Position Paper, Workshop on Evolutionary Computation with Variable Size Representation, ICGA, 20 July 1997, East Lansing, Michigan, USA

Fitness Causes Bloat in Variable Size Representations

W. B. Langdon
School of Computer Science, The University of Birmingham, Birmingham B15 2TT, UK
W.B.Langdon@cs.bham.ac.uk http://www.cs.bham.ac.uk/~wbl
Tel: +44 (0) 121 414 4791, Fax: +44 (0) 121 414 4281

Technical Report: CSRP-97-14 14 May 1997

Keywords: genetic programming, structural complexity, introns, Occam's Razor, MDL, Price's Theorem

Abstract

We argue based upon the numbers of representations of given length, that increase in representation length is inherent in using a fixed evaluation function with a discrete but variable length representation. Two examples of this are analysed, including the use of Price's Theorem. Both examples confirm the tendency for solutions to grow in size is caused by fitness based selection.

1 Introduction

The tendency for programs in genetic programming (GP) populations to grow in length has been widely reported. This tendency has gone under various names such as "bloat", "fluff" and increasing "structural complexity". The principal explanation advanced for bloat has been the growth of "introns" or "redundancy", i.e. code which has no effect on the operation of the program which contains it. Such introns are said to protect the program containing them from crossover [Blickle and Thiele, 1994]. [McPhee and Miller, 1995] show, with some function sets, longer programs can "replicate" more "accurately" when using crossover. I.e. offspring produced by crossover between longer programs are more likely to behave as their parents than children of shorter programs. [Rosca and Ballard, 1996] provides a detailed analysis of bloat using tree schemata specifically for GP.

In this paper we advance a more general explanation which should apply generally to any discrete variable length representation and generally to any progressive search technique. That is bloat is not specific to genetic programming applied to trees and tree based crossover but should also be found with other genetic operators and non-population based stochastic search techniques such as simulated annealing and stochastic iterated hill climbing.

2 Bloat in Variable Length Representations

In general with variable length discrete representations there are multiple ways of representing a given behaviour. If the evaluation function is static and concerned only with the quality of a partial solution and not with its representation then all these representations have equal worth. If the search strategy were unbiased, each of these would be equally likely to be found. In general there are many more long ways to represent a specific behaviour than short representations of the same behaviour. Thus we would expect a predominance of long representations.

Practical search techniques are biased. There are two common forms of bias when using variable length representations. Firstly search techniques often commence with simple (i.e. short) representations, i.e. they have an in built bias in favour of short representations. Secondly they have a bias in favour of continuing the search from previously discovered high fitness representations and retaining them as points for future search. I.e. there is a bias in favour of representations that do at least as well as their initiating point(s).

On problems of interest, finding improved solutions is relatively easy initially but becomes increasingly more difficult. In these circumstances, especially with a discrete fitness function, there is little chance of finding a representation that does better than the representation(s) from which it was created. So the selection bias favours representations which have the same fitness as those from which they were created.

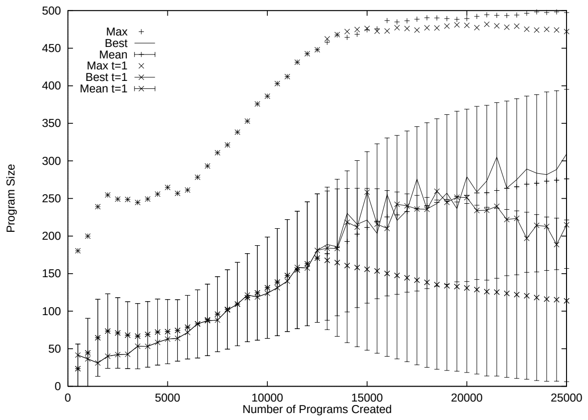

Figure 1: Evolution of maximum and population mean program length. Error bars indicate one standard deviation. Solid line is the length of the "best" program in the population. Means of 50 runs showing effect of removing fitness selection.

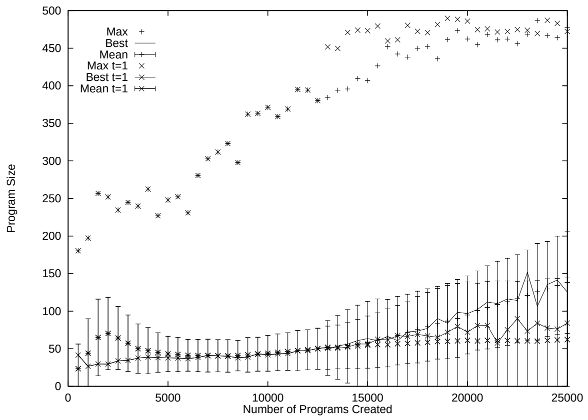

Figure 2: Evolution of maximum and population mean program length using mutation in place of crossover.

In general the easiest way to create one representation from another and retain the same fitness is for the new representation to represent identical behaviour. Thus, in the absence of improved solutions, the search may become a random search for new representations of the best solution found so far. As we said above, there are many more long representations than short ones for the same solution, so such a random search (other things being equal) will find more long representations than short ones. In GP this has become known as bloat.

3 The Artificial Ant Problem

We conducted two sets of experiments using the Artificial Ant Problem [Koza, 1992, pages 147–155]. Firstly solving it using GP and crossover [Langdon and Poli, 1997] and secondly replacing crossover with sub tree based mutation. In each case bloat occurs with fitness based selection, it does not occur without fitness based selection and bloat stops when fitness selection is removed after a period of evolution (cf. Figures 1 and 2).

3.1 Fitness is Necessary for Bloat – Price's Theorem Applied to Representation Length

Price's Covariance and Selection Theorem [Price, 1970] from population genetics relates the expected change in frequency of a gene Δq in a population from one generation to the next, to the covariance of the gene's frequency in the original population with the number of offspring z produced by individuals in that population (see Equation 1).

$$\Delta q = \frac{\mathrm{Cov}(z,q)}{\overline{z}} \tag{1}$$

If Equation 1 held exactly a plot of covariance vs. change in mean length would be a straight line. Figure 4 shows good agreement between theory and measurement, as does the solid line plot in Figure 3 until generation 20. After Generation 20 the rise in program length is smaller than predicted by Equation 1. The bulk of the discrepancy is due to the restriction on program length, as can be seen from the dotted line in Figure 3.

Where fitness selection is not used each individual in the population has the same fitness and so the covariance is always zero. Therefore Price's Theorem predicts on average there will be no change in length, as we have observed.

3.2 Non-Disruptive Crossover and Program Length

Detailed examination of the creation of new programs by crossover with the same fitness as their first parent shows on average they are always longer than it. From generation 4 to 23 they are on average

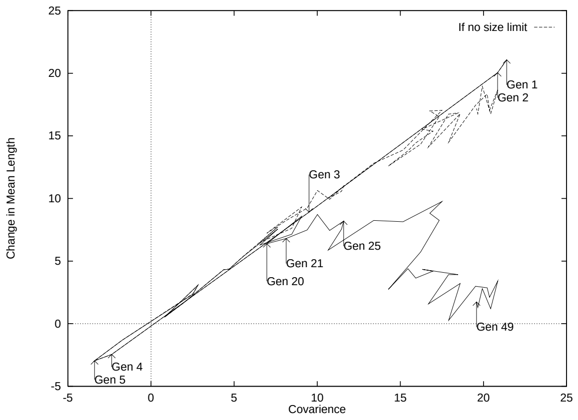

Figure 3: Covariance of program length and normalised rank based fitness v. change in mean length in next generation. Means of 50 runs.

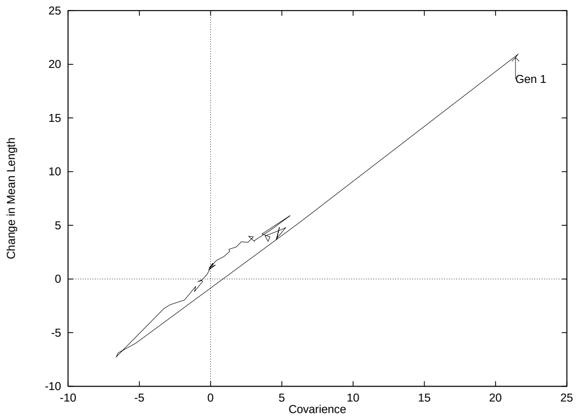

Figure 4: Covariance of length and fitness v. change in length in next generation. Means of 50 runs using mutation in place of crossover.

about 3 program nodes longer. Presumably due to the average program length approaching the upper limit, the bias falls to about 1 after generation 33 but remains positive.

4 Conclusions

We have generalised existing explanations for the widely observed growth in GP program size with successive generations (*bloat*) to give a simple statistical argument which should be generally applicable both to GP and other systems using discrete variable length representations and static evaluation functions. We have summarised two experiments, using different search techniques, which support this. Briefly our argument is, in general simple static evaluation functions quickly drive search to converge, in the sense of concentrating the search on trial solutions with the same fitness as previously found trial solutions. In general variable length allows many more long representations of a given solution than short ones of the same solution. Thus (in the absence of a parsimony bias) we expect longer representations to occur more often and so representation length to tend to increase. I.e. fitness based selection leads to bloat.

Acknowledgements

This research was funded by the Defence Research Agency in Malvern.

References

[Blickle and Thiele, 1994] Tobias Blickle and Lothar Thiele. Genetic programming and redundancy. In J. Hopf, editor, *Genetic Algorithms within the Framework of Evolutionary Computation (Workshop at KI-94, Saarbrücken)*, pages 33–38, Im Stadtwald, Building 44, D-66123 Saarbrücken, Germany, 1994. Max-Planck-Institut für Informatik (MPI-I-94-241).

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

[Langdon and Poli, 1997] W. B. Langdon and R. Poli. Fitness causes bloat. Technical Report CSRP-97-09, University of Birmingham, 26 March 1997. Submitted to WSC2.

[McPhee and Miller, 1995] Nicholas Freitag McPhee and Justin Darwin Miller. Accurate replication in genetic programming. In L. Eshelman, editor, *Genetic Algorithms: Proceedings of the Sixth International Conference (ICGA95)*, Morgan Kaufmann.

[Price, 1970] George R. Price. Selection and covariance. *Nature*, 227, August 1:520–521, 1970.

[Rosca and Ballard, 1996] Justinian P. Rosca and Dana H. Ballard. Complexity drift in evolutionary computation with tree representations. Technical Report NRL5, University of Rochester, Computer Science Department, Rochester, NY, USA, December 1996.