Analysis of genetic diversity through population history

Nicholas Freitag McPhee and **Nicholas J. Hopper**
Division of Science and Mathematics
University of Minnesota, Morris
Morris, MN USA-56267
(320) 589-6300
{mcphee, hoppernj}@mrs.umn.edu
http://www.mrs.umn.edu/{~mcphee, ~hoppernj}

Abstract

The idea that diversity in the population of a genetic algorithm affects the algorithm's search efficiency is widely accepted. However, little is known about the amount of node level diversity present in Genetic Programming (GP) runs. In this paper, we introduce several techniques for measuring the diversity of a population based on the genetic history of the individuals. We then apply these measures to the genetic histories of several runs of four different problems. The results of this analysis show that a surprisingly small amount of diversity is present in the final population of a GP run. We conclude by suggesting a variety of other potential applications of these measures.

1 INTRODUCTION

Progress in evolution depends fundamentally on the existence of variation in a population. Unfortunately, a key problem in many Evolutionary Computation (EC) systems is the loss of diversity through premature convergence. This lack of diversity often leads to stagnation, as the system finds itself trapped in local optima, lacking the genetic diversity needed to escape.

When Genetic Programming (GP) was introduced, many hoped that the use of a (fairly) unrestricted tree structure would reduce this tendency toward premature convergence. Subsequent experience has suggested that while the tree representation certainly has advantages, lack of diversity in GP populations is indeed a significant issue.

1.1 MEASURING GENETIC DIVERSITY

One of the challenges, however, is getting a reasonable measure of diversity. This is especially difficult in GP because the relationship between syntactic diversity (i.e., diversity of tree structures) and semantic diversity (i.e., diversity of behaviors) is typically quite complex. Ultimately one wants diverse behaviors in order to find more fit behaviors. This has to be achieved, however, via manipulation of specific tree structures, and so some sort of syntactic diversity is necessary to obtain the desired semantic diversity.

While diversity was recognized as an issue from early on (e.g., [Koza 1992]), little work has been done on quantifying diversity in GP runs. Our goal in this paper is to develop a set of measures that can be used to quantify various aspects of the syntactic diversity in a GP run. We are specifically interested in syntactic diversity, since its absence has serious implications for semantic diversity. In particular, we're interested in diversity at a node level, which we will call *genetic diversity*. This allows us to track the movement of nodes and subtrees through the population over time, and to determine which nodes and subtrees are shared among individuals.

It's important to note that a lack of genetic diversity (as represented by nodes) doesn't necessarily imply stagnation. As long as every function and terminal occurs at least once in a population, it is theoretically possible to construct any possible tree via, for example, crossover. As a practical matter, however, certain structures are very difficult to generate from a given set of structures and recombination operators. The MAX problem, for example, is very sensitive to the choice of operator at the root node, and once that choice is made it is very hard to change using standard subtree crossover [Gathercole and Ross 1996]. As another example, Langdon and Poli have shown that it is extremely difficult to move from one local maximum to another in the Artificial Ant problem using just point mutation [Langdon and Poli 1998].

Thus, while a lack of genetic diversity doesn't logically imply stagnation, it tends to lead to stagnation in practice. Ultimately, it isn't sufficient for particular operators to exist in the population. Instead specific *nodes* must exist in the population, placing particular operators (such as $*$) in specific locations (such as the root position) relative to other nodes.

Note, also, that convergence is not a bad thing *per se*, as there is an important tradeoff between exploration and exploitation [Holland 1975]. If a run converges strongly on the "correct" answer, for example, that may be seen as a good thing. The problem we're concerned with is *premature* convergence, where a run converges on a non-optimal solution, preventing further exploration.

Our primary interest here is measuring diversity when using just crossover. As a result we limited our recombination to standard GP sub-tree crossover. As is discussed in Section 6, one could extend this work in a variety of ways by exploring the effect of other recombination operators.

In this paper we present several different methods of measuring the actual genetic diversity in GP runs, and apply these measures to multiple runs of four different problems. In each case, we find that the genetic diversity in the populations is surprisingly low, providing further evidence that diversity is indeed a serious problem in GP. As an extreme example, in one run of the Artificial Ant Problem *every* node of *every* individual in the final population came from a *single* individual in the initial population. Thus, while several members of the initial population may have had descendents in the final population, only one of them had any surviving genetic material.

These new measures are also important because they provide new ways to quantitatively measure the impact of GP variants on genetic diversity, as discussed in Section 6.

1.2 REVIEW OF RELATED WORK

Early on, [Koza 1992] presented the notion of *variety* to show that throughout his GP runs, populations contained a significant number of different trees (when considered as S-expressions). This notion (and variations thereof) has subsequently been used by many authors as the *de facto* measure of diversity. Unfortunately, as is discussed in [Rosca 1997], it's not clear how semantically different these trees really were, or to what degree they sampled the space of possible trees (measured either syntactically or semantically). In particular, many syntactically different GP trees can have the same semantics (and therefore the same fitnesses) through mechanisms like symmetry of operators and the existence of introns.

Because of these difficulties Rosca suggests that we instead measure variety with respect to semantic properties such as fitness [Rosca 1997]. These measures provide useful high level views of the state of the population, but don't address the actual diversity of genetic material in the population.

Population size	500, steady state
Selection Method	Tournament selection Tournament size t=10
Maximum Generation	50
Generative Method	Ramped half-and-half, max initial depth 4
Maximum created depth	10
Internal crossover rate	90%
Mutation rate	0
Reproduction rate	0
Random # generator	Java 1.1 `java.util.Random`

Table 1: Common parameters for all runs

While there has been research tracking changes in the size and shape of trees during a run (e.g., [Soule and Foster 1997]), we are not aware of any previous research that tracks diversity through a run at the node level.

2 EXPERIMENTAL SETUP

The measures described in this paper were applied to the results of 20 runs each on a set of four problems: the MAX problem, sequence induction on a quartic polynomial, the Artificial Ant problem with the Santa Fe trail, and symbolic regression on a quartic polynomial. Except where noted, the parameters are those in Table 1.[1]

2.1 FITNESS REPRESENTATION

For all but the MAX problem we used lexical fitness, with the tuple components listed in the respective tables. For these non-MAX problems we implemented a parsimony pressure by setting the last element of the fitness tuple to the number of nodes in the individual (with fewer being better). This always gives preference to an individual with better performance on the problem, but selects the more parsimonious among two individuals with equal performance. To discourage early convergence to very small trees, the size was normalized by setting this component to 0 if it was less than a certain threshold arbitrarily determined for each problem; these thresholds are listed in the parameter tables.

For the two regression problems (sequence induction and quartic polynomial), we limit the total error accumulated on the set of fitness cases to a maximum of 10^{10}. We also set the total error to that maximum value if some test case yields a NaN value (by, e.g., dividing by 0).

[1] The parameters described here were not tuned for optimal performance on the test problems; they simply provide a convenient reference point.

Function set	$\{+, *, -\}$
Terminal set	$\{j, 0, 1, 2, 3\}$
Fitness cases	$j \in \{0, 1, 2, ...20\}$
Fitness measure	(difference, hits, size)
Size normalization threshold	100

Table 2: Sequence Induction parameters

Function set	{`if-food-ahead`, `begin2`}
Terminal set	{`(move)`, `(turn-left)`, `(turn-right)`}
Fitness measure	(food eaten, size)
Size normalization threshold	30

Table 3: Artificial Ant parameters

2.2 THE MAX PROBLEM

The MAX problem was first described in [Gathercole and Ross 1996]. It consists of the set of problems MAX-depth-D-{Function Set}-{Terminal Set} where the depth is restricted to D and the goal is to maximize the tree value. In these experiments, the MAX-depth-7-{*,+}-{0.5} problem was used. The ramped half-and-half generative method was used with the maximum initial depth set to 5.

2.3 SEQUENCE INDUCTION

The sequence induction problem is an instance of the symbolic regression problem in which the domain of the independent variable is constrained to the natural numbers {0,1,2,3...}. We used the sequence $S_j = 5j^4 + 4j^3 + 3j^2 + 2j + 1$, as described in [Koza 1992]. The fitness measure used was the tuple `(difference, hits, size)`. `difference` was the sum of the absolute values of the difference between the individual and the target over the cases $j \in \{0, 1, 2, ...20\}$. `hits` represented the number of cases where the individual's value exactly matched the value of the target sequence. Other parameters for the sequence induction problem are listed in Table 2.

2.4 THE ARTIFICIAL ANT PROBLEM

The Artificial Ant problem evolves a program which navigates a near-sighted and motion-constrained artificial ant along an irregular trail of food [Koza 1992]. There are several well-known trails used for this problem; the experiments described here use the Santa Fe trail, which lies on a 32×32 toroidal grid and contains 89 pieces of food. The ant is constrained to 400 moves, where a move is one of turning left, turning right, or moving straight ahead. The available non-terminals were `if-food-ahead` and `begin2`. `if-food-ahead` takes two arguments and evaluates the first if there is food ahead and the second otherwise. `begin2` takes two arguments and evaluates them sequentially. The terminals were the side-effecting thunks `(move)`, `(turn-right)`, and `(turn-left)`.

Function set	{`+, -, *, /, sin, cos, exp, log`}
Terminal set	{`x`}
Fitness cases	20 evenly spaced points from the interval [-5, 5)
Fitness measure	(difference, hits, size)
Size normalization threshold	100

Table 4: Symbolic regression with quartic polynomial as target

2.5 QUARTIC POLYNOMIAL

The quartic polynomial $x^4 + x^3 + x^2 + x$ [Koza 1992] was used as the target for a symbolic regression problem. The function set was {`+, -, *, /, sin, cos, exp, log`}.[2] The terminal set consisted of `x`, the independent variable. Fitness was the ordered triple `(difference, hits, size)`. `difference` was the sum of the absolute values of the differences between the actual value at each fitness case and the value of the individual at that point. `hits` was the number of fitness cases in which the difference between the actual value and the individual's value was less than 0.05. Other parameters for the Quartic Polynomial problem are listed in Table 4.

3 NODES IN THE FINAL POPULATION

The simplest of our measures merely measure the number of different nodes in the population. In crossover, some nodes are copied untouched (i.e., the copy has the same children as the original), while other nodes have one of their children modified when they are copied. To be able to distinguish between these two cases, we introduce the notions of root and non-root parent.

In standard GP crossover, an individual has two parents. One of these, which we will call the *root parent*, contributes the rooted portion of the tree. The other, which we will call the *non-root parent*, contributes a subtree that is swapped into the root parent. Figure 3, for example, depicts a tree resulting from crossover between the trees in Figures 1 and 2.

[2] Instead of using protected division and log, we take advantage of the fact that Java generates a NaN (Not a Number) value in the cases of division by zero and $\log 0$. We then normalize these NaN values to some very poor fitness (e.g., the maximum error).

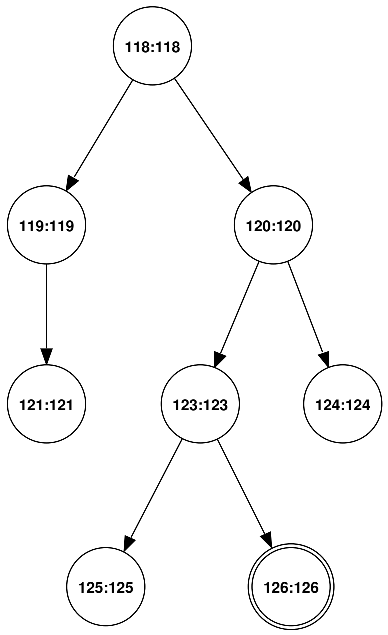

Figure 1: The root parent for an example of genetic and memory IDs. The labeling scheme is `ID:memID`, with a double outline around the crossover points

In the case where the crossover point is the root node, the child gets its root (and indeed its entire structure) from the other parent, and we call that parent the root parent.

Given these definitions, we note that the only new nodes constructed in crossover are along the path from the root of the root parent to the crossover point (indicated by a dashed line in Figure 3). All other nodes are copied unchanged. To distinguish these two cases we give every node two labels (`ID` and `memID`). Every node generated in the initial population is given a unique value which is used for both `ID` and `memID` for that node. When we perform crossover, any node along the path from the root of the root parent to the crossover point in the new individual is given the same `ID` as the corresponding node in the root parent since the two nodes share a common ancestry. Since these nodes represented new nodes (in the sense they have different children), every newly constructed node receives a new, unique `memID`. (See the labels in Figures 1, 2, and 3 for an example of the process.) This scheme allows us to check two nodes for common ancestry by comparing their `IDs` and to compare their `memIDs` to determine whether two nodes are indeed two copies of the same node.

Counting the number of distinct ID (either `ID` or `memID`) values present in the final population provides an indication of the amount of genetic material present at the end of a run. As Table 5 indicates, these counts are surprisingly small.

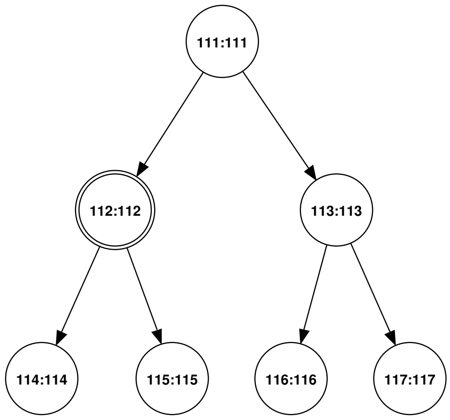

Figure 2: The non-root parent for an example of genetic and memory IDs. The labeling scheme is `ID:memID`, with a double outline around the crossover point.

	N	`IDs`		`memIDs`	
Problem		Avg.	σ	Avg.	σ
MAX	20	27.05	5.78	3374	349
SeqInd	20	33.65	11.16	2096	580
Ants	20	18.30	8.82	1879	252
Quartic	20	16.90	4.44	1852	325
All	80	23.98	10.41	2300	741

Table 5: Number of runs (N), average (Avg) and standard deviation (σ) of genetically distinct nodes and distinct nodes in memory in the final population for each problem.

In fact, the average number of genetically distinct nodes in the final population over all runs collected for this paper was only 24, or *less than 1%* of the estimated 3128 distinct nodes in the initial population. That this happened after just 50 generations represents a dramatic loss of diversity in relatively few generations.

4 INDIVIDUALS CONTRIBUTING GENETICALLY

By tracking both parents of every individual, we can construct the set of individuals from the initial population having any descendents in the final population. However, it is possible for one individual to be a descendent of another individual without sharing any genetic material with that ancestor, through a sequence of subtree swaps in the intermediate lineage which eliminate the ancestor's contribution. Thus an indicator of the genetic diversity in the final population is the number of individuals in the initial population contributing genetic material to the final population. If this number is relatively small, this indicates that the GP

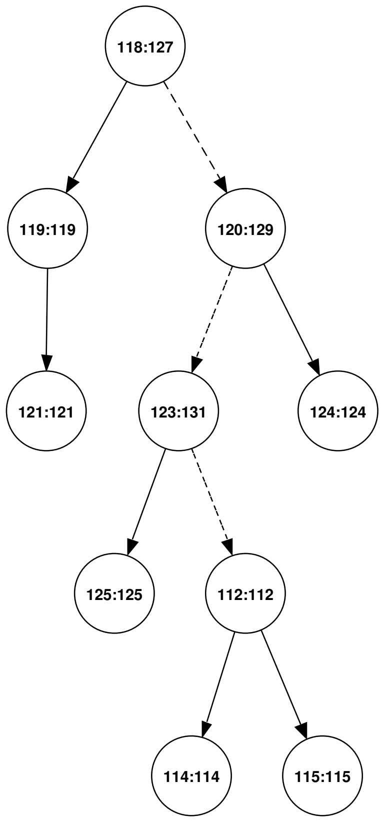

Figure 3: The child resulting from crossover of the trees in Figures 1 and 2. The dashed line represents the path from the root to the crossover point. The nodes along that path (except for the crossover point) retain their original `IDs`, but are assigned new `memIDs`.

algorithm is spending most of its time permuting a small number of (not necessarily optimal) individuals rather than searching the space of possible programs.

By tracking both the root and non-root parents of every individual, we traced each individual in the final population back through all its ancestors to create the set S of ancestors from the initial population for each run. Using the set G of genetically distinct nodes (`IDs`) computed in Section 3, we were able to determine for each individual in S whether it contributed any genetic material to the final population (by searching the tree for `IDs` found in the set G). The number of individuals in S contributing nodes to G indicates how many of the initial individuals the algorithm is still manipulating; as Table 6 indicates, these numbers are surprisingly small. For all runs, the average number of individuals contributing genetic material to the final populations was less than 7. In one run of the artificial ant problem, *a single individual* from the initial population of 500 individuals accounted for *all* of the genetic material

Problem	N	Avg.	σ
MAX	20	5.10	1.33
SeqInd	20	7.80	2.59
Ant	20	5.05	2.56
Quartic	20	6.40	1.76
All	80	6.09	2.38

Table 6: Number of runs (N), Average number of initial individuals contributing genetic material to the final population (Avg), and standard deviation (σ) for each problem.

present in the 50th generation, indicating that GP was no longer taking advantage of the diversity present in the initial population. This one individual was sufficient to ensure that the final population contained instances of all the terminals and non-terminals, but since the best individual in the run was non-optimal, it would appear that other sorts of diversity (such as structural diversity) were lacking.

5 EVE: ANALYSIS OF THE PARENT RELATION

Several researchers have noted an asymmetry between the root portions of GP trees and the portions nearer the leaves [Gathercole and Ross 1996, Rosca 1997, Poli and Langdon 1998, McPhee *et al.* 1998]. In particular, there is a marked tendency for populations to converge quickly on a particular root structure, which is then used by the entire population throughout the remainder of the run. In this analysis we quantify certain aspects of this convergence, finding that the population converges *very* strongly over a substantial rooted section of the trees.

5.1 DEFINITION OF *EVE*

As mentioned in Section 3, every individual has both a root and a non-root parent, and as a consequence every individual in the final population has a linear sequence of root ancestors that leads back to a single root ancestor in the initial population. One question, then, is how many individuals from the initial population are root ancestors of some member of the final population. Perhaps surprisingly, in 72 of our 80 runs there was a *single* root ancestor from which *every* individual in the final population descended. This means that all 500 root parent lineages from the final population eventually converge and then follow up together in a single lineage to the initial population.

Each of the individuals from the point of convergence to the initial population are then root ancestors of every member of the final population. In runs where there is such a single root ancestor in the initial population (i.e., these lineages do converge), we call the common root ancestor at the point of convergence *Eve*. Eve is then the latest individual (in the evolutionary history) that is a common root ancestor of every individual in the population.

5.2 MEASURING COMMONALITY WITH EVES

Given that 90% of our runs had an Eve, the question arises of how much material is shared between these Eves and the individuals in the final population. In order to determine how extensive Eve's influence was, we compared Eve to every individual in the final population.

For a given individual t in the final population, we compared Eve to t starting with the root node. To simplify the presentation we use the standard heap numbering scheme for binary trees, where the root node of a tree is labelled 1, and the left and right children of a node labelled n are labelled $2 * n$ and $2 * n + 1$, respectively.[3] Given this labelling scheme, we use the following algorithm to quantify the degree to which the rooted portion of Eve is duplicated in an individual t in the final population. Here `compare`(n, t, Eve) compares t and *Eve* in a depth-first fashion starting at node number n, and updates `count` as long as node IDs match.

```
Procedure compare(n, t, Eve)
if ID (t, n) = ID (Eve, n)
    increment count[n]
    compare(2 * n, t, Eve).
    compare(2 * n + 1, t, Eve).
fi.
```

The results of these calculations on the runs with a single Eve are summarized up to depth four in Table 7. These results highlight the extreme lack of diversity at and near the root node in the final populations. Substantial portions of the population (mostly greater than 70%, and greater than 60% in all cases) share the *top four levels* of their tree with Eve, in a system which constrains the trees to a maximum depth of 10; in fact, well over half the population even share their fifth level nodes with Eve. These numbers suggest that it is *enormously* difficult for a population to change near the root in the later generations of a run.

Also alarming is the speed with which the root appears to converge. Among the runs in which there was a single Eve, on average that Eve was the 8102st individual created in the run (standard deviation 6726), indicating that the root structure for the entire population was decided on average by the 16th generation. In one run of the Artificial Ant problem, Eve was the second individual created after the initial population, so that the root structure of the majority of the population was essentially unchanged for the duration of the run. Fixing this root structure at such an early stage likely makes it difficult for the run to escape local optima in later generations; the phenomenon also seems to reduce GP to essentially blind random search of the permutations at depths below 5 for the remainder of the run.

[3]Note that the inclusion of unary operators in the Quartic Polynomial problem causes some numbers to be skipped in this labelling scheme.

	MAX		Seq Ind		Ants		Quartic	
N	**E**	**O**	**E**	**O**	**E**	**O**	**E**	**O**
1	20	499	20	497	18	496	14	497
2	20	498	19	484	16	429	12	446
3	20	499	20	484	18	477	14	486
4	20	496	18	432	13	322	11	414
5	20	498	19	460	13	390	9	436
6	20	498	20	477	17	443	12	409
7	20	498	19	454	17	437	12	394
8	20	488	16	403	8	319	8	394
9	20	494	15	401	7	318	7	382
10	20	495	18	404	7	437	5	450
11	20	496	18	424	8	427	6	455
12	20	496	16	371	13	437	9	349
13	20	494	17	461	14	434	9	360
14	20	492	14	406	12	421	9	301
15	20	496	15	448	14	423	8	334

Table 7: Number of runs containing Eves, and number of individuals sharing nodes with those Eves for four problems. The **N** column is the node number as described in the text (**N**=1 is the root node, **N**=2 or 3 is the second level, 4-7 are the third level, etc.). The **E** column is the number of runs (out of 20) having a single Eve whose structure from the root to node **N** has been passed unchanged to at least one individual in the final population. The **O** column is the average number of individuals (out of 500) which share this structure from the root to node **N**.

6 CONCLUSIONS: MAJOR LOSS OF DIVERSITY

The data presented in the preceding sections clearly indicates that, at least on these four test problems, the loss of diversity is typically both rapid and dramatic.

- In Section 3 we found that less than 1% of the nodes generated in the initial population survive to the final population.
- In Section 4 we found that on average less than seven out of the 500 individuals from the initial population contributed *any* genetic material to the final population. In one extreme example, we found that only *one*

individual from the initial population contributed any genetic material to the final population.

- In Section 5 we found that in 90% of our runs, there was a single individual (Eve) who was the root ancestor of every individual in the final population. Further, we found that the majority of the final population typically shared as much as the top five levels (out of a rarely reached maximum of 10) with Eve.

These results clearly show strong convergence, but is it truly *premature* convergence? A quick look at the fitnesses of the best individuals from the runs suggests strongly that for the most part this convergence was indeed premature. In 80 runs, only 15 actually found an optimal solution, over half of which (8) were from the much easier MAX problem.

In short, standard GP with basic subtree crossover as its only recombination operator appears unable to take advantage of more than a tiny fraction of the genetic material present in the initial population.[4]

In light of the data presented in the preceding sections, there are several important questions to be answered about diversity and GP: What causes this lack of diversity? What impact does this have on performance? How might these measures be used to improve the situation?

6.1 CAUSES OF DIVERSITY LOSS

It's unclear why we're seeing such a dramatic loss in diversity across these 80 runs, but the most likely culprits seems to be hitchhiking and genetic drift.[5]

The key idea in hitchhiking is that not all loci for genetic material have the same significance. Thus individuals which have the "correct" genetic material in the most significant loci tend to do well, especially in the early stages of a run when significant variation in fitness exists. The benefit of this is that the population is quickly dominated by individuals with "correct" material in those significant loci. The problem, though, is that *all* the material from those individuals tends to spread through the population, including "incorrect" material from less significant loci.

In the MAX problem, for example, imagine an individual from the initial population with a large number of multiplications near the root. Such an individual is likely to do well, and its root structure could easily spread through the population. If, however, it had a single "incorrect" addition near the root, this would also spread, even though it is not optimal. Worse, once it had spread, the dynamics of the problem would make the "mistake" extremely difficult to fix.

It may well be that this sort of hitchhiking is one of the main causes of the observed lack of diversity. One way to test this conjecture would be to explore the effect of different selection pressures on genetic diversity (by, e.g., changing the tournament sizes). Another approach would be to study the effects of altering the tree representation or the set of recombination operators in an effort to alter the effects of hitchhiking (e.g., AppGP [McPhee *et al.* 1998]) or kinds of hitchhiking (e.g., Stack-Based GP [Perkis 1994]).

Another likely culprit here is genetic drift, since the size of the set of different nodes in the population is non-increasing. As a simple example, the initial population of a Quartic Polynomial run will contain a large number of `X` nodes, since `X` is the only terminal in that problem. Over time, however, the number of *different* `X`'s will tend to decrease, since once the last copy of a given `X` is lost, there is no way to retrieve it using just crossover.

The measures of the number of different nodes in the final population (Section 3) would seem to be particularly affected by this. Two runs of the Quartic Polynomial problem, for example, had only 10 distinct nodes in the final population. In each case four of the eight different functions had been lost, and in each case the large number of `X` nodes in the initial population (well over 1,000) had been reduced to five different surviving `X` nodes. While hitchhiking probably causes much of the initial loss (`X`s attached to successful root sections spread, while `X`s attached to unsuccessful root sections are eliminated), drift probably continues to whittle this number down as `X`s are replaced by other `X`s via crossover.

6.2 IMPACT OF DIVERSITY ON PERFORMANCE

As mentioned in Section 1.1, this loss of diversity doesn't logically necessitate stagnation, but as a practical matter it does appear to have a variety of negative effects. The Eve measurements, for example, strongly suggest that the higher levels of GP trees tend to become fixed fairly early, and that as the run progresses more and more levels become fixed. The stability of these "fixed" sections implies that rarely is crossover in the fixed section of the tree successful. Since this fixed section tends to grow downward with time, successful crossovers will be increasingly restricted to swapping small subtrees. In the extreme case this reduces GP to a blind search where, given the fixed rooted structure, GP attempts to discover the "best" set of small subtrees (including leaves) to graft onto that structure by

[4] Note also that it is possible (if not likely) that the presence of introns could cause some of our measures to in some cases *over* estimate the diversity of *functional* nodes.

[5] One anonymous reviewer pointed out that our results resemble those obtained in biology using coalescent theory. While we haven't had time to fully explore the connection, it appears that coalescent theory may indeed prove valuable in better understanding these results, especially in terms of genetic drift.

randomly shuffling small subtrees around the population.

It seems plausible that this loss of diversity corresponds to the common tendency for fitnesses in a run to flatten out. The early improvements in fitness likely correspond to the part of the run where the population is still fairly diverse. The flattening out of the fitnesses in the later parts of the run would then result from a loss of diversity in the population, and the corresponding reduction of GP to something like a blind search mechanism. Further research would, however, be necessary to confirm this hypothesis.

6.3 APPLICATIONS OF THESE MEASURES

There have been many techniques proposed for improving the performance of GP, and many have specifically claimed improved diversity as an advantage. If diversity is measured empirically, however, it is typically in terms of variety (an extremely coarse grained measure) or fitness (which cannot address the structural specifics of a run). One potentially valuable application of these measures would be to quantify the effect these GP variants have on genetic diversity.

In the following we discuss several changes that have been proposed specifically to address the issue of diversity, or which we think might have a significant effect on genetic diversity.

6.3.1 Alter the population size

One possibility is that the results we present are artifacts of our choice of population size (500 individuals), and it would perhaps be useful to repeat these experiments with a variety of different population sizes. Preliminary results, based on five runs of Quartic Polynomial with 1000 individuals, suggest that doubling the population size does not lead to corresponding increases in diversity. For example, the average number of individuals from the initial population contributing genetic material to the final population was 9.0 (compared to 6.4 for our runs with 500 individuals). As a percentage of the number of individuals in the initial population, this is in fact smaller than for the runs with 500 individuals.

6.3.2 Alter the parsimony pressure

It has been suggested that introns might be a source of genetic diversity in GP runs (e.g., [Angeline 1994]). Since we used a parsimony pressure in these experiments (which would tend to reduce the number and size of introns), it's possible that this parsimony pressure is partly responsible for the reported loss of diversity. As a preliminary attempt to asses this effect, we ran five runs of Quartic Polynomial without parsimony pressure and found that there was no statistically significant difference between these results and those with parsimony pressure.

6.3.3 Alter the selection pressure

As mentioned above, selection pressure affects convergence rates, and could well affect the genetic diversity.

6.3.4 Add mutation

In these experiments we restricted recombination to standard subtree crossover. Adding mutation (as described, e.g., in [O'Reilly and Oppacher 1996]) would no doubt change many of these results. Mutation, for example, can make it possible for a node that had disappeared to reappear later in the run. The diversity measures presented here should be useful in assessing the actual impact of different mutation schemes. As an example, mutation that simply replaces a subtree with a new, randomly generated subtree would be unlikely to successfully alter the common rooted section of the tree. Point mutation, on the other hand, might be better able to "fix" nodes near that root that had converged to "incorrect" choices.

6.3.5 Decimation

One approach to periodically increasing the diversity during a run is decimation [Koza 1992], where a substantial percentage of the population is eliminated (either at regular intervals or in response to a lack of progress) and replaced by new, randomly generated individuals. While this would clearly lead to large jumps in some of our diversity measures, it's possible that these jumps would in fact be quite temporary and have little lasting effect. In the case of the Eve measures, for example, it seems unlikely that a large group of randomly generated (and presumably not terribly fit) individuals would be able to alter the previously "settled" root structure.

6.3.6 Demes

It seems likely that the use of separate sub-populations (demes) with limited interaction between demes would affect the diversity as measured here. In particular, the use of demes might better enable GP to explore a variety of possible root structures. Each deme would presumably settle initially on a single root structure, but the sharing between demes would introduce individuals with high fitnesses but different root structures. If the fitnesses of the different individuals were close, both alternatives would have an opportunity to persist long enough to perhaps trade some useful genetic material. This differs from decimation, where the fitnesses of the newly created individuals are likely to be much lower than those of the survivors.

6.3.7 Different GP representations and recombination operators

Over the years a wide variety of different GP representations and recombination operators have been proposed, several of which specifically claim to address diversity issues. The measures presented here would be a useful tool for assessing these claims and, more generally, providing a quantitative measure of the impact of these variants on genetic diversity.

6.4 IN SUMMARY…

In this paper we presented a collection of measures that allow us to quantify various aspects of node level genetic diversity in a GP population and its genetic history. Applying these measures to the results of a number of runs on several test problems clearly shows a profound loss of diversity over the course of the runs, indicating that standard GP with basic subtree crossover as its only recombination operator is unable to take advantage of more than a tiny fraction of the genetic material present in the initial population. These measures should also prove useful in helping compare the effects the many proposed GP variants have on population diversity.

Acknowledgments

Hopper's work was supported by the University of Minnesota Undergraduate Research Opportunities Program. A grant from the University of Minnesota Graduate School Grant-in-aid program funded the purchase of computing equipment used in these experiments.

We are grateful for the helpful comments of several anonymous reviewers.

References

Angeline, Peter John (1994). Genetic programming and emergent intelligence. In: *Advances in Genetic Programming* (Kenneth E. Kinnear, Jr., Ed.). Chap. 4, pp. 75–98. MIT Press.

Gathercole, Chris and Peter Ross (1996). An adverse interaction between crossover and restricted tree depth in genetic programming. In: *Genetic Programming 1996: Proceedings of the First Annual Conference* (John R. Koza et al., Eds.). MIT Press. Stanford University, CA, USA. pp. 291–296.

Holland, John H. (1975). *Adaptation in natural and artificial systems*. University of Michigan Press. Ann Arbor, Michigan, USA.

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

Langdon, W. B. and R. Poli (1998). Why ants are hard. In: *Genetic Programming 1998: Proceedings of the Third Annual Conference* (John R. Koza et al., Eds.). Morgan Kaufmann. University of Wisconsin, Madison, Wisconsin, USA. pp. 193–201.

McPhee, Nicholas Freitag, Nicholas J. Hopper and Mitchell L. Reierson (1998). Impact of types on essentially typeless problems in GP. In: *Genetic Programming 1998: Proceedings of the Third Annual Conference* (John R. Koza et al., Eds.). Morgan Kaufmann. University of Wisconsin, Madison, Wisconsin, USA. pp. 232–240.

O'Reilly, Una-May and Franz Oppacher (1996). A comparative analysis of GP. In: *Advances in Genetic Programming 2* (Peter J. Angeline and K. E. Kinnear, Jr., Eds.). Chap. 2, pp. 23–44. MIT Press. Cambridge, MA, USA.

Perkis, Tim (1994). Stack-based genetic programming. In: *Proceedings of the 1994 IEEE World Congress on Computational Intelligence*. Vol. 1. IEEE Press. Orlando, Florida, USA. pp. 148–153.

Poli, Riccardo and William B. Langdon (1998). On the search properties of different crossover operators in genetic programming. In: *Genetic Programming 1998: Proceedings of the Third Annual Conference* (John R. Koza et al., Eds.). Morgan Kaufmann. University of Wisconsin, Madison, Wisconsin, USA. pp. 293–301.

Rosca, Justinian P. (1997). Hierarchical Learning with Procedural Abstraction Mechanisms. PhD thesis. University of Rochester. Rochester, NY 14627.

Soule, Terence and James A. Foster (1997). Code size and depth flows in genetic programming. In: *Genetic Programming 1997: Proceedings of the Second Annual Conference* (John R. Koza et al., Eds.). Morgan Kaufmann. Stanford University, CA, USA. pp. 313–320.