The Evolution of General Intelligent Behavior

Daniel J. Parks, R. Kent Vail, Erin M. Harpel and Frank W. Moore

Miami University
Systems Analysis Department
230 Kreger Hall, Oxford, OH 45056
{parksdj, harpelem, moorefw}@muohio.edu
(513) 529-5928

Abstract

The evolution of general intelligent behavior continues to be an important goal of genetic programming research. This paper advocates the development of more appropriate problem definitions and models, using the classic artificial ant problem as an example. Our research demonstrates that generalized trail-following behavior can be achieved in the artificial ant through the evolution of a reactive control system.

1. Introduction

A. The Traditional Artificial Ant Problem Defined

The Artificial Ant (Koza 1992, pp. 147-155) is a classic evolutionary computation problem whose goal is the discovery of a trail following behavior for an ant that moves along an M-by-N toroidal grid. Food exists at various points on this grid until the ant "consumes" it by entering the unit square containing the food. Instead of being randomly scattered around the grid, the food is arranged in a trail with occasional gaps and turns.

In the traditional formulation of this problem (Jefferson *et* al 1991), the artificial ant had the ability to move forward, to turn to the left or right, and to sense if food is ahead and subsequently decide between one of two options. These abilities were encoded as the function and terminal set for a genetic programming solution. Arbitrary compositions of these functions were provided through the use of two other common functions, PROGN2 and PROGN3. The ant's performance was measured directly by the number of pieces of food consumed in the allotted number of simulated time-steps. A perfect ant consumed all pieces of food on the trail, i.e., the path the ant took passed very close to each point where food existed.

B. The Helpless Ant and its Limited World

As shown by other researchers [(Koza 1992), (Kuscu 1998), (Langdon and Poli 1998)], the artificial ant (as traditionally defined and implemented) was not an ideal candidate for producing generalized trail-following behavior, for several reasons. First, the traditional ant had no *a priori* knowledge of its environment. Second, the only way the traditional ant could learn about its environment was by using the IFFOODAHEAD function (defined below); the only information this single sensing function provided was obtained by answering the simple yes-or-no question, "Is there food directly in front of me?" Third, the evolved programs were often *overtrained* (Droste 1997) or specialized for one specific training environment (Kuscu 1998); that is, an artificial ant trained on a single trail of food simply memorized the location of food on that particular trail, and was incapable of learning *general* food foraging behavior. In effect, the solutions provided by the traditional problem definition achieved only *local* optimization, rather than *global* (Langdon and Poli 1998). Efforts have been made to overcome some of the limitations stemming from the training environment of the program [e.g., (Kuscu 1998)]; however, our experience with similar strategy optimization problems (Moore 1997) has shown that the problem of bolstering generalization of evolved solutions cannot be solved simply by providing a test-bed of training cases before releasing the ant to scoot around in unknown territory.

The model of the traditional ant's world was also quite limited. During the run of a program, it was generally limited to the integer X and Y values of the ant, the ant's current direction (North, South, East, or West), the count of food that the ant had consumed, and the time-steps taken thus far in the simulation. In short, the traditional ant was forced to live in a discrete representation of a highly constrained and unrealistic world.

Thus, *the traditional artificial ant's failure to produce global, general solutions was a direct result of the limitations of the traditional problem definition.*

C. Freeing the Ant

To overcome these limitations, we redefined the artificial ant problem in a manner that would evolve a

"SuperAnt". The SuperAnt would be capable of *general intelligent action*, exhibited through the ability to navigate previously unseen trails of food in an environment where the limitations and bounds of the traditional artificial ant problem do not exist. Our new formulation necessitated several changes in both the SuperAnt's environment and its function/terminal set, including the following:

i. POSITION – The SuperAnt is placed on an XY plane that is measured in real values (i.e., using floating point representation) rather than discrete integer values.

ii. DIRECTION – Whereas the traditional artificial ant could only look one of four discrete directions, the SuperAnt may face any direction within the range of 0-360 degrees.

iii. MOVE – This terminal causes the SuperAnt to move forward one unit of distance in the current direction during one time-step of the simulation.

iv. ROTATE – This function takes one argument from the gene and causes the direction of the SuperAnt to change that amount. The change value is in the range of -1.0 -- 1.0, where +/-1.0 represents +/-180 degrees, and 0.0 represents 0 degrees.

v. SNIFF – This terminal allows the SuperAnt to learn about its environment. If the piece of food closest to the SuperAnt's position lies within certain parameters of distance and angle relative to the its current direction, the difference of angle between the SuperAnt's direction and its line-of-sight angle to the piece of food is returned. If there is no such piece of food, then a random value over the possible range is returned. [Note: When biological ants follow trails, such as pheromone trails, they are guided in a more continuous fashion (Holldobler and Wilson 1990), rather than "connecting the dots" by searching for a chemical deposit at a discrete location. In spite of this obvious difference, the SuperAnt's behavior comes much closer in practice to that of the biological ant than traditional artificial ant models.] The range of possible values this return value may take is the same as the range of values for ROTATE.

vi. IFFOODAHEAD – This function is similar to the traditional definition, with the exception that the piece of food in question may be either directly ahead of the SuperAnt (an unlikely occurrence), or within some reasonable range of being directly ahead. If such a piece of food is found, then the *true* branch of the if-then-else structure is evaluated; otherwise, the *false* branch is evaluated.

vii. The two other functions in the set, PROGN2 and PROGN3, are similar to those of the traditional artificial ant problem, with the exception that values from the subtrees that are evaluated are averaged, and the result is returned to the calling function, rather than returning 0.0 every time.

D. The Nature of the Problem Changes After Liberation

With the enhanced function and terminal set, this SuperAnt may seem less like an ant and more like a dog (see SNIFF): in effect, SuperAnt learns to sniff its way from one piece of food to the next. Additionally, the problem domain moves from one of simply finding a solution to a fixed problem over a fixed circumstance [e.g., The Pac Man problem (Koza 1992), (Rosca and Ballard 1996)] to one that could evolve to a generalized behavior over many circumstances [(Moore 1997), (Kuscu 1998)]. The new formulation evolves a reactive control system capable of effective interaction with the environment [e.g., (Brooks 1991), (Howley 1996), (Parker 1998)]. This approach is appropriate: the intent of enhancing the function/terminal set for the problem is to allow the program to learn about its environment [e.g., (Banzhaf, Nordin, and Olmer 1997), (Bennett 1997)], thereby being less dependent on "genetically memorized" knowledge and more dependent on the evolution of general behaviors supported by the enhanced function/terminal set.

2. Implementation of the New Function and Terminal Set

The relevant sections of code were written in C and used with the *lilGP* system (Zonger and Punch 1996). Many of the technical details of the function and terminal sets are related more to computational geometry than to artificial intelligence; we ultimately used several algorithms from (Sedgewick 1990).

Several parameters to the simulation that greatly affect the outcome of the program must be specified in advance. These parameters include such items as the range within which the SuperAnt can smell food, the angle to either side of the SuperAnt's line-of-sight in which it can sense food, and so on. In the runs of the program/simulation conducted to date, every attempt was made to specify "reasonable" values for these parameters; admittedly, however, there may be much to question and investigate to determine what is "reasonable" and what is not. The *lilGP* system comes with an implementation of the traditional artificial ant, as well as a parameter file that models the problem as presented by Koza (Koza 1992). Although we began this research using values from this file, it became necessary to change several of the parameters, primarily due to the fact that the artificial ant existed on a discrete grid, whereas the SuperAnt uses a real-valued plane. One parameter that is the same for both ants is the distance moved in one time-step of the simulation: each moves exactly one unit.

Additional SuperAnt parameters that were unnecessary for the traditional ant problem include the angle and distance within which the SuperAnt's sensing functions returned accurate values.

Another important factor that was overlooked during the early stages of our research was the distance that specified how close the SuperAnt has to get to a piece of food before that food is considered to have been eaten (i.e., before a "hit" is registered). If the distance were too large, then the SuperAnt would have an unfair advantage over the traditional ant, in that it would be able to consume more food with less effort. For this reason, we used a value of only 1/4 of one unit of movement.

3. Modification of the Training Set

To bolster the ability of best-of-run programs to exhibit optimized behavior on previously unseen trails of food, SuperAnts were evolved using a training population of multiple trails. Each trail in the training population significantly differed from other trails; variety in the training population minimized the potential for overtraining on a single trail (or a single type of trail), and rendered memorized solutions typical of the traditional artificial ant problem useless. Aggregate raw program fitness equaled the total amount of food eaten by the ant when independently executed on each of the trails in the training population. Each program was allowed a limited number of time steps to complete each trail.

4. Results

Two best-of-run programs were independently evolved against a set of six training trails, using the enhanced function set described above. Each run used a population size of M = 1000 programs per generation and ran for G = 50 generations. These best-of-run programs, dubbed "Bud1" and "Bud2", were subsequently tested against two previously unseen trails to determine whether they were indeed capable of general trail-following behavior. This section summarizes the results of training and testing for these two programs. For the sake of comparison, results for random programs against the two test trails are also provided.

A. *FITNESS OF THE BEST-OF-RUN PROGRAM VS. THE TRAINING TRAILS.*

```
Bud1 ate 441 out of 534 pieces of
   food (83%) over the 6 training
   trails.

Bud2 ate 447 out of 534 pieces of
   food (84%) over the 6 training
   trails.
```

B. *FITNESS OF THE BEST-OF-RUN PROGRAM WHEN TESTED AGAINST THE TWO PREVIOUSLY UNSEEN TEST TRAILS.*

```
Bud1 ate 69 pieces of food (78%)
   on test trail 0, and 77 pieces
   of food (87%) on test trail 1.

Bud2 ate 75 pieces of food (84%)
   on test trail 0, and 79 pieces
   of food (89%) on test trail 1.
```

C. *AVERAGE FITNESS OF A RANDOM POPULATION OF PROGRAMS AGAINST THE SAME TWO TEST TRAILS.*

```
Test Trail 0: programs from a
   random population (size 1000)
   averaged 9 pieces of food
   (10%).

Test Trail 1: programs from a
   random population (size 1000)
   averaged 13.8 pieces of food
   (16%).
```

As these results show, the SuperAnts evolved during this investigation exhibit general food foraging behavior. Because they are evolved from a set of primitive functions typical of a reactive control system, SuperAnts can successfully find and eat food located on previously unseen trails of arbitrary complexity.

5. Summary and Conclusions

The artificial ant problem is a classic problem for artificial intelligence and specifically for genetic programming. Previous work has shown that solutions to problems for a single trail of food are achievable, but do not generalize. Part of the problem lies in the way that the problem has traditionally been defined. By resetting the problem in a new context, we have shown that alternate and improved definitions of the problem exist. This paper establishes the viability of SuperAnt as a candidate solution to the problem of evolving an artificial ant that is capable of general intelligent action. Our success stems from reformulating the Artificial Ant problem as a genetic programming system that evolves a reactive control program capable of successfully locating food along previously unseen trails.

Future research may include a more extensive look at what effect the specification of additional parameters will have on the work. (This is not the same as a study of the parameters and operators of genetic evolution [e.g., (O'Reilly and Oppacher 1996), (Teller 1996), (Angeline 1996)]. Rather, it deals with functions and variables such as EAT_RADIUS, SNIFF_DISTANCE, etc. that are specific to this problem.) One other potentially fertile field would be the combination of this research, involving modification of the agent, with ideas presented in (Kuscu 1998), which deal with modification of the environment, to identify additional ways to enhance a generalized behavior for this class of optimization problems.

The most exciting aspect of this research effort stems from the fresh new approach taken towards the evolution of programs that begin to exhibit general intelligent behavior. Evolutionary computation in general, and genetic programming in particular, have long suffered from the inability of evolved programs to behave optimally (or near-optimally) in test scenarios that were not explicitly anticipated by the training population. By modeling the classic Artificial Ant problem as a reactive control problem, we have transformed a problem that was once considered to be extremely difficult into a problem that could easily be solved using genetic programming. Future research will evolve reactive control systems representing optimized solutions to difficult problems drawn from a wide variety of problem domains.

6. Bibliography

Angeline, P. J. 1996. "Two Self-Adaptive Crossover Operators for Genetic Programming", in Angeline, P. J. and K. E. Kinnear, Jr. (eds.), *Advances in Genetic Programming Volume 2*, pp. 89-119, The MIT Press.

Banzhaf, W., P. Nordin, and M. Olmer 1997. "Generating Adaptive Behavior for a Real Robot using Function Regression within Genetic Programming", in Koza, J.R. *et al* (eds.), *Genetic Programming 1997: Proceedings of the Second Annual Conference, July 13-16, 1997, Stanford University*, pp. 35-43, Morgan Kaufmann Publishers.

Bennett, F. H. III 1997. "A Multi-Skilled Robot that Recognizes and Responds to Different Problem Environments", in Koza, J.R. *et al* (eds.), *Genetic Programming 1997: Proceedings of the Second Annual Conference, July 13-16, 1997, Stanford University*, pp. 44-51, Morgan Kaufmann Publishers.

R. Brooks, 1991. "Intelligence without representation", in *Artificial Intelligence*, Vol. 47, pp. 139--160, Jan. 1991.

Droste, S. 1997. "Efficient Genetic Programming for Finding Good Generalizing Boolean Functions", in Koza, J.R. *et al* (eds.), *Genetic Programming 1997: Proceedings of the Second Annual Conference, July 13-16, 1997, Stanford University*, pp. 82-87, Morgan Kaufmann Publishers.

Hanke, S. 1997. "Chromatic Search Trees Revisited", Technical Report 00090, *Institut fur Informatik Freiberg*, June 12, 1997.

Holldobler, B. and E. O. Wilson 1990. *The Ants*, Belknap Press/Harvard University Press.

Howley 1996. "Genetic Programming of Near-Minimum-Time Spacecraft Attitude Maneuvers", in Koza, J.R. *et al* (eds.), *Genetic Programming 1996: Proceedings of the First Annual Conference, Cambridge, MA*, pp. 98-109, The MIT Press.

Jefferson, D. *et al* 1991. "Evolution as a theme in artificial life: the Genesys/Tracker system", in *Artificial Life II*, Addison-Wesley.

Koza, J. R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, The MIT Press.

Kuscu, I. 1998. "Evolving a Generalised Behavior: Artificial Ant Problem Revisited", in Porto, V. W., N. Saravanan, D. Waagen, and A. E. Eiben (eds.), *Evolutionary Programming VII: 7th International Conference Proceedings/EP98, San Diego, CA, USA, March 25-27, 1998*, pp. 799-808, Springer-Verlag.

Langdon, W. B. and R. Poli 1998. "Why Ants are Hard", in Koza, J. R. *et al*, *Genetic Programming 1998: Proceedings of the Third Annual*

Conference, July 22-25, 1998, University of Wisconsin, Madison, pp. 193-201, Morgan Kaufmann Publishers.

Moore, F. W. 1997. *A Methodology for Strategy Optimization Under Uncertainty*, Ph.D. Dissertation, Wright State University, Dayton, OH.

O'Reilly, U. and F. Oppacher 1996. "A Comparative Analysis of Genetic Programming", in Angeline, P. J. and K. E. Kinnear, Jr. (eds.), *Advances in Genetic Programming Volume* 2, pp. 23-44, The MIT Press.

Parker, G. B. 1998. "Generating Arachnid Robot Gaits with Cyclic Genetic Algorithms", in Koza, J. R. *et al, Genetic Programming 1998: Proceedings of the Third Annual Conference, July 22-25, 1998, University of Wisconsin, Madison*, pp. 576-583, Morgan Kaufmann Publishers.

Rosca, J. P. and D. H. Ballard 1996. "Discovery of Subroutines in Genetic Programming", in Angeline, P. J. and K. E. Kinnear, Jr. (eds.), *Advances in Genetic Programming Volume* 2, pp. 177-201, The MIT Press.

Sedgewick, R. 1990. *Algorithms in C*, Addison-Wesley Publishing Company.

Teller, A. 1996. "Evolving Programmers: The Co-evolution of Intelligent Recombination Operators", in Angeline, P. J. and K. E. Kinnear, Jr. (eds.), *Advances in Genetic Programming Volume* 2, pp. 45-68, The MIT Press.

Zongker, D. and B. Punch 1996. "*lilGP* 1.01 User's Manual", Technical Report, Michigan State University. Available on-line at http://garage.cps.msu.edu/software/lil-gp/lilgp-index.html.