Genetic Programming with Multi-Layered Population Structure

Taku Hasegawa
Graduate School of Engineering
Osaka Prefecture University
Gakuen-cho 1-1, Naka-ku
Sakai, Osaka 599-8531
hasegawa@ss.cs.osakafu-u.ac.jp

Naoki Mori
Graduate School of Engineering
Osaka Prefecture University
Gakuen-cho 1-1, Naka-ku
Sakai, Osaka 599-8531
mori@cs.osakafu-u.ac.jp

Keinosuke Matsumoto
Graduate School of Engineering
Osaka Prefecture University
Gakuen-cho 1-1, Naka-ku
Sakai, Osaka 599-8531
matsu@cs.osakafu-u.ac.jp

ABSTRACT

This paper focus on the control of building blocks in the population of Genetic Programming (GP). We propose a GP algorithm that employs multi-layered population and searches solutions by using local search and crossover. The computational experiments were carried out by taking several classical Boolean problems as examples .

CCS CONCEPTS

•**Theory of computation → Genetic programming;**

KEYWORDS

Genetic Programming, Population Structure, Local Search

ACM Reference format:
Taku Hasegawa, Naoki Mori, and Keinosuke Matsumoto. 2017. Genetic Programming with Multi-Layered Population Structure. In *Proceedings of the Genetic and Evolutionary Computation Conference 2017, Berlin, Germany, July 15–19, 2017 (GECCO '17),* 2 pages.
DOI: http://dx.doi.org/10.1145/3067695.3076048

1 INTRODUCTION

It is important for valid search of Evolutionary Computations (ECs) to store efficient building blocks and combine these blocks efficiently. Therefore, EC should keep a diversity of building blocks and select them to combine effectively. This paper, aimed at building block control, presents a GP algorithm called "Genetic Programming with Multi-Layered Population Structure (MLPS-GP)" MLPS-GP employs multi-layered population like the pyramid-like population and searches solutions using local search and crossover. The computational experiments were carried out by taking a classical Boolean problem as an example. According to our experimental results, we demonstrated that MLPS-GP found an optimal solution efficiently

GECCO '17, Berlin, Germany

DOI: http://dx.doi.org/10.1145/3067695.3076048

Algorithm 1 ITERATE-MLPS-GP

```
1:  Create random solution s (Section 2.1)
2:  Apply local search (Section 2.2)
3:  if solution ∉ mlps then
4:      Copy tree of s to solution s'
5:      Calculate tree depth d of s'
6:      Add s' to P_d (Section 2.4)
7:  for all P_i ∈ mlps do
8:      Crossover s with P_i (Section 2.3)
9:      if the fitness of s has improved then
10:         if s ∉ mlps then
11:             Calculate tree depth d of s
12:             subset ← ∪_{j=0}^{(min(lev_max, d−1))} P_j
13:             if the fitness of s is the best in subset then
14:                 Copy tree of s to solution s''
15:                 if P_d ∉ mlps then
16:                     Add .P_d to mlps
17:                 Add s'' to P_d
18:     n_count ← n_count + 1
19:     Calculate tree depth d_s of s''
20:     if d_s < i + 1 then break
```

Figure 1: One iteration of MLPS-GP optimization. ***mlps*** **is an ordered set of populations and** $lev_{\max}$ **is the highest level in MLPS.**

2 GENETIC PROGRAMMING WITH MULTI-LAYERED POPULATION STRUCTURE

The MLPS-GP does not have a single population of solutions, but instead a several-layered structure. We call the proposed population a "Multi-Layered Population Structure (MLPS)". Each layer of an MLPS comprises a population of solutions that are of the same depth. Fig. 1 provides a high-level summary of the MLPS-GP. It shows how the MLPS-GP constructs populations iteratively and uses them to improve the randomly generated solutions using hill climbing and crossover. In Fig. 1, *mlps* is an ordered set of populations and $lev_{\max}$ is the highest level in MLPS.

2.1 Initialization

In this study, we adopt the Probabilistic Tree Creation (PTC) [3] to consider the diversity of tree structures. The PTC algorithm in

MLPS-GP can be described as follows. The sets of functions F are divided into a non-terminal node set N and a terminal node set T. At initialization, we choose new child nodes from N or T based on the probability p_d, which is given by Equation (1):

$$p_d = r_{\text{damp}}^{d-1} \tag{1}$$

The term r_{damp} represents a damping rate and d represents the depth of the current node. The term p_d is the probability that a non-terminal node is chosen as a new child from N. The user also provides a maximum tree depth bound D_{init}. While users can tune r_{dump} according to the problem structure, we have set its value to 0.9 in this study.

2.2 Local Search

MLPS-GP uses a local search method based on the First Improvement Hill Climber (FIHC) [2].In the local search method, one locus of the target individual is selected randomly, and the fitness values of all types of allele genes in that locus are checked. Next, the original allele for the locus is replaced with the most improving allele gene in the above process. To avoid waste of evaluations, same loci are never tested until other loci are replaced and the solution improves the fitness. Iteration of the process continues until such a change no longer results in a strict fitness improvement. In this study, the local search is applied to a non-terminal node.

2.3 Crossover

In the crossover, after applying local search, an individual is called "recipient". Subtrees of recipient are replaced by a solution in MLPS, called donor. A candidate set of positions replaced by donors are positions of terminal node of recipient immediately after local search. This candidate set is never changed during an iteration. A position of root node of the subtree is selected from this candidate set in a random order. If the fitness of recipient after being replaced is higher than that of the recipient before being replaced, the recipient keeps the subtree. After a subtree has been replaced by a donor, the position is still considered as an element of the candidate set, while terminal nodes of connected donor are not put into candidate set. If the donor cannot provide a new solution with a higher fitness than the recipient in the position, it reverts to the original subtree.

2.4 Multi-Layered Population Structure (MLPS)

MLPS-GP maintains a multi-layered structure of the populations, with deeper and more optimized solutions in the higher levels. We denote the level number of the layers in the MLPS from bottom to top. The bottom level is 0, and each layer represents as one population. In MLPS-GP, new solutions with depth i are added into level i of the MLPS. In order to add a new solution to a level of the MLPS, two conditions must be met: it should not already exist in the MLPS and it should have the best fitness among the solutions of the MLPS excluding the solutions in lower levels. If the depth of the individuals is larger than those of the top layer, MLPS-GP creates a new layer above the current top layer. The other condition for iterating MLPS-GP is that all solutions with tree depth 0 are added into level 0 before search of MLPS-GP.

Table 1: Comparison of the mean evaluations to success as well as successful rate for D/F and MLPS-GP. The mean represents the average number of evaluations required ($\times 10^4$) and std. represents the standard deviation of them ($\times 10^4$).

method (setting)	mean (std.)	successful rate
D/F	469.24 (290.33)	0.06
MLPS ($D_{\text{init}} = 3$)	411.67 (161.24)	0.97
MLPS ($D_{\text{init}} = 4$)	245.38(117.01)	1.0
MLPS ($D_{\text{init}} = 5$)	208.85(70.86)	1.0

3 EXPERIMENT

The experimental validation of the approach proceeds by comparing the performance of MLPS-GP to Density/Fitness Pareto Optimization (D/F) [1]. D/F which is one of the state-of-the-art algorithm in tree-based GP.

We ran each method for 100 trials, where each run was limited to 10 million evaluations for Even Parity-8 Problem (Par-8). The non-terminal node set $N = AND, OR, NAND, NOR$ contained standard boolean operators. The setting of D/F was decided in accordance with the setting written in the original paper [1].

3.1 Comparative Performance

The experimental results comprise the average number of evaluations required to be success for each algorithm and successful rate which is the ratio of the number of global optima found by each algorithm. Table. 1 shows the results. From Table. 1, the MLPS-GP also outperforms the D/F by a wide margin in the Par-8. MLPS-GP can find the optimal solution in the almost every trials whereas the successful rate of D/F is only 6%. This result shows the effectiveness of MLPS-GP

4 CONCLUSION

This paper presents a novel tree-based GP method, MLPS-GP. This method can be expected to store efficient and various types of building blocks and control these blocks using Multi-layer Population Structure. In addition, the computational experiments were carried out taking several classical Boolean problems as examples and these result showed MLPS-GP outperforms D/F, which is one of the state-of-the-art GP for Par-8. In our future researches, we plan to analyze the detail of search dynamics of MLPS-GP and individuals in each layer of MLPS-GP.

A part of this work was supported by JSPS KAKENHI Grant, Grant-in-Aid for Scientific Research(C), 26330282 and by JSPS KAKENHI Grant, Grant-in-Aid for JSPS Fellows, 16J10941.

REFERENCES

[1] Armand R. Burks and William F. Punch. 2015. An Efficient Structural Diversity Technique for Genetic Programming. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation (GECCO '15)*. ACM, New York, NY, USA, 991–998. DOI: http://dx.doi.org/10.1145/2739480.2754649

[2] H. Hoos and T. Stützle. 2004. *Stochastic Local Search: Foundations & Applications*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[3] Sean Luke. 2000. Two Fast Tree-Creation Algorithms for Genetic Programming. *IEEE Transactions on Evolutionary Computation* 4, 3 (Sept. 2000), 274–283. http://ieeexplore.ieee.org/iel5/4235/18897/00873237.pdf