# Evolving SoccerBots: A Retrospective

**Sean Luke**
seanl@cs.umd.edu
http://www.cs.umd.edu/~seanl

Department of Computer Science
University of Maryland
College Park, MD 20742 USA

## ABSTRACT

In the RoboCup97 robot soccer tournament, we entered a team of softbot programs whose player strategies had been entirely learned by computer. Our team beat other human-coded competitors and received the RoboCup97 Scientific Challenge award. This paper discusses our approach, and details various ways that, in retrospect, it could have been improved.

## 1 Introduction

The RoboCup simulation tournament pits teams of softbot programs against each other in simulated soccer matches, promoting artificial intelligence strategies that are robust in dynamic, noisy environments. For the RoboCup 1997 simulation tournament, most teams refined well-understood robotics techniques in order to win the competition. Almost every team used hand-coded decision algorithms, though some optimized a few low-level functions (like ball interception) with backpropagation or decision trees. In contrast, we ([Luke *et al* 1998a]) entered a softbot team whose high-level strategies were entirely learned through *genetic programming*, an evolutionary computation method.

For many reasons, it is not easy to combine evolutionary computation with real-time environments like robotics. Hence, our goal was relatively modest: to produce a team which played at all. As it turned out, we were pleasantly surprised with the results. Our evolved teams learned to disperse throughout the field, pass, kick to the goal, defend the goal, and coordinate with and defer to other teammates. At the IJCAI/RoboCup97 competition our team managed to win its first two matches against human-coded opponents, and took home the RoboCup97 Scientific Challenge award.

As this was an early attempt to apply genetic programming to such a difficult robotics endeavor, there are certainly many things we feel we could have improved. As other researchers are now trying similar approaches, in this paper I discuss what we did, and more importantly, what we could have done better.

## 2 What We Did

Our goal was to use genetic programming (GP) [Koza 1992] to evolve high-level decision behaviors for an entire team of RoboCup softbots. Genetic programming is an evolutionary computation method which searches for good Lisp programs to solve some particular problem. Lisp programs can be thought of as trees, where each node in the tree is a Lisp function: a node's children are arguments to its function. GP searches for the most fit program for a given problem (in our case, operating a soccer softbot in the RoboCup Soccer Server). To do this, it first creates a random population of program trees, then tests them by executing their code in the environment, selects the fitter ones, and "breeds" them (by swapping random subtrees among highly-fit programs, or replacing subtrees with randomly generated ones) to produce a new generation of trees, which go on to be tested, selected, bred, and so on. After repeating this process many times (50 is common, with a population size of 2000), GP often can produce highly fit solutions for the given problem.

We supplied our genetic programming system with a set of low-level atomic functions designed for the soccer environment (vectors to the goal and ball and to other players, if-then style decision functions, predicates indicating state of play, vector math functions, etc.) The GP system used this function set to build its programs. In our approach, these program trees did not represent individual players but entire *teams*. We tried two strategies here. First, we tried *heterogeneous* teams, where each GP program had different sections which were assigned to different players (allowing each player to develop its own unique strategy). We also tried *homogeneous* teams, where each GP program was a single soccer-playing program that every player in the team used. In the end, our homogeneous teams played better by competition-time.

A team's fitness was determined by playing actual Soccer Server matches against other teams in the population. This approach, recently known as *competitive fitness* [Angeline and Pollack 1993], allows the problem to naturally get harder as the individuals in the population get better: this is a very nice attribute, and it worked well for us.

Evolving GP with robotics is a major challenge. A pop-

ulation of 2000 teams, running for 50 generations means at least 50,000 10-minute-long soccer matches—this could potentially take a *very* long time. In order to get results in only a few months' evaluation time, we took a variety of measures to make it possible to run with a small population (under 200), with short matches (20 seconds on average) and a small number of generations (about 40), running in parallel.

During the run, we made many interesting, though anecdotal, observations. In the beginning, teams moved arbitrarily, though most teams contained one or two players which would go to the ball and kick it to the goal (because vectors to the ball and to the goal were basic functions in the function set). Initial teams had very poor defense, which led to an early successful strategy: everyone runs to the ball and kicks it toward the goal ("kiddie-soccer"). Later this strategy would fail as teams began to develop simple defenses, with a few players protecting the goal. Ultimately teams learned to distribute throughout the field with more global strategies.

The details of our approach, compromises, and results can be found in [Luke *et al* 1998a] and (with more GP-specific details) [Luke *et al* 1998b].

# 3 What We Could Have Done Better

For this first-time attempt, I am very pleased with the results. Nonetheless, we made many compromises that in retrospect I feel may have been too conservative.

**Our Population Size Was Too Small.** This is perhaps an obvious complaint, but it needs to be said: a population size of 200 or less is simply too small for this problem. To compensate for a small population size, we had to make many compromises in terms of selection, breeding, and fitness evaluation which no doubt resulted in lower-fitness teams. In the future one should make larger populations a first priority.

**Our Teams Competed Too Infrequently.** In order to reduce the number of evaluations, we evaluated the fitness of a population by pairing off teams and having each pair play a game. Each team's fitness was based on a *single* game with an opposing team. The disadvantage of this is that teams often don't get a fair assessment. In the future I would determine fitness by playing against many more members of the population, perhaps using a single-elimination tournament.

**We Evolved Teams And Not Individuals.** The advantage of evolving a whole team at once is that you don't have to deal with the *credit assignment problem*: determining which players get credit or blame for the whole team's success or failure. But it can require more evolution time to come up with a whole team that works well together (because all the separate pieces must be evolved together). With the introduction of the goalie in the 1998 RoboCup tournament, this approach will be even more difficult.

I think that a more interesting approach may be to evolve attackers, middlemen, defenders, and goalies in separate populations, assessing their fitness by grouping together players from different populations and playing them as a team against other such teams. At the very least, credit assignment can be dealt with by giving every player an equal share of credit for success or failure.

This approach would move GP soccer from competitive fitness into true *co-evolution*: attackers are evolving to beat goalies better, defenders are evolving to block attackers better, and so on. This approach is interesting because it naturally takes advantage of the close relationship between competition and cooperation. Players from different populations must cooperate with one another in order to successfully compete against other teams.

**Our Function Set Was Too Biased, And Provided No Internal State.** It is often the case in GP that the choice of atomic functions can make all the difference in GP's success or failure. We were very conservative with our function set, biasing it somewhat towards "soccer-specific" functions, rather than using a more generalized approach. This increased the likelihood that we would get a *feasible* soccer strategy, but it's often the case that such environment-specific function sets eliminate truly *good* strategies because they include the (incorrect) biases of their creators. This is a compromise which is difficult to gauge. In retrospect, I would have included more general vector and decision functions.

Another problem with our function set was that it provided no internal state. This meant that our players were wholly reactive, acting on only the current state of the world. In contrast, other entrants were able to devise sophisticated setplays because their players had internal state that helped them to perform a complex sequence of actions in concert. We decided not to use internal state because it is difficult to use with GP. But I think this was too serious a compromise: as a result our teams were able to devise only simple strategies.

Despite these compromises, I think the project was a success. In the end we were fortunate to have good results and interesting observations along the way. In the future I hope address these compromises with new, even more competitive GP learning strategies. In the meantime, I wish any new GP RoboCup98 entrants the best.

# Bibliography


Angeline, P. and J. Pollack. 1993. Competitive Environments Evolve Better Solutions for Complex Tasks. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, S. Forrest, ed. 264–270. Morgan Kaufmann, San Mateo CA.

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

Luke, S. *et al*. 1998 (a). Co-evolving Soccer Softbot Team Coordination with Genetic Programming. *In RoboCup-97: Robot Soccer World Cup I (Lecture Notes in Artificial Intelligence No. 1395)*, H. Kitano, ed. Berlin: Springer-Verlag. 398–411.

Luke, S. *et al*. 1998 (b). Genetic Programming Produced Competitive Soccer Softbot Teams for RoboCup97. *Genetic Programming 1998: Proceedings of the Third Annual Conference*, Koza, J.R. *et al*, eds. San Fransisco: Morgan Kaufmann.