FOGP 1999

Linear Increase in Tree Height Leads to Sub-Quadratic Bloat

W. B. Langdon
Centrum voor Wiskunde en Informatica, Kruislaan 413, NL-1098 SJ Amsterdam
W.B.Langdon@cwi.nl http://www.cwi.nl/~bill/ Tel: +31 20 592 4093, Fax: +31 20 592 4199

1 INTRODUCTION

It has been known for some time that programs within GP populations tend to rapidly increase in size as the population evolves [6]. If unchecked this consumes excessive machine resources and so is usually addressed either by enforcing a size or depth limit on the programs or by an explicit size component in the GP fitness. We show with standard subtree crossover and no limits programs grow in depth at a linear rate leading sub-quadratic increase in size. It might be assumed that such limits have little effect but GP populations are effected surprisingly quickly.

2 DISTRIBUTION OF FITNESSES

In general the distribution of fitness values does not change much with their length, provided they are bigger than some problem and fitness level dependent threshold [5, 2]. (Figure 1 shows an example distribution of fitness against size). Thus the number of programs with a given fitness is distributed like the total number of programs. The number of programs rises ≈ exponentially with program length. Also most programs have a maximum depth near $2\sqrt{\pi(\text{internal nodes})}$ (ignoring terms $O(N^{1/4})$) [1]. See dotted lines on Figure 2.

After a period GP (or any other stochastic search technique) will find it difficult to improve on the best trial solution it has found so far and instead most of the trial solutions it finds will be of the same or worse performance. Selection will discard those that are worse, leaving active only those that are as good as the best-so-far. In the absence of bias, the more plentiful programs with the current level of performance are more likely to be found [4]. But as the previous paragraph has argued, the distribution of these is similar to the distribution of trees, therefore we expect the search to evolve in the direction of the most popular tree shape. [6] confirms this in various diverse problems when using GP with standard crossover.

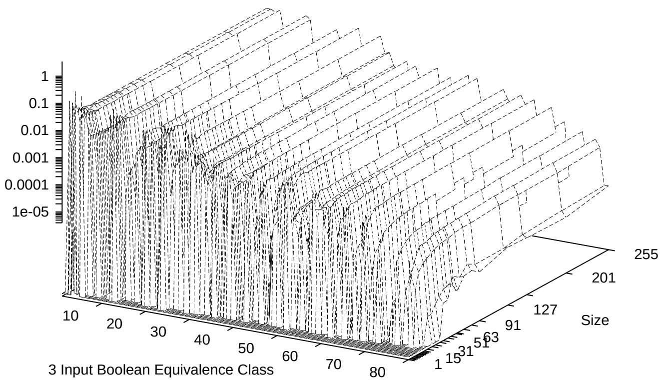

Figure 1: Proportion of NAND trees which yield each 3 input logic function.

3 EXPERIMENTS

3.1 EVOLUTION OF DEPTH

In four benchmark problems (quintic & sextic polynomials, 6 & 11 multiplexors) and 3 different ways of creating the initial populations (2 half-and-half and uniform [3]) both program depth and the spread of depths in the population in runs using standard crossover grow rapidly but apparently linearly. On average trees grow about one level per generation. Table 1 gives the population mean and max program depths (i.e. distance from root to furthest leaf) and their average rate of increase over the last 38 generations of the runs.

3.2 EVOLUTION OF SHAPE

Figure 2 shows in the absence of size or depth limits, program shape (dashed line) evolves towards the ridge in the distribution of programs [6].

3.3 SUB-QUADRATIC BLOAT

As discussed in [6, 3] and Section 2, if the programs within the population remain close to the ridge in the number of programs versus their shape and they increase their depth at a constant rate this leads to a prediction of sub-quadratic growth in their lengths'. (For modest size programs we expect size O(gens$^{1.3}$) rising to a limit of quadratic growth for $|\text{program}| \gg 1000$ cf. [1, Table II]. Over the last 38 generations the mean measured values are near O(gens$^{1.25}$) for the quintic and sextic problems (which are solved with binary trees). See last columns in Table 1, which gives the exponent in a power law fit of mean program size in population over last 38 generations v. generation.

Table 1: Program Depth, standard crossover 50 runs

Prob	Initia-lisation	Final pop ave	max	Growth/gen ave	max	Size exp (SD)
Qu	R2–6	54	181	1.2	4.0	1.31 (.28)
	R5–8	43	128	0.8	2.4	1.30 (.35)
	U3–25	101	332	2.2	7.0	1.28 (.21)
Sex	R2–6	47	150	1.1	3.5	1.46 (.37)
	R5–8	45	131	0.9	2.5	1.24 (.30)
	U3–63	98	312	1.9	5.8	1.18 (.18)
6 Mux	R2–6	39	101	0.8	2.1	1.25 (.21)
	R5–8	40	97	0.7	1.9	1.31 (.34)
	U2–25	56	172	1.2	3.6	1.29 (.20)
11 Mux	R2–6	34	107	0.7	2.1	1.22 (.16)
	R5–8	32	90	0.5	1.4	1.15 (.26)
	U2–25	45	157	0.9	2.9	1.23 (.15)

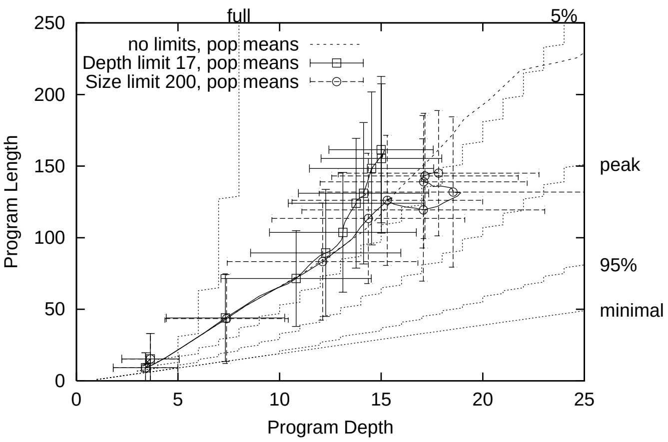

Figure 2: Evolution of size and maximum depth. Standard deviations shown every 5 generations.

3.4 DEPTH AND SIZE LIMITS

Using the average depth of the initial population and rate of increase in depth from the table we can estimate how long it will take for bloat to take the population on average to the depth limit (17). $\Delta\text{gens} = (17 - 3.64)/1.2 = 12$. The 3 curves in Figure 2 give the mean sizes and depths. They lie almost on top of each other until the fourth tick mark (corresponding to generation 12). Co-incidentally this is also the point where the size limited population diverges from the unlimited population.

There is a lot of variation between runs. In this run the programs are fuller than the ridge and so the size limited case diverges rather sooner than the peak curve predicts. (In fact shortly after the biggest program reaches the limit, 200).

4 CONCLUSIONS

Average growth in program depth when using standard subtree crossover is near linear in these problems. When combined with the known distribution of number of programs of any given size and depth, this yields a prediction of sub-quadratic growth in program size which we have measured. This indicates GP populations using standard crossover (and no parsimony techniques) will quickly reach bounds on size or depth commonly used. While crude quantitative predictions concerning the effect of depths restrictions are surprisingly accurate, size limits appear to more variable, perhaps due to greater sensitive about when bloat starts, variability between runs and greater sensitivity to individuals within the population. However a qualitative estimate can still be made.

References

[1] P. Flajolet and A. Oldyzko. The average height of binary trees and other simple trees. *Journal of Computer and System Sciences*, 25:171–213, 1982.

[2] W. B. Langdon. Scaling of program tree fitness spaces. 31 Jan. 1999.

[3] W. B. Langdon. Size fair and homologous tree genetic programming crossovers. *GECCO-99.*

[4] W. B. Langdon and R. Poli. Fitness causes bloat. In P. K. Chawdhry, *et. al.* eds. *Soft Computing in Engineering Design and Manufacturing*, 1997.

[5] W. B. Langdon and R. Poli. Boolean functions fitness spaces. In R. Poli, *et. al.* eds. *Proceedings of EuroGP'99*, *LNCS 1598.* Springer-Verlag.

[6] W. B. Langdon, T. Soule, R. Poli, and J. A. Foster. The evolution of size and shape. In L. Spector, *et. al.* eds. *Advances in Genetic Programming 3.*