4 = \ln(x+1) + \ln(x^2+1)$	0.38	0.008	0.36	0.008≈
$f_5 = \sqrt{(x)}$	0.99	0	1.0	0≈

* Symbol +, ≈ mean the MP-SLGEP is respectively significantly better than,competitive with SLGEP according to the Wilcoxon rank-sum test with 5% significance value.

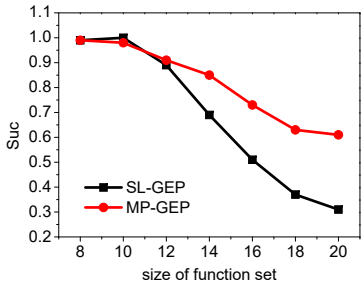

Figure 1: The performance of SL-GEP and MP-GEP with different function sets.

4 CONCLUSIONS

This paper proposed an efficiency multi-population genetic programming framework with adaptively weighted building blocks to improve the performance of GP on symbolic regression problems. As for future work, we plan to integrate the proposed framework with GPU to further improve the search efficacy and to apply the new algorithm to real applications.

ACKNOWLEDGMENT

This work is supported under the National Natural Science Foundation of China (Grant No. 61602181) and the Fundamental Research Funds for the Central Universities (Grant No. 2017ZD053, 106112017CDJXY170003).

REFERENCES

[1] John R. Koza and Riccardo Poli. 2005. *Genetic Programming*. 127–164 pages.
[2] Alberto Moraglio, Krzysztof Krawiec, and Colin G. Johnson. 2012. Geometric Semantic Genetic Programming. In *Parallel Problem Solving from Nature - PPSN XII*, Carlos A. Coello Coello, Vincenzo Cutello, Kalyanmoy Deb, Stephanie Forrest, Giuseppe Nicosia, and Mario Pavone (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 21–31.
[3] Michael D Schmidt and Hod Lipson. 2009. Distilling Free-Form Natural Laws from Experimental Data. *Science* 324, 5923 (2009), 81–85.
[4] Jinghui Zhong, Wentong Cai, Michael Lees, and Linbo Luo. 2017. Automatic model construction for the behavior of human crowds. *Applied Soft Computing* 56 (2017), 368–378.
[5] Jinghui Zhong, Liang Feng, and Yew Soon Ong. 2017. Gene Expression Programming: A Survey [Review Article]. *IEEE Computational Intelligence Magazine* 12, 3 (2017), 54–72.
[6] Jinghui Zhong, Yew Soon Ong, and Wentong Cai. 2016. Self-Learning Gene Expression Programming. *IEEE Transactions on Evolutionary Computation* 20, 1 (2016), 65–80.