Better Trained Ants

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-98-08
28 February 1998

Abstract

The problem of programming an artificial ant to follow the Santa Fe trail has been repeatedly used as a benchmark problem. Recently we have shown performance of several techniques is not much better than the best performance obtainable using uniform random search. We suggested that this could be because the program fitness landscape is difficult for hill climbers and the problem is also difficult for Genetic Algorithms as it contains multiple levels of deception.

Here we redefine the problem so the ant is obliged to traverse the trail in approximately the correct order. A simple genetic programming system, with no size or depth restriction, is show to perform approximately three times better with the improved training function.

1 Introduction

The problem of programming an artificial ant to follow the Santa Fe trail has been repeatedly used as a benchmark problem. Recently we have shown performance of several techniques is not much better than the best performance obtainable using uniform random search [Langdon and Poli, 1998]. We suggested that this was may be because the program fitness landscape is difficult for hill climbers and the problem contains multiple levels of deception which also makes it difficult for Genetic Algorithms.

Analysis of high scoring non-optimal programs suggests many reach high rewards even though they exhibit poor trail following behaviour. Typically they achieve high scores by following the trail for a while and then losing it at a corner or gap. They then execute a random search until they stumble into the trail at some later point and recommence following it. The random search may give them a better score than a competing program which successfully navigated the same bend or gap but lost the trail later.

Here we redefine the problem. The same trail is used but we only place food onto the grid as the ant following along the trail nears it. This makes it difficult for an ant which moves away from the trail to find another part of it by random search. This changes the fitness landscape. We anticipate that almost all optimal points within it will retain their scores and many previously high scoring non-optimal points will be given reduced fitness by the new training regime. This is expected to make the landscape less deceptive and so easier for genetic algorithms. Removal of false peaks may also benefit hill climbing techniques.

The Ant problem and the GP we use to evolve solutions to it are briefly described in Section 2, our results are given and discussed in Section 3, and in Section 4 we give our conclusions.

2 The Artificial Ant Problem

The genetic programming system and the artificial ant problem [Koza, 1992, pages 147–155] were set up identically to [Langdon and Poli, 1998] except the new training technique was used (see Table 1).

The program primitives are the same as before except now food items on the trail are numbered in the order we expect the ant to eat them. Only the first x are placed on the grid initially. As the ant moves and eats them new food pellets are added to the grid. When it eats food pellet n then we ensure all the uneaten food pellets up to pellet $n + x$ are on the grid. 50 (or 100 in the case of $x = 5$) independent runs where carried out with each value of x.

Table 1: Ant Problem

Objective:	Find an ant that follows the “Santa Fe trail”
Terminal set:	Left, Right, Move
Functions set:	IfFoodAhead, Prog2, Prog3
Fitness cases:	The Santa Fe trail
Fitness:	Food eaten
Selection:	Tournament group size of 7, non-elitist, generational
Wrapper:	Program repeatedly executed for 600 time steps.
Population Size:	500
Initial population:	Created using “ramped half-and-half” with a max depth of 6
Parameters:	90% crossover, 10% reproduction, no size limit
Termination:	Maximum number of generations G = 50

3 Results

As expected the improved training technique made the task of evolving suitable programs easier. This can be seen in Figure 1 which plots the estimated minimum number of individuals that the GP with a population size of 500 needs to create in order to have a probability of at least 99% of finding at least one solution. (This is known as “Effort” required, cf. [Koza, 1992, page 194]).

Figure 2 plots effort calculated from 50 runs (or 100 for $x = 89$, size limit=200) for the original Santa Fe trail and the same trail but with x set to five. No run found any of the very smallest solutions (of length 11). Considering first the original problem, the minimium estimate for the Effort is 189,000 and occurs with a size limit of 100 however values for 25–100 seem similar at about two thirds of the Effort required when no size limit is imposed. Also Effort values for size limits of 200–500 appear to be much the same as when there is no size limit. Now considering the case where food is only placed on the grid as the ant approaches it in along the trail (i.e. $x = 5$), the minimium estimate for the Effort is 104,000 and occurs with a size limit of 50 however values for 25–100 seem similar at about 80% of the Effort required when no size limit is imposed. Again Effort values for size limits of 200–500 appear to be much the same as with no size limit.

Changing amount of food on the trail has no obvious effect on the size of solutions.

4 Conclusions

By making a modest change to the Santa Fe trail problem we have made it better at training GP. In terms of the Effort required it is approximately three times easier and GP performs about as well as the best results previously obtained using variable length hill climbers [Langdon, 1998] and size restricted EP search [Chellapilla, 1997]. By enforcing an optimal size limit on the programs being evolved we are able to do marginally better. Surprisingly GP performance is only marginally affected by a size limit and is roughly constant for wide ranges in maximium size.

This indicates GP is still not greatly out performing random search.

References

[Chellapilla, 1997] Kumar Chellapilla. Evolutionary programming with tree mutations: Evolving computer programs without crossover. In John R. Koza, *et al*, editors, *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 431–438, 13-16 July 1997. Morgan Kaufmann.

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

[Langdon and Poli, 1998] W. B. Langdon and R. Poli. Why ants are hard. Technical Report CSRP-98-4, University of Birmingham, School of Computer Science, January 1998. Submitted to GP-98.

[Langdon, 1998] W. B. Langdon. The evolution of size in variable length representations. In *1998 IEEE International Conference on Evolutionary Computation*, Anchorage, Alaska, USA, 5-9 May 1998. Forthcoming.

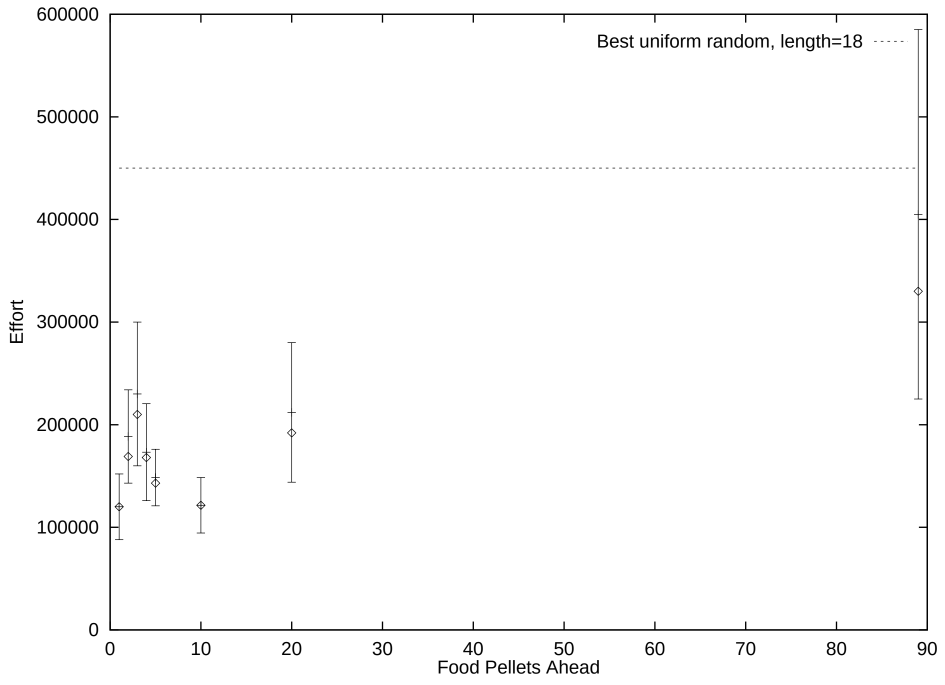

Figure 1: Effort v. no. food pellets ahead the ant can see and eat. Error bars indicate estimates given by one standard deviation above and below minimum measured figure. Original Santa Fe trail always allows all 89 food pellets to be seen.

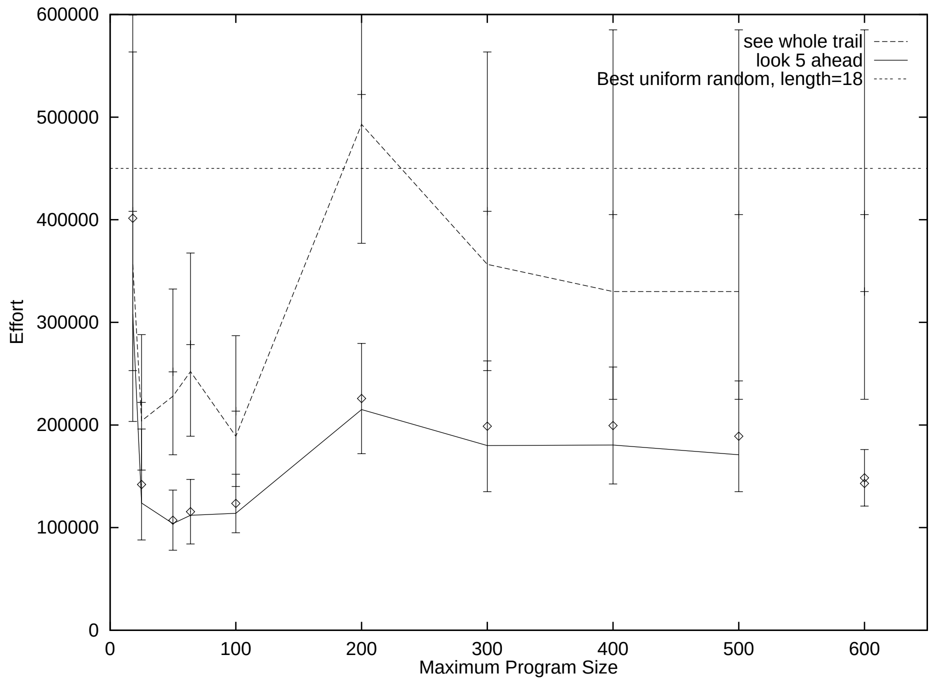

Figure 2: Effort v. maximium program size (11, 18, 25, 50, 64, 100, 200, 300, 400 and 500) for original Santa Fe trail and ant restructed to load ahead of 5. (For comparison, cf. Figure 1, values for runs without a size restriction are plotted at 600). Error bars indicate estimates given by one standard deviation above and below minimum measured figure.