Influences of Function Sets in Genetic Programming

Jen-Shiang Wang
E. L. Ginzton Lab
Stanford University
Stanford, California 94305
jenwang@stanford.edu

ABSTRACT

Genetic programming (GP) provides a good tool for solving sequence induction and symbolic regression (i.e. function identification) problems. However, the influence of the function set in these problems has not been extensively studied. In this paper, I compare the efficiency of six function sets, using GP to find the numerical approximation of a number sequence and symbolic regressions of functions with one and two variables. The best individuals, the average fitness, and the performance curves of these function sets are presented for the efficiency comparison. The efficiencies of function sets with and without automatically defined functions (ADFs) are also compared. The characteristics of the best function set are given.

1. Introduction

Finite element methods and other numerical integration methods are major tools for solving complicated physics and engineering design problems. However, these methods typically require a lot of computational effort, and the results usually do not provide sufficient understanding of the underlying physics. On the other hand, if there is an analytic or empirical model with approximation formulae, we can have a better understanding of the problem and the computation can be reduced significantly due to the simplicity of these approximation formulae. Typically, these formulae were derived from theories and were modified according to the experimental data by humans. Now, it is possible for a computer to retrieve approximation formulae by itself from the experimental data. This automated discovery was successfully demonstrated by Koza (1992), who used an evolutionary algorithm called genetic programming (GP) to rediscover a simple formula from a set of data.

GP used a set of simple functions to build a function tree and explore the best function tree based on the Darwin selection rule. Since the simple functions (like `+`, `*`, `log`) are the basic building blocks of GP, the search efficiency will be influenced by the selection of functions in the function set. The selection of proper function sets with high converging efficiency is relatively important for physics or engineering problems as stated above because there are usually several parameters or variables. In this paper, I will compare the efficiencies of six function sets for three types of problem in several aspects, including the average fitness (error), the best individual, and the performance curve.

The Background and previous work are provided in Section 2. The problem statement of this research is stated in Section 3 and methods for solving these problems are specified in Section 4. The results are presented in Section 5 and are discussed in Section 6. The conclusion and the future work are given in the end.

2. Background

Genetic programming (GP) is an evolutionary algorithm that can automatically synthesize a complicated structure from a set of simple structures without being explicitly programmed (Koza 1992). GP has been used to solve numerous problems, including optimal control, automatic programming, game strategy development, and symbolic regression problems. For the symbolic regression problem, Koza (1992) did some pioneer work for functions with one variable. Later, Streeter and Becker (2001) extended Koza's work to symbolic regressions of functions with two variables and also used GP to induct the asymptotic formula of a number sequence with a complicated expansion. However, in these works, the influences of function sets were not explicitly studied.

Soule and Heckendorn (2001) studied the influence of function sets in the symbolic regression problems. However, only one one-variable function was studied in their work. This paper will extend their work to study the influence of function sets on symbolic regressions for functions with two variables. Moreover, the influence of the function sets in the sequence induction problem will be studied since sequences are quite commonly used in physics and engineering problems. Streeter and Becker (2001) suggested to include automatically defined functions (ADFs) in the function set to improve the search efficiency. Function sets with and without ADFs will be compared in this paper.

3. Problem Statement

Sequence induction involves finding a mathematical (asymptotic) expression to calculate the value of the sequence at a given step. Symbolic regression relates finding a mathematical expression from a finite set of values of the independent variables and the associated values of the dependent variables. The goal of this research is to compare the efficiencies of different function sets in inducting the asymptotic formula of a sequence and in finding the symbolic regression of a known function. Three categories of problems will be examined in this paper.

The first category is to induct the asymptotic formula of a number sequence. The selected number sequence is the natural logarithm of factorial function `log(n!)`. The sequence can be represented by Stirling's Series (Arfken 1985):

$$\log(\mathrm{n!}) = \frac{1}{2}\log 2\pi + (n+\frac{1}{2})\log n - n + \frac{1}{12n} - \frac{1}{360n^3} + \frac{1}{1260n^5} - \ldots \qquad (1)$$

The reason for choosing this function instead of the factorial function `n!` is that the error function is easier to determine. The value of the factorial function grows dramatically when `n` is large. The terms with larger values will affect the selection of the asymptotic function more than the terms with lower numbers do. The logarithm function reduces the weighting of the large-number terms such that the error can be reasonably distributed over all the components.

The second category is to find the symbolic regression of a function with one variable. The selected function is the hyperbolic tangent `tanh(X)`. This function can be represented as:

$$\tanh(\mathrm{x}) = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)} = \frac{x+\frac{1}{3!}x^3+\frac{1}{5!}x^5+\ldots}{1+\frac{1}{2!}x^2+\frac{1}{4!}x^4+\ldots} \qquad (2)$$

The last category is to find the symbolic regression of a function with two variables. The selected function is X^{Y}. The function can be represented as:

$$x^y = \exp(y * \log x) \qquad (3)$$

Notice that this function cannot be expanded as a product of a summation of a function of variable x and a function of y. Thus, this function cannot be represented as a polynomial expansion by a regular way.

4. Methods

The method for finding the numerical approximation or symbolic regression is the basic genetic programming described in Koza's book (Koza 1992). The tableaus of the three problems are shown in Table 1, 2, and 3.

For the consistency, these problems used the same set of parameters: The population size (M) is 1000; The maximum number of generations (G) is 151; The maximum size in depth is 10; The crossover rate for leaves is 0.10; The crossover rate for nodes is 0.80; The reproduction rate is 0.10; The mutation rate is 0.00. 60 trials of each problem with each function set and 1080 (60×3×6) trials in total were performed to gather the statistical data. The random seeds for these 60 trials were selected randomly between 0 and 10^8.

Other than the difference in target function, the major differences among these problems are the number and the range of data points. In the first problem, the independent variables are 50 consecutive integers from 1 to 50. These data points are used as the training data for GP to induct the asymptotic formula. The second problem has 200 data points with the absolute values of the independent variables from 10^{-6} to 10^4. The purpose of this wide range is to keep the function value stay at 1 when the independent variable is large (i.e. the value of the function is 1 when `X` is large). The last problem has 100 data points. Independent variable `X` is from the interval [0,1], and independent variable `Y` is also from the interval [0,1].

. Table 1 Tableau for the sequence induction problem with `log(n!)` as the target function

Objective:	Find an asymptotic expansion of a number sequence, in symbolic form, that fits a given sample of 50 (n_i, s_i) data points, where the target function is `log(n!)`.
Terminal set:	n (the independent variable).
Function set:	The function sets are listed in Table 4
Fitness cases:	The given sample of 50 data points (n_i, s_i) where the n_i are consecutive integers from 1 to 50.
Raw fitness:	The sum, taken over the 50 fitness cases, of the absolute value of difference between the value of the dependent variable produced by the S-expression and the target value s_i of the dependent variable.
Standardized fitness:	Eqauls raw fitness for this problem.
Hits:	Number of fitness cases for which the value of the dependent variable produced by the S-expression comes within 0.05 of the target value s_i of the dependent variable.
Wrapper:	None.
Parameters:	M = 1,000. G= 151.
Success predicate:	An S-expression scores 50 hits or the raw fitness is below 1.

. Table 2 Tableau for the symbolic regression problem with `tanh(x)` as the target function

Objective:	Find a function of one independent variable and one dependent variable, in symbolic form, that fits a given sample of **200** (x_i, y_i) data points, where the target function is `tanh(X)`.
Terminal set:	`X` (the independent variable).
Function set:	The function sets are listed in Table 4
Fitness cases:	The given sample of 200 data points (x_i, y_i) where the x_i are listed below: $\{\pm1, \pm2, \pm3, \pm4, \pm5, \pm6, \pm7, \pm8, \pm9, \pm10\} * 10^n$, n = -6,-5,-4,-3,-2,-1 ,0,1,2,3
Raw fitness:	The sum, taken over the 200 fitness cases, of the absolute value of difference between the value of the dependent variable produced by the S-expression and the target value y_i of the dependent variable.
Standardized fitness:	Eqauls raw fitness for this problem.
Hits:	Number of fitness cases for which the value of the dependent variable produced by the S-expression comes within 0.01 of the target value y_i of the dependent variable.
Wrapper:	None.
Parameters:	M = 1,000. G= 151.
Success predicate:	An S-expression scores 200 hits or the raw fitness is below 1.

Table 3 Tableau for the regression problem with x^y as the target function

Objective:	Find a function of two independent variables and one dependent variable, in symbolic form, that fits a given sample of **100** (x_i, y_i, z_i) data points, where the target function is X^Y.
Terminal set:	`X`,`Y` (the independent variables).
Function set:	The function sets are listed in Table 4
Fitness cases:	The given sample of 100 data points (x_i, y_i, z_i) where the (x_i, y_i) come from the interval $\{[0, 1] \times [0,1]\}$
Raw fitness:	The sum, taken over the 100 fitness cases, of the absolute value of difference between the value of the dependent variable produced by the S-expression and the target value z_i of the dependent variable.
Standardized fitness:	Eqauls raw fitness for this problem.
Hits:	Number of fitness cases for which the value of the dependent variable produced by the S-expression comes within 0.01 of the target value z_i of the dependent variable.
Wrapper:	None.
Parameters:	M = 1,000. G= 151.

Success predicate:	An S-expression scores 100 hits.

The function sets (FS) to be tested are listed in Table 4. Here `%` is the protected division function, which returns `zero` when the denominator is zero. `RLOG` is the protected logarithm function, which returns the logarithm of the absolute value of the argument, and returns `−1000` when the argument is zero. As can be seen from the expansions in Section 3, the target functions were selected intentionally to contain either the logarithm function or the exponentiation function. The purpose of this similarity is to provide a more common criterion for comparisons. Based on this similarity, FS1 was selected to contain both logarithm and exponentiation functions. FS2 is a modified version of FS1 with an ADF. FS3 contains two periodic functions `SIN` and `COS` in addition to the functions in FS1. FS4 is the modified version of FS3 with an ADF. Since most functions can be represented by a Tyler's expansion or a Padé approximation, it is possible to use basic functions to compose a complex function. Thus, FS5 only contains the basic functions `{+, -, *, %}`. FS6 is the modified version of FS5 with an ADF. The last two function sets are used to test the ability of GP to find the polynomial approximation of a complicate function.

Table 4 Function sets

Number	Function Set
FS1	`+, -, *, %, EXP,` and `RLOG.`
FS2	`+, -, *, %, EXP, RLOG,` and `ADF0.` The terminal set for `ADF0: ARG0 and ARG1` The function set for `ADF0: +, -, *, %, EXP,` and `RLOG.`
FS3	`+, -, *, %, EXP, RLOG, SIN,` and `COS.`
FS4	`+, -, *, %, EXP, RLOG, SIN, COS,` and `ADF1.` The terminal set for `ADF1: ARG0 and ARG1` The function set for `ADF1: +, -, *, %, EXP, RLOG, SIN,` and `COS.`
FS5	`+, -, *,` and `%.`
FS6	`+, -, *, %,` and `ADF2.` The terminal set for `ADF2: ARG0 and ARG1` The function set for `ADF2: +, -, *,` and `%.`

5. Results

As mentioned in Section 4, 60 trials were performed for each problem and for each function set. For each trial, there is a best individual, so there are 60 best individuals in total. The average hit and the average error (which is the same as the average raw fitness) of these 60 individuals can be used as the first comparison criterion. The best individual (and its error) among these 60 trials can be used as the second criterion. The third criterion is the performance curve. These three criteria will be presented in the following subsections. Because there are no individuals satisfying the success predicate in FS5 and FS6, only the performance cures of FS1, FS2, FS3, and FS4 will be presented.

5.1. Asymptotic formula of `log(n!)`

The average errors and the best individuals of those six function sets defined in Table 4 are listed in Table 5. The best values are highlighted in boldface. As described in Table 1 to Table 3, hits represent the number of matched points. So the larger the hit is, the better the function set is. The error represents the difference between the target function and the retrieved function. So the smaller the error is, the better the function set is. As shown in Table 5, FS1 outperforms the other function sets in all aspects. The expression of the best individual in FS1 is similar to Stirling's series in Eq (1). After FS1, the performance rankings from the best to the worst are FS3, FS2, FS4, FS5, and FS6. For the comparison between function sets with and without ADFs, the function sets without ADF (FS1, FS3, and FS5) all outperformed their ADF counterparts (FS2, FS4, and FS6). Including ADF in the function set does improve but reduces the convergence efficiency. The best-individual errors of FS5 and FS6 are relatively large, which means that GP could not retrieve good polynomial expansions in this problem

The performance curves of FS1 to FS4 are shown in Figure 1. The small probabilities of success in all four cases reveal that the problem is difficult to solve. The minimum numbers of individual to be processed are from 6,336,000 (FS1) to 18,224,000 (FS2). Overall, FS1 has the best convergence efficiency and the function set with ADF also converge worse than their counterparts without ADF. These results are consistent with the results shown in Table 5.

Table 5 Average errors and the best individuals of six functions sets with `log(n!)` as the target function

	Average		Best individual		
	Hits	Error	Expression (after simplification if applicable)	Hits	Error
FS 1	**19.4**	**14.743**	`0.93565 + 0.49573*RLOG(X)+X*RLOG(X)- X`	**50**	**0.367**
FS 2	8.1	48.303	**`Main:`** `(X+.238144)*RLOG((X+.23814*RLOG(X)))-ADF0(X,-0.166/RLOG(X))` **`ADF0:`** `(ARG0^2*ARG1-ARG1^3*ARG0+ARG1^4-ARG0^2)/ARG0/ARG1`	**50**	0.607
FS 3	15.2	44.438	`0.69183*RLOG(0.75829+X)+RLOG(0.75829+X)*X-0.17991*RLOG(0.83033+X)+0.12446-X`	**50**	0.382
FS 4	6.8	86.490	`NA`*	47	0.974
FS 5	1.7	85.178	`NA`*	11	5.510
FS 6	0.7	146.21	`NA`*	4	19.100

*NA means the best individual cannot be represented in a simple or short form.

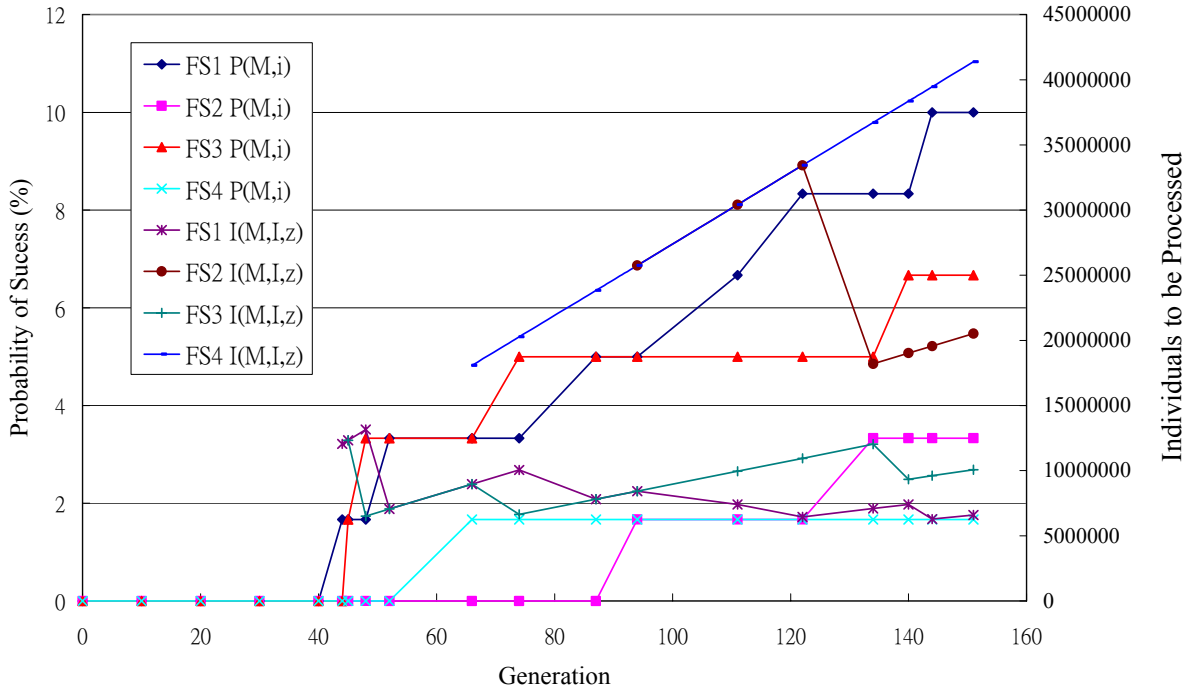

Figure 1. Performance curves of FS1 to FS4 for the symbolic regression problem with `log(n!)` as the target function.

5.2. Symbolic regression of `tanh(X)`

The average errors and the best individuals of the six function sets are listed in Table 6. FS1 outperforms the other function sets in the average values, while FS3 ranks first in the best individual. After FS1 and FS3, the performance rankings (based on the average error) from the best to the worst are FS2, FS4, FS5, and FS6. Surprisingly, none of these six best individuals has an exact or similar expansion form like the one in Eq (2). For the comparison between function sets with and without ADFs, FS1 performs better than its ADF counterpart FS2 in all aspects. FS3 performs better than its ADF counterpart FS4 in all aspects except the average hits. FS5 performs worse than or in even with its non-ADF counterpart FS3 in most aspects except the average error. The best-individual errors of

FS5 and FS6 are relatively large compared to the others, which means that GP could not retrieve good polynomial expansions in this problem

The performance curves of FS1 to FS4 are shown in Figure 2. The small probabilities of success in all of these cases imply that the problem is difficult to solve. The minimum numbers of individual to be processed are quite large, from 5,427,000 (FS3) to 19,992,000 (FS4). Overall, FS3 has the best convergence efficiency. Again, the function sets with ADF converge worse than their non-ADF counterparts.

Table 6 Average errors and the best individuals of six function sets with `tanh(X)` as the target function

	Average		Best individual		
	Hits	Error	Expression (after simplification if applicable)	Hits	Error
FS 1	**121**	**33.351**	`X*(EXP(X)+X*EXP(-1/X))/(X*EXP(X)+X^2*EXP(-1/X)+1)`	**200**	0.131
FS 2	104	43.671	NA*	**200**	0.148
FS 3	92.8	43.944	`((EXP(RLOG(X)))/(X+(((SIN(EXP(RLOG(X))))/X)/(EXP(EXP(RLOG(X))))))` `= |x|/(X+((SIN(|X|)/X)/EXP(|x|)))`	**200**	**0.088**

FS 4	96.0	49.770	**`Main:`** `(X/(((EXP(RLOG(X)))`
			`-(-0.22128/((EXP(EXP(RLOG(X))))`
			`-(EXP(-0.22128/(EXP(RLOG(X)))))))`
			`-(-0.22128/(EXP(RLOG((EXP(RLOG(X)))`
			`-((-0.22128)/(EXP(EXP(RLOG(X))))))))))`
			`ADF1:` ARG0
				194	0.396

FS 5	80.8	71.852	NA*	83	62.7
FS 6	89.7	74.044	NA*	83	60.18

*NA means the best individual cannot be represented in a simple or short form.

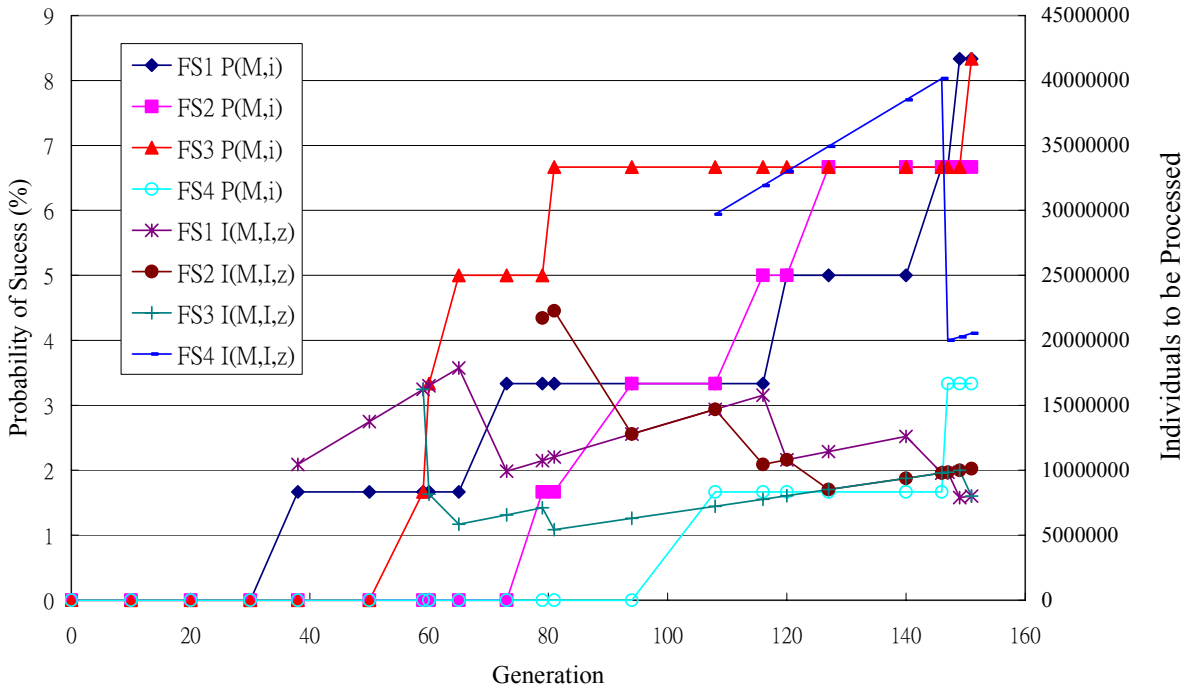

Figure 2. Performance curves of FS1 to FS4 for the symbolic regression problem with `tanh(X)` as the target function.

5.3. Symbolic regression of X^Y

The average errors and the best individuals of the six function sets are listed in Table 7. FS1 has the highest average hits, while FS3 ranks first in the average error. After FS1 and FS3, the performance ranking (based on the average error) from the best to the worst are FS4, FS5, FS2, and FS6. The best individuals of FS1, FS2, FS3 and FS4 are exactly the same as the expansion of X^Y shown in Eq (3). For the comparison between function sets with and without ADF, FS1 and FS3 perform better than their ADF counterparts FS2 and FS4 in the average hits and the average error. FS5 performs better than its ADF counterpart FS4 in all aspects. Although it does meet the success predicate, the best individual in FS5 has an error of a reasonable value. It shows the possibility of using GP to retrieve the Padé approximation of functions that were treated as nonexpendable.

The performance curves of FS1 to FS4 are shown in Figure 3. Because the best individuals satisfying the success predicate were found within generation 60, the performance curves were drawn from generation 0 to generation 60. The minimum numbers of individual to be processed are from 53,000 (FS2) to 275,000 (FS4), which are about two magnitudes smaller than the values in previous problems. Although FS2 converges fastest in the first few generations, FS1 has the overall best convergence efficiency. Overall, the function sets with ADF converge worse than their non-ADF counterparts.

Table 6 Average errors and the best individuals of six functions sets with X^Y as the target function

	Average		Best individual		
	Hits	Error	Expression (after simplification if applicable)	Hits	Error
FS 1	**53.4**	5.138	`EXP(Y * RLOG(X))`	**100**	**0.0**
FS 2	27.7	10.73	**`Main:`** `EXP(Y * RLOG(X))` **`ADF0:`** `ARG0+ARG1`	**100**	**0.0**
FS 3	53.0	**2.623**	`EXP(Y * RLOG(X))`	**100**	**0.0**
FS 4	32.2	6.770	**`Main:`** `EXP(Y * RLOG(X))` **`ADF1:`** `ARG0 – ARG1`	**100**	**0.0**
FS 5	19.4	8.961	`X/((Y-Y^2*X+X)*(Y-Y*X)*(0.8259537924/Y^3*X*(Y-Y^2)^2*(Y-Y*X)+0.92038)+X)`	79	0.687
FS 6	13.8	12.11	**`Main:`** `X/(0.78967*Y+0.78967*X/(Y+0.78967))` **`ADF2:`** `ARG1`	41	3.03

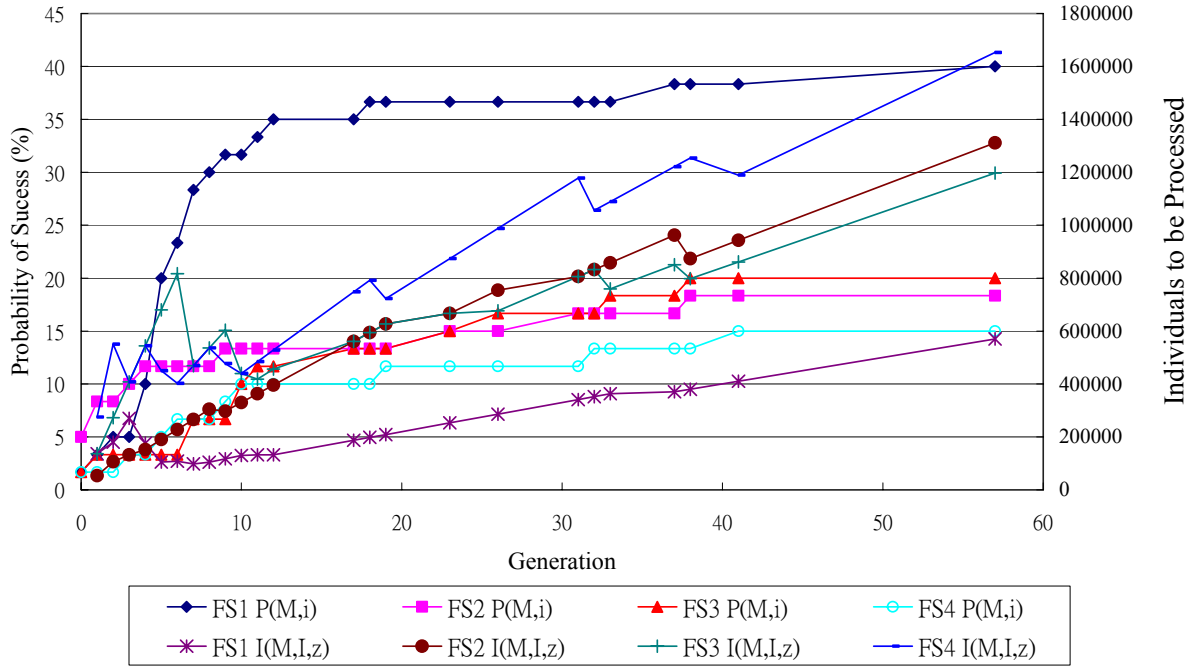

Figure 3. Performance curves of FS1 to FS4 for the symbolic regression problem with x^y as the target function.

6. Discussion of Results

Overall, FS1 has the best performance. It has the highest average hits in all three problems and the lowest errors in two of the problems. The best individual of FS1 outperforms others in two of the problems. From the performance curves, the convergence speed of FS1 is fastest in two of the problems. Among the six function sets, FS1 is the smallest function set that contains the basic elements of the target function, namely `EXP`, `RLOG`, and the basic arithmetic operators. The similarity between the target function and the elements in the function set helps finding the best individual. Having the smallest number of elements also increases the possibility of finding the better individuals, contributing a better average performance. This property could be further verified by testing a variety of target functions, including functions with periodicity (like `SIN(3X) + COS(5X)`) and functions with abrupt changes (like the flattop function).

Interestingly, in the problem with `tanh(X)` as the target function, the overall best individual is retrieved from FS3. As shown in Table 6, this best individual contains the function `SIN`, a function that is not in FS1. This result shows that including extra functions helps finding the best individual in some problems. In addition, including extra functions in the function set may also improve the average performance, which is supported by the fact that FS3 achieved the lowest average error in the problem with X^Y as the target function. These properties should be further verified by testing target functions with more variables.

Surprisingly, the result showed that including ADFs in the function set did not improve the performance. On the contrary, adding ADF reduced the average performance and retarded the convergence. These downgraded performances reveal that these problems do not have regularities such that ADFs can be used to improve the convergence efficiency.

As mentioned in Section 4, FS5 and FS6 are used to test ability of GP to find the Padé approximation or Taylor's expansion. There were no good approximation formulae found for `log(n!)` and `tanh(X)`. A possible explanation for this is that the ranges of the independent variables are too large in these problems. In the first problem, the independent variables are integers from 1 to 50 (Table1). In the second problem, the absolute values of the independent variables are from 0.000001 to 10,000 (Table2). When the range of the independent variables is large, it requires more terms to fit the data. Increasing the tree size may help finding a good approximation since more terms can be included. The problem with `tanh(X)` as the target function should be more easily solved if the range is reduced. The influence of the sampling points is a worthwhile topic to look at. A better understanding of the influence of the sampling points may explain why the exact form of `tanh(X)` was not retrieved.

7. Conclusion

This paper examined the influence of the function set when using genetic programming (GP) to solve the sequence induction problem and the symbolic regression problem. Three types of problems (sequences, functions with one variable, and functions with two variable) along with six function sets were tested statistically. Sixty trials for each problem and for each function set were performed to retrieve the average error, the best individual, and the performance curve, which were used to evaluate the performance of the function set. The results showed that adding automatically defined functions (ADFs) in the function set did not improve the performance, implying that these problems did not have regularities so that ADF can be used to improve the performance. Two characteristics were found in the best function set. First, the function set contained elements similar to the target function. Second, the function set had the smallest number of elements in the set. In the last problem, a Padé expansion of a nonexpendable function X^Y was successfully retrieved, demonstrating the potential of using GP to decouple multi-variable functions.

8. Future Work

The work done in this research suggested several possible extensions. First, by studying more target functions and more function sets could provide broader understanding of the influence of the function set. Second, by studying the influence of the sampling points together with the influence of the function could help determine the best configuration for efficient searching. Finally, for the long run, the efficiency of the function set should be tested in real engineering problems.

Bibliography

Arfken, George 1985. *Mathematical Methods for Physicists, 3rd Edition*. Orlando, FL: Academic Press.

Koza, John R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: The MIT Press.

Soule, Terence and Heckendorn, Robert E. 2001. Function Sets in Genetic Programming. In Spector, L., Goodman, E. D., We, A., Langdon, W. B., Voigt, H.-M., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M. H., and Burke, E. (editors). *Proceedings of the Genetic and Evolutionary Computation Conference* San Francisco ,CA: Morgan Kaufmann Publishers. Page 190.

Streeter, Matthew and Becker, Lee A. 2001.Automated Discovery of Numerical Approximation Formulae Via Genetic Programming. In Spector, L., Goodman, E. D., We, A., Langdon, W. B., Voigt, H.-M., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M. H., and Burke, E. (editors). *Proceedings of the Genetic and Evolutionary Computation Conference* San Francisco ,CA: Morgan Kaufmann Publishers. Pages 147-154.