Information Sciences 436–437 (2018) 352–366

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Semantic tournament selection for genetic programming based on statistical analysis of error vectors

Thi Huong Chu[a], Quang Uy Nguyen[a,*], Michael O'Neill[b]

[a] Faculty of IT, Le Quy Don Technical University, Hanoi, Vietnam
[b] Natural Computing Research & Applications Group and Lero, School of Business, University College Dublin, Ireland

ARTICLE INFO

Article history:
Received 13 March 2017
Revised 12 January 2018
Accepted 13 January 2018

Keywords:
Genetic programming
Tournament selection
Statistical test
Code bloat
Semantics

ABSTRACT

The selection mechanism plays a very important role in the performance of Genetic Programming (GP). Among several selection techniques, tournament selection is often considered the most popular. Standard tournament selection randomly selects a set of individuals from the population and the individual with the best fitness value is chosen as the winner. However, an opportunity exists to enhance tournament selection as the standard approach ignores finer-grained semantics which can be collected during GP program execution. In the case of symbolic regression problems, the error vectors on the training fitness cases can be used in a more detailed quantitative comparison. In this paper we introduce the use of a statistical test into GP tournament selection that utilizes information from the individual's error vector, and three variants of the selection strategy are proposed. We tested these methods on twenty five regression problems and their noisy variants. The experimental results demonstrate the benefit of the proposed methods in reducing GP code growth and improving the generalisation behaviour of GP solutions when compared to standard tournament selection, a similar selection technique and a state of the art bloat control approach.

1. Introduction

Genetic Programming (GP) is a biologically inspired method of using a computer to evolve solutions, in the form of computer programs, for a problem [24,37]. To solve a problem using a GP system, a population of individuals is first initialised. The population is then evolved, under fitness based selection, through a number of generations by applying genetic operators. The evolutionary process terminates when a desired solution is found or when the maximum number of generations is exceeded.

There are several factors that can affect the performance of GP for a given problem. These factors include the size of the population, the fitness evaluation of individuals, the selection mechanisms for reproduction and the genetic operators for modifying individuals. Amongst these, selection plays a critical role in GP performance [4]. To date, there have been many selection schemes proposed [23] and the most widely used selection in GP is tournament selection [11].

Tournament selection compares the fitness values of sampled individuals. The individual with the best fitness is then selected as the winner. This implementation is simple and its effectiveness has been widely evidenced [11]. However, the

* Corresponding author.
E-mail addresses: huongktqs@lqdtu.edu.vn (T.H. Chu), quanguyhn@lqdtu.edu.vn (Q.U. Nguyen), m.oneill@ucd.ie (M. O'Neill).

https://doi.org/10.1016/j.ins.2018.01.030
0020-0255/

standard approach only uses the fitness value while ignoring information from the error vectors of individuals in all fitness cases. Consequently, some information that is potentially useful for GP search may be lost. Recent research has shown that significant benefit could be gained by using semantic information of GP individuals (e.g., [21,22,28,31,35]). The genetic search operators of crossover and mutation can be modified to improve the semantic locality of search [9,30,34]. In addition, the preservation of semantic diversity is a desirable feature of an evolving GP population to avoid local optima [5,12], thus, it is also attractive to examine whether using the error vectors of individuals on the fitness cases during selection can improve GP performance.

In our preliminary research [6], we have proposed two forms of semantic tournament selection that are based on statistical analysis of the error vectors of individuals. The experimental results on a set of GP benchmark problems showed the benefit of the proposed techniques [6]. In this paper, we extend this research with the main contributions of this paper being:

- We introduce the use of statistical analysis of GP error vectors to create novel forms of tournament selection. Based on a Wilcoxon signed rank test, three variants of tournament selection are proposed to exploit semantic diversity and to explore the potential of the approach to control program bloat.
- The performance of the selection strategies are examined on a large set of regression problems employing the original problems and noisy variants. We observe that the new selection techniques help to reduce the code growth and improve the generalization ability of the evolved solutions when compared to standard tournament selection and a state of the art method for controlling code bloat in GP.
- The simplicity of the design of the proposed selection strategies allows for further improvements. In this paper, the addition of a state of the art crossover operator is observed to further enhance performance.

In the next section, we present the background of the paper. Section 3 reviews the related work on improving tournament selection in GP. Three proposed tournament selection strategies are presented in Section 4. Section 5 presents the experimental settings adopted in the paper. Section 6 analyses and compares the performance of the proposed selection strategies with standard tournament selection. The approach is further enhanced through it's coupling to a state of the art crossover strategy in Section 7. Section 8 investigates the ability of the proposed techniques on noisy datasets. Finally, Section 9 concludes the paper and highlights some future work.

2. Background

This section presents some important concepts used in the proposed selection strategies, including the semantics of a GP individual, the error vector of an individual, and the Wilcoxon signed rank test.

In GP, it is common to define the semantics of a program simply as its behaviour with respect to a set of input values [27,31]. Formally, the semantics of a program is defined as follows:

Definition 2.1. Let $K = (k_1, k_2, \ldots k_N)$ be the fitness cases of the problem. The *program semantics* $S(P)$ of a program P is the vector of output values obtained by running P on all fitness cases.

$$S(P) = (P(k_1), P(k_2), \ldots, P(k_N)), \text{ for i} = 1, 2, \ldots, \text{N}.$$

This definition is valid for problems where a set of fitness cases is defined. The error vector of an individual is calculated by comparing the semantic vector with the target output of the problem. More precisely, the error vector of an individual is defined as:

Definition 2.2. Let $S = (s_1, s_2, \ldots s_N)$ be the semantics of an individual P and $Y = (y_1, y_2, \ldots y_N)$ be the target output of the problem on N fitness cases. The *error vector* $E(P)$ of a program P is a vector of N elements calculated as follows.

$E(P) = (|s_1 - y_1|, |s_2 - y_2|, \ldots, |s_N - y_N|)$.

In this study, the error vectors of individuals competing in a tournament are compared using a Wilcoxon signed rank test. The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test used when comparing two related samples to assess whether their population mean ranks differ [20]. This test is used as an alternative to the paired Student's t-test when the population cannot be assumed to be normally distributed. Let N be the sample size of the test and $x_{1,i}$ and $x_{2,i}$ denote the ith pair sample. Let H_0: be the hypothesis that difference between the pairs follows a symmetric distribution around zero and H_1: be the hypothesis that difference between the pairs does not follow a symmetric distribution around zero. The test is performed as follows:

1. For $i = 1, \ldots, N$, calculate $|x_{2,i} - x_{1,i}|$, and $sgn(x_{2,i} - x_{1,i})$, where sgn is the sign function:

$$sgn(x) := \begin{cases} -1 & \text{if } x < 0, \\ 0 & \text{if } x = 0, \\ 1 & \text{if } x > 0. \end{cases} \quad (1)$$

2. Exclude pairs with $|x_{2,i} - x_{1,i}| = 0$. Let N_r be the reduced sample size.
3. Order the remaining N_r pairs from smallest absolute difference to largest absolute difference, $|x_{2,i} - x_{1,i}|$.

4. Rank the pairs, starting with the smallest as 1st. Ties receive a rank equal to the average of the ranks they span. Let R_i denote the rank.
5. Calculate the test statistic W, the sum of the signed ranks:

$$W = \sum_{i=1}^{N_r} [sgn(x_{2,i} - x_{1,i}) \cdot R_i] \tag{2}$$

6. The value of W is compared to a threshold to decide if the null hypothesis H_0 is rejected.

In practice, *p-value* is often calculated from the test. This value is defined as the probability of obtaining a result equal to or more extreme than what was actually observed, when the null hypothesis is true [38]. If *p-value* is smaller than a threshold (called the critical value), the null hypothesis is rejected and two data set are significantly different. Otherwise, we can not reject the null hypothesis. In statistics, two popular values of *p-value* (0.01 and 0.05) are often used. These values correspond to the confident level of 99% and 95% to reject the null hypothesis, respectively.

3. Related work

Selection is a key factor that affects the performance of Evolutionary Algorithms (EAs) [10]. Commonly used selection strategies in EAs include fitness proportionate selection, rank selection, and tournament selection [4]. The most popular selection method in GP is tournament selection [45]. In standard tournament selection, a number of individuals (tournament size) are randomly selected from the population. These individuals are compared with each other and the winner (in terms of better fitness) is selected to go to the mating pool. The advantage of tournament selection is that it allows the adjustment of the selection pressure by tuning the tournament size. A small value of the tournament size leads to a low selection pressure while a large one results in a high selection pressure.

Since tournament selection is the most popular selection method in GP, there have been many studies to analyse its behaviour and improve its effectiveness. The majority of the early studies have focused on sampling and selecting. Gathercole et al. [14] analysed the selection frequency of each individual and the likelihood of not-selected and not-sampled individuals in tournament selection with different values of the tournament size. Sokolov and Whitley proposed unbiased tournament selection [41] where all individuals have a fair chance to participate in a tournament.

Xie and his colleagues conducted a series of studies to investigate tournament selection in GP. Xie indicated that standard tournament selection can lead to the result in which the individuals with bad fitness could be selected multiple times while the individuals with good fitness not selected any time [44]. Thus, he proposed the fully covered tournament selection method [44] which excludes the sampled individuals in the next tournament to ensure that each individual has an equal chance to participate into tournaments. Next, Xie et al. [45,47] analysed the performance of no-replacement tournament selection in which no individual can be sampled multiple times into the same tournament. Another problem in tournament selection is that some individuals are not sampled at all when using small values of the tournament size. However, Xie at al. [48] showed that the not-sampled issue does not seriously affect the selection performance in standard tournament selection.

Overall, previous research has shown that sampling strategies have a minor impact on GP performance. Consequently, researchers have paid more attention to the second step in tournament selection: selection. Goldberg and Deb introduced binary tournament selection [15] in which two individuals are selected at random, and the individual with better fitness could be selected with probability p, $0.5 < p \leq 1.0$. Back [3] ranked the best individual first and the selection probability of an individual of rank j is calculated by:

$$N^{-k}((N - j + 1)^k - (N - j)^k) \tag{3}$$

where k is the tournament size and N is the population size. Conversely, Blickle and Thiele [4] ranked the worst individual first and introduced the cumulative fitness distribution, $S(f_j)$, which denotes the number of individuals with fitness value f_j or worse. Finally, selection probability of an individual with rank j is calculated as:

$$\left(\frac{S(f_j)}{N}\right)^k - \left(\frac{S(f_{j-1})}{N}\right)^k. \tag{4}$$

Julstrom and Robinson also ranked the worst individual first and proposed weighted k-tournaments method [19]. A parameter w between 0 and 1 is chosen and the selection probability of an individual with rank j is calculated by the formula:

$$\frac{k(1-w)}{N^k(1-w^k)}((j-1) + w(N-j))^{k-1} \tag{5}$$

Later, Hingee and Hutter [18] introduced the polynomial rank scheme of degree d for calculating the probability of an individual of rank j as follows:

$$P(I = j) = \sum_{t=1}^{d+1} a_t j^{t-1} \tag{6}$$

Algorithm 1: Statistics tournament selection with random.

```
Input: Tour size, Population.
Output: The winner individual.
A ⟵ RandomIndividual();
for i ← 1 to TourSize do
    B ⟵ RandomIndividual();
    sample1 ⟵ Error(A);
    sample2 ⟵ Error(B);
    p − value ⟵ Testing(sample1, sample2);
    if p − value <alpha then
        A ⟵ GetBetterFitness(A, B);
    else
        A ⟵ GetRandom(A, B);
    end
end
TheWinnerIndividual ⟵A ;
```

They also showed that every probabilistic tournament is equivalent to a unique polynomial rank scheme.

Recently, researchers have focused on adapting the selection pressure. Xie and Zhang [46] proposed a method for automatically tuning the selection pressure based on the fitness rank distribution of the population. In each generation, they clustered the population into S clusters. Next, they sampled K clusters from S clusters with replacement and selected the winner among K sampled clusters. Finally, a random individual is returned from the winning cluster. In grammatical evolution, Forstenlechner et al. introduced semantic clustering selection [12]. The individuals in a GP population are clustered based on the similarity of their error vectors. Then, parents are drawn from the same cluster to improve semantic locality. Moreover, semantic diversity is managed through the preservation of the existence of multiple clusters.

More recently, Helmuth and Matheson proposed lexicase selection [17]. The idea is to evaluate the goodness of an individual based on part of fitness cases instead of all fitness cases. Each time a parent must be selected, lexicase selection randomly shuffles the list of fitness cases. Then, it removes any individual that did not achieve the best error value on the first fitness case. If more than one individual remains in the population, the first fitness case is removed and this process is repeated with the next fitness case. This technique was then extended to the real-valued regression problem by La Cava et al. [25] and was proved to maintain better diversity compared to standard tournament selection [16,33].

In this paper, we propose a new method for selecting the winner in tournament selection using the statistical analysis of the semantics of GP programs. Specifically, we focus our attention on the error vector produced in symbolic regression problems. The most similar approach in the literature was Semantic in Selection (SiS) technique [13] which calculates the semantic similarity of parents and selects parents which are semantically dissimilar (i.e., have large differences between their semantic vectors). Rather than computing differences in semantic vectors we perform a statistical analysis based on error vectors to ascertain semantic diversity of the individuals competing in a tournament. A detailed description of our method will be presented in the next section.

4. Methods

This section presents three statistics tournament selection techniques using the Wilcoxon signed rank test. The objective is to select breeding parents based on the statistical test instead of on their fitness values. The first method is called *statistics tournament selection with random* [6] and shortened as TS-R. The main objective of TS-R is to promote the semantic diversity of GP population and the process of TS-R is similar to standard tournament selection. However, instead of using the fitness value for comparing between individuals in the tournament, a statistical test was applied to the error vectors of these individuals. For a pair of individuals, if the test shows that the individuals are different, then the individual with better fitness value is considered as the winner. Conversely, if the test confirms that two individuals are not different, a random individual is selected from the pair. Next, the winner is tested against other individuals in the tournament size. The detailed description of TS-R is presented in Algorithm 1.

In Algorithm 1, function *RandomIndividual*() returns a random individual from the GP population. Function *Error*(A) assigns the error vector of individual A to *sample*1 [1] and function *Testing*(*sample*1, *sample*2) performs the Wilcoxon signed rank test. Two last functions, *GetBetterFitness*(A, B) and *GetRandom*(A, B) find the better fitness individual among A and B or return a random individual between two, respectively. Finally, *alpha* is the critical value used to decide if the null hypothesis

[1] To reduce the computational time, the error vectors of all individuals are calculated at the beginning of each generation. These error vectors are then stored for using in every statistical test in the same generation.

Algorithm 2: Statistics tournament selection with size.

```
Input: Tour size, Population.
Output: The winner individual.
A ⟵ RandomIndividual();
for i ← 1 to TourSize do
    B ⟵ RandomIndividual();
    sample1 ⟵ Error(A);
    sample2 ⟵ Error(B);
    p − value ⟵ Testing(sample1, sample2);
    if p − value <alpha then
        A ⟵ GetBetterFitness(A, B);
    else
        A ⟵ GetSmallerSize(A, B);
    end
end
TheWinnerIndividual ⟵A ;
```

Algorithm 3: Statistics tournament selection with probability.

```
Input: Tour size, Population.
Output: The winner individual.
A ⟵ RandomIndividual();
for i ← 1 to TourSize do
    B ⟵ RandomIndividual();
    sample1 ⟵ Error(A);
    sample2 ⟵ Error(B);
    p − value ⟵ Testing(sample1, sample2);
    A ⟵ GetBetterWithProbability(A, B, p − value);
end
TheWinnerIndividual ⟵A ;
```

is rejected. If *p-value* is smaller than *alpha*, then the null hypothesis is rejected and the better fitness individual is selected as the winner. If the test can not reject the null hypothesis, then a random individual is selected from the pair.

The second proposed tournament selection is called *statistics tournament selection with size* [6] and shorted as TS-S. TS-S is similar to TS-R in the objective of promoting diversity. Moreover, TS-S also aims at reducing the code growth in GP population. In TS-S, if the statistical test can not reject the null hypothesis, then the individual with smaller size is selected from the pair. The detailed description of TS-S is presented in Algorithm 2 in which function *GetSmallerSize*(*A, B*) returns the individual with smaller size among *A* and *B*. If the size of *A* and *B* are tie, then the first individual will be return.

The third tournament selection method is called *statistics tournament selection with probability* and shorted as TS-P. This technique is different from TS-R and TS-S in which it does not rely on the critical value to decide the winner. Instead, TS-P uses *p-value* as the probability to select the winner. The better and the worse fitness individual is selected with the probability of 1− *p-value* and *p-value*, respectively. The detailed description of TS-P is presented in Algorithm 3 . In this algorithm, $GetBetterWithProbability(A, B, p - value)$ return the better fitness individual between *A* and *B* with the probability of 1− *p-value* and return the worse fitness individual with the probability of *p-value*.

In three proposed tournament selection techniques, a series of the Wilcoxon signed rank test is applied on the error vectors of sampled individuals. Potentially, there are two limitations. First, the overuse of statistical tests may result in the significant difference being detected by chance [7]. This, subsequently, may affect the performance of the statistics tournament selection. In Section 6 we show that, on average, approximate 50% to 70% of the Wilcoxon test in TS-R and TS-S is significant. This large number may help to alleviate the impact of the test that are affected by chance.

Second, the overhead of computational time that results from executing a series of the statistical test. We compare the computational time of the selection step in statistics tournament selection and in standard tournament selection and find that the execution time of the selection step in statistics tournament selection is greater than that in standard tournament selection. However, statistics tournament selection often helps to remarkably reduce the code growth of GP population. Subsequently, the GP system using statistics tournament selection often runs faster than the system that uses standard tournament selection.

Table 1
Problems for testing statistics tournament selection techniques.

Abbreviation	Name	No. features	No. training	No. testing
A. Benchmarking Problems				
F1	korns-11	5	20	20
F2	korns-12	5	20	20
F3	korns-14	5	20	20
F4	vladislavleva-2	1	100	221
F5	vladislavleva-4	5	500	500
F6	vladislavleva-6	2	30	93636
F7	vladislavleva-8	2	50	1089
F8	korns-1	5	1000	1000
F9	korns-2	5	1000	1000
F10	korns-3	5	1000	1000
F11	korns-4	5	1000	1000
F12	korns-11	5	1000	1000
F13	korns-12	5	1000	1000
F14	korns-14	5	1000	1000
F15	korns-15	5	1000	1000
B. UCI Problems				
F16	airfoil_self_noise	5	800	703
F17	casp	9	100	100
F18	ccpp	4	1000	1000
F19	wpbc	31	100	98
F20	3D_spatial_network	3	750	750
F21	protein_Tertiary_Structure	9	1000	1000
F22	yacht_hydrodynamics	6	160	148
F23	slump_test_Compressive	7	50	53
F24	slump_test_FLOW	7	50	53
F25	slump_test_SLUMP	7	50	53

Table 2
Evolutionary parameter values.

Parameters	Value
Population size	500
Generations	100
Tournament size	3, 5, 7
Crossover, mutation probability	0.9; 0.1
Function set	$+, -, *, /, sin, cos$
Terminal set	$X_1, X_2, \ldots, X_n$
Initial Max depth	6
Max depth	17
Max depth of mutation tree	15
Raw fitness	mean absolute error on all fitness cases
Trials per treatment	100 independent runs for each value
Elitism	Copy the best individual to the next generation.

5. Experimental settings

We tested the proposed tournament selection techniques on twenty-five regression problems. Among them, fifteen problems are GP benchmark problems recommended in the literature [43] and an additional ten problems were taken from UCI machine learning repository [2]. For each problem, we also created a noisy version from the original (noiseless) form that results in twenty-five noisy datasets. Totally, fifty datasets were used for the experiments. The detailed description of the tested problems including the abbreviation, their name, number of features, number of training and testing samples are presented in Table 1.

The GP parameters used for our experiments are shown in Table 2. The terminal set for each problem includes N variables corresponding to the number of features of that problem. The raw fitness is the mean of absolute error on all fitness cases. Therefore, smaller values are better. Three popular values of tournament size (referred to as tour-size hereafter) including 3, 5 and 7 were tested. [2] The critical value in the Wilcoxon test in TS-R and TS-S is $alpha = 0.05$. For each problem and each parameter setting, 100 runs were performed.

For statistical analysis, we followed Derrac [8] to employ the Friedman's test on the results. If the Friedman test shows that at least one technique is significantly different from the others with confident level of 95%, we conducted a post-hoc

[2] Due to the space limitation, we only show in this paper the results with tour-size=3 and tour-size=7. The results with tour-size=5 are presented in the supplement of the paper at https://github.com/chuthihuong/GP.

Table 3
The median of testing error with tour-size=3 (the left) and tour-size=7 (the right). Bold indicates the best (lowest) value. The result is marked + if it significantly smaller than GP. Conversely, it is marked−.

Pro	GP	SiS	TS-R	TS-S	TS-P	GP	SiS	TS-R	TS-S	TS-P
F1	8.55	9.06	5.31^{+}	**3.93**$^{+}$	6.68^{+}	11.3	10.2	6.13^{+}	**3.97**$^{+}$	6.18^{+}
F2	0.96	0.99	0.88	**0.81**$^{+}$	0.92	0.98	1.00	0.89^{+}	**0.82**$^{+}$	0.98
F3	33.4	34.2	15.7^{+}	**14.2**$^{+}$	16.7^{+}	33.4	34.2	14.8^{+}	**13.5**$^{+}$	16.7^{+}
F4	0.06	0.06	0.06	**0.05**	0.06	0.06	0.05	0.05	0.06	**0.04**$^{+}$
F5	0.135	0.137^{-}	0.136	**0.131**	0.135	0.135	0.135	0.135	**0.130**$^{+}$	0.131
F6	1.78	**1.45**	1.89	1.90	1.63	1.34	**1.21**	1.64	1.97	1.62
F7	1.70	1.69	1.74	**1.56**$^{+}$	1.75	1.77	1.75	1.80	**1.54**$^{+}$	1.72
F8	6.20	**4.52**	6.75	6.90	5.52	7.38	**6.79**	6.96	7.38	6.85
F9	1.66	1.63	1.61	1.58^{+}	**1.57**	1.71	1.66	**1.59**$^{+}$	1.59^{+}	1.61
F10	41.1	38.9	46.1	**35.8**	39.1	56.5	42.2	38.1	**34.1**	52.9
F11	0.085	0.086	0.086	**0.084**	0.085	0.084	0.085	0.085	**0.083**	0.084
F12	7.39	7.33^{+}	**7.32**$^{+}$	7.32^{+}	7.33	7.42	7.41	7.35^{+}	**7.33**$^{+}$	7.37^{+}
F13	0.876	0.875	0.873	**0.872**$^{+}$	0.875	0.878	0.875	0.874^{+}	**0.871**$^{+}$	0.878
F14	128.7	130.5	130.0	128.4	**126.2**	127.8	129.7	124.7	**122.7**	124.3
F15	4.95	5.32	5.14	**4.74**	4.90	4.23	4.43	4.32	**3.92**	4.68
F16	20.9	**20.8**	27.1	25.5	27.8	**18.8**	23.7	24.6	24.5	24.2
F17	4.99	4.93	4.90	**4.78**$^{+}$	4.87	4.93	4.99	4.91	**4.70**$^{+}$	4.88
F18	**8.72**	9.07	10.1	10.1	11.1	8.70	**8.31**	10.3	8.79	8.88
F19	41.3	41.1	38.8^{+}	**36.1**$^{+}$	39.9^{+}	42.1	42.2	38.5^{+}	**36.8**$^{+}$	39.5^{+}
F20	**9.54**	9.91	9.75	9.56	9.75	**9.42**	9.57	9.68	9.52	9.53
F21	**4.35**	4.38	4.44	4.37	4.42	**4.27**	4.29	4.32	4.30	4.29
F22	2.11	2.03	2.40	**1.93**	2.09	1.86	1.81	1.97	1.90	**1.73**
F23	7.74	7.24	7.99	5.89^{+}	7.90	6.65	7.37	6.63	8.04^{-}	6.01
F24	16.7	18.4	15.8	**14.9**$^{+}$	18.1	17.5	18.3	15.5^{+}	**13.3**$^{+}$	16.4
F25	8.70	8.44	8.31	**7.99**$^{+}$	8.48	8.89	8.43	**7.99**$^{+}$	8.40^{+}	8.46^{+}

analysis with the BonferroniDunn correction of the *p-value* for each comparison [8]. In the following tables, if the result of a method is significantly better than GP with standard tournament selection (shorthanded as GP hereafter), this result is marked + at the end. Conversely, if it is significantly worse compared to GP, this result is marked−.

The source code of all tested methods are available for download.[3] All techniques were implemented in Java with the exception of neatGP for which we used the implementation in Python.[4] Moreover, the same computing platform (Operating system: Windows 7 Ultimate (64bit), RAM 16.0GB, Intel®Core^{TM}i7-4790 CPU@3.60GHz) was used in every experiment in this paper.

We divided our experiment into three sets. The first aims at investigating the performance of three variants of tournament selection based on statistical analysis. The second attempts to improve the performance of the proposed selection strategy through its combination with a state of the art semantic crossover operator [36]. The third set of experiments examine the performance of the strategies on noisy instances of the problems.

6. Performance analysis of statistics tournament selection

This section analyses the performance of the statistics tournament selection methods and compares them with GP and semantic in selection (SiS) by Galvan-Lopez et al.[13]. The first metric used in the comparison is the generalisation ability of the tested methods [1,42]. The median of the testing error across 100 runs is shown in Table 3. We can see that the testing error of SiS and GP are roughly equal. The difference between the testing error of two techniques is often marginal. SiS is only significant better than GP on F12 and GP is only significantly better than SiS on F5 with tour-size=3. Conversely, the testing error of three statistics tournament selection are often smaller than that of GP. Among three statistics tournament selection, the performance of TS-S is the best on the testing data. TS-S achieved the best performance on 17 problems with tour-size=3 and 14 problems with tour-size=7.

In terms of statistical comparison using Friedman's test, the proposed tournament selection is often significantly better (with the confident level of 95%) than GP. For tour-size=3, TS-R is significantly better than GP on 4 problems, TS-S is significantly better than GP on 12 problems and TS-P is significantly better than GP on 3 problems. For tour-size=7, these values are 9,12 and 6 problems, respectively. Conversely, GP is only significantly better than TS-S on one problem with the tour-size=7 (F23).

The second metric is the average size of their solutions. These values are presented in Table 4. While the solutions found by SiS are often as complex as those found GP, the solutions found by statistics tournament selection are simpler than those of GP and SiS. Statistical test using Friedman's test also show that the size of the solutions obtained by TS-R, TS-S and TS-P

[3] https://github.com/chuthihuong/GP.
[4] http://www.tree-lab.org/index.php/resources-2/downloads/open-source-tools.

Table 4
The average of solution's size with tour-size=3 (the left) and tour-size=7 (the right). Bold indicates the best (lowest) value. The result is marked + if it significantly smaller than GP. Conversely, it is marked−.

Pro	GP	SiS	TS-R	TS-S	TS-P	GP	SiS	TS-R	TS-S	TS-P
F1	280	276	244	**121**$^+$	238$^+$	286	292	264	**100**$^+$	259
F2	169	170	130$^+$	**35**$^+$	148$^+$	160	173	150	**37**$^+$	169
F3	263	270	262	**124**$^+$	277	262	276	278	**98**$^+$	277
F4	171	190	225	**79**$^+$	197	184	177	217	**80**$^+$	194
F5	89	78	91	**53**$^+$	105	91	85	93	**42**$^+$	111
F6	167	156	141$^+$	**50**$^+$	159	137	151	134	**37**$^+$	145
F7	142	138	128	**48**$^+$	147	133	136	149	**38**$^+$	157
F8	180	166	174	**108**$^+$	184	247	215	197	**101**$^+$	210
F9	166	141	129$^+$	**73**$^+$	140	227	176$^+$	161$^+$	**74**$^+$	166$^+$
F10	161	145	162	**108**$^+$	162	184	172	165	**101**$^+$	175
F11	148	139	149	**76**$^+$	152	149	151	154	**66**$^+$	159
F12	259	222$^+$	179$^+$	**92**$^+$	188$^+$	306	258	211$^+$	**89**$^+$	226$^+$
F13	169	146	119$^+$	**32**$^+$	135$^+$	161	152	117$^+$	**29**$^+$	152
F14	254	210	268	**168**$^+$	273	332	284	300	**164**$^+$	287
F15	155	132	147	**112**$^+$	163	157	150	133	**84**$^+$	156
F16	200	184	193	**152**$^+$	189	262	252	260	**203**$^+$	276
F17	207	182	168	**50**$^+$	165	230	220	192	**39**$^+$	192
F18	160	150	165	**119**$^+$	165	226	207	206	**132**$^+$	203
F19	208	200	119$^+$	**16**$^+$	152$^+$	313	280	100$^+$	**11**$^+$	182$^+$
F20	198	177	193	**125**$^+$	199	299	290	264	**178**$^+$	282
F21	178	149$^+$	179	**97**$^+$	179	236	196	219	**71**$^+$	199
F22	186	171	178	**105**$^+$	182	194	185	190	**85**$^+$	184
F23	160	159	132$^+$	**56**$^+$	134$^+$	204	200	149$^+$	**24**$^+$	155$^+$
F24	164	168	115$^+$	**45**$^+$	137	220	195	131$^+$	**20**$^+$	155$^+$
F25	170	167	111$^+$	**31**$^+$	138$^+$	226	205	132$^+$	**22**$^+$	157$^+$

Table 5
Average semantic distance with tour-size=3. Bold indicates the value of SiS or TS-S is greater than the value of GP.

Pro	GP	SiS	TS-S	Pro	GP	SiS	TS-S
F1	2.42	**8.93**	**3.76**	F14	11.22	**11.88**	**12.30**
F2	0.42	**1.60**	**0.43**	F15	10.36	**12.28**	**12.79**
F3	7.01	**19.73**	5.02	F16	71.08	**101.43**	**78.42**
F4	1.05	**1.55**	**1.24**	F17	60.91	13.52	**136.99**
F5	0.07	**0.63**	**0.10**	F18	105.55	**366.87**	**123.34**
F6	0.99	**1.58**	**1.03**	F19	35.12	**51.21**	7.83
F7	0.44	**0.58**	**0.70**	F20	12.43	**21.87**	**15.25**
F8	38.28	**426.28**	**57.13**	F21	62.09	20.84	**96.44**
F9	8.79	**16.88**	**8.86**	F22	6.83	**7.56**	**11.61**
F10	8.16	**12.66**	**10.65**	F23	42.25	29.08	**53.12**
F11	4.32	**5.18**	**6.13**	F24	43.10	**44.05**	**80.74**
F12	6.12	**7.10**	**8.81**	F25	37.19	17.49	**41.42**
F13	2.81	**3.86**	**10.69**				

is significantly smaller than that of GP on most problem. Especially, the size of the solutions of TS-S is always much smaller than that of GP on all problems. This provides a reason partially explaining why the performance of TS-S on the testing data is better than other techniques in Table 3 following the Occam Razor principle [29].

We also measured the semantic distance between parents and their children of GP, SiS and TS-S [5] using the similar approach to Nguyen et al. [32]. This information shows the ability of a method to discover different areas in the search space. The semantic distance between a pair of individuals (e.g. a parent and a child) averaged over the population and over 100 runs with tour-size=3 is presented in Table 5. Apparently, both SiS and TS-S maintained higher semantic diversity compared to GP. TS-S and SiS preserved better semantic diversity than GP on 23 and 21 problems, respectively. These results show that TS-S achieved one of its objective in enhancing semantic diversity of GP population.

The last result in this section is the percentage of rejecting the null hypothesis ($N_{rejnull}$) in the Wilcoxon test of TS-S and TS-R with tour-size=3. This value is calculated in Eq. (7)

$$N_{rejnull} = \frac{N_p}{N_{test}} \tag{7}$$

[5] We focus on analysing TS-S since this is the best selection approach among three proposed techniques.

Table 6
Average percentage of rejecting the null hypothesis in Wilcoxon test of TS-R and TS-S with tour-size=3.

Pro	TS-R	TS-S	Pro	TS-R	TS-S	Pro	TS-R	TS-S
F1	25.32%	50.91%	F10	71.95%	81.93%	F19	39.83%	86.09%
F2	19.68%	53.97%	F11	62.18%	84.25%	F20	75.61%	84.27%
F3	30.75%	51.26%	F12	43.31%	69.85%	F21	66.13%	77.81%
F4	63.60%	74.61%	F13	38.82%	78.64%	F22	71.36%	80.85%
F5	59.65%	68.34%	F14	66.87%	78.65%	F23	43.22%	77.46%
F6	39.80%	47.88%	F15	85.17%	87.53%	F24	42.26%	73.06%
F7	33.69%	63.44%	F16	82.01%	87.30%	F25	35.80%	79.95%
F8	70.29%	88.25%	F17	43.98%	69.74%			
F9	62.91%	77.23%	F18	82.29%	87.28%			

Table 7
Median of testing error with tour-size=3 (the left) and tour-size=7 (the right). Bold indicates the best (lowest) value. The result is marked + if it significantly smaller than GP. Conversely, it is marked−.

Pro	GP	neatGP	TS-S	RDO	TS-RDO	GP	neatGP	TS-S	RDO	TS-RDO
F1	8.55	12.5$^-$	**3.93**$^+$	8.91	4.19$^+$	11.3	12.5	3.97$^+$	8.88	4.52$^+$
F2	0.96	0.84	**0.81**	1.17	0.97	0.98	0.84	0.82	1.19	0.96
3	33.4	32.2	14.2$^+$	3.73$^+$	**1.61**$^+$	33.4	32.2	13.5$^+$	5.92$^+$	**1.87**$^+$
F4	0.06	0.12$^-$	0.05	0.02$^+$	**0.02**$^+$	0.06	0.12$^-$	0.06	0.02$^+$	**0.02**$^+$
F5	0.135	0.135	**0.131**	0.14	0.14	0.13	0.13	0.13	0.14	0.14$^-$
F6	1.78	1.74	1.90	0.00$^+$	**0.00**$^+$	1.34	1.74$^-$	1.97	0.00$^+$	**0.00**$^+$
F7	1.698	1.61	1.56	1.38$^+$	**1.06**$^+$	1.77	1.61	1.54$^+$	**1.15**$^+$	**1.33**$^+$
F8	6.20	7.41$^-$	6.90	**0.00**$^+$	0.00$^+$	7.38	7.41	7.38	0.00$^+$	**0.00**$^+$
F9	1.66	2.41	1.58	**0.01**$^+$	0.11$^+$	1.71	2.41	1.59	**0.23**$^+$	0.23$^+$
F10	41.1	41.0	35.8	0.00$^+$	**0.00**$^+$	56.5	41.0	34.1	0.47$^+$	**0.00**$^+$
F11	0.08	0.30$^-$	0.08	**0.00**$^+$	0.08$^+$	0.08	0.30$^-$	0.08	**0.00**$^+$	0.08
F12	7.39	7.34	**7.32**$^+$	7.49$^-$	7.32	7.42	7.34$^+$	7.33$^+$	7.40	**7.28**$^+$
F13	0.88	0.87	**0.87**$^+$	0.88	0.87	0.88	0.87$^+$	0.87$^+$	0.88	0.87$^+$
F14	128.7	131.3$^-$	128.4	124.3	**121.8**$^+$	127.8	131.3$^-$	122.7$^+$	125.9	**121.7**$^+$
F15	4.95	5.92	4.74	**3.24**$^+$	**3.24**$^+$	4.23	5.92$^-$	3.92	**3.24**$^+$	**3.24**$^+$
F16	20.9	33.7$^-$	25.5	6.11$^+$	**5.75**$^+$	18.8	33.7$^-$	24.5	6.46$^+$	**5.91**$^+$
F17	4.99	4.95	**4.78**$^+$	5.50$^-$	4.85	4.93	4.95	4.70$^+$	5.63$^-$	4.74$^+$
F18	8.72	28.49$^-$	10.18	**3.56**$^+$	3.56$^+$	8.70	28.49$^-$	8.79	3.63$^+$	**3.61**$^+$
F19	41.3	38.3$^+$	36.1$^+$	41.5	**32.3**$^+$	42.1	38.3$^+$	36.8$^+$	39.1$^+$	**32.2**$^+$
F20	9.54	**9.18**	9.56	12.0$^-$	11.4$^-$	9.42	**9.18**	9.52	12.2$^-$	11.4$^-$
F21	4.35	4.52$^-$	4.37	4.19$^+$	**4.17**$^+$	4.27	4.52$^-$	4.30	4.24	**4.18**$^+$
F22	2.11	3.29$^-$	1.93	**1.09**$^+$	1.15$^+$	1.86	3.29$^-$	1.90	1.23$^+$	**1.23**$^+$
F23	7.74	8.44	5.89$^+$	5.72	**4.32**$^+$	6.65	8.44$^-$	8.04$^-$	6.84	**4.03**$^+$
F24	16.7	17.7	14.9$^+$	22.2$^-$	**14.8**	17.5	17.7	13.3$^+$	23.3	14.3$^+$
F25	8.70	8.89	**7.99**$^+$	12.2$^-$	8.13	8.89	8.89	8.40$^+$	16.0$^-$	**7.07**$^+$

in which N_p is the number of the Wilcoxon test rejecting the null hypothesis and N_{test} is the total number of the Wilcoxon test conducted in each selection technique. Table 6 shows that there is a large number of the test that rejected the null hypothesis. This value of TS-R is often from 30% to 70% and the value of TS-S is slightly higher (from 50% to nearly 90%). Thus, the statistical test will have a considerable impact to the selection process in TS-R and TS-S [7].

Overall, three statistics tournament selection methods find simpler solutions and generalize better on unseen data. Particularly, the solutions found by TS-S are much less complex than those of GP. Moreover, the generalization ability of TS-S is also better compared to GP and SiS.

7. Combining semantic tournament selection with semantic crossover

In this section, we present an improvement of TS-S (the best approach among three proposed selection techniques) performance by combining this technique with a recently proposed crossover - random desired operator (RDO) [36]. In other words, we used RDO instead of standard crossover in TS-S. The resulting GP system is called statistics tournament selection with random desired operator and referred to as TS-RDO. The reason for combining RDO with TS-S is that the training error of TS-S is often worse than GP [6]. Moreover, RDO has been showed to perform well on the training data [32,36]. We predict that the coupling of these semantic selection and semantic crossover strategies in the form of TS-RDO will lead to the improved performance.

We compare TS-RDO with TS-S, neatGP (a state of the art bloat control approach) [26], RDO [36] and GP. The setting for RDO is similar to the setting in [32]. The result on the testing data of these methods is shown in Table 7. It can be seen that TS-RDO achieved the best result among five tested techniques. The testing error of TS-RDO is smallest on 12 and 14 problems with tour-size=3 and tour-size=7, respectively. Furthermore, TS-RDO is more frequently significantly better than

Table 8
Average of solutions size with tour-size=3 (the left) and tour-size=7 (the right). The best (lowest) value is printed bold. The result is marked + if it significantly smaller than GP. Conversely, it is marked−.

Pro	GP	neatGP	TS-S	RDO	TS-RDO	GP	neatGP	TS-S	RDO	TS-RDO
F1	280	124$^+$	121$^+$	238$^+$	**78**$^+$	286	124$^+$	100$^+$	219$^+$	**56**$^+$
F2	169	60$^+$	**35**$^+$	174$^-$	80$^+$	160	60$^+$	37$^+$	167$^-$	47$^+$
F3	263	112$^+$	124$^+$	153$^+$	**59**$^+$	262	112$^+$	98$^+$	169$^+$	**48**$^+$
F4	171	**60**$^+$	79$^+$	320$^-$	207$^-$	184	**60**$^+$	80$^+$	311$^-$	146$^+$
F5	89	**12**$^+$	53$^+$	49$^+$	23$^+$	91	**12**$^+$	42$^+$	46$^+$	13$^+$
F6	167	45$^+$	50$^+$	40$^+$	**20**$^+$	137	45$^+$	37$^+$	50$^+$	**18**$^+$
F7	142	50$^+$	**48**$^+$	240$^-$	83$^+$	133	50$^+$	38$^+$	197$^-$	73$^+$
F8	180	118$^+$	108$^+$	21$^+$	**12**$^+$	247	118$^+$	101$^+$	14$^+$	**9**$^+$
F9	166	62$^+$	73$^+$	53$^+$	**36**$^+$	227	62$^+$	74$^+$	74$^+$	**37**$^+$
F10	161	60$^+$	108$^+$	71$^+$	**51**$^+$	184	60$^+$	101$^+$	105$^+$	**57**$^+$
F11	148	44$^+$	76$^+$	28$^+$	**15**$^+$	149	44$^+$	66$^+$	39$^+$	**13**$^+$
F12	259	67$^+$	92$^+$	181$^+$	**55**$^+$	306	67$^+$	89$^+$	162$^+$	**32**$^+$
F13	169	49$^+$	32$^+$	142$^+$	**22**$^+$	161	49$^+$	29$^+$	113$^+$	**17**$^+$
F14	254	**66**$^+$	168$^+$	160$^+$	72$^+$	332	66$^+$	164$^+$	165$^+$	**55**$^+$
F15	155	58$^+$	112$^+$	53$^+$	**40**$^+$	157	58$^+$	84$^+$	45$^+$	**35**$^+$
F16	200	**103**$^+$	152$^+$	279$^-$	186$^+$	262	**103**$^+$	203$^+$	326$^-$	165$^+$
F17	207	62$^+$	**50**$^+$	207$^+$	106$^+$	230	62$^+$	39$^+$	247$^-$	81$^+$
F18	160	**71**$^+$	119$^+$	305$^-$	196$^-$	226	**71**$^+$	132$^+$	380$^-$	178$^+$
F19	208	79$^+$	16$^+$	83$^+$	**9**$^+$	313	79$^+$	11$^+$	84$^+$	**7**$^+$
F20	198	**87**$^+$	125$^+$	328$^-$	237$^-$	299	**87**$^+$	178$^+$	387$^-$	208$^+$
F21	178	**63**$^+$	97$^+$	199$^-$	113$^+$	236	**63**$^+$	71$^+$	243$^-$	102$^+$
F22	186	83$^+$	105$^+$	139$^+$	**58**$^+$	194	83$^+$	85$^+$	127$^+$	**46**$^+$
F23	160	55$^+$	56$^+$	245$^-$	118$^+$	204	**55**$^+$	24$^+$	286$^-$	84$^+$
F24	164	68$^+$	**45**$^+$	240$^-$	97$^+$	220	68$^+$	20$^+$	291$^-$	46$^+$
F25	170	63$^+$	**31**$^+$	227$^-$	92$^+$	226	63$^+$	22$^+$	265$^-$	63$^+$

GP compared to TS-S. TS-RDO is significantly better than GP on 17 problems with tour-size=3 and on 21 problems with tours-size=7 while these values of TS-S are only 10 and 11, respectively. RDO achieved the second best (behind TS-RDO) result. The testing error of RDO is often smaller than that of GP on all problems. This is consistent with the result in Pawlak et al. [32,36] where RDO has been reported to perform well on unseen data.

Among five examined methods in Table 7, the performance of neatGP is worst. neatGP is only significantly better than GP on 2 and 4 problems, while it is worse than GP on 9 and 10 problems correspondingly with tour-size=3 and tour-size=7. This result is slightly different with the result in Trujillo et al. [26] where neatGP is showed performing equally well on unseen data compared to GP. The reason could be that the tested problems in our experiments are more difficult than those in [26]. Further research needed to examine this.

The average size of the solutions are presented in Table 8. The size of the solutions obtained by both neatGP and TS-S are significantly smaller than that of GP. Comparing between neatGP and TS-S, the table shows that their solution's size is roughly equal. For RDO, its solutions are also often smaller than the solutions of GP. However, this seems only true on the benchmarking problems. On most of UCI problems, RDO's solutions are more complex than the solutions of GP. The best technique in Table 8 is TS-RDO. This method achieved the best result on most problem regarding to the solution's size. The average size of the solutions found by TS-RDO is the smallest one on 12 and 13 problems with tour-size=3 and tour-size=7, respectively.

Overall, TS-RDO improves the testing error, and further reduces the size of the solutions compared to TS-S. Moreover, this technique performs better than both RDO and neatGP, two recently proposed methods for improving GP performance and reducing GP code bloat.

8. Performance analysis on the noisy data

This section investigates the performance of five methods in Section 7 on the noisy data. In data mining, it has been observed that the problems will become harder when they are incorporated with noise [39,40]. We created a noisy dataset from the original one by adding 10% Gaussian noise with zero mean and one standard deviation in the all features and objective function of the problems in Table 1. Moreover, the noise is installed for both the training and the testing data. The testing error on the noisy data is shown in Table 9.

There are some interesting results observed from Table 9. First, TS-RDO performs slightly more consist on the noisy data compared to the noiseless data. The best testing error is mostly achieved by TS-RDO on all problems with both values of the tournament size. Second, the performance of RDO on the noisy data is not as good as on the noiseless data. This technique only achieved the best performance on one problem with tour-size=7. This evidenced that RDO is prone to be over-fitted on the noisy problems. Third, the performance of TS-S is also robust and more consistent than on the noiseless data. This method is significantly better than GP on 11 and 13 problems with tour-size=3 and tour-size=7 while these values on the noiseless data are only 10 and 11, respectively. Last, neatGP is still the worst method regarding to the generalization ability.

Table 9
Median of testing error on the noisy data with tour-size=3 (the left) and tour-size=7 (the right). Bold indicates the best (lowest) value. The result is marked + if it significantly smaller than GP. Conversely, it is marked−.

Pro	GP	neatGP	TS-S	RDO	TS-RDO	GP	neatGP	TS-S	RDO	TS-RDO
F1	9.68	13.1$^-$	**5.88**$^+$	10.3	7.99	9.19	13.1$^-$	**5.13**$^+$	10.2	6.53$^+$
F2	0.92	0.84	**0.81**	1.17$^-$	1.01$^-$	0.90	0.84	**0.79**$^+$	1.14$^-$	0.92
F3	29.6	32.2	15.9$^+$	7.06$^+$	**6.28**$^+$	34.8	32.2	16.8$^+$	7.30$^+$	**6.28**$^+$
F4	0.15	0.19$^-$	0.145	0.143	**0.136**$^+$	0.151	0.19$^-$	0.147	0.143	**0.138**$^+$
F5	0.14	0.14	**0.139**$^+$	0.141$^-$	0.14	0.14	0.14	**0.137**$^+$	0.14	0.14
F6	2.14	2.19	2.10	4.03$^-$	**1.36**$^+$	2.22	2.19	2.07$^+$	2.71	**1.39**$^+$
F7	1.74	1.73	1.71	1.64	**1.61**	1.79	1.73	1.64$^+$	1.78	**1.51**$^+$
F8	66.9	66.9	**66.8**$^+$	68.3$^-$	66.8$^+$	67.1	66.9	66.8$^+$	68.0	**66.6**$^+$
F9	5.52	5.68	5.34	5.19	**5.04**	5.49	5.68	5.24	**4.98**$^+$	5.10$^+$
F10	63.8	56.4	56.2	49.2$^+$	**46.3**$^+$	57.0	56.4	55.0	49.6$^+$	**47.1**$^+$
F11	0.20	0.32$^-$	0.20	0.20	**0.20**$^+$	0.20	0.32$^-$	0.20	0.20$^+$	**0.20**$^+$
F12	7.40	7.41	**7.31**$^+$	7.54$^-$	7.36	7.44	7.41$^+$	7.34$^+$	7.44	**7.30**$^+$
F13	0.90	0.90	**0.90**$^+$	0.91	0.90	0.90	0.90	**0.90**$^+$	0.90	0.90$^+$
F14	122.8	128.8$^-$	122.6	122.8	**122.6**$^+$	122.8	128.8$^-$	122.6	122.8	**122.5**$^+$
F15	5.01	6.21$^-$	5.07	4.15$^+$	**4.12**$^+$	4.20	6.21$^-$	4.13	4.13$^+$	**4.12**$^+$
F16	36.4	36.3	37.2	12.0$^+$	**11.4**$^+$	34.5	36.3	37.5	12.1$^+$	**11.55**$^+$
F17	5.61	5.45	5.42$^+$	6.41$^-$	**5.34**$^+$	5.70	5.45	5.30$^+$	6.55$^-$	**5.26**$^+$
F18	48.3	52.9$^-$	46.6	37.8$^+$	**36.8**$^+$	48.0	52.9$^-$	46.3	38.6$^+$	**36.7**$^+$
F19	42.8	40.2$^+$	37.7$^+$	39.5$^+$	**35.6**$^+$	44.9	40.2$^+$	37.7$^+$	38.4$^+$	**35.6**$^+$
F20	9.22	**8.72**$^+$	9.13	11.1$^-$	10.5$^-$	9.33	**8.72**$^+$	9.16	11.7$^-$	10.3$^-$
F21	4.53	4.67$^-$	4.54	4.36$^+$	**4.32**$^+$	4.49	4.67$^-$	4.50	4.42	**4.35**$^+$
F22	5.91	6.19$^-$	5.80	5.97	**5.43**$^+$	5.87	6.19$^-$	5.91	5.93	**5.50**$^+$
F23	8.84	9.15	**5.81**$^+$	8.96	6.07$^+$	7.52	9.15$^-$	9.19$^-$	9.29$^-$	**6.01**$^+$
F24	20.2	19.1	17.1$^+$	23.7	**16.5**$^+$	22.7	19.1	16.9$^+$	29.1	**16.0**$^+$
F25	9.36	9.42	8.38$^+$	14.8$^-$	**8.00**$^+$	9.58	9.42	8.50$^+$	13.9$^-$	**7.38**$^+$

Table 10
Average running time in seconds on noisy data with tour-size=3 (the left) and tour-size=7 (the right). Bold indicates the best (lowest) value. The result is marked + if it significantly smaller than GP. Conversely, it is marked−.

Pro	GP	neatGP	TS-S	RDO	TS-RDO	GP	neatGP	TS-S	RDO	TS-RDO
F1	4	863$^-$	**1**$^+$	32$^-$	10$^-$	3	863$^-$	**2**$^+$	34$^-$	9$^-$
F2	3	501$^-$	**1**$^+$	29$^-$	10$^-$	2	501$^-$	**1**$^+$	32$^-$	10$^-$
F3	4	831$^-$	**1**$^+$	29$^-$	11$^-$	2	831$^-$	**2**	32$^-$	11$^-$
F4	12	686$^-$	**6**$^+$	160$^-$	117$^-$	9	686$^-$	**8**	144$^-$	116$^-$
F5	20	177$^-$	**18**	690$^-$	556$^-$	**24**	177$^-$	25	627$^-$	595$^-$
F6	3	522$^-$	**1**$^+$	104$^-$	85$^-$	17	522$^-$	**2**$^+$	142$^-$	86$^-$
F7	4	448$^-$	**2**$^+$	75$^-$	34$^-$	3	448$^-$	**3**	93$^-$	37$^-$
F8	51	2439$^-$	**35**$^+$	1647$^-$	1240$^-$	**69**	2439$^-$	72	1674$^-$	1259$^-$
F9	49	584$^-$	**36**	1762$^-$	1430$^-$	**53**	584$^-$	71	1755$^-$	1556$^-$
F10	67	710$^-$	**59**	1937$^-$	1536$^-$	**67**	710$^-$	90	1856$^-$	1738$^-$
F11	69	573$^-$	**59**	2095$^-$	1239$^-$	**72**	573$^-$	87	1924$^-$	1257$^-$
F12	88	678$^-$	**62**	1558$^-$	1118$^-$	**91**	678$^-$	104	1387$^-$	1078$^-$
F13	69	396$^-$	**36**$^+$	1697$^-$	1235$^-$	69	396$^-$	**61**	1530$^-$	1259$^-$
F14	109	867$^-$	**106**	1571$^-$	1134$^-$	**117**	867$^-$	136	1745$^-$	1204$^-$
F15	54	672$^-$	**48**	1478$^-$	1228$^-$	**50**	672$^-$	74$^-$	1390$^-$	1255$^-$
F16	**27**	1081$^-$	32	1195$^-$	1050$^-$	**39**	1081$^-$	58	1602$^-$	1072$^-$
F17	9	398$^-$	**4**$^+$	137$^-$	65$^-$	8	398$^-$	**6**	192$^-$	67$^-$
F18	**36**	821$^-$	37	1782$^-$	1282$^-$	**40**	821$^-$	62$^-$	2293$^-$	1406$^-$
F19	8	706$^-$	**1**$^+$	135$^-$	57$^-$	9	706$^-$	**3**$^+$	199$^-$	58$^-$
F20	94	879$^-$	**41**$^+$	1755$^-$	1084$^-$	120	879$^-$	**66**$^+$	2280$^-$	1063$^-$
F21	58	535$^-$	**45**	1645$^-$	1436$^-$	**66**	535$^-$	66	2083$^-$	1512$^-$
F22	8	670$^-$	**5**$^+$	184$^-$	133$^-$	**8**	670$^-$	9	222$^-$	136$^-$
F23	5	365$^-$	**2**$^+$	68$^-$	31$^-$	3	365$^-$	**2**	81$^-$	31$^-$
F24	4	367$^-$	**1**$^+$	53$^-$	30$^-$	3	367$^-$	**2**	103$^-$	28$^-$
F25	4	395$^-$	**1**$^+$	53$^-$	27$^-$	4	395$^-$	**2**	78$^-$	25$^-$

Since the solution's size of the tested methods on the noisy data is similar to the result on the noiseless data, it is shown in the supplement. Alternately, Fig. 1 presents the testing error and the population size on four typical problems over the evolutionary process. [6] It can be seen that TS-RDO often achieved the lowest testing error over the whole evolution process. TS-RDO quickly achieved a good testing error and often kept improving this value. On one problem, F25, the testing error of

[6] The figures for other problems are presented in the supplement of the paper.

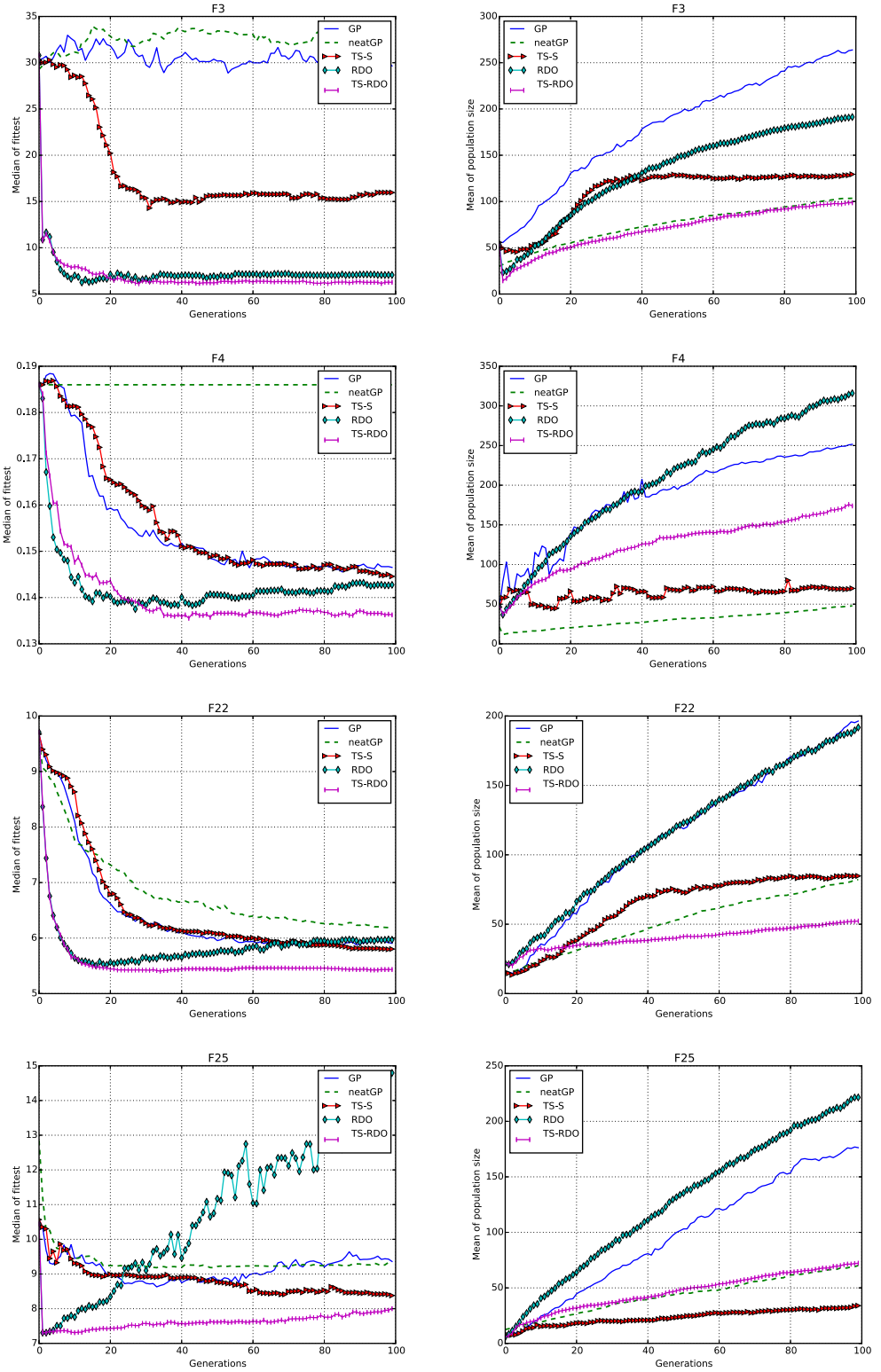

Fig. 1. Testing error and Population size over the generations with tour-size=3.

Table 11
Average execution time of a run (shorted as Run) and average execution time of selection step (shorted as Tour) of GP and TS-S in seconds on noisy data with tour-size=3.

Pro	GP		TS-S		Pro	GP		TS-S	
	Run	Tour	Run	Tour		Run	Tour	Run	Tour
F1	4	0.02	1	0.3	F14	109	0.01	106	14.8
F2	3	0.02	1	0.3	F15	54	0.02	48	16.5
F3	4	0.02	1	0.2	F16	27	0.01	32	10.7
F4	12	0.02	6	0.9	F17	9	0.01	4	1.3
F5	20	0.01	18	5.4	F18	36	0.01	37	14.1
F6	3	0.01	1	0.2	F19	8	0.01	1	0.9
F7	4	0.01	2	0.5	F20	94	0.01	41	10.2
F8	51	0.02	35	17.0	F21	58	0.01	45	14.3
F9	49	0.02	36	17.2	F22	8	0.01	5	2.0
F10	67	0.01	59	16.2	F23	5	0.02	2	0.6
F11	69	0.01	59	15.8	F24	4	0.02	1	0.6
F12	88	0.02	62	16.1	F25	4	0.02	1	0.6
F13	69	0.02	36	16.8					

Table 12
Median of testing error (the left) and average running time (the right) in seconds on noisy data with tour-size=3 of GP, TS-S and TS-S-100. Bold indicates the best (lowest) value. The result is marked + if it significantly smaller than GP. Conversely, it is marked−.

Pro	F.cases	GP	TS-S	TS-S-100	GP	TS-S	TS-S-100
F5	500	0.14	0.14^{+}	**0.14**	20.40	17.67	**10.41**$^{+}$
F8	1000	66.95	66.84^{+}	**66.81**$^{+}$	51.42	35.03	**9.11**$^{+}$
F9	1000	5.52	5.34^{+}	**5.33**	48.53	36.19^{+}	**11.09**$^{+}$
F10	1000	63.89	56.21^{+}	**55.04**$^{+}$	66.99	59.42^{+}	**31.00**$^{+}$
F11	1000	0.2	**0.20**$^{+}$	0.2	68.84	58.67^{+}	**32.63**$^{+}$
F12	1000	7.4	7.31^{+}	**7.25**$^{+}$	87.78	61.51^{+}	**26.29**$^{+}$
F13	1000	0.9	0.90^{+}	**0.90**$^{+}$	69.03	35.92^{+}	**14.90**$^{+}$
F14	1000	122.86	122.67	**122.59**	108.65	106.48	**69.57**$^{+}$
F15	1000	5.01	5.07^{-}	**4.13**	54.01	48.16	**18.70**$^{+}$
F16	800	**36.44**	37.20^{-}	55.82^{-}	27.33	32.30	**18.02**$^{+}$
F18	1000	48.33	**46.60**$^{+}$	47.49	36.02	36.75	**16.57**$^{+}$
F20	750	9.22	9.13	**8.80**$^{+}$	93.63	41.50	**23.61**$^{+}$
F21	1000	**4.53**	4.54^{-}	4.67^{-}	58.21	45.36	**17.06**$^{+}$
F22	160	5.91	5.80^{+}	**5.80**	7.63	4.89	**3.34**$^{+}$

TS-RDO slightly goes up at the last generations. However, it does not go as high as that of RDO. The second best technique is often RDO. However, this crossover is over-fitted after few generations particularly on the UCI problems like F22 and F25. The figure also shows that the testing error of GP and neatGP are usually much higher than others. Last, TS-S is the technique that has less over-fitted impact compared to others. This selection is only the technique that the trend of the testing error is mostly downward.

In terms of the growth of the population size, three methods including neatGP, TS-S and TS-RDO do not incur much code growth in GP population. The population size of these techniques is only slightly increased during the course of the evolution. Conversely, the population size of GP and RDO is quickly grown and it is much higher than that of neatGP, TS-S and TS-RDO.

The last experimental result analysed in this paper is the average running time of the five methods. The average running time over 100 runs is shown in Table 10. Apparently, TS-S is often the fastest system among all tested methods especially with tour-size=3. This is not surprising since, the code growth of TS-S's population is much less than GP (Fig. 1). All other techniques need a longer running time compared to GP. For neatGP, since we used its implementation in Python, the computational time is larger than that of GP and others. RDO also requires a longer time to run compared to GP and this is consistent with the result in previous research [32]. Finally, although TS-RDO is slower than GP, its execution time has been considerably reduced compared to RDO. Thus, by combining TS-S with RDO, we achieved better testing error compared to all tested methods. Moreover, this technique also helps to reduce GP code growth and consequently lessens the computational expensive of RDO.

Table 11 presents the average execution time of the selection step in GP, TS-S and the average execution time of a run. It can be seen that the selection step in TS-S is slower than that of GP. However, this overhead is mostly acceptable. Moreover, since TS-S helps to reduce the code growth of GP population, the overall computational time of a run of TS-S is often smaller than that of GP.

We also observed that the execution time of the selection step in TS-S is often higher on the problems with a large number of fitness cases (F8,F9, etc.) than on the problems with a small number of fitness cases (F1, F1, etc.). On the first problems group, it is possible to further reduce the computational time of TS-S by conducting the statistical test on only a subset of the fitness cases. An extra experiment was conducted to examine this hypothesis by applying the Wilcoxon test on 100 random values of fitness cases on the problems where the number of fitness cases is greater than 100. The results of this experiment (Table 12) show that this technique (TS-S-100) helps to further reduce the running time of TS-S (about 50%) while its testing error is mostly preserved. As a result, the average running time of TS-S-100 is always much smaller than that of GP.

9. Conclusions and future work

In this paper, we introduced the idea of using a statistical test as part of an approach to semantic selection, which utilizes the error vectors of GP individuals. We proposed three variations of tournament selection that employed statistical analysis of these semantic vectors to select the winner for the mating pool. The proposed techniques aim at enhancing the semantic diversity and reducing the code bloat in GP population. The effectiveness of the approach was examined on a large number of symbolic regression problems including GP benchmark problems and additional problem instances drawn from UCI dataset. In the experimental results we observed that, the proposed techniques especially TS-S was better than standard tournament selection and neatGP (the state of the art method for controlling GP code bloat) in improving GP generalisation and reducing GP code growth.

One of the advantages of the proposed method is its simplicity in design and implementation. This allows it to be further improved by combining it with advanced techniques in GP. In this paper, the best approach to semantic tournament selection, TS-S, was combined with the recently proposed semantic crossover, RDO. The resulting method, TS-RDO, achieved better testing error and reduced code growth to a greater extent. In addition, noisy instances of each problem were generated and performance of the various strategies examined demonstrating that the proposed methods have a good ability to perform well on noisy problems.

There are a number of research areas for future work. First, the experimental results showed that the performance of the proposed methods are robust with various values of the tournament size. However, it would be better if the tournament size could be self-adapted in the learning process [16]. Secondly, statistical analysis was used in this paper only to enhance selection. It is also possible that statistical analysis can be employed in other phases of the GP algorithm, for example, in model selection [49]. In terms of applications, TS-S and TS-RDO can be applied to any problem domain where the output is a single real-valued number. In this paper, we focused exclusively on GP's most popular problem domain, symbolic regression, in the future we will extend semantic selection to popular problem domains such as classification and program synthesis.

Acknowledgement

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2014.09. MON acknowledges the support of Science Foundation Ireland grants 13/IA/1850 and 13/RC/2094.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.ins.2018.01.030.

References

[1] E. Alpaydin, Introduction to Machine Learning, 3, MIT Press, 2014.
[2] K. Bache, M. Lichman, UCI machine learning repository, 2013, http://archive.ics.uci.edu/ml.
[3] T. Bäck, Selective pressure in evolutionary algorithms: acharacterization of selection mechanisms, in: Evolutionary Computation, IEEE Press, 1994, pp. 57–62.
[4] T. Blickle, L. Thiele, A comparison of selection schemes used in evolutionary algorithms, Evol. Comput. 4 (4) (1996) 361–394.
[5] M. Castelli, L. Manzoni, S. Silva, L. Vanneschi, A. Popovic, The influence of population size in geometric semantic gp, Swarm Evol. Comput. 32 (2017) 110–120.
[6] T.H. Chu, Q.U. Nguyen, M. ONeill, Tournament selection based on statistical test in genetic programming, in: International Conference on Parallel Problem Solving from Nature, Springer, 2016, pp. 303–312.
[7] G. Cumming, Understanding The New Statistics: Effect Sizes, Confidence Intervals, and Meta-Analysis, Routledge, 2012.
[8] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (1) (2011).
[9] T. Dou, P. Rockett, Semantic-based local search in multiobjective genetic programming, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion, ACM, 2017, pp. 225–226.
[10] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing, Springer, 2003.
[11] Y. Fang, J. Li, A review of tournament selection in genetic programming, in: Proceedings of the International Conferences on Advances in Computation and Intelligence, ISICA 2010, Springer, 2010, pp. 181–192.
[12] S. Forstenlechner, M. Nicolau, D. Fagan, M. O'Neill, Introducing semantic-clustering selection in grammatical evolution, in: GECCO 2015 Semantic Methods in Genetic Programming (SMGP'15) Workshop, ACM, 2015, pp. 1277–1284.
[13] E. Galvan-Lopez, B. Cody-Kenny, L. Trujillo, A. Kattan, Using semantics in the selection mechanism in genetic programming: a simple method for promoting semantic diversity, in: L.G. de la Fraga (Ed.), 2013 IEEE Conference on Evolutionary Computation, 1, 2013, pp. 2972–2979.

[14] C. Gathercole, An investigation of supervised learning in genetic programming, 1998 Ph.D. thesis.
[15] D.E. Goldberg, K. Deb, A comparative analysis of selection schemes used in genetic algorithms, Found.Genet.Algorithms 1 (1991) 69–93.
[16] T. Helmuth, N.F. McPhee, L. Spector, Effects of lexicase and tournament selection on diversity recovery and maintenance, in: Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion (GECCO), ACM, 2016, pp. 983–990.
[17] T. Helmuth, L. Spector, J. Matheson, Solving uncompromising problems with lexicase selection, IEEE Trans. Evol. Comput. 19 (5) (2015) 630–643.
[18] K. Hingee, M. Hutter, Equivalence of probabilistic tournament and polynomial ranking selection, in: 2008 IEEE Congress on Evolutionary Computation, 2008, pp. 564–571.
[19] B.A. Julstrom, D.H. Robinson, Simulating exponential normalization with weighted k-tournaments, in: Evolutionary Computation, 1, 2000, pp. 227–231.
[20] G.K. Kanji, 100 Statistical Tests, SAGE Publications, 1999.
[21] A. Kattan, A. Agapitos, Y.-S. Ong, A.A. Alghamedi, M. O'Neill, Gp made faster with semantic surrogate modelling, Inf. Sci. 355–356 (2016) 169–185.
[22] A. Kattan, Y.-S. Ong, Surrogate genetic programming: a semantic aware evolutionary search, Inf. Sci. 296 (2015) 345–359.
[23] J.-J. Kim, B.-T. Zhang, Effects of selection schemes in genetic programming for time series prediction, in: Proceedings of the Congress on Evolutionary Computation, 1, 1999, pp. 252–258.
[24] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, 1, The MIT Press, 1992.
[25] W. La Cava, L. Spector, K. Danai, Epsilon-lexicase selection for regression, in: Proceedings of the 2016 on Genetic and Evolutionary Computation Conference (GECCO), ACM, 2016, pp. 741–748.
[26] T. Leonardo, M. Luis, G.-L. Edgar, S. Sara, Neat genetic programming: controlling bloat naturally, Inf. Sci. 333 (2016) 21–43.
[27] N. McPhee, B. Ohs, T. Hutchison, Semantic building blocks in genetic programming, in: Proceedings of 11th European Conference on Genetic Programming, Springer, 2008, pp. 134–145.
[28] E. Naredo, L. Trujillo, P. Legrand, S. Silva, L. Munoz, Evolving genetic programming classifiers with novelty search, Inf. Sci. 369 (2016) 347–367.
[29] S. Needham, D.L. Dowe, Message length as an effective ockham's razor in decision tree induction, in: Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics, 2001, pp. 257–275.
[30] Q.U. Nguyen, X.H. Nguyen, M. O'Neill, R.I. McKay, N.P. Dao, On the roles of semantic locality of crossover in genetic programming, Inf. Sci. 235 (2013) 195–213.
[31] Q.U. Nguyen, X.H. Nguyen, M. O'Neill, R.I. McKay, E. Galvan-Lopez, Semantically-based crossover in genetic programming: application to real-valued symbolic regression, Genet. Program. Evolvable Mach. 12 (2) (2011) 91–119.
[32] Q.U. Nguyen, T.A. Pham, X.H. Nguyen, J. McDermott, Subtree semantic geometric crossover for genetic programming, Genet. Program. Evolvable Mach. 17 (1) (2016) 25–53.
[33] K. Oksanen, T. Hu, Lexicase selection promotes effective search and behavioural diversity of solutions in linear genetic programming, in: 2017 IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 169–176.
[34] T.P. Pawlak, K. Krawiec, Progress properties and fitness bounds for geometric semantic search operators, Genet. Program. Evolvable Mach. 17 (1) (2016) 5–23.
[35] T.P. Pawlak, B. Wieloch, K. Krawiec, Semantic backpropagation for designing search operators in genetic programming, IEEE Trans. Evol. Comput. 19 (3) (2015) 326–340.
[36] T.P. Pawlak, B. Wieloch, K. Krawiec, Semantic backpropagation for designing search operators in genetic programming, IEEE Trans. Evol. Comput. 19 (3) (2015) 326–340.
[37] R. Poli, W.B. Langdon, N.F. McPhee, A Field Guide to Genetic Programming, Lulu Enterprises, UK Ltd, 2008. Available at http://www.gp-field-guide.org.uk.
[38] D.L. Rumpf, Statistics for dummies, Technometrics 46 (3) (2004).
[39] J.A. Sáez, M. Galar, J. Luengo, F. Herrera, Tackling the problem of classification with noisy data using multiple classifier systems: analysis of the performance and robustness, Inf. Sci. (Ny) 247 (2013) 1–20.
[40] J.A. Sáez, M. Galar, J. Luengo, F. Herrera, Analyzing the presence of noise in multi-class problems: alleviating its influence with the one-vs-one decomposition, Knowl. Inf. Syst. 38 (1) (2014) 179–206.
[41] A. Sokolov, D. Whitley, Unbiased tournament selection, in: Proceedings of the 7th annual conference on Genetic and evolutionary computation, ACM, 2005, pp. 1131–1138.
[42] L. Trujillo, G. Olague, E. Lutton, F.F. de Vega, L. Dozal, E. Clemente, Speciation in behavioral space for evolutionary robotics, J. Intell. Rob. Syst. 64 (3–4) (2011) 323–351.
[43] D.R. White, J. McDermott, M. Castelli, L. Manzoni, B.W. Goldman, G. Kronberger, W. Jaskowski, U.-M. O'Reilly, S. Luke, Better GP benchmarks: community survey results and proposals, Genet. Program. Evolvable Mach. 14 (1) (2013) 3–29.
[44] H. Xie, Diversity control in gp with adf for regression tasks, in: Australasian Joint Conference on Artificial Intelligence, Springer, 2005, pp. 1253–1257.
[45] H. Xie, M. Zhang, Impacts of sampling strategies in tournament selection for genetic programming, Soft Comput. 16 (4) (2012) 615–633.
[46] H. Xie, M. Zhang, Parent selection pressure auto-tuning for tournament selection in genetic programming, IEEE Trans. Evol. Comput. 17 (1) (2013) 1–19.
[47] H. Xie, M. Zhang, P. Andreae, M. Johnson, An analysis of multi-sampled issue and no-replacement tournament selection, in: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, ACM, 2008, pp. 1323–1330.
[48] H. Xie, M. Zhang, P. Andreae, M. Johnston, Is the not-sampled issue in tournament selection critical? in: 2008 IEEE Congress on Evolutionary Computation, 2008, pp. 3710–3717.
[49] J. Zegklitz, P. Posik, Model selection and overfitting in genetic programming: empirical study, in: Proceedings of the Companion Publication of the 2015 on Genetic and Evolutionary Computation Conference (GECCO), ACM, 2015, pp. 1527–1528.