Under the Hood of Grammatical Evolution.

Michael O' Neill
Dept. of Computer Science And Information Systems
University of Limerick
Ireland
Michael.ONeill@ul.ie
Phone: +353-61-202745

Conor Ryan
Dept. of Computer Science And Information Systems
University of Limerick
Ireland
Conor.Ryan@ul.ie
Phone: +353-61-202755

ABSTRACT

Grammatical Evolution (GE) is a grammar based GA to generate computer programs which has been shown to be comparable with GP when applied to a diverse array of problems. GE has the distinction that its input is a BNF, which permits it to generate programs in any language, of arbitrary complexity, including loops, multiple line functions etc. Part of the power of GE is that it is closer to natural DNA than GP, and thus can benefit from natural phenomena such as a separation of search and solution spaces through a genotype to phenotype mapping, and a genetic code degeneracy which can give rise to silent mutations(Mutations that have no effect on the phenotype).

We have previously shown how runs of GE are competitive with GP, and in this paper we analyse characteristics such as genotypic diversity, and individual genotypic length, in an attempt to shed light on the power of the system. Results indicate that GE can use certain features of the system to its benefit if and when necessary.

1 Introduction

Grammatical Evolution (GE) is a grammar based, variable length, linear genome system which is capable of generating code in any language. Rather than the functions and terminals associated with GP, GE takes a BNF specification of a language, or subset thereof, from which it can subsequently generate compilable code. In order to produce code using the BNF a genotype to phenotype mapping process is employed which produces a program from the genotypic binary string.

GE has been successfully applied to a number of diverse problem domains such as, Symbolic Regression, Finding Trigonometric Identities, and the Santa Fe Trail [Ryan et al. 98a] [Ryan et al. 98b] [Ryan, O'Neill 98] [O'Neill, Ryan 99]. The results compared favorably with systems such as GP, and in the case of the Santa Fe Trail problem have been shown to outperform GP [O'Neill, Ryan 99].

This paper analyses some of the distinctive features of GE, which draw inspiration from molecular biology, to examine their effect on the system, and how the system as a whole may benefit from the modeling of further biological principles. Features such as the extent of wrapping, which involves re-using parts of the genome which have already being expressed, actual genome length in comparison with the effective genome length, and the genotypic diversity maintained during a run will be analysed. Before a description of the problem domains tackled, and results obtained thereon, a brief description of GE follows.

2 Grammatical Evolution

When tackling any problem with GE a suitable BNF definition must first be decided upon. The BNF can be either the specification of an entire language, or perhaps more usefully a subset of a language geared towards the problem at hand. For example, the BNF for the Santa Fe Ant Trail is as follows;

```
N = {<code>,<line>,<if-statement>,
       <op>,<if-true>,<if-false>}

T = {left(),right(),move(),food_ahead(),
       else,if,{,},(,),;}

S = <code>
```

And P can be represented as:

```
(1) <code> :: =  <line>                (A)
                |<code><line>          (B)

(2) <line> :: =  <if-statement>        (A)
                |<op>                  (B)
```

```
(3) <if-statement> :: =
     if(food_ahead()){<line>} else{<line>}

(4) <op> :: =    left();              (A)
               | right();             (B)
               | move();              (C)
```

where the operations, $left()$, $right()$, $move()$, and $food\_ahead()$, are all functions written in the C programming language, and N is the set of non-terminals, T, the set of terminals, P, a set of production rules that map the elements of N to T, and S is a start symbol which is a member of N.

The genotype is then used to map the start symbol onto terminals by reading codons of 8 bits to generate a corresponding integer value, from which an appropriate production rule is selected. A rule is selected by using the following, (Integer Codon Value) MOD (Number of Production Rules for the current non-terminal). Considering rule #5 from above, i.e. given the non-terminal $< op >$ there are three production rules to select from. If we assume the codon being read produces the integer 6, then $6\ MOD\ 3 = 0$ would select rule $(A)\ left()$. Each time a production rule has to be selected to map from a non-terminal, another codon is read, and in this way, the system traverses the genome.

During the genotype to phenotype mapping process it is possible for individuals to run out of codons, and in this case we wrap the individual, and reuse the codons. This is quite an unusual approach in EA's, as it is entirely possible for certain codons to be used two or more times. This technique of wrapping the individual draws inspiration from the gene overlapping phenomenon which has been observed in many organisms in nature [Elseth 95]. In GE, each time the same codon is expressed it will always generate the same integer value but, depending on the current non-terminal, may have a different effect, that is, it may select a different production rule. What is crucial, however, is that each time a particular individual is mapped from its genotype to its phenotype, the same output is generated. This is because the same choices are made each time. It is possible that an incomplete mapping could occur, even after wrapping, and in this event the individual in question is given the lowest fitness value possible, then the selection and replacement mechanisms should operate accordingly to increase the likelihood that this individual is removed from the population. An incomplete mapping could arise if the integer codon values expressed by the genotype were applying the same production rules over and over. For example given an individual with three codons, if the first codon specified rule B from below,

```
(1) <code> :: =  <line>               (A)
                |<code><line>         (B)
```

and the second, and third also specified this same rule, even after wrapping the mapping process would be incomplete and would carry on indefinitely unless stopped. Those individuals that do not undergo a complete mapping after a predetermined time are given the lowest possible fitness value, and in order to reduce the number of invalid individuals being passed from generation to generation a Steady State replacement mechanism is employed. A consequence of the Steady State method is its tendency to maintain fit individuals at the expense of less fit, and in particular, invalid individuals.

An overview of how GE operates in comparison with a biological genetic system can be seen in Figure 1. It can be seen that genes are expressed as proteins which produce a phenotype. More than one protein may be responsible for producing any one phenotype, and a similar situation occurs in GE. Here a codon results in the selection of rules which map non-terminals onto either more non-terminals, or terminals, it is a combination of these terminals and non-terminals that produce the phenotype, or program.

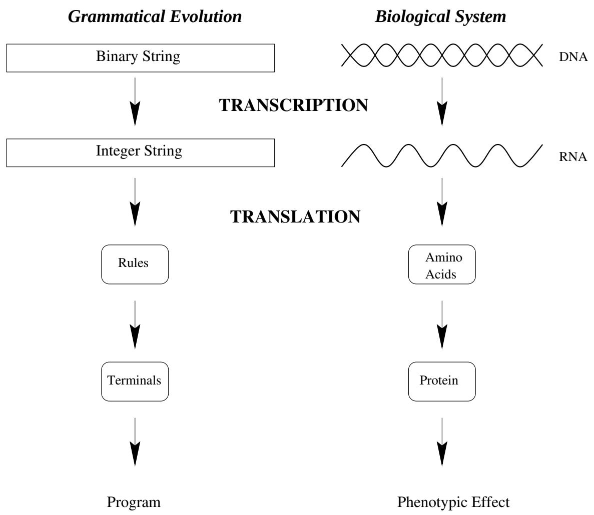

Figure 1 A comparison between Grammatical Evolution and Biological System.

3 The Problem Spaces

In order to obtain useful data to allow an effective analysis of GE, two problem domains were tackled at which GE was previously found to be successful, namely a Symbolic Regression problem [Ryan et al. 98a], and the Santa Fe Ant Trail [O'Neill, Ryan 99]. A brief description of these problem domains follows.

3.1 Symbolic Regression

GE was applied to a symbolic regression problem summarised in Table 1, which had been previously solved in [Ryan et al. 98a]. The system is given a set of input and output pairs, and must determine the function that maps one onto the other. The particular function examined is

$$f(X) = X^4 + X^3 + X^2 + X$$

with the input values in the range $[-1.. + 1]$. As the target function has no constant values, only the variable X, it was decided to use only one terminal operand, i.e. X. So as not to unduly bias the system towards the correct solution, the terminal operators set consisted of more than the binary addition and multiplication operators, which alone would be sufficient to reach the target function. To determine the fitness of an individual program 20 test points were applied in the input range, and the fitness was taken as the sum of the error.

Objective :	Find a function of one independent variable and one dependent variable,in symbolic form that fits a given sample of 20 (x_i, y_i) data points, where the target function is the quartic polynomial $X^4 + X^3 + X^2 + X$
Terminal Operands:	X (the independent variable)
Terminal Operators	The binary operators $+, *,$ $-$ and, the unary operators Sin, Cos, Exp and Log
Fitness cases	The given sample of 20 data points in the interval $[-1, +1]$
Raw Fitness	The sum, taken over the 20 fitness cases, of the error
Standardised Fitness	Same as raw fitness
Hits	The number of fitness cases for which the error is less than 0.01
Wrapper	Standard productions to generate C functions
Parameters	M = 500, G = 21

Table 1 Grammatical Evolution Tableau for Symbolic Regression

3.2 Santa Fe Ant Trail

The Santa Fe Ant Trail is a standard problem tackled in the area of Genetic Programming, and can be considered a deceptive planning problem with many local and global optima [Langdon, Poli 98]. GE has been previously shown to outperform GP on this problem [O'Neill, Ryan 99]. The objective is to find a computer program to control an artificial ant so that it can find all 89 pieces of food located on a non-continuous trail within a specified number of time steps. The code evolved is then executed in a loop until the number of time steps allowed has elapsed. The ant can only turn left, right, move forward one square, and may also look ahead one square in the direction it's facing to determine if that square contains a piece of food. The actions left, right, and move each take one time step to execute.

While there are many possible fitness cases to the Santa Fe trail only one case was taken for the purposes of this experiment. The ant started in the top left hand corner of the grid facing the first piece of food on the trail. A summary of the problem can be seen in Table 2.

Objective :	Find a computer program to control an artificial ant so that it can find all 89 pieces of food located on the Santa Fe Trail.
Terminal Operators:	left(), right(), move(), food_ahead()
Fitness cases	One fitness case
Raw Fitness	Number of pieces of food before the ant times out with 615 operations.
Standardised Fitness	Total number of pieces of food less the raw fitness.
Hits	Same as raw fitness.
Wrapper	Standard productions to generate C functions
Parameters	M = 500, G = 21

Table 2 Grammatical Evolution Tableau for the Santa Fe Trail

4 Results

In order to establish the effects that wrapping may be having on the system, two sets of experiments were carried out where the wrapping feature was turned on and off. Twenty runs for each experiment set on both problem domains were carried out, and the results produced now follow.

4.1 Symbolic Regression

Cumulative frequency measures in the case where wrapping is turned on and off can be seen in Figure 2. In the case where wrapping is disabled, there is a slight increase in performance of the system on this problem.

The average number of individuals undergoing wrapping at each generation can be seen in Figure 4.

A genotypic diversity measure which we have termed the *mean variety*, was carried out for the twenty runs. This measure was obtained by calculating the average of the variances at each bit locus on the genome, and is given by the following formula:

$$Mean\ Variety = \frac{\sum_{1}^{L} \frac{\sum_{i=1}^{N} (Xi-\mu)^2}{N}}{L}$$

Where Xi is the locus value at position i, μ is the mean of Xi at locus i, N is the sample population size, and L is the number of loci analysed.

With a population size of 500 this meant that the greater the variance in a population the closer the mean variety measure is to 0.25. The aim of this measure is to attempt to establish how different the individual genotypes in any given population are, see Figure 8 for these results. Another measure of variety has also been given in Figure 7, which is simply the number of unique individuals in a population, and can be used to some extent to illustrate the genetic diversity within a population[Langdon 98]. From these results it would appear that wrapping is having very little effect on the number of unique individuals, with the mean variety measure decreasing when wrapping is absent.

A comparison of the average actual genome lengths to the average effective genome lengths is given in Figure 3. The effective genome length is a measure of the number of expressed genes during the genotype to phenotype mapping process, the actual length being the number of genes on the chromosome. The actual length is on average longer than the effective length for this problem, and when wrapping is turned off the actual genome lengths increase significantly.

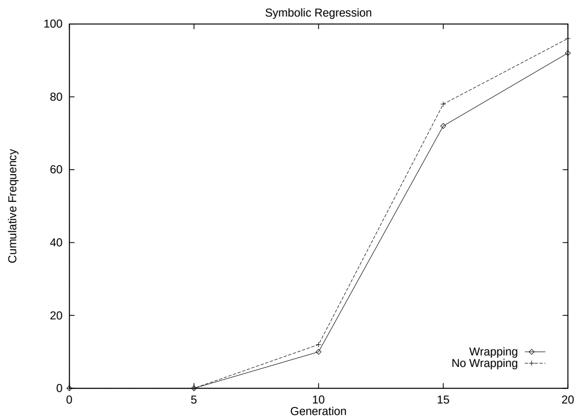

Figure 2 Cumulative frequency measures for the Symbolic Regression problem.

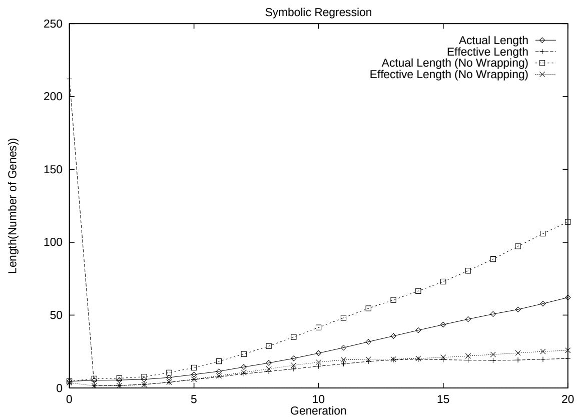

Figure 3 Actual versus Effective Genome Length for Symbolic Regression.

4.2 Santa Fe Ant Trail

Cumulative frequency measures in the case where wrapping is turned on and off can be seen in Figure 5. In the case where wrapping is disabled there are no successful individuals created within the 20 generations examined.

The average number of individuals undergoing wrapping at each generation is significantly greater than for the Symbolic Regression problem, see Figure 4.

The mean variety measure described earlier is given for this problem in Figure 8, and the number of unique individuals at each generation can be seen in Figure 7. The number of

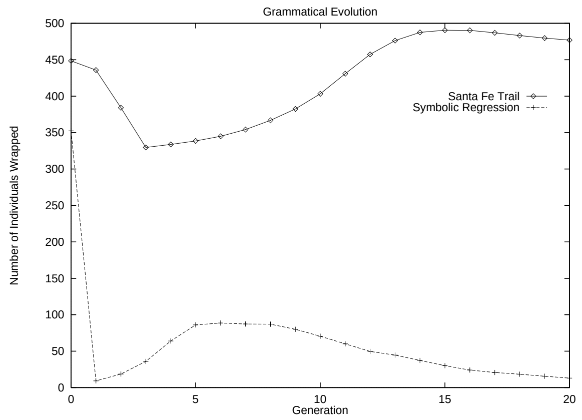

Figure 4 Number of individuals wrapped.

unique individuals is much greater when wrapping is enabled, and the variety within the population is less than when wrapping is disabled.

A comparison of the average actual genome lengths to the average effective genome lengths is given in Figure 6. Given the fact that the wrapping feature was exploited to a greater extent in this problem the effective lengths of the genomes is greater than the actual lengths when wrapping was allowed.

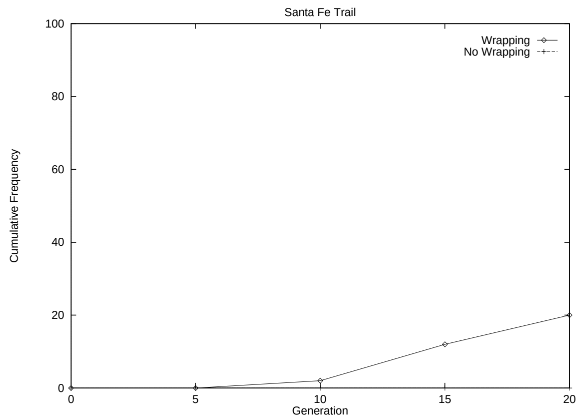

Figure 5 Cumulative frequency measures for the Santa Fe Trail problem.

5 Discussion

A comparison of the number of invalid individuals produced in both problem domains examined here shows that, invalid individuals are restricted almost exclusively to the initial population, see Figure 9. The effective removal of these unwanted individuals could be attributed to selection and replacement mechanisms used in the GA, in particular the Steady State replacement strategy, and as such is an example of the effective use of the illegal individual replacement constraint to evolve legal phenotypes, as described in [Yu and Bentley 98]. A significant increase in performance of the system was noted in [Ryan, O'Neill 98] when a Steady State replacement mechanism was employed over the previous generational mechanism. The poorer performance of the

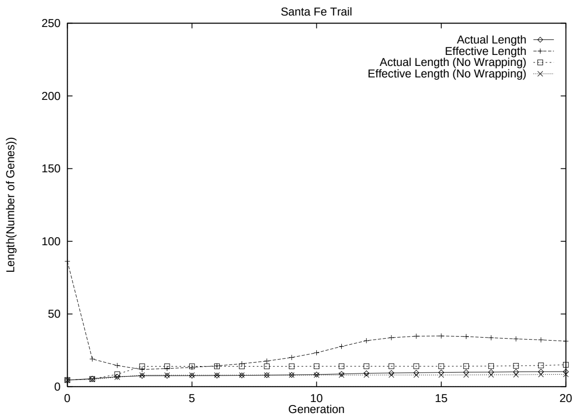

Figure 6 Actual versus Effective Genome Length for the Santa Fe Trail.

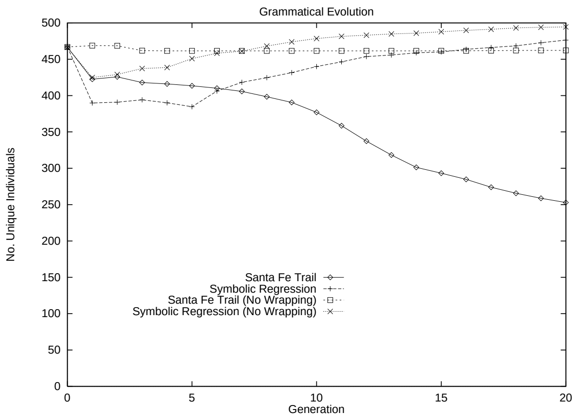

Figure 7 Number of unique individuals.

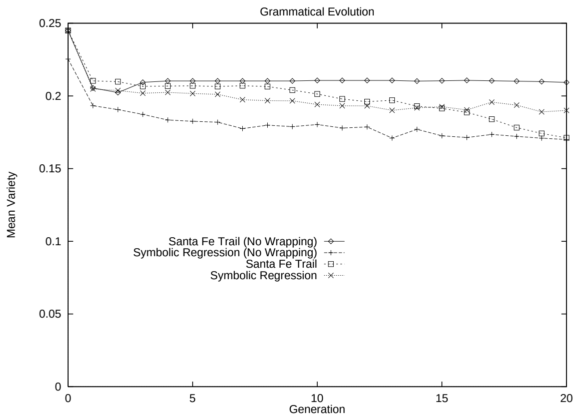

Figure 8 Mean variety for each generation.

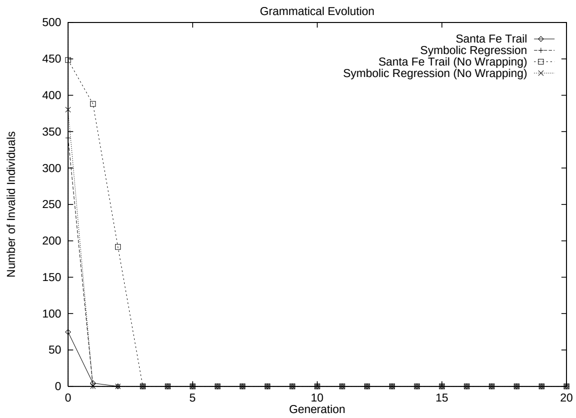

Figure 9 Number of invalid individuals for each generation.

system, when using the generational replacement mechanism, could be partially attributed to the propagation of a large number of invalid individuals from generation to generation which overwhelm the useful population.

The data obtained for the number of unique individuals, and the mean variety measure, while not perfect indicators of genetic diversity, would suggest that wrapping does not have a significant effect on the genetic diversity within a population. Wrapping does however, have a large effect on the probability of success within the 20 generations examined in the case of the Santa Fe Trail problem. This could be due to the regular repetitive nature of the required solution which could be exploited effectively by the wrapping mechanism. Future work will involve analysing these runs over longer periods of generations to determine if a solution to the Santa Fe Trail without wrapping could be obtained, by simply using longer chromosomes, and in particular by the use of the duplication operator with useful codons.

In [Banzhaf 94] it was suggested that a genotype to phenotype mapping process could be responsible for maintaining genetic diversity in a population by virtue of *neutral mutations*, what we refer to as silent mutations, according to Kimura's neutral theory of molecular evolution [Kimura 83]. The genetic code degeneracy in GE, just like in the genetic code observed in molecular biology, could be playing a vital role in maintaining the genetic diversity that has been observed here, and as such, an investigation into this phenomenon will be carried out. The fact that GE is capable of ensuring genetic diversity throughout a run, could prove to be a very useful feature if the system was to be applied to problems that required an adaptive property, such as in dynamic problem domains.

A comparison of GE to GP was carried out in [O'Neill, Ryan 99] on the Santa Fe Trail, and it was found that while GP suffered from bloat in the case where solution lengths were not minimised by being taken into account in the fitness function, GE suffered no such ill effects. Wrapping, one of the more unusual features of GE has been shown to be used by the system in the case of this problem domain. As to why wrapping is employed in this case is unclear, and does not appear to be responsible for restricting bloat, as no significant increase in genotypic lengths was observed when wrapping was switched off. However, when wrapping was disabled in the case of the Symbolic Regression problem it was found that the chromosome lengths increased by a large amount. In this case the wrapping feature appears to restrict the length of the chromosomes.

6 Conclusions

A detailed analysis of some of GE's features has been described, in particular focus was directed towards wrapping. The results indicate that wrapping was essential for success in the window of generations analysed on the Santa Fe Trail problem, and was useful in restricting genotypic lengths in the case of Symbolic Regression.

Using the mean variety measure has suggested that genetic diversity is being maintained during runs of the system. The fact that genetic diversity is maintained could be attributed to the fact that the search space has been separated from the solution space via the genotype to phenotype mapping process, and in particular it is thought that the degenerate genetic code is responsible for this phenomenon. In light of this, it would be interesting to examine in detail what effect the genetic code degeneracy is having on the performance of the system, and if it is being exploited during the evolutionary process to maintain genetic diversity.

Taking on board these findings, and the fact that GE has proved successful across diverse problem domains, indications are that GE is a powerful approach to generating programs in any language.

7 Acknowledgement

Many thanks to Alan Sheahan for discussions on this work.

References

[Banzhaf 94] Banzhaf, W. 1994. Genotype-Phenotype Mapping and Neutral Variation - A case study in Genetic Programming. In *Parallel Problem Solving from Nature III*. Springer.

[Elseth 95] Elseth Gerald D., Baumgardner Kandy D. 1995. Principles of Modern Genetics. *West Publishing Company*

[Kimura 83] Kimura, M. 1983. The Neutral Theory of Molecular Evolution. Cambridge University Press.

[Koza 92] Koza, J. 1992. *Genetic Programming*. MIT Press.

[Langdon 98] Langdon, W. 1998. Genetic Programming and Data Structures. Kluwer Academic Publishers.

[Langdon, Poli 98] Langdon, W. & Poli, R. Why Ant's Are Hard. In *Proceedings of Genetic Programming 1998*, pages 193-201 .

[O'Neill, Ryan 99] O'Neill M., Ryan C. Evolving Multi-line Compilable C Programs. In *Proceedings of the Second European Workshop on Genetic Programming 1999*.

[Ryan et al. 98a] Ryan C., Collins J.J., O'Neill M. 1998. Grammatical Evolution: Evolving Programs for an Arbitrary Language. *Lecture Notes in Computer Science 1391, Proceedings of the First European Workshop on Genetic Programming*, pages 83-95 . Springer-Verlag.

[Ryan et al. 98b] Ryan C., O'Neill M., Collins J.J. 1998. Grammatical Evolution: Solving Trigonometric Identities. In *Proceedings of Mendel '98: 4th International Conference on Genetic Algorithms, Optimization Problems, Fuzzy Logic, Neural Networks and Rough Sets*, pages 111-119.

[Ryan, O'Neill 98] Ryan C., O'Neill M. Grammatical Evolution: A Steady State Approach. In *Late Breaking Papers, Genetic Programming 1998*, pages 180-185.

[Yu and Bentley 98] Yu, T. and Bentley, P. 1998 Methods to Evolve Legal Phenotypes. In *Lecture Notes in Computer Science 1498, Proceedings of Parallel Problem Solving from Nature V*. Pages 280-291. Springer.