Improving Means and Variances of Best-of-Run Programs in Genetic Programming

Frank W. Moore
Computer Science Dept., University of Dayton
300 College Park, Dayton, OH 45469-2160
fmoore@udcps.cps.udayton.edu

ABSTRACT

Genetic programming (GP) systems have traditionally used a fixed training population to evolve best-of-run programs according to problem-specific fitness criteria. The ideal GP training population would be sufficiently representative of each of the potentially difficult situations encountered during subsequent program use to allow the resulting best-of-run programs to handle each test situation in an optimized manner. Practical considerations limit the size of the training population, thus reducing the percentage of situations explicitly anticipated by that population. As a result, best-of-run programs may fail to exhibit sufficiently optimized performance during subsequent program testing. This paper summarizes an investigation into the effects of creating a new randomly generated training population prior to the fitness evaluation of each generation of programs. Test results suggest that this alternative approach to training can bolster *generalization* of evolved solutions, improving the *mean* program performance while significantly reducing *variance* in the fitness of best-of-run programs.

1. Introduction

Genetic programming (GP) systems have traditionally used a fixed training population to evolve programs according to problem-specific fitness criteria. Koza ([17], p. 95) explains his rationale for using fixed training populations during program evolution in the following manner: *"One can minimize the effect of selecting a particular selection of fitness cases by computing fitness using a different set of fitness cases in each generation. Because the potential benefit of this approach is offset by the inconvenience associated with noncomparability of performance of a particular individual across generations, we do not use this approach in [17]. Instead, fitness cases are chosen at the beginning of each run and not varied from generation to generation."*

Best-of-run programs evolved using a fixed training population frequently exhibit optimal (or near-optimal) performance in competitive survival environments explicitly represented by that training population. Unfortunately, subsequent performance of these programs is generally less than optimal when situations arise that were not explicitly anticipated during program evolution. Recent attempts to bolster program *generalization* (the ability to correctly handle situations not explicitly anticipated during training) have considered the possibility of randomly changing the training population prior to the fitness evaluation of each generation of programs. This paper summarizes related research, demonstrates that the new methodology can improve *mean* fitness of best-of-run programs evolved by GP systems while significantly reducing *variance* in that fitness (when subsequently tested against arbitrary test populations), and suggests topics for future investigation.

2. Current Approaches

Several researchers have investigated the effects of changing the training population during program evolution. This section summarizes the results of these previous studies.

Co-evolution in GP systems is the process of simultaneously evolving two separate program populations. With co-evolution, the fitness of programs in each program population is evaluated against a training population of selected individuals from the other program population. As co-evolution proceeds, the average fitness of individuals from each program population typically improves. Since new training cases are selected from the opposing population prior to the fitness evaluation of each generation of programs, the average fitness of

each training population simultaneously increases.

Miller [22] used genetic algorithms (GAs) to co-evolve strategies for playing the Repeated Prisoner's Dilemma game. Hillis [16] used GAs to simultaneously evolve the candidate solutions a given problem, as well as the training population used to evaluate those candidate solutions. The fitness of each element of the training population was proportionate to the difficulty it presented to the solution population. As the fitness of the solution population improved, training cases simultaneously evolved to identify weak points in those solutions. Angeline and Pollack [2] used GP to evolve optimized programs for playing the game of Tic Tac Toe (Naughts and Crosses). The fitness of each program was estimated by playing it against other programs from the same generation in a single elimination tournament. The use of a competitive fitness function contributed to the accelerated evolution of more robust best-of-run programs. Siegel [31] subsequently described a GP solution to the word sense disambiguation problem that exploited competition to adjust the probability that each element of a fixed training population was selected for inclusion in the training set for the next generation of programs.

Since the fitness landscapes of each population of co-evolutionary systems continuously change (a phenomenon known as the "Red Queen effect"), the behavior of such systems can be erratic [32] and difficult to analyze [6, 12]. Best-of-run programs produced by the co-evolution of closed program populations often fail to successfully generalize against competitors from outside the training population [19, 11]. Various modifications to the standard co-evolutionary model have been shown to improve the robustness of competitively evolved programs [10]. Co-evolution continues to be an important topic for on-going GP research [7, 1, 21].

Luke [20] has pointed out that "the decision to randomize training cases primarily reflects trying to achieve a balance between generalization and difficulty in training. The appeal of randomizing one's training set is that it bolsters generalization during evolution". Luke and Spector identified a GP solution to the Wumpus World problem in which the fitness cases were changed only once per generation. Spector noted that the resulting best-of-run program effectively generalized to process previously unseen worlds, and pointed out one of the drawbacks of changing fitness cases during program evolution: "Changing fitness cases during the run can make it *much* more difficult to figure out what's going on during program evolution. We've gotten some really weird computational results ... that only begin to make sense when you start thinking about how GP is in this case simultaneously searching the space of programs and (randomly) the space of worlds. Bad programs can get 'lucky' fitness cases, and good programs can get 'unlucky'. Even if the same worlds are used across the entire generation, they may be particularly well-fitted to some particular weak strategies while presenting serious difficulties for stronger, more general strategies" [33]. McPhee reported similar results for various regression problems [23].

Gathercole and Ross [14] summarized GP systems that use comparatively small training populations to evolve solutions to various supervised learning classification problems. Their results indicated that GP could find better solutions with smaller population sizes and less computational effort by *not* using the entire training set to evaluate each generation [15]. Their GP system used Dynamic Subset Selection (DSS) to select a new subset of fitness cases prior to the evaluation of each generation of programs. DSS was shown to improve the average fitness of best-of-run programs, and consistently produced best-of-run programs exhibiting near-optimal performance [13]. The resulting best-of-run programs were more "robust" in that they were more likely to generalize well for previously-unseen test scenarios.

Daida *et al* [9] described a GP system that used dynamic selection of fitness cases to evolve programs for analysis of remotely sensed images. The amount of computation required to train the resulting GP system was significantly reduced by varying fitness cases during the course of each run. Bersano-Begey and Daida [4] described a methodology that divides complex tasks into subtasks, and checks for and supports the elimination of ambiguity within the set of fitness cases. Their preliminary results for the Wall Following Robot problem demonstrated that a non-exhaustive set of fitness cases can be systematically organized to produce robust best-of-run programs.

Each of these results supports the following conclusion: *Changing the training population*

during program evolution can significantly improve the performance of best-of-run programs during subsequent program testing.

3. Application to Missile Countermeasures Optimization

The MCO System [24] used GP to evolve programs that combine maneuvers with additional countermeasures (including chaff, flares, and jamming) to optimize aircraft survivability against attack by a single surface-launched anti-aircraft missile (SAM), in light of uncertainty about the type and/or current state of that missile. The MCO problem is a representative example of the more general task of identifying a methodology for strategy optimization under uncertainty. A GP solution to an abstracted, two-dimensional MCO problem known as the Extended Two-Dimensional Purser/Evader (E2DPE) problem (Figure 1) was introduced by Moore and Garcia [25].

The goal of the MCO System was to use GP to optimize aircraft survivability. The fitness of each program was evaluated by simulating several encounters between an F-16C aircraft evader [35, 18] and various types of SAM pursuers [8]. Prior to the start of each simulated encounter, the SAM used the initial state of the aircraft to predict an *intercept point.* The SAM was then launched at maximum thrust in the direction of the intercept point. If the aircraft failed to maneuver, the SAM destroyed the aircraft at (or very close to) the intercept point. If the aircraft maneuvered, the SAM relied upon the highly effective *proportional navigation* technique [37, 5] to pursue the aircraft. Proportional navigation caused the SAM to accelerate in the direction perpendicular the line-of-sight from the SAM to the aircraft; the magnitude of this acceleration was calculated by the equation

$$n_c = N' V_c (d\lambda/dt)$$

where N' is a unitless designer-chosen gain known as the *effective navigation ratio*, and V_c is the closing velocity vector (the negative rate of change of the distance from the SAM to the aircraft). The time derivative of the line-of-sight angle λ is known as the *line-of-sight rate.* The effective navigation ratio for a given guidance system, also known as the *guidance law gain*, may be determined mathematically from a complex series of computations [3]; for practical guidance systems, optimal values for N' range between 3 and 5 [29].

The aircraft maneuvered by executing specific combinations of thrusting and turning forces in specific sequences. The *optimal strategy* for the aircraft was to combine maneuvers with ECM in a manner that maximized its survivability, regardless of its initial state and the relative launch position of the SAM. Note that by rotating the reference coordinate system at the launch site of the SAM, the initial SAM/aircraft line-of-sight angle λ_0 was considered constant for all SAM/aircraft pairs. For this reason, the only variables necessary to describe the initial configuration of each confrontation were the *line-of-sight distance* between the aircraft and the SAM, and the *velocity vector* of the aircraft at the time the SAM was launched.

As shown in Figure 1, each fitness case in the MCO System was identified by a unique combination of two floating-point values ranging from 0.0 to 1.0. The first value, from set J, identifies the initial line-of-sight distance from the SAM to the aircraft. If D_{min} and D_{max} denote the minimum and maximum effective launch distances for the SAM, then the initial line-of-sight distance d_0 for fitness case P may be calculated by the equation

$$d_0 = D_{min} + (J_P * (D_{max} - D_{min}))$$

D_{min} and D_{max} depend upon the type of SAM. The second value, from set K, identifies the angle that the initial velocity vector of the incoming aircraft makes with the line-of-sight from the aircraft to the SAM. Let Θ_0 denote this angle. If Θ_{min} and Θ_{max} denote the minimum and maximum initial value of Θ, then Θ_0 for fitness case P may be calculated by the equation

$$\Theta_0 = \Theta_{min} + (K_P * (\Theta_{max} - \Theta_{min}))$$

To maintain the relative geometry illustrated in Figure 1, Θ_{min} and Θ_{max} described a range of values between 10 and 80 degrees. The magnitude of the initial aircraft velocity vector (its "speed") was assumed to be the equal to the F-16C's typical attack speed for each encounter; thus, each fitness case corresponded to a unique combination of one value from set J and one value from K.

The initial solution to the E2DPE problem [25] used fixed training populations – i.e., fixed values for sets J and K -- to evolve best-of-run programs. To determine the impact of randomly changing the training population prior to the fitness evaluation of each generation of

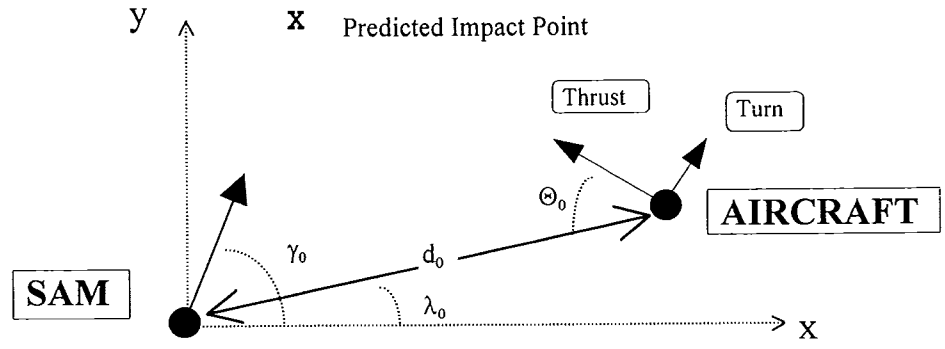

Figure 1. The Missile Countermeasures Optimization Problem (Initial Conditions).

programs, the MCO System was modified in three ways:

a) The modified system specified the initial aircraft/SAM line-of-sight distance by assigning to each fitness case a *randomly generated J value* in the [0.1, 0.9] interval continuum.
b) The modified system assigned to each fitness case a *randomly generated K value* in the [0.1, 0.9] interval continuum to specify the angle that the initial velocity vector of the incoming aircraft makes with the line-of-sight from the aircraft to the SAM.
c) The modified system randomly generated new set values for J and K immediately prior to the fitness evaluation of *each generation* of each run, using a finite, uniform distribution over possible values.

Since GP is a probabilistic optimization process, multiple runs are necessary to analyze the impact of specific changes to the GP methodology. To determine whether the brittleness of best-of-run programs evolved by the MCO System could be reduced by randomly generating a new training population prior to the fitness evaluation of each generation of programs, ten pairs of Training Runs were performed. Each pair of runs differed in only one respect: the first run used a fixed set of fitness cases, while the second run used a new set of randomly generated fitness cases to evaluate each generation of programs. All other conditions were identical for each pair of runs. Note that regardless of whether fixed or randomly generated training were used, the results of any given training run could be replicated by using an identical value for the random number seed [28].

Two different fixed training populations were created; Training Runs 1-5 used the first training population, while the second was used during Training Runs 6-10. Each of these fixed training populations was defined by the cross-product of values from sets J and K (as described above). 16 fitness cases were used during Training Runs 1-5, while Training Runs 6-10 used 32 fitness cases. To determine how well each best-of-run program generalized to handle arbitrary aircraft/SAM encounters, each program was subsequently tested against a large, representative fixed test population of SA-15 pursuers. The results of these tests, indicating the percentage of SAM attacks survived by an aircraft executing the maneuvers specified by the corresponding best-of-run program, are tabulated in Figure 2.

As these results indicate, best-of-run programs evolved using fixed fitness cases during Training Runs 6-10 frequently proved to be more brittle during subsequent testing than programs evolved using fixed fitness cases during Training Runs 1-5. In particular, programs optimized during Training Runs 7 and 9 survived a much smaller percentage of SAMs from the test population than any of the programs evolved during Training Runs 1-5. The poor performance of Training Runs 7 and 9 may be attributed to *overtraining* with a less representative fixed training population.

These test results suggest that the brittleness of best-of-run programs may be reduced by careful selection of the fixed fitness cases used to train the MCO System. For many GP applications involving complex and dynamic test environments, however, identification of a fixed training population suitable for evolving robust best-of-run programs can prove to be an extraordinarily complex task [9]. Preliminary experiments for the more sophisticated MCO problem of evolving programs that combine

Training Population	Tests 1	2	3	4	5	6	7	8	9	10
FIXED:	85.9	85.2	82.8	78.1	86.7	89.1	70.3	82.8	68.0	85.9
RANDOM:	82.0	86.7	84.4	80.5	80.5	82.0	87.5	82.8	89.1	89.1

Figure 2. Survivability (%) of Best-of-Run Programs Against the Test Population

maneuvers with additional countermeasures (such as chaff, flares, and jamming) under various conditions of uncertainty [27] appear to substantiate this claim.

A comparison of the *mean* and *variance* of MCO System test results is revealing. For fixed training cases, Tests 1-5 resulted in a mean aircraft survivability of 83.7% with a variance of 9.7%. When a new training population was generated prior to the fitness evaluation of each generation, however, Tests 1-5 resulted in a mean survivability of 82.8% and a variance of only 5.8%. Similarly, for fixed training cases, Tests 6-10 resulted in a mean fitness of 79.2% and a huge variance of 71.2%, while for randomly generated training populations Tests 6-10 resulted in a mean fitness of 86.1% and a variance of 9.5%. These results strongly suggest that *variance* in the fitness of best-of-run programs can be significantly reduced by changing the training population during program evolution.

Figure 2 also illustrates that the use of random fitness cases in Tests 1-5 did *not* appear to improve the mean fitness of best-of-run programs; the average survivability of programs evolved using either fixed or random fitness cases in Tests 1-5 differed by less than 1%. This result is attributed to the fact that the fixed training population of Training Runs 1-5 provided the MCO System with a reasonably accurate representation of the test environment. In contrast, the mean performance of programs evolved using randomly generated training sets during Training Runs 6-10 was *nearly 7% better* than that of programs evolved using fixed training sets, when subsequently tested against a large, representative test population. By randomly generating a new training population prior to the fitness evaluation of each generation of programs, the modified MCO System was able to overcome the potential lack of representativeness of the relatively small fixed training population [26]. The results of Tests 6-10 suggest that the new training methodology may also prove useful in improving mean program performance, potentially reducing the brittleness of best-of-run programs evolved for a variety of GP applications.

4. Generalization of Results

The results summarized above suggest that the brittleness of best-of-run programs evolved by GP systems can be reduced by creating a new training population of randomly generated fitness cases prior to the evaluation of each generation of programs. The methodology summarized in this paper is particularly beneficial when the corresponding fixed training population does not adequately represent the full range of difficult situations that may arise during subsequent program testing. Statistical analysis of results for the MCO System indicate a significant reduction in *variance* due to this approach to program training. Reduction in variance means that a much higher percentage of best-of-run programs are likely to represent sufficiently optimized solutions – i.e., the likelihood that GP will evolve a program that is incapable of exhibiting optimized performance will be significantly reduced. Improvement in the mean fitness of best-of-run programs indicates that this alternative training methodology can also help overcome the detrimental effects of using a small, less representative fixed training population. Future investigations into the MCO problem may be able to exploit formal methods of *statistical inference* [35] and *analysis of variance* [30] to identify more powerful training sets. These results, together with complementary results reported by other GP researchers (as summarized above), strongly encourage the application of this methodology to a wide range of complex strategy optimization problems.

Acknowledgments

Much of the research described in this paper was performed at Wright State University (WSU) under the direction of Oscar Garcia, with support from the Dayton Area Graduate Studies Institute. Gary Lamont (Air Force Institute of Technology), Mateen Rizki (WSU), Ray Siferd (WSU), and Thomas Sudkamp (WSU) provided additional feedback. This paper has also benefited from lively debate on the Genetic Programming electronic mailing list; special thanks to Tom Bersano-Begey, Arjun Bhandari, Kumar Chellapilla, Jason Daida, David Faulkner, Chris Gathercole, David Goldberg, Frank Hoffman, Ibrahim Kuscu, Bill Langdon, Sean Luke, Nic McPhee, Ric Riolo, David Schaffer, Eric Siegel, and Lee Spector.

Bibliography

[1] Ahluwalia, M., L. Bull, and T. C. Fogarty, 1997. "Co-evolving Functions in Genetic Programming: A Comparison in ADF Selection Strategies", in Koza, J. R. *et al* (eds.), *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pp. 3-8, Morgan Kaufmann Publishers, Inc.

[2] Angeline, P. and J. Pollack, 1993. "Competitive environments evolve better solutions for complex tasks", in Forrest, S. (ed.), *Genetic Algorithms: Proceedings of the Fifth International Conference (GA 93)*, pp. 264-270, San Mateo, CA, Morgan Kaufmann Publishers, Inc.

[3] Barron, R. L., 1995. "Reduced-Computation End-Game Steering Laws for Predictive Guidance", in Journal of Guidance, Control, and Dynamics, Vol. 18 No 2, March-April 1995, pp. 306-315, American Institute of Aeronautics and Astronautics, Inc.

[4] Bersano-Begey, T. F. and J. M. Daida, 1997. "A Discussion on Generality and Robustness and a Framework for Fitness Set Construction in Genetic Programming to Promote Robustness", in Koza, J. R. (ed.), *Late-Breaking Papers at the Genetic Programming 1997 Conference, Stanford University, July 13-16, 1997*, pp. 11-18, Stanford Bookstore.

[5] Bryson, A. E. and Y. Ho, 1969. *Applied Optimal Control*, Blaisdell Publishing Company.

[6] Cliff, D. and G. F. Miller, 1995. "Tracking the red queen: Measurements of adaptive progress in co-evolutionary simulations", in Moran, F., A. Moreno, J. J. Merelo, and P. Chacon (eds.), *Advances in Artificial Life: Proceedings of the Third European Conference on Artificial Life*, pp. 200-218, Springer-Verlag.

[7] Cliff, D. and G. F. Miller, 1996. "Co-evolution of Pursuit and Evasion II: Simulation Methods and Results", in Maes, P. *et al* (eds.), *From Animals to Animats IV: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*, MIT Press-Bradford Books.

[8] Cullen, T. and C. Foss, 1995. *Jane's Land-Based Air Defence: 1995-1996*, Jane's Information Group, Inc.

[9] Daida, J. M., T. F. Bersano-Begey, S. J. Ross and J. F. Vesecky, 1996. "Computer-Assisted Design of Image Classification Algorithms: Dynamic and Static Fitness Evaluations in a Scaffolded Genetic Programming Environment", in Koza, J. R., D. E. Goldberg, D. B. Fogel, and R. L. Riolo (eds.), *Genetic Programming 1996: Proceedings of the First Annual Conference*, pp. 279-284, MIT Press.

[10] Darwen, P. J. and X. Yao, 1995. "On Evolving Robust Strategies for Iterated Prisoner's Dilemma", in Yao, X. (ed.), *Progress in Evolutionary Computation*, Lecture Notes in Artificial Intelligence Vol. 956, pp. 276-292.

[11] Fogel, D. B., 1993. "Evolving behaviors in the iterated prisoner's dilemma", Evolutionary Computation, Vol. 1 No. 1, pp. 77-97.

[12] Floreano, D. and S. Nolfi, 1997. "God Save the Red Queen! Competition in Evolutionary Robotics", in Koza, J. R. *et al* (eds.), *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pp. 398-406, Morgan Kaufmann Publishers, Inc.

[13] Gathercole, C., 1997. Responses to messages posted on the Genetic Programming electronic mailing list, March 25, 1997 and September 1, 1997.

[14] Gathercole, C. and P. Ross, 1994. "Dynamic Training Subset Selection for Supervised Learning in Genetic Programming", in Davidor, Y., H.-P. Schwefel and R. Manner (eds.), *Parallel*

Problem Solving from Nature III, Springer-Verlag.

[15] Gathercole, C. and P. Ross, 1997. "Small Populations over Many Generations can *beat* Large Populations over Few Generations in Genetic Programming", in Koza, J. R. *et al* (eds.), *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pp. 111-118, Morgan Kaufmann Publishers, Inc.

[16] Hillis, D., 1992. "Co-evolving parasites improves simulated evolution as an optimization procedure", in Langton, C., C. Taylor, J. Farmer, and S. Rasmussen (eds.), *Artificial Life II*, pp. 313-324, Addison-Wesley Publishing Company, Inc.

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

[18] Lambert, M. and K. Munson (eds.), 1994. *Jane's All the World's Aircraft: 1994-1995*, Jane's Information Group, Inc.

[19] Lindgren, K., 1991. "Evolutionary Phenomena in Simple Dynamics", *Santa Fe Institute Studies in the Sciences of Complexity*, Vol. 10, pp. 295-312, Addison-Wesley.

[20] Luke, S., 1997. Response to a message posted on the Genetic Programming electronic mailing list, March 25, 1997.

[21] McNutt, G., 1997. "Using Co-Evolution to Produce Robust Robot Control", in Koza, J. R. (ed.), *Late-Breaking Papers at the Genetic Programming 1997 Conference, Stanford University, July 13-16, 1997*, pp. 141-149, Stanford Bookstore.

[22] Miller, J. H., 1989. "The Co-evolution of Automata in the Repeated Prisoner's Dilemma", Santa Fe Institute Report 89-003.

[23] McPhee, N., 1997. Response to a message posted on the Genetic Programming electronic mailing list, September 1, 1997.

[24] Moore, F. W., 1997. A Methodology for Strategy Optimization Under Uncertainty, Ph.D. Dissertation, Computer Science & Engineering, Wright State University.

[25] Moore. F. W. and O. N. Garcia, 1997a. "A Genetic Programming Approach to Strategy Optimization in the Extended Two-Dimensional Pursuer/Evader Problem", in Koza, J. R. *et al* (eds.), *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pp. 249-254, Morgan Kaufmann Publishers, Inc.

[26] Moore. F. W. and O. N. Garcia, 1997b. "A New Methodology for Reducing Brittleness in Genetic Programming", in *Proceedings of the National Aerospace and Electronics 1997 Conference* (NAECON-97), IEEE Press.

[27] Moore. F. W. and O. N. Garcia, 1998. "A Genetic Programming Methodology for Missile Countermeasures Optimization Under Uncertainty", submitted to the Seventh Annual Conference on Evolutionary Programming (EP-98), March 25-27, 1998, acceptance pending.

[28] Park, S. K. and K. W. Miller, 1988. "Random number generators: Good ones are hard to find", in Communications of the ACM, Vol. 31 No. 10 (October 1988), Association for Computing Machinery.

[29] Ramo, S. and A. Pucket, 1959. *Guided Missile Engineering*, McGraw-Hill, pp. 176-180.

[30] Rider, P. R., H. L. Harter, and M. D. Lum, 1956. "An elementary approach to the analysis of variance", WADC Technical Report 56-20, Wright-Patterson Air Force Base, OH.

[31] Siegel, E. V., 1994. "Competitvely Evolving Decision Trees", in Kinnear, K. E. Jr. (ed.), *Advances in Genetic Programming*, pp. 409-423, MIT Press.

[32] Sims, K., 1994. "Evolving 3-D Morphology and Behavior by Competition", in Brooks, R. and P. Maes (eds.), *Proceedings of the Fourth Workshop on Artificial Life*, pp. 28-39, MIT Press.

[33] Spector, L., 1997. Responses to messages posted on the Genetic Programming electronic mailing list, March 25, 1997 and September 1, 1997.

[34] Stevens, B. L. and F. L. Lewis, 1992. *Aircraft Control and Simulation*, John Wiley & Sons.

[35] Winer, B. J., D. R. Brown, and K. M. Michels, 1991. *Statistical Principles in Experimental Design*, Third Edition, McGraw-Hill, Inc.

[36] Yuan, C. L., 1948. "Homing and Navigation Courses of Automatic Target-Seeking Devices", in Journal of Applied Physics, Vol. 19, December 1948, pp. 1122-1128.