The Assessment and Application of Lineage Information in Genetic Programs for Producing Better Models

Gary D. Boetticher, Kim Kaminsky, *Member, IEEE*

***Abstract*—One of the challenges in data mining, and in particular Genetic Programs, is to provide sufficient coverage of the search space in order to produce an acceptable model. Traditionally, Genetic Programs generate equations (chromosomes) and consider all chromosomes within a population for breeding purposes. Considering the enormity of the search space for complex problems, it is imperative to examine Genetic Programs breeding efforts in order to produce better solutions with less training.**

This research examines chromosome lineage within Genetic Programs in order to identify breeding patterns. Fitness values for chromosomes are sorted, then partitioned into five classes. Initial experiments reveal a distinct difference between upper, middle, and lower classes. Based upon initial results, a novel Genetic Programming process is proposed which breeds a new generation exclusively from the top 20 percent of a population. A second set of experiments statistically validate this proposed approach.

I. Introduction

Traditionally, Genetic Programs (GP) solve problems by generating a set of mathematical equations, or chromosomes, that represent a mapping between two sets of variables. Collectively, these chromosomes form a population. GPs repetitively breed new generations of chromosomes seeking to find an optimal, or at least satisfactory, solution.

GPs are frequently deployed for identifying patterns in large, complex, noisy datasets where the corresponding search space is extremely large. Finding a solution within this search space is an extremely difficult challenge. As might be expected, GPs struggle at providing search space coverage. For example, there are more than $8.51*10^{37}$ ways of constructing an equation tree of height 5 consisting of 4 variables. Running a GP experiment with a population of 1000 equations (chromosomes) for 1000 generations produces at most one million possible equations. This experiment would cover less than 10^{-30} percent of all the possibilities.

The previous example assumes that GPs are static in structure. It is well known that chromosomes rapidly increase in size as the population evolves [1]–[4], thus greatly increasing the expanse of the search space and reducing the probability that a solution will be found.

Solving large problems using GPs consumes excessive amounts of computer resources. Though Genetic Programs may successfully evolve solutions to complex problems, their use may sometimes be cost-prohibitive.

What is desired is a more efficient approach to exploring the search space. This may be accomplished qualitatively by focusing the search efforts or quantitatively by increasing the number of searches.

This research explores the qualitative approach by examining the breeding patterns of a GP. Some key questions addressed are, ***Does chromosome lineage information provide any insight into the effectiveness of solving problems? If so, how could these insights be utilized to make better breeding decisions?***

Gaining a better understanding about a chromosome's lineage, in terms of how fitness values propagate over generations, could be beneficial in several ways. Greater emphasis could be placed on those chromosomes that produce better offspring. Secondly, the utility of such a discovery could focus the search efforts, thus reducing the training time, and requiring less computing resources. All these benefits are immensely important when applying GPs to large, complex, noisy problem spaces.

To explore the role chromosome lineage plays in the breeding process, five initial experiments are conducted using synthetic datasets. Chromosomes are clustered into different classes (e.g. upper, middle, and lower classes). Each of these classes is tracked over a generation to determine whether certain classes are prone to producing good (or poor) solutions.

Based upon the results of the initial set of experiments, an alternative breeding approach is proposed that focuses on those chromosomes with a solid pedigree. A second set of experiments examines this novel approach along with a traditional approach to determine the merit of focusing on a certain portion of a GP population.

II. Related Research

McPhee et al. [5] analyze node level genetic diversity in a GP population over its genetic history. They observe that there is a profound loss of diversity over the evolution process indicating that a standard GP does not perform opportunistic breeding.

Burke et al. [6] use lineage selection to increase diversity by reducing the selection pressure from "most fit" to "fit and

Manuscript received May 1, 2006 and revised July 30, 2006. This work was supported in part by the Institute for Space Systems Operations (ISSO).

G. D. Boetticher is with the University of Houston – Clear Lake, Houston, TX 77058 (e-mail boetticher@uhcl.edu)(Phone: 281.283.3805)

K. Kaminsky was with the University of Houston – Clear Lake, Houston, TX 77058. She is currently with Quorum Business Solutions.

diverse." They find that introducing diversity can avoid getting trapped in local optima.

Both McPhee and Burke use lineage information as a method to promote diversity within the population. This research uses lineage information more as a mechanism to improve the selection process.

III. How Genetic Programs Work

Genetic Programs solve problems by genetically breeding a population of individuals, or chromosomes, over a series of generations. Inspired by theories of evolution, Genetic Programs use the analogy of evolutionary operators on chromosomes to optimize a fitness function. A fitness function assesses the goodness of a chromosome (represented as an equation) in terms of how well (or poorly) that equation fits a given dataset. The goodness of a chromosome serves as the basis for propagation decisions.

An implementation of a Genetic Program starts with a population of individual equations, usually represented as tree structures. Each tree, or chromosome, can be viewed as a potential solution to the given problem (training data). Each node on the tree represents a *gene*, or some trait within a problem. Programmatically, each gene corresponds to either an operator or an operand. Collectively, the set of the genes would make up a mathematical expression.

Collectively the set of chromosomes, which represent potential solutions, are known as a population. This population 'reproduces' to create a future generation. Each iteration of a Genetic Program produces a new generation of individuals.

After the population has been initialized and the fitness of each individual has been evaluated, the selection of parent chromosomes occurs. During selection, the fittest individuals are selected to engage in reproduction. A fitness function evaluates the individuals and ranks them in terms of performance. An example of a fitness function is:

$$Fitness = 1 + e^{(7*(1-n-k)/(n-1)*Se^2/Sy^2)} \quad (1)$$

where

k corresponds to the number of inputs;
n represents the number of valid results;
Se is the standard error; and
Sy equals the standard deviation.

This equation ranges from 1 through 1067. These values are scaled to span 1 through 1000.

Chromosomes are paired together by their fitness for breeding purposes. After selecting two individuals within a population, several reproductive type steps occur. One example is crossover. The crossover process takes a subtree from each chromosome parent, chooses a random branch, and then crosses over the genetic material. Crossover occurs at one point, or at several points, within each chromosome.

A second step in the breeding process is mutation. Mutation randomly selects an operator or operand node within an equation tree and modifies the value. Mutation promotes diversity within a population by randomly adding in gene variations and prevents a solution from falling into local minima or maxima, which is a problem experienced by most optimization algorithms [7], [8].

This process of creating a population of individuals, ranking the individuals by fitness, and recombining these individuals to produce a better set of solutions is called a generation. The modeling process spans multiple generations until an acceptable solution is found; the experiment runs for a specified number of generations, or the user terminates the run. Figure 1 shows the general Genetic Programming algorithm.

```
The Genetic Program

Initialize the Population
Evaluate Fitness
Repeat
    Select Parents
    Perform Crossover
    Perform Mutation
Until a Desired Fitness is Reached
```

Figure 1: GP Algorithm [7], [9]

IV. Proposed Research

In traditional GP modeling process, once a new generation is created, all legacy information about the previous generation is discarded. Perhaps this discarded ancestral fitness information could offer valuable clues on how to make better propagation decisions. This research statistically analyzes a chromosome's lineage, in terms of the fitness values across generations. If there is a correlation between fitness values across generations, then it would be possible to focus only on those chromosomes with a good pedigree. This could lead to more accurate GP solutions requiring less training time.

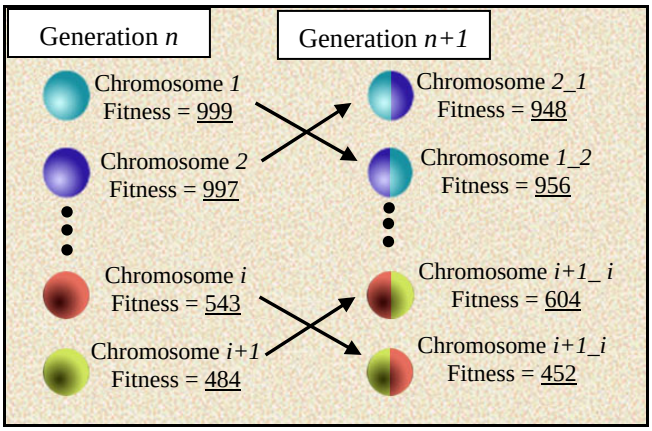

Figure 2: Tracking Fitness values For Two Generations

Figure 2 illustrates the tracking of fitness values across multiple generations. In the Figure above, two of the most fit

chromosomes (1 and 2 in generation *n*) produce fitter children in generation *n + 1*.

It is possible to trace lineage back several generations (e.g. grandparents, great-grandparents). However, this research only considers the previous generation in the lineage assessment process.

V. Initial Experiments

Five experiments are conducted using synthetic data sets where the dependent values are clearly defined. Using synthetic data reduces the fuzziness of problems and makes it possible to adequately test the theory. The Genetic Program must not solve the equations easily, since this experiment requires a large number of samples. However, the Genetic Program needs the ability to model the equation easily. A difficult problem causes a Genetic Program to get trapped in local minima. In this case, the Genetic Program may not grow closer to the solution and the differences in fitness values over time may not be clearly illustrated. Therefore, the five equations chosen may be solved by the Genetic Program easily within a few generations. To keep the Genetic Program from solving the programs, a very small random number is added to each dependent variable for all instances. This makes it impossible for the Genetic Program to solve the problem, while allowing the Genetic Program to come very close to the actual solution.

The datasets are based on the following five equations:

$$Z = W + X + Y \quad (2)$$

$$Z = 2 * X + Y - W \quad (3)$$

$$Z = X / Y \quad (4)$$

$$Z = X^3 \quad (5)$$

$$Z = W^2 + W * X - Y \quad (6)$$

The complexity of each equation becomes progressively complicated for each experiment.

All experiments define a generation as 1,000 chromosomes. The selection method pairs chromosomes based on their fitness rank, with the top two fittest individuals mating, then the next two, etc.

All trials run for 50 generations. For every generation, the 1,000 chromosomes are sorted by fitness values then divided into five distinct groups of 200 chromosomes each. The fitness values of the offspring are recorded for each pair of parent chromosomes and the average of all fitness values within a group is calculated.

The next step compares the offspring's fitness values for those parents who had the best 200 chromosomes (the *best class*) with the offspring's fitness values of those parents who had the middle 200 chromosomes (the *middle class*), along with those offspring whose parents had the lowest 200 fitness values (the *worst class*).

VI. Initial Experiment Results

Figure 3 depicts the results from running the Genetic Program against equation (2). The *x-axis* represents the number of generations (1 through 50). The *y-axis* shows the average fitness values for the *best class*, *middle class*, and *worst class* groups. The black line (the one in the 300-400 range) represents average fitness values of the offspring for the *best class* parents. The pink line (the middle line) is the average fitness values of the offspring for the *middle class* parent chromosomes. The navy blue line shows the average fitness values of the offspring for the worst class parents. Inspecting Figure 2, it is clear that there is a distinction between best, middle, and worst class groups. At no time do any of the group averages intersect. A t-test reveals these differences as statistically significant.

Figure 3: Results from the First Experiment

Figures 4 shows the results for the GP modeling equation (3). The x-axis represents the number of generations (1 through 50). The y-axis shows the average fitness values for the *best class*, *middle class*, and *worst class* groups. The black line (the one in the 500-700 range) represents average fitness values of the offspring for the upper class parents. The pink line (the middle line) is the average fitness values of the offspring for the *middle class* parent chromosomes. The navy blue line shows the average fitness values of the offspring for the worst class parents. Once again, it is clear that there is a distinction between best, middle, and worst class groups. At no time do any of the group averages intersect. A t-test reveals the differences between the classes as statistically significant.

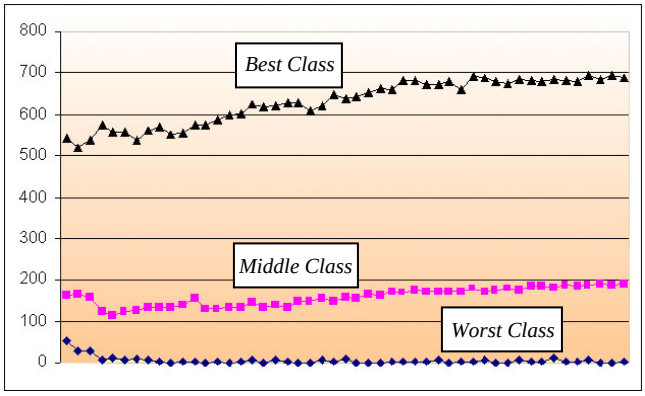

Figure 4: Results from the Second Experiment

Figure 5 illustrates the differences between the three groups of chromosomes for the third experiment for equation (4). The x-axis represents the number of generations (1 through 50). The y-axis shows the average fitness values for all three classes. The black line (top line) represents the *best class* group. The pink line corresponds to the average fitness values of the offspring for the middle set of parent chromosomes. The navy blue line (bottom line) shows the average fitness values of the offspring of the worst class parents. Once again, there is a distinct difference between the *best class* and the other two groups. The *middle class* group's fitness values are generally higher than the fitness values of the offspring of the *worst class* parents. T-tests indicate that the differences between these groups are significant in all cases.

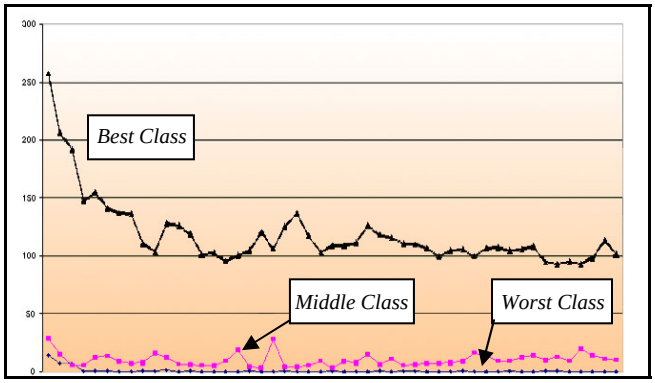

Figure 5: Results from the Third Experiment

Figure 6 shows the results for the third experiment which model equation (5). The average fitness values of the offspring of the best chromosomes are far superior to the other two groups. Compared to the previous experiments, this *middle* and *worst class* parents are not as distinguishable. The t-test analysis indicates that the differences between these groups are significant in all cases where the highest fitness values differ from the other groups.

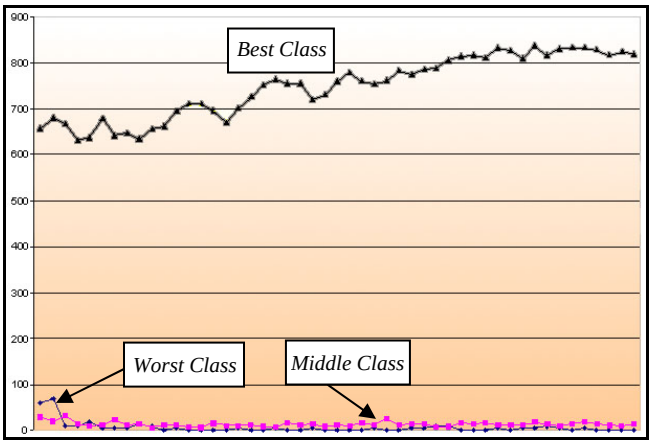

Figure 6: Results from the Fourth Experiment

Figure 7 depicts the results for the fifth experiment (equation 6). The black line shows average fitness values of the offspring of the set of the best 200 chromosomes. These offspring are much higher than the other (middle and worst class) lines. The t-tests indicate that the superiority of the *best class* parents is statistically significant to each of the other classes. The *middle class* and *worst class* results are comparable.

Figure 7: Results from the Fifth Experiment

VII. APPLIED EXPERIMENTS

The previous set of experiments demonstrate the statistical superiority of chromosomes from the *best class* versus those chromosomes from the middle and lower classes with respect to average fitness values. The next question is whether this information may be leveraged for building better GP models in fewer generations.

To answer this question, two experiments are conducted which compare a vanilla-based GP (all chromosomes participate in the formation of the next generation) versus a lineage-based GP which siphons the top 20 percent of the population, replicates this group 5 times, then breeds the next generation. The intent is to determine whether a lineage-based approach produces better models in less time.

These experiments use datasets based on the following

equations.

$$Z = Sin(W) + Sin(X) + Sin(Y) \quad (7)$$

$$Z = \log_{10} * (W^{X}) + (Y * Z) \quad (8)$$

The previous set of experiments perturbed the data so the models would not prematurely finish. These two experiments do not perturb data so that those models may finish early. This makes it possible to assess how quickly an approach converges to a solution. However, these two equations are a bit more complicated than the previous 5 equations so that the GP would not converge too quickly.

Both experiments consist of 20 trials. Every trial uses a population of 1000 chromosomes with a maximum chromosome length of 2000 characters and runs for up to 50 generations.

Table I shows the results for modeling equation 7. It shows the final fitness value with 1000 as the maximum, the final r-squared, and the generation in which it stopped. The last row shows the average for each respective column.

TABLE I
ORGINAL VS. LINEAGE APPROACH FOR EQUATION 7

Original GP			Lineage-Based GP		
Fitness	Final r^2	Gen.	Fitness	Final r^2	Gen.
1000	1.0000	8	1000	1.0000	12
1000	1.0000	9	1000	1.0000	2
1000	1.0000	6	1000	1.0000	6
1000	1.0000	8	1000	1.0000	3
1000	1.0000	4	1000	1.0000	4
1000	1.0000	6	1000	1.0000	6
1000	1.0000	21	1000	1.0000	21
1000	1.0000	4	1000	1.0000	6
1000	1.0000	9	1000	1.0000	3
1000	1.0000	6	1000	1.0000	6
309	0.8316	50	934	0.9900	50
265	0.8096	50	838	0.9746	50
232	0.7906	50	807	0.9691	50
228	0.7878	50	771	0.9625	50
188	0.7601	50	470	0.8919	50
147	0.7251	50	226	0.7867	50
137	0.7150	50	195	0.7656	50
121	0.6973	50	195	0.7656	50
111	0.6846	50	195	0.7656	50
98	0.6665	50	187	0.7593	50
591.8	**0.87341**	**29.1**	**740.9**	**0.9315**	**28.5**

Both approaches converge to an exact solution 10 out of 20 times. The lineage-based GP approach produces superior results over the traditional GP approach in terms of fitness, r-squared, and number of generations. The r-squared results for the lineage approach are statistically superior (α=0.10) over the traditional approach.

Table II shows the results of the two approaches for modeling equation 8. As in the previous table, each row represents each trial with the last showing the averages of the 20 trials.

TABLE II
ORGINAL VS. LINEAGE APPROACH FOR EQUATION 8

Original GP			Lineage-Based GP		
Fitness	Final r^2	Gen.	Fitness	Final r^2	Gen.
662	0.9408	50	1000	1.0000	22
467	0.8907	50	815	0.9705	50
353	0.8506	50	656	0.9394	50
339	0.8448	50	624	0.9323	50
338	0.8447	50	471	0.8923	50
327	0.8400	50	443	0.8834	50
310	0.8321	50	376	0.8601	50
288	0.8218	50	354	0.8510	50
222	0.7845	50	337	0.8440	50
192	0.7633	50	318	0.8357	50
138	0.7162	50	307	0.8307	50
116	0.6912	50	282	0.8181	50
94	0.6600	50	164	0.7405	50
77	0.6320	50	161	0.7379	50
56	0.5852	50	157	0.7350	50
55	0.5836	50	123	0.6990	50
54	0.5802	50	111	0.6840	50
51	0.5699	50	97	0.6651	50
46	0.5568	50	73	0.6233	50
32	0.4996	50	60	0.5959	50
210.9	**0.7244**	**50.0**	**346.5**	**0.8069**	**48.6**

The lineage-based approach is statistically superior (α=0.05) than the traditional approach in terms of the fitness value and the r-squared.

The lineage-based approach is a bit quicker to complete to a solution. This is a modest claim since the lineage approach converged in only one trial.

VIII. DISCUSSION

The initial set of experiments clearly demonstrate the statistical superiority of the *best class* group over the other classes. As the complexity increases over the progression of experiments, the gulf widens between the *best class* parents and the other two classes.

These initial results spawned an applied experiment to assess the feasibility of building GP models which focus on the top 20 percent of a population. The results from the applied experiments demonstrate the superiority of a lineage-based modeling process.

Although the results are superior, there are some ways the lineage approach could be improved. The nature of the lineage-based discards 80 percent of population. The initial generation uses randomly generated equations. One approach under consideration is whether to postpone the lineage approach until about generation 4; this would allow for the population to form distinct boundaries between classes.

A second performance strategy considers the elimination of duplicate chromosomes. Both techniques produce redundant chromosomes (equations). The strategy deployed in both techniques sorts the equations and removes duplicates every 10 generations. Since the first several generations are critical to the success of a particular trial, it may be prudent to eliminate duplicates and replenish the population for every generation.

IX. Conclusions

The initial set of experiments demonstrate that fitness pedigree plays a significant role in producing statistically superior offspring. This first set of experiments lay the groundwork for the second set of experiments which examine lineage-based GP relative to traditional GP models.

This second set of experiments show that a lineage-based GP, which focuses breeding effort on the top 20 percent of the population, produces superior results in less time.

X. Future Directions

This research examines the lineage for one generation back (parents). It may be interesting to explore 2 or more generations back to determine whether the chromosomal relationship decays over subsequent generations. Exploring the degree of decay would help determine the impact of ancestors upon their descendents.

References

[1] Angeline, Peter John, Genetic programming and emergent intelligence. In Kenneth E. Kinnear, Jr., editor, *Advances in Genetic Programming*, chapter 4, Pp. 75-98. MIT Press, 1994.

[2] Angeline, Peter J., Genetic Programming's Continued Evolution. In K. E. Kinnear, Jr., editor, Advances in Genetic Programming Volume 2, Chapter 1, pages 1-20. MIT Press, 1996.

[3] Koza, John. (1997) Genetic Programming. In A. Kent, J.G.Williams, editor, Encyclopedia of Computer Science and Technology.

[4] Langdon, W.B., Banzhaf, W., "Genetic Programming Bloat without Semantics," The 6th International Conference on Parallel Problem Solving from Nature, Paris France, 2000, Pp. 201-210.

[5] McPhee, Nicholas Freitag, Nicholas J. Hopper, "Analysis of Genetic Diversity through Population History," GECCO99: Proceedings of the Genetic and Evolutionary Computation Conference, July 1999.

[6] Burke, E., Gustafson, S., Kendall, G., and N. Krasnogor, "Is Increased Diversity Beneficial in Genetic Programming: An Analysis of the Effects on Fitness," Proceedings of the Congress on Evolutionary Computation, Australia. IEEE Press, 2003, Pp. 1398-1405.

[7] Grefenstette, John J. Incorporating Problem Specific Knowledge into Genetic Algorithms. In L. Davis, editor, Genetic Algorithms and Simulated Annealing, Chapter 4, Pp. 42-60. Morgan Kaufmann Publishers, Inc., 1987.

[8] Whitley, Darrel. (1993) A Genetic Algorithm Tutorial. Technical Report CS-93-103, November 10, 1993. Department of Computer Science, Colorado State University.

[9] Goldberg, David. (1989) Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, USA.