The Simple Soccer Machine Learning Environment

Jeff Riley
School of Computer Science and Information Technology
RMIT University
Melbourne, AUSTRALIA
jeff.riley@optushome.com.au

Abstract- The RoboCup simulated soccer league is a dynamic, complex and uncertain environment which presents many challenges to machine learning techniques. The asynchronous design of the RoboCup simulation environment can create long and unpredictable delays in the effects of actions, often causing onerous training times. A new environment known as Simple Soccer is proposed which, while retaining much of the dynamics and complexity of RoboCup, provides more certainty and less delay, thus increasing the viability of machine learning techniques. The goal of this work is to create an environment complex and dynamic enough that while low-level tactics may differ due to the removal of systematic uncertainty, high-level strategies directly applicable to the RoboCup environment can be developed. This paper specifies the Simple Soccer environment, and presents some initial learning results.

1 Introduction

The RoboCup simulated soccer league is an important and useful tool for multi-agent and machine learning research (Kitano 1995, Kitano 1997) which provides a distributed, multi-agent environment in which agents have an incomplete and uncertain world view. The agent perception and action cycles in the RoboCup environment are asynchronous, sometimes resulting in long and unpredictable delays in the completion of actions in response to some stimuli. These delays, as well as the uncertain and incomplete world view of the agents, can increase the learning cycle of some machine learning techniques onerously.

There is a large body of work in the area of the application of machine learning techniques to the challenges of RoboCup (e.g. Uchibe 1999, Stone 1999, Stone 2001, Riedmiller 2001, Luke 1998a, Luke 1998b), but because the RoboCup environment is so large, complex and unpredictable the extent to which such techniques can meet these challenges is not certain. More progress would be made more quickly if the complexity and uncertainty could be reduced. Simple Soccer is proposed as environment that reduces complexity and uncertainty sufficiently to increase the viability of machine learning techniques, yet retains sufficient complexity and dynamics to allow learnings from Simple Soccer to be directly transferrable to the RoboCup environment.

2 Goal

The primary goal of the work presented in this paper is to create an environment complex and dynamic enough that while low-level tactics may differ due to the removal of systematic uncertainty, high-level strategies directly applicable to the RoboCup environment can be developed, and to investigate the usefulness of using such a simplified system to produce coarsely trained players that can be further trained in the more uncertain RoboCup environment.

3 Simple Soccer

The environment developed for this work, named Simple Soccer, was inspired by the Ascii Soccer environment (Balch 1995). The soccer field in Simple Soccer is represented by a two-dimensional grid with player and the ball locations specified by discrete grid co-ordinates, or *cells*. Whereas for Ascii Soccer the goal area is the entire width of the playing field at each end, the goal area for Simple Soccer is a defined area at each end of the field, more in keeping with the RoboCup field.

Player movement and sensory capabilities in Simple Soccer are more aligned with those of RoboCup than Ascii Soccer. Whereas for Ascii Soccer player movement is limited to eight discrete directions, in Simple Soccer players may move in any direction, specified by a real-valued angle from -360.0^o to +360.0^o relative to the player's current facing direction. Similarly, the ball can be kicked in any direction. At the completion of an action, player and ball final locations are quantized to discrete cells. Unlike Ascii Soccer, in which a player's sensory capability is limited to the cells immediately adjacent to the player, players in Simple Soccer have a field of vision similar to that of RoboCup. A player's field of vision is a cone-like area specified by the depth of the field and the angle around the centre-line. Players are presented with the cell co-ordinates, direction and distance (number of cells) of any object of interest (ball, player or goal) in the player's field of vision.

The specific actions available to players are:

turn(*direction*): the player turns through the angle specified by *direction*.

dash(*direction, power, face*): the player dashes in the direction specified with the power specified.

kick(*direction, power, face*): if the ball is within a kickable distance from the player, the player kicks the ball in the direction specified with the power specified

For each of these actions:

direction is specified in degrees in a clockwise direction relative to the direction the player is facing.

power is specified as a percentage of maximum power and determines the number of cells the player or ball will travel as a result of the action.

face, where specified, if true, causes the player to turn to face in the direction specified.

A Simple Soccer unit of time is a single *tick* which corresponds to one iteration of the program's main loop. At each tick the ball and players are moved, if necessary, a single cell (as a result of a previous action) and each player is presented with their new (visual) view of the state of the game, whereupon each player determines what action, if any, is to be taken and that action is begun (any previous action still in progress is superseded by the new action).

There may be a maximum of eleven players per team in Simple Soccer, and team sizes may be uneven. There is no referee; there are no free kicks for *offside* or other rule violations. The ball is never out of bounds; the boundaries (except for the goal areas) are hard barriers. Unlike the RoboCup environment, in Simple Soccer there is no uncertainty (or introduced randomness) in response to actions, there is no momentum or stamina, and there is no loss of clarity of vision over distance etc

4 Experiment Description

The experiments performed are based on those described in previous work involving the RoboCup simulation league (Riley 2002).

4.1 Overview

Learning classifier systems (Holland 1986) are an example of genetic algorithms (Holland 1975) incorporated into models of complex systems, where the classifier systems are used as models of behaviour ranging from simple stimulus-response to more complex cognitive behaviour. Classifier systems implement hierarchies of internal models that represent the environment, and the genetic algorithm uses intermittent feedback from the environment in order to discover the rules that represent those hierarchies.

This work implements a method involving the use of a messy genetic algorithm (Goldberg 1989) and a fuzzy inference system (Zadeh 1965) in which the messy genetic algorithm is used to determine, by simulated evolution, the fuzzy ruleset which defines the set of behaviours exhibited by reactive agents (players) in response to stimuli.

In addition to the Simple Soccer primitives of ***turn***, ***dash*** and ***kick***, a set of mid-level actions constructed from those primitives is provided for the agent being evolved. These are:

runTowardBall: the player *dashes* once in the direction of the ball, provided the direction to the ball is known.

goToBall: the player *dashes* towards ball until it is within kicking distance of the ball, provided the direction to the ball is known.

runTowardGoal: the player *dashes* once in the direction of its goal, provided the direction to the goal is known.

kickTowardGoal: the player *kicks* the ball once towards its goal, provided the direction to the goal is known.

dribble: the player *kicks* the ball once in the direction it is facing, then *dashes* once in that direction.

dribbleTowardGoal: the player *kicks* the ball once in the direction of its goal, then *dashes* once in that direction, provided the direction to the goal is known.

doNothing: the player takes no new action.

The player will perform one of these actions in response to external stimuli; the specific response being determined by the fuzzy ruleset. If no action is indicated given the information known by the player (that is, no rule fires) and the ball is not visible to the player, the player will dash in a randomly chosen direction in an effort to locate the ball.

The external stimuli used as input to the fuzzy inference system is the visual information supplied by the Simple Soccer server.

4.2 Detail

Input variables for the fuzzy rules developed by this method are fuzzy interpretations of the visual stimuli supplied to the agent by the soccer server. Output variables are the fuzzy actions to be taken by the agent. The universe of discourse of both input and output variables are covered by fuzzy sets, the parameters of which are predefined and fixed. Each input is fuzzified to have a degree of membership in the fuzzy sets appropriate to the input variable.

The encoding scheme implemented for this method exploits the capability of messy genetic algorithms to encode information of variable structure and length. The basic element of the coding of the fuzzy rules is a triplet representing a fuzzy clause and connector, with the first element denoting the input variable, the second the fuzzy set membership (or fuzzy variable) of this input variable, and the third the clause connector. The rule consequent gene is specially coded to distinguish it from premise genes allowing multiple rules, or a ruleset, to be encoded onto a single chromosome. Chromosomes are not fixed length: the length of each chromosome in the population varies with the length of individual rules and the number of rules on the chromosome. The number of clauses in a rule and the number of rules in a ruleset is only limited by the maximum size of a chromosome. The minimum size of a rule is two clauses (one premise and one consequent), and the minimum number of rules in a ruleset is one.

The set of input variables for the premise clauses is:

(Ball, Goal)

and for the consequent clauses:

(turn, kick, dash, runTowardBall, goToBall, runTowardGoal, kickTowardGoal, dribbleTowardGoal, dribble, doNothing)

The fuzzy variables for each of the fuzzy sets DISTANCE, POWER and DIRECTION which describe the input or action variables for both the premise and consequent clauses are:

DISTANCE: (At, Very Near, Near, Slightly Near, Medium, Slightly Far, Far, Very Far)

POWER: (Very Low, Low, Slightly Low, Medium, Slightly High, High, Very High)

DIRECTION: (Left180, Very Left, Left, Slightly Left, Straight, Slightly Right, Right, Very Right, Right180)

Each of these can be further modified by the use of a *not* operator.

The set of possible clause connectors is:

(and, or, *)

where * indicates the connector is not used – for example, in the final premise and consequent clauses of a rule. An example chromosome and corresponding rules are shown in Figure 1.

The genetic operators implemented are cut, splice and mutation. Cut and splice are analogous to the crossover operation of classic genetic algorithms; the mutation operator is the same as that of the classic genetic algorithm. Since chromosomes are variable in length and can contain multiple rules, each chromosome represents a complete ruleset.

No explicit *don't care* values are implemented for any attributes in this method. Since messy genetic algorithms encode information of variable structure and length not all attributes, particularly premise variables, need be present in any rule, or indeed in the entire ruleset. In other words the format of the messy genetic algorithm implies *don't care* values for all attributes since any attribute (premise variable) may be omitted from any or all rules, so generalisation is an implicit feature of this method.

5 Results

Some initial trials for both RoboCup and Simple Soccer were performed for comparison. Each trial consisted of a population of 200 randomly initialised chromosomes evolved over 10 generations. The best performing player from the Simple Soccer trials was then used to seed the population for a second RoboCup trial. For this seeded trial, 50 individuals from the initial population were initialised to the seed chromosome, with the remainder randomly initialised. The RoboCup seeded trial was also conducted over 10 generations.

(B,N,O)	(B,nF,A)	(G,N,*)	▨ (RB,S,*)	(B,A,A)	(G,vN,*)	▨ (KG,M,*)	(B,F,*)	▨ (GB,vF,*)

Premise	▨ Consequent

Rule 1: if *Ball* is *Near* or *Ball* is *not Far* and *Goal* is *Near* then *runTowardBall Slow*
Rule 2: if *Ball* is *At* and *Goal* is *Very Near* then *kickTowardGoal Medium*
Rule 3: if *Ball* is *Far* then *goToBall Very Fast*

Figure 1 Chromosome and corresponding rules

For each of these trials:

- The *Roulette Wheel* method of selection for crossover was used, and the probability of crossover occurring after selection was *0.8*.
- Each generation was mutated by selecting *10%* of the population for possible mutation, then subjecting those selected individuals to a probability of mutation of *0.35*, so a maximum of *3.5%* of the population was mutated. For each individual, a single gene was randomly selected for mutation: for a premise gene the input variable, fuzzy variable or connector was mutated; and for a consequent gene the input variable or fuzzy variable was mutated. Mutation consisted of replacement by a randomly selected value.

Individuals were rewarded, in order of importance, for

- the number of goals scored in a game
- the number of times the ball was kicked during a game

A game was played with the only player on the field being the agent under evaluation. The player was placed randomly on its half of the field and oriented so that it was facing the end of the field to which it was kicking.

A Simple Soccer game was terminated when any of the following was true:

- the target fitness of *0.05* was reached
- *120* seconds expired
- *5* seconds elapsed with no *dash* executed
- *5* seconds elapsed with no *kick* executed

Similarly, a RoboCup game was terminated when any of the following was true:

- the target fitness of *0.05* was reached
- *120* seconds expired
- *5* seconds elapsed with no *dash* executed
- *10* seconds elapsed with no *kick* executed
- the ball was kicked out of play

The increase in time to wait for a kick to be executed for RoboCup reflects the uncertainty of that environment.

Two methods of terminating the evolutionary search were implemented. The first stops the search when a specified maximum number of generations have occurred; the second stops the search when the best fitness in the current population becomes less than a specified threshold. Both methods were active, with the first to be encountered terminating the search.

Figure 2 shows the average fitness of the population after each generation for each of the trials. Figure 3 shows the best individual fitness from the population after each generation for each of the trials.

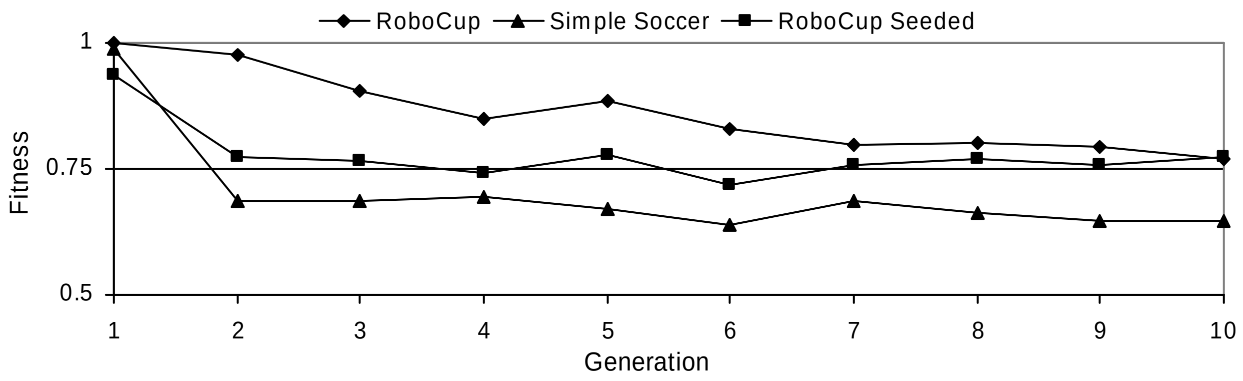

Figure 2 Average fitness per generation.

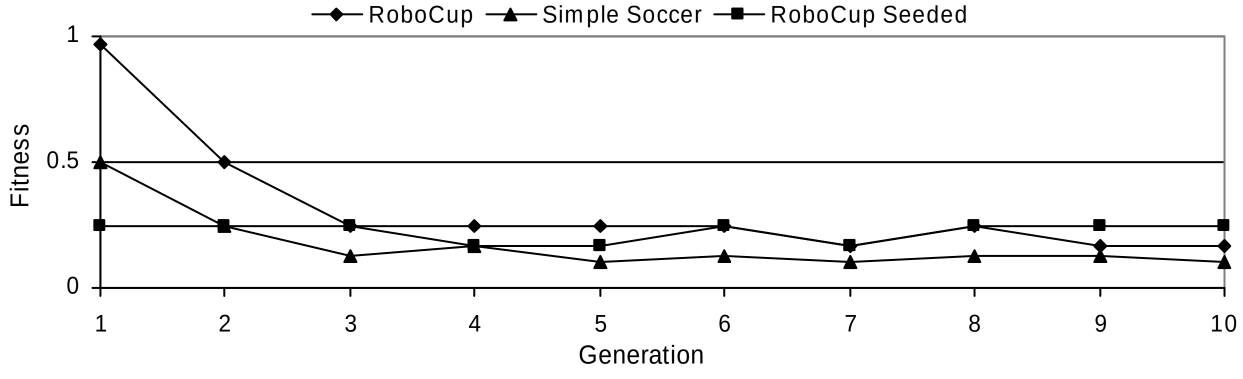

Figure 3 Best fitness per generation.

Table 1 shows the training times for each of the trials. It is clear from these times that players in the Simple Soccer environment are able to learn much faster than in the RoboCup environment. This is almost certainly due to the removal of uncertainty from the environment.

	Gens	**Elapsed Minutes**	**Minutes/ Generation**
RoboCup	10	2430	243
Simple Soccer	10	340	34
RoboCup Seeded	10	2280	228

Table 1 Training times.

The best performing individual from the Simple Soccer trial was defined by the following rules:

if Goal is Medium Distant or Ball is Very Left or
Ball is Far or goal is Slightly Left and
Goal is Very Right and Ball is Right180 or
Goal is Slightly Near
then runTowardBall Slightly Low

if Goal is Straight or Goal is Very Right or
Ball is not Slightly Left or Goal is Very Near or
Goal is Slightly Right and Goal is Left
then kickTowardGoal Soft

if Ball is not Slightly Left or Goal is Slightly Left
then dribbleTowardGoal Hard

if Goal is Slightly Left or Goal is Slightly Far and
Ball is Near or Ball is Far and Ball is Left or
Goal is At and Goal is Right180 or
Goal is Medium Distant and Ball is Very Far and
Goal is Straight
then turn Left

if Ball is Left180
then doNothing

The player defined by this ruleset achieved a fitness value of *0.1* during training by kicking *5* goals in the allotted time of *120* seconds. In subsequent tests in the Simple Soccer environment kicked one or more goals in 56% of the trials conducted, whereas the same player tested in the RoboCup environment kicked one or more goals in 44% of the trials conducted.

6 Conclusions

The goal of this work was to create an environment with similar complexity and dynamics to the RoboCup simulated soccer environment but with reduced uncertainty, both in agents' perception and in their interaction with the environment. The motivation was to create an environment in which the training times of machine learning techniques would be reduced sufficiently so as to improve the viability of such techniques. The Simple Soccer environment was proposed as a solution, and through some sample experiments it was shown that the Simple Soccer environment does aid in the reduction of training times for some machine learning techniques. High-level strategies learned in the more certain Simple Soccer environment are directly transferrable to the RoboCup environment, and when used as the starting point for further learning can help to reduce the training time in the RoboCup environment.
Further work needs to be done in order to generalise these results to other machine learning techniques, as well as work to identify how best to transfer the learning from Simple Soccer to RoboCup in a way that maximises the benefit.

Bibliography

Balch, T. *The Ascii Robot Soccer Home Page.* http://www-2.cs.cmu.edu/~trb/soccer/ 1995.

Goldberg, D., Korb, B., and Deb, K. *Messy Genetic Algorithms: Motivation, Analysis, and First Results.* In Complex Systems, 3, 1989.

Holland, J. *Adaptation in Natural and Artificial Systems.* Ann Arbor: The University of Michigan Press, 1975.

Holland, J., Holyoak, K., Nisbett, R., Thagard, P. *Induction: Processes of Inference, Learning, and Discovery.* MIT Press, 1986.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., and Osawa, E. *RoboCup: The Robot World Cup Initiative.* In Proceedings of the 1995 International Joint Conference on Artificial Intelligence (IJCAI'95), Workshop on Entertainment and AI/ALife, Montreal, Canada, 1995.

Kitano, H., Tambe, M., Stone, P., Veloso, M., Coradeschi, S., Osawa, E., Matsubara, H., Noda, I. and Asada, M. *The RoboCup Synthetic agent Challenge 97.* In Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, pp24-29, San Francisco CA, USA, 1997.

Luke, S. *Genetic Programming Produced Competitive Soccer Softbot Teams for RoboCup97.* In Proceedings of the Third Conference on Genetic Programming, Madison WI, USA, 1998(a).

Luke, S. *Evolving SoccerBots: A Retrospective.* In Proceedings of the Twelfth Annual Conference of the Japanese Society for Artificial Intelligence, Tokyo, Japan, 1998(b).

Riedmiller, M., Merke, A., Meier, D., Hoffman, A., Sinner, A., Thate, O. and Ehrmann, R. *Karlsruhe Brainstormers – a Reinforcement Learning Approach to Robotic Soccer.* In Peter Stone, Tucket Balch and Gerhard Kraetszchmar, editors, RoboCup-2000: Robot Soccer World Cup IV. Springer Verlag, Berlin, 2001

Riley, J. and Ciesielski, V. *Evolving Fuzzy Rules for Reactive Agents in Dynamic Environments.* In Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution And Learning , Singapore, 2002.

Stone, P. and Sutton, R. *Scaling Reinforcement Learning Toward RoboCup Soccer.* In Proceedings of the Eighteenth International Conference on Machine Learning, Williamstown MA, USA, 2001.

Stone, P. and Veloso, M. *Team-partitioned, Opaque-transition Reinforcement Learning.* In Proceedings of the Third International Conference on Autonomous Agents, Seattle WA, USA, 1999.

Uchibe, E. *Cooperative Behavior Acquisition by Learning and Evolution in a Multi-Agent Environment for Mobile Robots.* PhD thesis, Osaka University, 1999.

Zadeh, L. *Fuzzy Sets.* Journal of Information and Control, Vol 8, 1965.