Predicting the Tide with Genetic Programming and Semantic-based Crossovers

Nguyen Quang Uy
Natural Computing Research and Applications Group
University College Dublin
Ireland, quanguyhn@gmail.com

Michael O'Neill
Natural Computing Research and Applications Group
University College Dublin
Ireland, m.oneill@ucd.ie

Nguyen Xuan Hoai
Faculty of Information Technology
Le Quy Don University,
Vietnam, nxhoai@gmail.com

***Abstract*—This paper proposes an improvement of a recently proposed semantic-based crossover, Semantic Similarity-based Crossover (SSC)[27]. The new crossover, called the Most Semantic Similarity-based Crossover (MSSC), is tested with Genetic Programming (GP) on a real world problem, as in predicting the tide in Venice Lagoon, Italy. The results are compared with GP using Standard Crossover (SC) and GP using validation sets. The comparative results show that while using validation sets give only limited effect, using semantic-based crossovers, especially MSSC, remarkably improve the ability of GP to predict time series for the tested problem. Further analysis on GP code bloat helps to explain the reason behind this superiority of MSSC.**

***Keywords*-Genetic Programming; Semantics; Crossover; Time Series;**

I. Introduction

In the field of Genetic Programming (GP) [22], [17], researchers have recently paid more attention to semantic information, with a dramatic increase in the number of publications (e.g., [11], [13], [14], [16], [15], [2], [20], [26], [27], [5]). Previously, most research has purely focused on syntactic aspects of GP representation. From a programmer's perspective, however, maintaining syntactic correctness is only one part of program construction: maintaining program semantic correctness is much more desirable. Thus incorporating semantic awareness in the GP evolutionary process could improve its performance, extending the applicability of GP to problems that are difficult with purely syntactic approaches.

In the realm of GP real world application, time series prediction has been considered as a major target [9], [25], [24], [19]. The previous work on GP application to time series prediction problems has forcused on financial data forecasting [9], [25], [19] or using artificial Mackey time series [24]. Their approach is to build up forecasting models by combining different variables that can represent the knowledge from time series data. To our best knowledge, incorporating semantics into GP systems has never been used in time series prediction. In this paper, we use a new semantics based crossovers to solve the problem of predicting the tide in Venice Lagood, Italy. This problem has been seen in [1] as a truely difficult time series to predict. The remainder of the paper is organised as follows. In the next section we review the literature on GP with semantics. The semantics similarity-based crossover (SSC), the new semantic-based crossover (MSSC) and the model to predict time series are presented in Section III. It is followed by a section detailing our experimental settings. The experimental results are shown and discussed in Section V. The last section concludes the paper and highlights some future work.

II. Related Work

Incorporating semantics into GP has recently been considered by a number of GP researchers. This results in a dramatic increase in the number of related publications. Generally, the work falls into three main strands:

1) using formal methods [11], [13], [14], [16], [15]
2) using grammars [28], [5], [6]
3) using structures such as GP trees [2], [20], [26], [27]

The first approach in a series of work [11], [13], [14] was advocated by Johnson. In this work, Abstract Interpretation and Model Checking are used to calculate/extract semantics. Then, semantic information is used to measure the fitness of individuals where it is unable to use traditional sample point fitness. Consequently, Katz and Peled used model checking to solve the Mutual Exclusion problem [16], [15]. Again, individuals' fitness is quantified through model checking. These formal methods have a strict mathematical foundation, that potentially may aid GP. Perhaps because of high complexity, however, these methods have seen only limited research despite the advocacy of Johnson [12]. Their main application to date has laid in evolving control strategies.

The second methodology uses Attribute Grammars as the main formalism. By adding attributes to a grammar, some useful semantic information about individuals can be generated. This information can then be used to delete bad individuals [6], or to prevent generating semantically invalid ones [28], [5]. The attributes used to represent semantics are, however, problem dependent, and it is not always easy to design such attributes for a new problem.

In the last category, controlling the GP operators is the major theme. In [2], the authors investigated the effect of semantic diversity on Boolean domains, checking the semantic equivalence between offspring and parents by transformating them to a canonical form, Reduced Ordered Binary Decision Diagrams (ROBDDs) [7]. This information is used to determine which offsprings are copied to the next generation. The method improved GP performance, presumably because it increased semantic

978-0-7695-4213-3/10 $25.00 © 2010 IEEE
DOI 10.1109/KSE.2010.7

diversity. The method has also been applied to mutation [4] and to the initialisation phase of GP [3].

While, most of previous research on semantics in GP were focused on combinatorial and boolean problems [5], [2], [20], [16], research on real-valued domains [26], [27], [18] is only recently considered. Krawiec and Lichocki [18] based the semantics of individuals on fitness cases, using it to guide the crossover operator (*Approximating Geometric Crossover* - AGC). AGC turned out to be not significantly better than standard crossover (SC) on their tested real-valued problems, and only slightly better on Boolean domains.

Uy et al. [26] proposed Semantics Aware Crossover (SAC) with an aim to promote semantic diversity. SAC is based on checking semantic equivalence of subtrees. It showed limited improvement on some real-valued problems; SAC was subsequently extended to Semantic Similarity based Crossover (SSC) [27], which performed better than both SC and SAC on tested real-valued regression problems [27]. However, in [27], SSC was only tested on some toy symbolic regression problems. Moreover, the performance of SSC could be dependent on some predetermined parameters (e.g. the semantic sensitivities). The objective of the work in this paper is to overcome this weakness by proposing an improved version of SSC resulting in a new crossover operator that will be called the Most Semantic Similarity-based Crossover (MSSC) in the sequel. More importantly, we test these crossovers, in comparison with standard crossover and validation sets method, on a real-world and hard problem.

III. Methods

This section briefly describes our Semantic Similarity based Crossover (SSC). Then, the new improvement of SSC, the Most Semantic Similarity based Crossover (MSSC), is detailed. Next, we present the time series prediction model used in this paper.

A. Semantic Similarity-based Crossover

Semantic Similarity based Crossover (SSC) [27] is inspired and extended from earlier research on Semantics Aware Crossover (SAC) [26]. SSC used in this paper is almost identical to that described in Uy et al. [27] with a slightly modified semantic distance measure. Since SSC operates on the semantics of subtrees, first a definition of subtree semantics is needed. Formally, the *Sampling Semantics* of any (sub)tree is defined as follows:

Let F be a function expressed by a (sub)tree T on a domain D. Let P be a set of points sampled from domain D, $P = \{p_1, p_2, ..., p_N\}$. Then the *Sampling Semantics* of T on P on domain D is the set $S = \{s_1, s_2, ..., s_N\}$ where $s_i = F(p_i), i = 1, 2, ..., N$.

The value of N depends on the problems. If it is too small, the approximate semantics might be too coarse-grained and not sufficiently accurate. If N is too big, the approximate semantics might be more accurate, but more time-consuming to measure. The choice of P is also crucial. If the members of P are too closely related to the GP function set (for example, π for trigonometric functions, or e for logarithmic functions), then the semantics might be misleading. In this paper, the number of points for evaluating *Sampling Semantics* is set as the number of fitness cases of the problem, and we choose the same set of points P as the fitness cases of the problem.

Based on *Sampling Semantics* (SS), we define a *Sampling Semantics Distance* between two subtrees. In the previous work [27], *Sampling Semantics Distance* (SSD) was defined as the sum of absolute difference of all values of SS. While the experiments showed that this kind of SSD is acceptable, it has a major weakness that the value of SSD is strongly dependent on the number of SS points (N) [27]. To remedy this drawback, in this paper, we use the mean of absolute distance as the SSD between subtrees. In other words, let $U = \{u_1, u_2, ..., u_N\}$ and $V = \{v_1, v_2, ..., v_N\}$ be the SS of $Subtree_1(St_1)$ and $Subtree_2(St_2)$ on the same set of evaluating values, then the SSD between St_1 and St_2 is defined as follows:

$$SSD(St_1, St_2) = \frac{|u_1 - v_1| + |u_2 - v_2| + ... + |u_N - v_N|}{N} \quad (1)$$

Thanks to SSD, a relationship known as *Semantic Similarity* is defined. The intuition behind semantic similarity is that exchange of subtrees is most likely to be beneficial if the two subtrees are not semantically identical, but also they are not too semantically dissimilar. Two subtrees are semantically similar on a domain if their SSD on the same set of points in that domain lies within a positive interval. The formal definition of semantic similarity (SSi) between subtrees St_1 and St_2 is as follows:

$$SSi(St_1, St_2) = \begin{array}{l} \textbf{if } \alpha < \mathrm{SSD}(St_1, St_2) < \beta \\ \quad \textbf{then } \text{true} \\ \quad \textbf{else } \text{false} \end{array}$$

here α and β are two predefined constants, known as the *lower* and *upper bounds* for semantic sensitivity, respectively. Conceivably, the best values for *lower* and *upper bound semantic sensitivity* might be problem dependent. However we strongly suspect that for almost any symbolic regression problem, there is a range of values that is appropriate [27]. The investigation of the effect of different semantic sensitivities on SSC performance is beyond the scope of this paper. In this paper, we set $\alpha = 10^{-3}$ and $\beta = 0.4$ which are good values found in the literature [27].

Inspired from the difficulty in designing an operator with high locality in GP, SSC was proposed with the main objective being to improve the locality of crossover. SSC is in fact an extension of SAC in two ways. Firstly, when two subtrees are selected for crossover, their semantic similarity, rather than semantic equivalence as in SAC, is checked. Secondly, semantic similarity is more difficult to satisfy than semantic equivalence, so repeated failures may occur. Thus SSC uses multiple trials to find a semantically

Algorithm 1: Semantic Similarity based Crossover

```
select Parent 1 P1;
select Parent 2 P2;
Count=0;
while Count<Max_Trial do
    choose a random crossover point Subtree1 in P1;
    choose a random crossover point Subtree2 in P2;
    generate a number of random points (P) on the
    problem domain;
    calculate the SSD between Subtree1 and Subtree2
    on P
    if Subtree1 is similar to Subtree2 then
        execute crossover;
        add the children to the new population;
        return true;
    else
        Count=Count+1;
if Count=Max_Trial then
    choose a random crossover point Subtree1 in P1;
    choose a random crossover point Subtree2 in P2;
    execute crossover;
    return true;
```

similar pairs, only reverting to random selection after passing a bound on the number of trials. Algorithm 1 shows how SSC operates in detail. In our experiments, the value of Max_Trial was set to 12, with this value having been calibrated by earlier experimental results.

B. The Most Semantic Similarity-based Crossover

MSSC further exploits the main idea of SSC. The purpose of MSSC is to avoid the manual determination of the semantic sensitivities in SSC, but still keep a semantic small change in child individual(s) after crossover. MSSC works as follows. Firstly, N subtree pairs are randomly selected from the two parents. The Sampling Semantic distance (SSD) of two subtrees in each pair is calculated. The pairs that have the smallest (most similar but not equivalent) semantic distance of two subtrees in N pairs is chosen for crossover. In MSSC, the concept of semantic equivalence is the same as in SAC. Algorithm 2 shows how MSSC works in detail. In the experiments of MSSC, the value of Max_Trial (TM) is again set as in SSC, and Extremal_Value is set to 10^6.

C. Time Series Prediction Model

The task of time series prediction is to estimate the value of the series in the future based on its values in the past. There are two models of time series prediction: one-step prediction and multi-step prediction. In one-step prediction, the task is to express the value of $x(t+1)$ as a function of previous of the N values of the time series, $x(t),...,x(t-N+1)$. That is finding the function F so that

$$x(t+1) = F(x(t),x(t-1),...,x(t-N+1)) \quad (2)$$

Algorithm 2: The Most Semantic Similarity based Crossover

```
select Parent 1 P1;
select Parent 2 P2;
Count=0;
Max=Extremal_Value;
while Count<Max_Trial do
    choose a random crossover point Subtree1 in P1;
    choose a random crossover point Subtree2 in P2;
    SD=SSD(Subtree1, Subtree2)
    if SD is less than Max then
        Max=SD;
        CrossPoint1=Subtree1;
        CrossPoint2=Subtree2;
Execute crossover by exchange the subtrees at
CrossPoint1 and CrossPoint2;
```

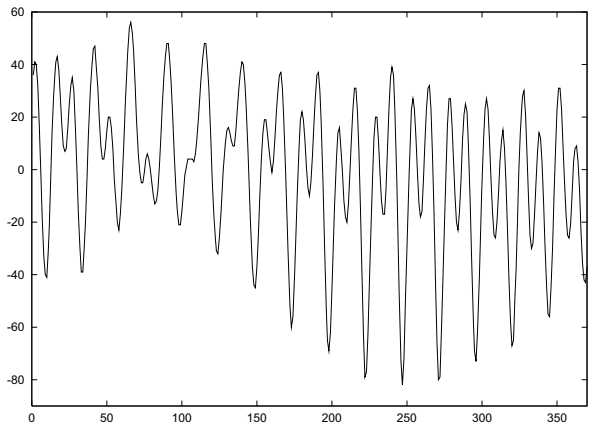

Figure 1. The plot of 370 points of the tide level in Venice Lagoon.

The purpose of multi-step prediction is to obtain predictions of several steps ahead into the future, $x(t+1),x(t+2),x(t+3),...$, starting from the information at current time slice t. In this paper, we only focus on one-step prediction and follows the work in [23] to set N as 8. It means that the main task is to ask GP to find model for $x(t+1)$ based on its 8 previous values: $x(t),x(t-1),...,x(t-7)$.

IV. Experimental Settings

A. Venice Lagoon Time Series Problem

The prediction of high tide has long been the subject of interest to humans. The motivation for such interest is the economic benefits of the prediction/forecasting system. Usually, high tide is the result of a combination of some chaotic climatic elements. Therefore, tide's behaviour is difficult to predict, because it depends on many factors [1]. Two main factors that most affect tide level are the astronomic and atmospheric agents. The problem has been approached by using time series analysis and nonlinear neural networks [10]. In this paper we use GP equipped with semantic-based crossovers to solve the problem.

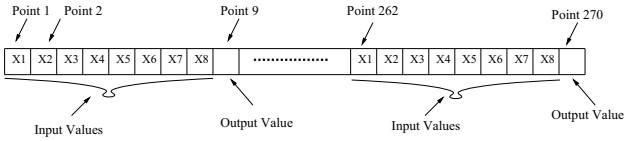

Figure 2. The plot of 270 points that are combined into 30 fitness cases.

The time series data to predict in this paper is high tide in Venice Lagoon, Italy. The data is measured in centimetres each hour along the years 1990 to 1995 [1]. We took 370 points (from position of 11 to position of 380) from this data set. 10 first values were discarded to ensure that the measurement system will be the stable state. This data is plotted in Figure 1. We divided these data into two sets, one for training and the other for testing. The first 270 values are used for the training set. They are combined into 30 fitness cases. Figure 2 shows 270 values of 30 these fitness cases. Similarly, 30 fitness cases were used for the testing set. We divided the testing set into 3 sets with the number of points in the future is 10^k $(k = 0,1,2)$. These 3 sets will be referred as Tk with $k = 0,1,2$ in the following text.

B. GP Parameters Settings

The GP parameters used for our experiments are shown in Table I. In this paper, 3 population sizes of 250, 500 and 1000, were tested. Correspondingly, three values of generation: 100, 50, 25 respectively, were used. These values fixed the number of fitness evaluations as 25000. The terminal set includes 8 variables, X_1, X_2, ..., X_8, and a constant 1. These eight variables present 8 values of time series in the past. Despite this being an experiment purely concerned with the prediction ability of crossover, we have retained mutation with a small rate in the system because the aim of the experiment is to study crossover in the context of a normal GP run. Our experiments were conducted on six configurations as follows:

1) Standard Crossover (SC): The fitness is measured as the error rate on the whole training set. The best-of-run individual is the individual with the lowest error rate on the training set in entire evolutionary time. This individual was then tested on the testing data set to give the result for solution prediction capacity of the run.
2) Standard Crossover with Validation (SCV): The training set is randomly divided into 2 (for each run): 67% is used for training (training set) and the remaining 33% is used for validating (validation set). At each generation the fitness of individuals is measured on the training set and this fitness is used for tournament selection. A two-objective trial (fitness and size of an individual) is conducted in order to extract a set of non-dominated individuals (the Pareto front). The individuals in the Pareto front are then evaluated on the validation set, with the best of run individual selected as the one of these with the smallest error rate on the validation set. This configuration is similar to the validation configuration in [8].
3) Semantic Similarity based Crossover (SSC): This configuration is similar to configuration 1 with only the difference is that SSC rather than SC is used.
4) Semantic Similarity based Crossover with Validation (SSCV): This configuration is similar to configuration 2 but with SSC replaces SC.
5) The Most Semantic Similarity based Crossover (MSSC): This configuration is similar to configuration 1 but MSSC is used instead of SC.
6) The Most Semantic Similarity based Crossover with Validation (MSSCV): This configuration is similar to connfiguration 2 but with MSSC rather than SC.

Table I
RUN AND EVOLUTIONARY PARAMETER VALUES.

Parameters	Value
Population size	250, 500, 1000
Generation	100, 50, 25
Selection	Tournament
Tournament size	3
Crossover probability	0.9
Mutation probability	0.05
Initial Max depth	6
Max depth	15
Max depth of mutation tree	15
Non-terminals	+, -, *, / (pro. one), sin, cos, exp, log (pro. one)
Terminals	X_1, X_2, ..., X_8, 1
Training set	30 fitness cases
Testing set	30 fitness cases
Raw fitness	mean of absolute error on all fitness cases
Trials per treatment	100 independent runs for each value

V. RESULTS AND DISCUSSION

This section presents the effect of semantic-based crossovers, in comparison with standard crossover and with validation set method, on GP performance, GP ability to predict, and GP code bloat.

A. On the GP Performance

To compare the three crossovers, we recorded classic performance metrics, the mean and the standard deviation of the best fitness. The results are shown in Table II. It can be seen from this table that both semantic-based crossovers helped to improve the performance of GP. It is reflected by the values of the mean best fitness found by SSC and MSSC were often smaller than one found by SC. The exception only lies in one case of SSC with population size of 250. It should, however, be noted from this table that while the improvement of SSC versus SC seemed not remarkable, the size of the improvement of MSSC over SC was much bigger.

Table II
MEAN AND STANDARD DEVIATION OF THE AVERAGE OF BEST FITNESS ON THE TRAINING SET (SMALLER VALUE IS BETTER)

Methods	Pop250		Pop500		Pop1000	
	Mean	Std	Mean	Std	Mean	Std
SC	4.76	1.58	4.93	1.50	5.39	1.22
SSC	5.01	1.61	4.76	1.22	4.99	1.15
MSSC	**4.02**	1.43	**4.00**	1.11	**4.33**	1.02
SCV	4.41	1.70	4.21	1.13	4.73	1.08
SSCV	4.31	1.79	4.16	1.18	4.72	0.99
MSSCV	**3.26**	1.39	**3.33**	1.19	**4.06**	1.03

Table III
THE NUMBER OF GOOD SOLUTIONS (GREATER VALUE IS BETTER)

Methods	Pop250			Pop500			Pop1000		
	T1	T2	T3	T1	T2	T3	T1	T2	T3
SC	12	12	13	24	21	20	27	26	22
SSC	5	5	4	25	22	23	33	32	27
MSSC	6	5	5	26	24	21	**48**	**47**	**45**
SCV	15	14	11	22	20	15	26	25	21
SSCV	16	14	13	16	16	16	27	28	23
MSSCV	20	18	16	29	26	25	**48**	**47**	**45**

The table also reveals that by using validation sets the mean best fitness was smaller than the case when they were not used. The reason, perhaps, is that GP was trained on a smaller set of data (20 fitness cases when using validation sets and 30 fitness cases in the other case). It is noteworthy that, even in this case, the semantic-based crossovers were still better than SC. Moreover, MSSC was still the best out of the three crossovers.

We also conducted a test of statistical significance of the improvement of SSC and MSSC in Table II over SC and of SSCV, MSSCV over SCV using Wilcoxon signed-rank test with the confidence level of 99%. In this table, if the improvements of SSC, MSSC over SC and of SSCV, MSSCV over SCV are statistically significant, the results are printed in **bold** face. The results of this test confirm the significant improvement of MSSC over SC and of MSSCV over SCV, while it reveals that the improvement of SSC over SC and SSCV over SCV is not statistically significant. We further conducted a test of statistical significant of the improvement of MSSC over SSC and MSSCV over SSCV with the confidence level of 99%. The result is (although not highlighted in this table) that MSSC and MSSCV are significant better than SSC and SSCV, respectively, in terms of the average of the best fitness. These results give clear evidence that MSSC not only helps to overcome the limitation of SSC but also helps to significantly improve GP performance, at least on solving this real world problem.

B. On the Ability to Predict

To examine and compare the ability of these methods to predict, we use a new performance metric to measure the quality of the solution of a run. For each run, the best individual (based on its fitness on the training data sets or on validation sets) was selected as the final solution of the run. This solution is then tested on the testing data sets. We define $\varepsilon = 5$ as a constant to determine the quality of a solution. For a solution with fitness ft on the testing sets, we define it as a good solution if the fitness on this set is less than ε. We counted the number of good solutions out of 100 runs and the results are shown on Table III.

It can be seen from this table that the prediction ability of the GP systems (both with and without using semantic-based crossovers) depends on the GP parameters settings. When using a small population size and a bigger number of generations, the prediction ability of GP is poorer than when using a large population with a smaller number of generations. The table also shows that using semantic-based crossovers could not help to improve the predictability of GP running with a small population and big number of generations. The number of good solutions of SSC and MSSC on population size of 250 are smaller than ones of SC. The reason behind this is that when the GP systems run for a big number of generations, they could be over-fitting on training data (Table II shows that the average of the best fitness of the GP systems run with a big number of generations is less than ones run with a small number of generations) and it was even worse when they are equipped with semantic based crossovers.

However, what is more important is that semantic-based crossovers helped to enhance the ability of GP to predict when overfitting is not such a serious problem (population size of 500, and population of 1000). The table shows that the number of good solutions found by SSC and MSSC were often slightly greater than ones found by SC on the setting of population size of 500. When the population size is 1000, MSSC helped to remarkably improve the ability of GP to predict. It can be observed that the number of good solutions found by MSSC on this population size setting is substantially greater than ones found by both SC and SSC.

The table also shows that using validation sets only gives a positive impact when the population size is 250. For this setting, the number of good solutions found by using crossovers with validation sets were often greater than ones without using validation sets. For other settings of population size (i.e. 500 and 1000), using validation sets gave very limited impact. The reason might be that using validation sets helps to prevent over-fitting when the GP systems were run for a long time, and when GP population were still under-fitting (population size of 500 and 1000), validation sets method was not essentially effective in improving the ability of GP to predict.

The second metric measures the ability of these methods to predict by recording the median of the best fitness on the testing data sets. The results of the median of the best fitness on testing sets are presented in Table IV. The results in this table are consistent with the ones in Table III. It again confirms the improvement of semantic-based crossovers, especially MSSC, when they were used with a large population and smaller number of generations.

Table IV
THE MEDIAN OF ERROR ON TESTING SETS (SMALLER VALUE IS BETTER)

Methods	Pop250			Pop500			Pop1000		
	T1	T2	T3	T1	T2	T3	T1	T2	T3
SC	7.2	7.4	7.9	6.6	6.8	7.0	5.9	6.1	6.2
SSC	8.5	8.5	8.7	6.5	6.6	6.8	5.7	5.8	6.0
MSSC	9.0	9.2	9.6	6.3	6.3	6.5	**5.0**	**5.1**	**5.2**
SCV	8.4	8.6	9.3	6.6	6.8	7.2	6.1	6.2	6.4
SSCV	8.2	8.5	9.2	6.5	6.7	6.7	6.2	6.4	6.4
MSSCV	7.4	7.6	7.9	6.3	6.4	6.2	**5.2**	**5.3**	**5.3**

Table V
MEAN OF POPULATION SIZE AND MEAN SIZE OF THE BEST FITNESS (THE NUMBER OF NODES, SMALLER VALUE IS BETTER)

Methods	Pop250		Pop500		Pop1000	
	Pop	Best	Pop	Best	Pop	Best
SC	74.4	116	42.8	64.8	27.7	32.8
SSC	71.1	116	43.5	69.5	28.1	36.1
MSSC	93.8	175	43.2	93.4	21.9	36.4
SCV	71.4	34.7	45.7	28.5	28.9	18.1
SSCV	76.2	36.9	44.3	25.4	26.6	16.7
MSSCV	80.2	34.2	45.9	27.3	21.6	14.3

The table also shows the over-fitting phenomenon of a GP run with a big number of generations and a small population and the limited effect of using validation sets method on the ability of GP to predict.

C. On the Code Bloat Effect

Since there is a strong correlation between the complexity of learnt solutions and their ability to generalise (Ockham's razor or Minimum Description Length [21]), statistics on population size and solution size were also recorded and analysed. This includes the average size of a GP population (Pop column) and the average size of the best fitness (on training sets with SC, SSC and MSSC or on validating sets with SCV, SSCV, and MSSCV) (Best column). The results are shown in Table V.

It can be seen from this table that GP bloats remarkably when it is run with a big number of generations (i.e. the population size of 250). Moreover, in these runs, MSSC produced more bloat than SSC and SC. This explains why the predictability of GP system was reduced and the ability to predict of MSSC is worse than SC in this experiment setting. The table also reveals that on the population of 500, three crossovers have almost the same code bloat effect. However, the size of the best fitness individuals found by MSSC is still greater than one of SC. Therefore, the ability of MSSC to predict is only slightly better than SC on this setting. Only when using a larger population (1000) and a smaller number of generations (25), MSSC produced less bloat than SC, even the best individuals' size was still a little greater than of SC. This is one of the main reasons for the significant improvement of the predictability of GP with MSSC over GP with SC on this experiment setting.

Although, the previous subsection shows that using a validation set method gives only a limited impact on improving the ability of GP to predict, using validation sets method helps to substantially reduce the size of the best individuals. This is confirmed by the results in Table V (Best column). The size of the best fitness when used with validation set method was always substantially smaller than without using this method. Therefore, the solutions found by a validation set method is more readable than one without using this method. Perhaps, this is the main advantage of using validation set method in this problem.

In conclusion, this section shows three important results. The first is that using a larger population and small number of generations is better, on the tested problem, than using a smaller population but bigger number generations. Secondly, semantic-based crossovers, especially MSSC, could help to significantly improve the ability to predict when GP is run with small a number of generations and a large population. Last, using validation could help to find less complex solutions, hence, more comprehensible.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a new semantic-based crossover called the Most Semantic Similarity based Crossover (MSSC). MSSC overcomes the limitation of a recently proposed semantic based crossover, Semantic Similarity based Crossover (SSC) by eliminating its dependence on the *upper semantic sensitivity*. We test these semantic-based crossovers on a real-world problem: predicting the tide in Venice Lagoon, Italy. The results are compared with the standard crossover. The validation sets method is also tested. The results shows that using semantic-based crossovers, especially MSSC, with a small number of generations helps to substantially improve the ability of GP to predict the tide. The results also show that using a validation set method helps to find the simpler solutions though it does not help to improve the prediction ability of GP.

In the near future, we are planning to use a validation method to dynamically prevent the over-fitting of semantic-based crossovers, especially MSSC.

ACKNOWLEDGMENT

This paper was funded under a Postgraduate Scholarship from the Irish Research Council for Science Engineering and Technology (IRCSET). The third author was party funded by The Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01.14.09 for doing this work.

REFERENCES

[1] Evolutionary and neural computation for time series prediction minisite. Website, 2005. http://tracer.uc3m.es/tws/TimeSeriesWeb/repo.html.

[2] L. Beadle and C. Johnson. Semantically driven crossover in genetic programming. In *Proceedings of the IEEE World Congress on Computational Intelligence*, pages 111–116. IEEE Press, 2008.

[3] L. Beadle and C. G. Johnson. Semantic analysis of program initialisation in genetic programming. *Genetic Programming and Evolvable Machines*, 10(3):307–337, Sep 2009.

[4] L. Beadle and C. G. Johnson. Semantically driven mutation in genetic programming. In A. Tyrrell, editor, *2009 IEEE Congress on Evolutionary Computation*, pages 1336–1342, Trondheim, Norway, 18-21 May 2009. IEEE Computational Intelligence Society, IEEE Press.

[5] R. Cleary and M. O'Neill. An attribute grammar decoder for the 01 multi-constrained knapsack problem. In *Proceedings of the Evolutionary Computation in Combinatorial Optimization*, pages 34–45. Springer Verlag, April 2005.

[6] M. de la Cruz Echeanda, A. O. de la Puente, and M. Alfonseca. Attribute grammar evolution. In *Proceedings of the IWINAC 2005*, pages 182–191. Springer Verlag Berlin Heidelberg, 2005.

[7] R. E.Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, C-35:677–691, 1986.

[8] C. Gagne, M. Schoenauer, M. Parizeau, and M. Tomassini. Genetic programming, validation sets, and parsimony pressure. In *Proceedings of the 9th European Conference on Genetic Programming*, volume 3905 of *Lecture Notes in Computer Science*, pages 109–120, Budapest, Hungary, April 2006. Springer.

[9] H. Iba and T. Sasaki. Using genetic programming to predict financial data. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, editors, *Proceedings of the Congress on Evolutionary Computation*, volume 1, pages 244–251, Mayflower Hotel, Washington D.C., USA, 6-9 July 1999. IEEE Press.

[10] F. S. E. G. J.M. Zaldivar, I.M. Galvan and A. Tomasin. Forecasting high waters at venice lagoon using chaotic time series analysis and nonlinear neural networks. *Journal of Hydroinformatics*, 021:61–84, 2000.

[11] C. Johnson. Deriving genetic programming fitness properties by static analysis. In *Proceedings of the 4th European Conference on Genetic Programming (EuroGP2002)*, pages 299–308. Springer, 2002.

[12] C. Johnson. Genetic programming with guaranteed constraints. In *Recent Advances in Soft Computing*, pages 134–140. The Nottingham Trent University, 2002.

[13] C. Johnson. What can automatic programming learn from theoretical computer science. In *Proceedings of the UK Workshop on Computational Intelligence*. University of Birmingham, 2002.

[14] C. Johnson. Genetic programming with fitness based on model checking. In *Proceedings of the 10th European Conference on Genetic Programming (EuroGP2002)*, pages 114–124. Springer, 2007.

[15] G. Katz and D. Peled. Genetic programming and model checking: Synthesizing new mutual exclusion algorithms. *Automated Technology for Verification and Analysis, Lecture Notes in Computer Science*, 5311:33–47, 2008.

[16] G. Katz and D. Peled. Model checking-based genetic programming with an application to mutual exclusion. *Tools and Algorithms for the Construction and Analysis of Systems*, 4963:141–156, 2008.

[17] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, Cambridge, Massachusetts, 1992.

[18] K. Krawiec and P. Lichocki. Approximating geometric crossover in semantic space. In F. Rothlauf, editor, *Genetic and Evolutionary Computation Conference, GECCO 2009, Proceedings, Montreal, Québec, Canada, July 8-12, 2009*, pages 987–994. ACM, 2009.

[19] J. Li, Z. Shi, and X. Li. Genetic programming with wavelet-based indicators for financial forecasting. *Transactions of the Institute of Measurement and Control*, 28(3):285–297, Aug. 2006.

[20] N. McPhee, B. Ohs, and T. Hutchison. Semantic building blocks in genetic programming. In *Proceedings of 11th European Conference on Genetic Programming*, pages 134–145. Springer, 2008.

[21] T. Mitchell. *Machine Learning*. McGraw Hill, New York, 1996.

[22] R. Poli, W. B. Langdon, and N. F. McPhee. *A Field Guide to Genetic Programming*. Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk`, 2008. (With contributions by J. R. Koza).

[23] R. Poli, N. McPhee, L. Citi, and E. Crane. Memory with memory in tree-based genetic programming. In L. Vanneschi, S. Gustafson, A. Moraglio, I. De Falco, and M. Ebner, editors, *Proceedings of the 12th European Conference on Genetic Programming, EuroGP 2009*, volume 5481 of *LNCS*, pages 25–36, Tuebingen, Apr. 15-17 2009. Springer.

[24] D. Rivero, J. R. Rabunal, J. Dorado, and A. Pazos. Time series forecast with anticipation using genetic programming. In J. Cabestany, A. Prieto, and F. S. Hernández, editors, *Computational Intelligence and Bioinspired Systems, 8th International Work-Conference on Artificial Neural Networks, IWANN 2005, Proceedings*, volume 3512 of *Lecture Notes in Computer Science*, pages 968–975, Vilanova i la Geltrú, Barcelona, Spain,, June 8-10 2005. Springer.

[25] M. Santini and A. Tettamanzi. Genetic programming for financial time series prediction. In J. F. Miller, M. Tomassini, P. L. Lanzi, C. Ryan, A. G. B. Tettamanzi, and W. B. Langdon, editors, *Genetic Programming, Proceedings of EuroGP'2001*, volume 2038 of *LNCS*, pages 361–370, Lake Como, Italy, 18-20 Apr. 2001. Springer-Verlag.

[26] N. Q. Uy, N. X. Hoai, and M. O'Neill. Semantic aware crossover for genetic programming: the case for real-valued function regression. In *Proceedings of EuroGP09*, pages 292–302. Springer, April 2009.

[27] N. Q. Uy, M. O'Neill, N. X. Hoai, B. McKay, and E. G. Lopez. Semantic similarity based crossover in GP: The case for real-valued function regression. In P. Collet, editor, *Evolution Artificielle, 9th International Conference*, Lecture Notes in Computer Science, pages 13–24, October 2009.

[28] M. L. Wong and K. S. Leung. An induction system that learns programs in different programming languages using genetic programming and logic grammars. In *Proceedings of the 7th IEEE International Conference on Tools with Artificial Intelligence*, 1995.