Redundant Genes and the Evolution of Robustness

Russell Thomason
Department of Computer Science
University of Idaho
Moscow, ID, 83844-1010
thom0398@uidaho.edu

Terence Soule*
Department of Computer Science
University of Idaho
Moscow, ID, 83844-1010
tsoule@cs.uidaho.edu

ABSTRACT

In this paper we demonstrate that pressure for robustness combined with function sets containing redundant genes can cause an evolutionary system to avoid a more fit solution in favor of a more robust solution. It is also shown that this trend depends significantly on the mutation rate used.

Categories and Subject Descriptors

I.2.2 [**Artificial Intelligence**]: Automatic Programming—*program synthesis*

General Terms

Genetic robustness

Keywords

Bloat, code bloat, robustness, redundant genes

1. INTRODUCTION

Genetic robustness is a measure of the invariance of fitness when an individual undergoes genetic changes [1]. Individuals with a smaller expected variance are more robust than individuals with a larger expected variance. Research has shown that there is significant evolutionary pressure to evolve genetically robust solutions and that this pressure has a significant effect on the course and trajectory of evolution.

In this paper we examine the inclusion of redundant genes in the initial function set. We define redundant genes as two or more sections of an individual that have the same affect on the individual's fitness. We hypothesize that the evolutionary trajectory of an evolving population will be deflected towards solutions that incorporate redundant genes.

2. EXPERIMENTAL MODEL

In these experiments individuals are linear, fixed length strings composed of four characters [A, X, Y, Z]. Fitness is determined by counting the number of A's and the combined number of X, Y and Z's. If the A count is larger, then the fitness is equal to the A count divided by the total length of the individual, multiplied by a weight coefficient. These solutions are defined as type-A solutions. If the XYZ count is larger, then the fitness is the XYZ count divided by the length of the individual with no weight coefficient used. These solutions are defined as type-XYZ solutions.

The goal is to create a string of either all A's or all X, Y and Z's. A solution composed entirely of A's is called an optimal type-A solution and a solution composed entirely of X, Y and Z's is called an optimal type-XYZ solution. Thus, this problem is similar to the max ones problem often seen in GA research. The X, Y and Z's act as the redundant genes in the initial set of primitives.

The weight coefficients tested are [1.00 1.01 1.03 1.05 1.10 1.15 1.20 1.25]. The first weight means that type-A solutions and type-XYZ solutions are evaluated equally. Each trial is run until an optimal solution is found. When this happens, the solution type of the optimal solution and the generation number are recorded. In each initial individual the number of A's exactly matches the combined number of X, Y and Z's and are distributed randomly. Thus, all individuals start with the worst possible fitness (0.5) because the A count and the XYZ count are equal.

3. EVOLUTIONARY ALGORITHM

For these experiments a steady state evolutionary algorithm is used. The population size is 100 individuals and each individual is a string of 100 characters. The algorithm is run for 500 trials on each combination of a mutation rate and a weight coefficient. Two point crossover is used. Parents are selected via tournament selection with a tournament size of three. Two offspring are created by crossover, mutation is applied, and the two least fit individuals in the population are replaced. Note that mutation includes the possibility of mutating a character into its original value. The mutation rates that are tested range from 0.005 to 0.10 in increments of 0.005.

4. EXPERIMENT

In this experiment the weight coefficients attempt to balance the preference for robustness versus fitness by increasing the value of the less robust type-A solutions. Figure 1 shows the number of trials that found optimal type-A solutions. Each line represents a different weight coefficient that is tested with each mutation rate.

Figure 1 shows that for low mutation rates, higher weight coefficients increase the number of trials that find optimal type-A solutions. As the mutation rate increases to a modest

*Address correspondence to Dr. Soule. This publication was made possible by NIH Grant P20 RR16448 from the COBRE Program of the National Center for Research Resources

GECCO'06 July 8-12, 2006, Seattle, Washington, USA
ACM 1-59593-186-4/06/0007.

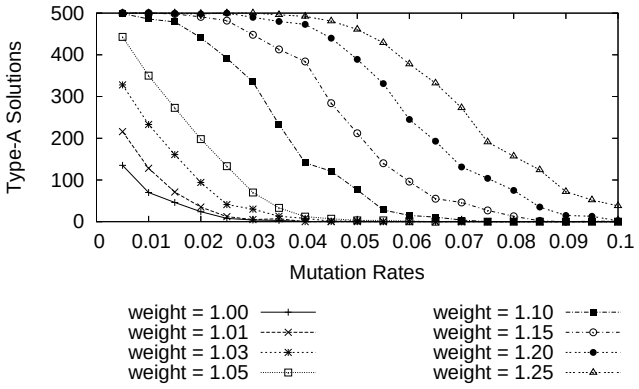

Figure 1: Number of trials that found optimal type-A solutions for different weight coefficients and different mutation rates.

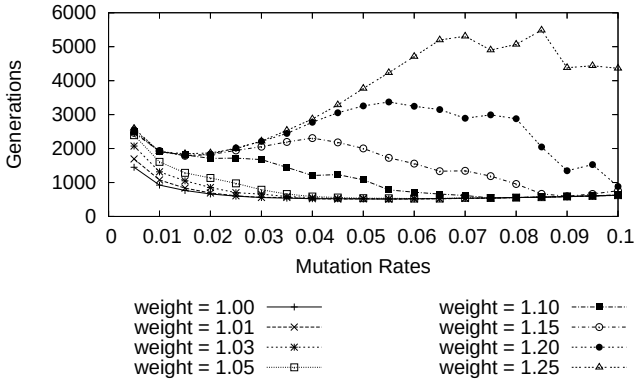

Figure 2: Average number of generations to find the first optimal solution (of either type) as a function of mutation rate and weight coefficient.

0.03, solutions that are 5% more valuable (weight coefficient = 1.05) are only found 14% of the time. At a mutation rate of 0.04, solutions worth 5% more are found only 2.4% of the time. We can control how many trials find type-A solutions by adjusting either the mutation rate or the weight coefficient. Even solutions that are 10% more valuable are still only found 3% of the time with a mutation rate of 0.06. Finally, for the rather high mutation rate of 0.10 the algorithm finds solutions that are 25% more valuable only 7.6% of the time and never finds solutions that are 20% more valuable. This shows strong pressure for the more robust type-XYZ solutions.

Figure 2 shows the average number of generations required for each trial to find the first optimal solution. For weight coefficients 1.05 and below the solutions quickly converge to almost all type-XYZ solutions, which mutation is more likely to produce anyway. The average number of generations required to find a solution decreases as the mutation rate increases, because individuals are more quickly changed into type-XYZ solutions. However, for the larger weight coefficients (above 1.05) the number of generations required to find the first optimal solution increases because the value of type-A solutions improves sufficiently that they become a significant, and eventually a predominant, percentage of the final solutions. When this happens, the average number of generations it takes to converge increases. This shows that there is a constant tension between higher fitness and higher robustness and the population takes longer to find an optimal solution of either type.

Note that the highest peak for each weight coefficient, which is the weight coefficient where it takes longest to find the first optimal solution, occurs at or around the mutation rate where the ratio of trials finding type-A to type-XYZ solutions is roughly 1 to 1. For example, Figure 1 shows that the line representing weight coefficient 1.25 created 278 type-A solutions (55.6% of the total trials) at a mutation rate of 0.07 and Figure 2 shows that the mutation rate that took the longest time to converge is 0.07. Similarly, the line representing weight coefficient 1.20 peaks around a mutation rate of 0.06 where 245 of the trials found type-A solutions (49% of the total trials) and in Figure 2 the line representing weight coefficient 1.15 peaks around a mutation rate of 0.045 which is where it produced 284 type-A solutions (56.8% of the total trials).

5. CONCLUSION

Our results show that the trajectory of evolution is heavily biased towards finding more robust solutions. Even when the less robust solutions are significantly more fit, by 10 to 25 percent, populations are still much more likely to converge on the less fit, but more robust solutions, for high mutation rates. In general, we found that there is a clear relationship between the mutation rate and the evolutionary pressure for robustness; as the mutation rate increases the probability that the population will converge on the more robust solution also increases.

These results do not support the hypothesis that neutrality (or neutral networks in the search space) automatically improve search performance. In these experiments the opposite result was found; neutral networks directed the trajectory of evolution away from the optimal solution. It appears that this redirection of the trajectory of evolution has two causes. The neutral networks, introduced in this research with redundant genes, represent plateaus in the search space. These plateaus consist of multiple, equivalent solutions, for example, a solutions where an X is replaced by a Y, etc. Because these plateaus consist of more than one solution they are both easier to find and harder to leave.

6. REFERENCES

[1] A. G. M. DeVisser, J. Hermission, G. P. Wagner, L. A. Meyers, H. Bagheri-Chaichain, J. L. Blanchard, L. Chao, J. M. Cheverud, S. F. Elena, W. Fontana, G. Gibson, T. F. Hansen, D. Krakauer, R. C. Lewontin, C. Ofria, S. H. Rice, G. von Dassow, and A. Wagner. Perspective: Evolution and detection of genetic robustness. *Evolution*, 57:1959–1972, 2003.