Shaped Prisoner’s Dilemma Automata

Wendy Ashlock and Daniel Ashlock

Abstract—Finite state automata are one of the common representations used to encode agents to play the iterated prisoner’s dilemma. A shape for a finite state automaton is a restriction on what transitions are permitted out of each state in the automata. Eight shapes, including a baseline shape with all possible transitions, are tested by enforcing the shape during the training of agents that play iterated prisoner’s dilemma with an evolutionary algorithm. All eight shapes yield distinct distributions of behaviors in the evolved agents.

I. INTRODUCTION

Previous work has shown that different representations for game playing agents can yield very different behaviors [6], [1]. In addition, it has been shown that changing the number of states allowed to a finite state machine can substantially change the behavior of evolved agents [2]. Both of these results suggest that when game playing agents are evolved as part of a behavioral or social model then the choice of agent representation must be checked to see if it reasonably models the real-world target. In this study, we show that it is possible to generate multiple, very different behaviors using finite state machines with the same number of states. The variation of behavior obtained in this study rivals that in the earlier studies on representation. This means that the process of specializing game playing agents to a particular range of behaviors may be possible using the finite state representation.

Inducing large behavioral variability in finite state machines results from restricting the structure of finite state machines by specifying a restricted collection of possible transitions from each state. This specification of permitted transitions is called a shape. Training of the agents is performed with an evolutionary algorithm. The restrictions on transitions are used in the creation of initial populations and when agents are mutated; the crossover operator used automatically preserves the restrictions. We examine eight shapes using a large, fixed number of states and achieve automatically preserves the restrictions. We examine eight shapes using a large, fixed number of states and achieve behaviors not seen in any of the earlier studies.

The following inequalities must be obeyed for a two-player simultaneous game to be considered prisoner’s dilemma:

\[
S \leq D \leq C \leq T
\]

and

\[
(S + T) \leq 2C.
\]

Iterated prisoner’s dilemma (IPD) has been used extensively as a model of how cooperation emerges among egoistic agents. Because of this, it is often used to model systems in the biological and social sciences [26], [18], [24], [14], [21]. Much research has been done on the evolution of prisoner’s dilemma playing agents, most of which has focused on the evolution of cooperation [22], [16], [15], [25], [13], [19], [12], [23]. This study continues this research into the evolution of cooperation using a variety of assessment techniques. It also examines the competitive ability granted to agents by the shape used in their evolution.

The remainder of this study is structured as follows. Section II describes the agent representation used to encode the game playing agents. Section III gives the design of experiments including the specification of the evolutionary algorithm and the analysis tools used. Results are given and discussed in Section IV and conclusions and possible next steps appear in Section V.

II. REPRESENTATION USED

The representation used in this study is a modified form of finite state machine using the Mealy architecture, which associated actions with transitions. Each state of the machine contains four pieces of information; the action to take if the opponent cooperates or defects and the next state to transition to if the opponent cooperates or defects. The modification of the finite state machine comes in the form of restricting which transitions are permitted. The states of the machine all have index numbers \(0 \ldots n - 1\). Using these, index numbers, we can specify the shapes used. We also compute the negative natural log of the fraction of the baseline space that the restricted shape occupies.

1) All transitions are permitted, this is the baseline shape which matches past experiments with finite state machines. \(-\log(size) = 0\)

2) The advance or return shape permits transitions to the state with the next highest index number \((mod n)\) or back to state zero. \(-\log(size) = 295.1\)
3) The **forward or stay** shape permits the machine to stay in the same state or go to a state one index number higher. Once the machine reaches the state of the highest index it is stuck there. $-\log(size) = 295.1$

4) The **tree** shape builds a minimum height binary tree but then makes transitions back to the root, instead of farther down the tree, when the ration of states runs out. $-\log(size) = 322.1$

5) The **forward 1 or 2** shape makes transitions to states one or two index numbers higher $(mod \ n)$. $-\log(size) = 295.1$

6) The **forward 0, 1 or 2** shape makes transitions to shapes zero, one or two index numbers higher $(mod \ n)$. $-\log(size) = 114.1$

7) The **± 1 or 2** shape makes transitions to shapes one or two index numbers higher or lower $(mod \ n)$. $-\log(size) = 239.7$

8) The **± 0, 1 or 2** shape makes transitions to shapes zero,
one, or two index numbers higher or lower \((mod \ n)\).
\(-Log(size) = 221.8\)

Examples of these shapes are shown in Figure 1, albeit with fewer states than are used in the experiments.

The machines used in experiments all have 80 states, a fairly large number needed to permit different shapes of machines to have scope to exhibit different behaviors. The shapes are used in the generation of initial population members and also when performing mutations. The mutation operator changes the machine’s initial state, initial action, or modifies one of the states with equal probability. Thus in an eighty-state machine each state has one chance in 82 of changing as do the initial state and action. When a state is modified there is an equal chance of changing either of the two actions and two transitions that make up the state. The mutation operator selects uniformly at random from possible values distinct from the current value of an action or transition. The mutation does not change values for which there is only one possibility, e.g. transitions out of the last state in the forward or stay shape. The experiments use two-point crossover. Crossover of two machines generated with a shape yields an automata that also fits within the constraints of that shape.

III. DESIGN OF EXPERIMENTS

The evolutionary algorithm uses a population of 36 finite state machines matching earlier studies [6], [1]. The machines have 80 states. Fitness is evaluated with a round robin tournament. Each machine plays each other machine for 150 rounds of iterated prisoner’s dilemma. Reproduction uses an elite of 24 machines. The remaining 12 machines are replaced by fitness proportional selection without replacement on the elite. Selected pairs of parents are copied and the copies undergo two point crossover and are then subjected to a single mutation. Evolution is run for 3200 generations with the elite members of the population saved in generations 50, 100, 200, 400, 800, 1600, and 3200. In addition, the average fitness of the population is recorded in each generation. For each of the eight shapes used, thirty independent runs of the evolutionary algorithm are performed. A number of analysis techniques are used on the resulting data.

A. Fitness Tracks

It is informative to examine the average fitness of each of the thirty replicates in an experiment over the course of evolution, but displaying them as individual graphs is difficult because the graphs overlap and obscure one another. If we assign colors to the average fitness scores then the thirty fitness tracks can all be displayed side-by-side in a three-dimension display with axes \(X \times Y \times color\). If they are also sorted by the average fitness over the entire course of evolution and displayed in sorted order, then we get snapshots of how fitness changes in each evolutionary run in easily comparable form. The color scale for fitness and examples fitness tracks appear in Figure 3.

B. Play Profiles

To assess an evolutionary system for training prisoner’s dilemma agents changes, it is important to evaluate the probability that a given population is cooperative. It has been established in [7] that when 150 rounds of iterated prisoner’s dilemma are used for fitness evaluation, with the 0, 1, 3, 5 payoff scheme, that an average score of 2.8 indicates that a cycle of sustained cooperation has been achieved by agents using a finite state machine representation.

In this study, we extend this cooperative measure using an extension of the assessment appearing in [4], [11]. This technique exploits that the average fitness of any population is a sum of pairs of payoffs with one of three values: \((1,1), (0,5), \text{or} (3,3)\). Thus, the mean value must be in the range \(1 \leq \mu \leq 3\). This region is then divided into ten equal intervals, with the highest interval corresponding the aforementioned definition of cooperation. At each epoch, the number of populations in each interval is recorded and the results are displayed as a \(7 \times 10\) table, where 7 is the number of epochs. In this study, this table is usually represented as a collection of horizontal, coloured bars. The colours help to distinguish between cooperative, non-cooperative, and mixed actions and their resulting scores. This same colour palette is used as in the fitness tracks shown in Figure 3. An example of a play profile, annotated with the values of scores for each bin, is shown in Figure 2.

C. Competitive Analysis

To assess the impact of the choice of decision variables the most fit agent from each population is placed in competition with every other such agent evolved using a different shape. If the score for two agents after 150 rounds of play is different, then victory is awarded to the higher score. The number of victories for agents evolved with each shape is modeled as a Bernoulli variable, using the normal approximation to the binomial, to estimate the probability of victory. This analysis is performed on agents from the last epoch, 3200 generations. When the confidence interval on the probability of victory does not include the probability \(p = 0.5\) then the shape corresponding to the higher probabilities of victory yields a significant competitive advantage to agents trained using it.

D. Fingerprints

A prisoner’s dilemma fingerprint is a function of two variables, \(x\) and \(y\) ranging from 0 to 1, whose value is the strategy’s expected score when playing against an infinite test suite of strategies called Joss-Am strategies. These strategies were chosen as a diverse set of opponents against which to assess behavior. If \(x + y \leq 1\), the Joss-Ann strategy plays C with a probability of \(x\) and D with a probability of \(y\) and TFT otherwise. If \(x + y \geq 1\), the Joss-Ann strategy plays C with a probability of \(1 - y\), D with a probability of \(1 - x\) and Psycho otherwise. Tit-for-tat returns the opponent’s last action; Psycho the opposite of the opponent’s last action. Note that in either case when \(x + y = 1\), the strategy is just.
the random strategy which cooperates with probability $x$ and defects with probability $y$. This study uses a sampling of the fingerprint function consisting of 25 numbers which are the values of the fingerprint function on a 5 by 5 evenly-spaced grid from 1/6 to 5/6 in both $x$ and $y$.

The theory of fingerprints are developed in detail in three theses [27], [20], [8]. These results are summarized and extended in [3]. A portion of the theory of fingerprints and an initial application to the visualization of evolved agents appears in [5]. One application of fingerprints is to place a metric-space structure on the space of prisoner's dilemma strategies, which is what is done here. The fingerprint results are displayed using nonlinear projection. This technique is a form of non-dimensional multi-metric scaling [17] which produces two dimensional renderings of high dimensional data with similar inter-point distances. The correlation coefficient between the distance matrices of the high-dimensional points and the projected points is maximized with an evolutionary algorithm. Fingerprints of the four strategies always cooperate, always defect, tit-for-tat, and psycho are added to the fingerprints of the most fit strategies in each run used in the competitive analysis.

IV. RESULTS AND DISCUSSION

Figure 3 gives the fitness tracks for all eight experiment while Figure 4 gives the play profiles. Populations evolved with the base shape partitions into a large number of cooperative populations and no populations in the completely uncooperative category. The only other shape that yielded no entirely uncooperative populations was the forward one or two shape, which is unique in having no cooperative populations at all. The tree shape is unique in having a majority of entirely uncooperative populations.

A. The Importance of Staying.

Contrasting the forward 1 or 2 and the forward 1 or 2 or stay experiments, the ability to stay in the same state increases the degree of cooperation from zero to a majority. The forward one or two shape is forced to use a large number of states in play - when the ability to stay permits the automata to get to a good place and stay there. This means that any behavior, in the machines with the ability to stay, has a much lower cross section to mutation and so is much less likely to be disrupted. In the ±1 or 2 experiments, adding the ability to stay increases the number of cooperative and uncooperative populations, emptying out the middle. Since ±1 or 2 permits short loops in the play space, the impact of permitting the machine to stay in the same state is less.

The tree shape does not permit staying and also forces the machine to deal with every possible move by an opponent with a different genetic loci over a window of moves as wide as the tree is tall. This means that any cooperation developed between tree-shaped agents is incrementally exploitable by single point mutants. Looking at the fitness track for the tree shape, individual populations are more and more likely to become uncooperative. The tree, with no ability to stay, becomes violently uncooperative as runs evaporate from higher categories into the uncooperative category.

B. Contrasting Tree and Plus One or Two

The two shapes yielding the least cooperation are the tree and plus one or two shapes. The tree shape induces behavior that moves to the all defect Nash equilibria of prisoner’s dilemma. The behavior of the plus one or two shape, however, is unprecedented. The scores are almost all in the range between 1.0 (defection) and 2.25 (random play). This is not a stable set of scores for any of the representations studied previously.

The tree shape, as a population evolves, will mostly use a single path through the tree because of diversity loss in the population. A mutation that turns a “C” on that path into a “D” grants the mutant a T payoff and diverts the path through the tree used by other agents. The slow degeneration into defection comes from serial exploration of distinct paths through the tree.

The unprecedented behavior of the plus one or two shape results from a similar situation. Like the tree shape, the plus one or two shape is forced to use a large number of states. The number of states used in a low-diversity population for the plus one or two shape is larger that in a population associated with the tree shape and the possible paths are more interwoven with one another. This population is capable of fixing (C,C) and (D,D) payoffs at various locations in
Fig. 3. Fitness tracks for 30 runs of the evolutionary algorithm using eight different shapes.
a consensus genome yielding an average score in the mid-range.

C. Cooperativeness of Forward or Stay

The forward or stay shape yielded agents that were more cooperative than any other, including the baseline shape. With 150 plays and 80 states, the machine must end up in a state that transitions to itself by roughly half way through. This state might be at any position in the genome, thought it will tend to have a high index. Note that, while the forward or stay shape yields the most cooperative results, it also produced four populations that were completely uncooperative. This is probably because the final state was an self-transitioning state that plays always defect.

D. Competitive Analysis

Figure 5 shows the result of the competitive analysis. Only one pair of shapes did not have a significant different in the competitive analysis - forward 0,1,2 and plus or minus one or two. All other pairs yield confidence intervals not including one half. The relation “agents from shape A are more competitive that agents from shape B” is transitive. This yields the following partial order, based on agent competitiveness,

$$\text{Tree} > Fwd1,2 > FwdR > FwdS > Fwd0,1,2 \ ; \ PM1,2 > PM0,1,2 > \text{Base}$$

There is no a-priori reason that this relation should be transitive, making this result unexpected and interesting. The top of the partial order are the two least cooperative shapes but after that correlation of competitive ability and cooperativeness breaks down. In any case, changing the shape has a large affect of the competitiveness of agents.

E. Fingerprints

A non-linear projection of the best agent from each run of the evolutionary algorithm and four reference agents appears in Figure 6. Many of the shapes yield agents that cluster...
Fig. 6. This picture is a nonlinear projection with correlation coefficient $\rho = 0.996$. of the most fit member of all thirty populations in each of the eight experiments. The position of four reference strategies are also shown.

Fig. 5. Shown are 95% confidence intervals for the probability the a type of strategy indexing the row will beat on indexing the column. The thing vertical line marks 50% probability which the black box shows the confidence interval. The labels appearing along the diagonal index both row and column.

V. CONCLUSIONS AND NEXT STEPS

This study demonstrates that changing the shape used to evolve IPD playing agents yields substantial control over the distribution of agents that evolve. This study makes a start on understanding how the control can be exercised, but substantial work remains to be done. The initial studies on the impact of changing representation originated in the observation that different representations yielded different results which, in turn, raises the issue that if an experiment is not controlled for choice of representation then its results may be driven by choice of representation rather than whatever phenomenon was being modeled. In addition there was no simple way to calibrate the impact of representation on results. The use of shape in this study reduces the problem of controlling behavior to a single representation and demonstrates that varying shape permits that representation to exhibit a variety of behaviors.

There are roughly $1.77 \times 10^{152}$ shapes for 80 state automata; this study examines 8. This suggests that this study represents a small scratch into a theory of shape, demonstration proof-of-concept. The nascent theory include the importance of being able to stay in the same state, the importance of forcing the use of many states, and the importance of short loops that can occur in the shapes that can increase or decrease index number for the next state. There are obvious next steps - other tree shapes, trees with the ability to stay, or branching structures leading into smaller...
shapes.

REFERENCES


