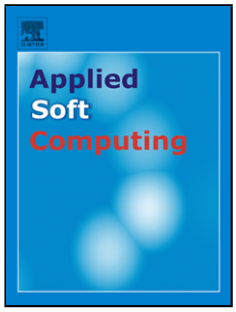

Title: Statistical Genetic Programming for Symbolic Regression

Author: Maryam Amir Haeri Mohammad Mehdi Ebadzadeh Gianluigi Folino

PII: S1568-4946(17)30393-9
DOI: http://dx.doi.org/doi:10.1016/j.asoc.2017.06.050
Reference: ASOC 4319

To appear in: *Applied Soft Computing*

Received date: 10-6-2016
Revised date: 29-4-2017
Accepted date: 25-6-2017

Please cite this article as: Maryam Amir Haeri, Mohammad Mehdi Ebadzadeh, Gianluigi Folino, Statistical Genetic Programming for Symbolic Regression, *<![CDATA[Applied Soft Computing Journal]]>* (2017), http://dx.doi.org/10.1016/j.asoc.2017.06.050

Statistical Genetic Programming for Symbolic Regression

Maryam Amir Haeri[a], Mohammad Mehdi Ebadzadeh[a], Gianluigi Folino[b]

[a]*Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran*
[b]*ICAR-CNR, Rende, Italy*

Abstract

In this paper, a new Genetic Programming (GP) algorithm for symbolic regression problems is proposed. The algorithm, named Statistical Genetic Programming (SGP), uses statistical information—such as variance, mean and correlation coefficient—to improve GP. To this end, we define *well-structured trees* as a tree with the following property: Nodes which are closer to the root have a higher correlation with the target. It is shown experimentally that on average, the trees with structures closer to well-structured trees are smaller than other trees. SGP biases the search process to find solutions whose structures are closer to a well-structured tree. For this purpose, it extends the terminal set by some small well-structured subtrees, and starts the search process in a search space that is limited to *semi–well-structured trees* (i.e., trees with at least one well-structured subtree). Moreover, SGP incorporates new genetic operators, i.e., *correlation-based mutation* and *correlation-based crossover*, which use the correlation between outputs of each subtree and the targets, to improve the functionality. Furthermore, we suggest a *variance-based editing* operator which reduces the size of the trees. SGP uses the new operators to explore the search space in a way that it obtains more accurate and smaller solutions in less time.

SGP is tested on several symbolic regression benchmarks. The results show that it increases the evolution rate, the accuracy of the solutions, and the generalization ability, and decreases the rate of code growth.

Keywords: Genetic Programming, Symbolic Regression, Well-Structured Subtree, Semi–Well-Structured Tree, Well-Structuredness Measure, Correlation Coefficient.

1. Introduction

In recent years, the problem of improving genetic programming (GP) has attracted many researchers. GP has three clearly identified challenges, *code growth*, *huge search space*, and *problem difficulty*.

- **Code growth (bloat):** Uncontrollable growth of the average tree size, without noticeable improvement in fitness, is named *bloat*. This phenomenon has two drawbacks: firstly, the evolution of large programs is computationally expensive; secondly, increasing the complexity of programs may decrease the ability of generalization.

  Several theories have been proposed on the causes of the bloat, including the *removal bias theory* [68], *replication accuracy theory* [45], *nature of program search space theory* [37, 39], and *crossover bias theory*[17, 57].

- **Huge Search Space:** The GP search space is huge due to its programs being variable-length. There are many *functionally-equivalent* solutions (i.e., with the same fitness value) in the search space. In other words, in the GP space, there exist many trees with the same phenotype[1] [18]. Despite the existence of many "redundant" solutions (i.e., solutions with the same phenotype but different genotypes) in the GP search space, most of the

Email addresses: haeri@aut.ac.ir (Maryam Amir Haeri), ebadzadeh@aut.ac.ir (Mohammad Mehdi Ebadzadeh), folino@icar.cnr.it (Gianluigi Folino)

[1]The attributes of a program are either structural (encoding related) or functional (behavioral). The structural attributes, also called the *genotype*, refers to the internal code of a program. On the other hand, the functional attributes, also called the *phenotype*, refers to the observable behavior of a program [11]. Most of the time, the phenotype of a GP tree is defined as its fitness value (the fitness is defined over the whole training data instances).