A Note on General Adaptation in Populations of Painting Robots

Dan Ashlock
Mathematics Department
Iowa State University,
Ames, Iowa 50011
danwell@iastate.edu

Elizabeth Blankenship
Computer Science Department
Iowa State University,
Ames, Iowa 50011
eblank@iastate.edu

Jonathan Gandrud
Computer Engineering
Iowa State University,
Ames, Iowa, 50011
jgandrud@iastate.edu

Abstract

A population of virtual robots is evolved to perform the task of competitively painting the floor of a toroidal room. Two robots are present in any given room and paint using distinct colors. The fitness of a robot is the amount of floor painted with its own color, a situation where maximal marginal fitness comes from painting over squares already painted in an opponent's color. The time required for a population to settle to a value close to its final average fitness is estimated experimentally at approximately 50 generations. Evolution is then continued well past this estimated settle-down point. The best robots in a given generation are saved at 500 and 5000 generations. The performance of highly evolved and less highly evolved robots is compared by placing the two types of robots into competition. The more evolved robots outperform the less evolved agents, with the empirical estimates of mean fitness differing by more than seven standard deviations. This occurs in spite of a lack of increased fitness of painting robots within their own populations during extended evolution. This result is somewhat at odds with biological dogma, demonstrating general adaptation to the task of painting against opponents never actually encountered. This experiment demonstrates that the quality of the agents as competitive painters is not completely documented by their own in-population fitness numbers. This sort of general adaptation in a competitive task has been observed before in another context, the iterated prisoner's dilemma. This study serves as additional evidence for a form of general adaptation in evolutionary computation systems using an agent-vs-agent competitive fitness function.

1 Introduction

Biologists view evolution as an essentially undirected process. Differential selection based on phenotype leads to change but that change is viewed as having at most a local direction tied to a specific adaptive feature or immediate situation. The idea that long term progress takes place within evolution is viewed with deep skepticism.

In this study an experiment is performed to document a form of long term progress in populations of virtual robots evolved to perform a competitive painting task. This sort of progress has been observed in another context [3] where agents were playing the iterated prisoner's dilemma(IPD) [5]. In this earlier study, populations of 10,000 agents played 150 rounds of IPD against their neighbors on a 100x100 toroidal grid. Evolution was run for 10,000 generations and the state of the grid (identity of players and their positions) was saved both in generation 1,000 and at generation 10,000. Evolution consisted of having any agent adopt the IPD strategy of any grid neighbor that out-scored it, together with a mean of one agent per generation undergoing a mutation of its IPD strategy. Populations from distinct eras and evolutionary lines were placed in competition by loading the left half of the grid with a generation 1,000 population and the right half of the grid with a generation 10,000 population. With reproduction only (no mutation) the simulation was run for an additional 50 generations and a majority vote was taken as to the type of agents remaining. In seven distinct variations of this experiment, using 30 replicates of each individual experiment for statistical power, the generation 10,000 agents had a significantly greater probability of dominating the world in all cases. The agents exhibited general adaptation to the IPD game, not just specific adaptation to their own populations.

This phenomenon of general rather than specific adaptation to a task has the potential to occur in any situation where agents compete at the task. It would be surprising if the phenomenon occurred for a simple task. The scope of this phenomenon is a focus of on-going study by the authors. This study checks for general adaptation in the context of pairs of virtual robots attempting to paint a square toroidal world, each using its own color.

2 Competitive Painting

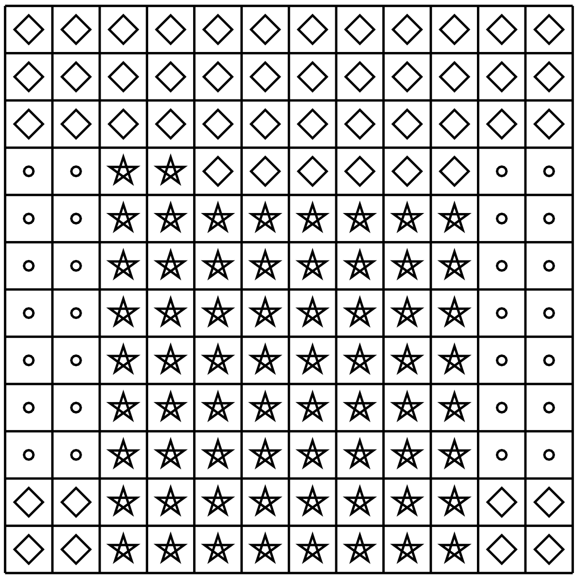

Figure 1: **A 12x12 board in which the first robot scored 52 and the second robot scored 92. The colors of paint used are diamonds and stars, with small circles denoting unpainted squares.**

The task set for the virtual robots in this study is to paint as much of a 12x12 toroidal world as possible. Two robots work simultaneously, each painting with the robots own color. This means that while the absolute fitness gain for painting a square is always one, the marginal score for painting over a square painted by another robot is two. The robots are permitted somewhat more than ample time to paint the world, leaving room for competitive behavior to arise. The robots are permitted to execute 288 actions during painting. In each time step a robot may turn left, turn right, or advance with the robots taking turns moving. The robot is considered to have painted a square if the robot occupies the square. The world starts unpainted, and the two robots are placed at random in the world. Such a random placement is termed a *fitness case.* Because the world is toroidal there are, considering position and heading, a few thousand fitness cases.

Each robot is allowed to know the color of the floor in the eight squares immediately adjacent to the robot. The robot controller must use this information to decide what action it wishes to take. An example of a painted board appears in Figure 1. In order to estimate a useful number of fitness cases, preliminary experiments were run with varying numbers of fitness cases. Populations of robots were evolved, in a manner described subsequently, for 200 generations using 1, 3, 5, or 12 fitness cases. The best robot from each of 100 runs was saved. The best-of-run robots were then tested against one another in each of the six possible pairs based on the number of fitness cases run. For each such pairing, two groups of 100 robots were loaded into the painting environment and their average fitness over 400 fitness cases was computed. All fitness cases involved one robot from each of the groups being compared and partners were re-assorted for each fitness case. A 95% confidence interval for the fitness of each group was computed assuming a Gaussian distribution of the mean. The results are shown in Figure 2. Based on these experiments, five fitness cases were judged to be sufficient. The additional competence gain between 5 and 12 was not significant while that from 3 to 12 was. In all subsequent experiments the population is divided into pairs selected uniformly at random five times. Each such collection of pairs is challenged with a single fitness case selected at random and the average of the score of a robot against five randomly selected opponents in five fitness cases form its fitness for use by the evolutionary algorithm.

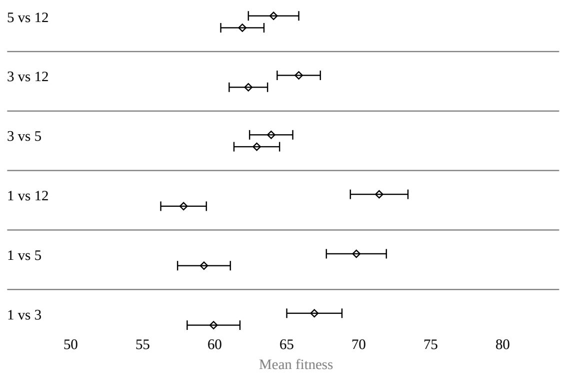

Figure 2: **Confidence intervals for comparative performance of paint bots evolved with different numbers of fitness cases per fitness evaluation during evolution. Performance is measured as mean fitness over 400 fitness cases with random assortment of paint bots in each fitness case.**

3 GP-Automata

The controllers for our painting robots are GP-Automata. A *GP-Automaton* is an augmented finite state ma-

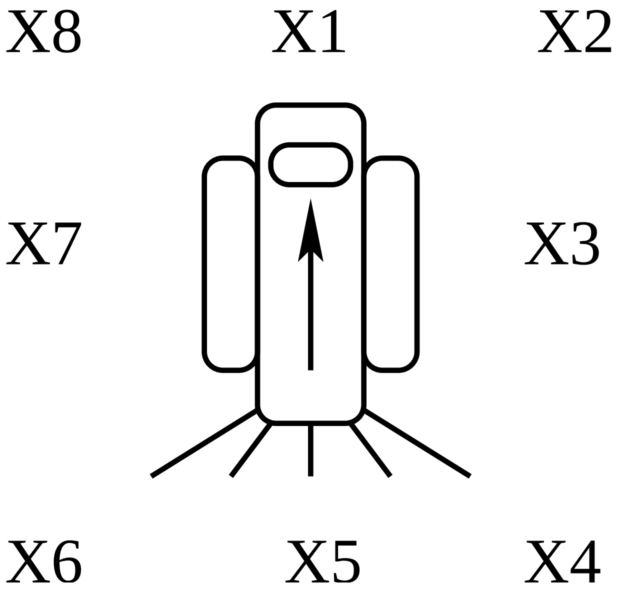

Figure 3: The arrangement of the sensor terminals used to inform the robot of the paint color of adjacent squares.

Start: 2→6			
State	If Even	If Odd	Deciders
0	1→0	3→5	(min x_6 x_8)
1	2→4	1→11	(Odd (∼ (ITE x_2 x_3 (Odd x_8))))
2	1→6	1→3	(<> x_6 (∼ x_5))
3	2→2	2→6	(min x_7 (<= x_6 (<> x_6 (∼ (ITE x_6 x_3 0)))))
4	3→2	3→1	(ITE (Odd x_4) x_2 x_6)
5	2→6	0→3	(Odd (∼ (ITE x_2 x_3 (Odd x_8))))
6	0→3	2→6	(Odd x_1)
7	3→6	2→0	x_6
8	2→5	2→10	x_2
9	2→6	1→2	(ITE (Odd -2) x_2 (Com x_5))
10	2→4	2→11	(<> (+ x_6 x_6) (ITE x_8 x_6 (Com x_8)))
11	3→9	2→7	(ITE 0 (Com x_7) (Odd x_2))

Figure 4: An example of a GP-Automaton evolved to control a painting robot.

chine. The augmentation consists of endowing each state with an integer formula that the state uses to interpret its inputs, the colors of the eight squares adjacent to the robot. These formulas are realized as parse trees, termed *deciders*, making GP-Automata an extension of the techniques of Genetic Programming [8, 9, 6, 7]. The parse trees use the set of operations and terminals given in Figure 5. Formally, a GP-Automaton consists of a collection of states and their associated deciders, together with a transition function and a response function. The transition function is used to determine what the next state of the GP-Automaton will be while the response function computes the output of the GP-Automaton. Both functions are conditioned on both the current state and on inputs from the environment, the colors of adjacent squares. The decider processes these environmental inputs before they are used by the transition and response functions. This permits the decider to function as a state-specific evolvable bandwidth compressor. The parse tree reduces the 3^8 possible paint color combinations to a single bit represented by the parity of the value returned by the parse tree. Since the deciders produce binary results, the parity of their integer value, the GP-Automata have a binary choice of next state and response. An example of a GP-Automaton of the type used in this work appears in Figure 4. GP-Automata are a flexible form of state conditioned evolvable software agents. Other studies using GP-Automata include [1, 4, 2].

Examining the GP-Automaton in Figure 4 we see it has twelve states. The initial state and action are displayed next to the word "Start" at the top of the box. The notation "2 → 6" is read: "output 2 and go to state 6". At the right hand side of each state is an integer arithmetic expression in LISP-like notation; the decider. Information about the color of squares adjacent to the robot is passed through the variables x_i, $i = 0 \ldots 7$. The relative positions of $x_0 \ldots x_7$ are shown in Figure 3. The sensors return the values -1 for unpainted squares, 0 for squares painted in the robot's own color, and 1 for squares painted with the opponent's color. To execute a state the decider is evaluated, returning an integer. If the integer is even, the **action** → **next state** pair from the "If Even" column is used. If odd, the action,state pair is taken from the "If Odd" column. The output is reported to the simulator and the internal state of the GP-Automaton is updated.

The GP-Automata used in this study have four possible outputs. The first three correspond to the actions used by the robots they are controlling according to the scheme 0=left, 1=right, 2=advance. The fourth action, 3=think, is a form of λ transition. A think action causes an immediate transition to the next state. That state is then executed. These think actions are permitted until eight have occurred consecutively. If the robot attempts a ninth consecutive think action then the robot's fitness evaluation ends. Think actions permit the robot to decide to make additional evaluations of available information about the color of adjacent squares and permit more flexible use of the deciders than would otherwise be possible.

The variation operators used with GP-Automata

	Arity	Semantics
I	0	Ephemeral integer constant.
$x_0 - x_7$	0	Input or sensor terminal.
$x_8, x9$	0	Relative position sensors.
$\sim$	1	Integer negation.
Com	1	Computes 1-x.
Odd	1	Predicate for oddness*
$+$	2	Integer addition.
$-$	2	Integer subtraction.
$=$	2	Equality*
$>$	2	Greater than*
$<$	2	Less than*
$>=$	2	Greater than or equal to*
$<=$	2	Less than or equal to*
Max	2	returns maximum of arguments
Min	2	returns minimum of arguments
ITE	3	If-then-else; if first argument** then return the second argument otherwise return the third argument

*returns 1 for true, zero for false
**all nonzero values are considered true

Figure 5: Operations and terminals of the integer valued parse tree language used in deciders.

are now described. The list of states forms the basis for the crossover operator. The states are treated as atomic objects, making the list of states a linear gene. In this study a two-point crossover of the list of states is used. The initial state and action are associated with the first state and follow it during crossover. This is very different from the more traditional sub-tree crossover typically used in genetic programming. There is no potential for crossover-driven bloat and the offspring of two identical parents are identical to those parents. Such a crossover operator is termed *pure* or *conservative.*

Eight different mutation operators are used. The type of each mutation is selected according to the following scheme to create a master mutation operator, used by the evolutionary algorithm. Ten percent of mutations modify the initial state of the GP-Automaton. Ten percent modify the initial action. Twenty percent modify a transition by replacing a next state selected uniformly at random with a new next state, also selected uniformly at random. Twenty percent modify a uniformly selected action with a new action selected uniformly at random. Ten percent of mutations replace a decider with a new decider generated at random. Ten percent perform subtree crossover on two deciders selected uniformly at random. Ten percent exchange two deciders. Finally, ten percent copy one decider over another decider.

The number of nodes (operations plus terminals) in an initial decider and the maximum size of a decider are both algorithm parameters. In this study new deciders have six nodes and all deciders have at most twelve nodes. This is less restrictive than it might seem because think actions permit the execution of multiple deciders. The only way that decider size can change is as a result of subtree crossover during mutation. If the size of a decider exceeds the maximum decider size then the decider is *chopped.* Chopping selects an argument (sub-tree) of the root node of a decider uniformly at random and uses that sub-tree to replace the decider, iteratively until the decider is small enough. The effect of this method of controlling code size is to have a moderate, implicit pressure toward economical solutions with solution size controlled by the decider size parameters and the number of states.

4 Experimental Design

The evolutionary algorithm used in this study operated on a population of 200 GP-Automata evaluated on five fitness cases of the competitive painting task. The five fitness cases were selected at random in each generation. The model of evolution used is single tournament selection with tournament size four. The population is shuffled into groups of four (tournaments). In each tournament the two most fit robots undergo reproduction replacing the two less fit. Reproduction consists of copying, followed application of crossover and mutation to the copies. The process of dividing the population into tournaments and performing reproduction within each tournament is called a *generation.* This model of evolution, using the smallest possible tournaments, representing relatively soft selection.

As explained previously, four initial sets of experiments were performed to select the number of fitness cases to be used in the main experiment. The experiment in which the agents were evolved with five fitness cases was examined to estimate the number of generations required for the system to approach its final fitness value. After generation 50 no large changes in fitness were observed in any of the 100 simulations. Plots of the mean and best fitness for several of these experiments are given in Figure 6. Since the goal is to document progress after a system has settled into a relatively stable state, the main experiment was run for 5,000 generations with currently best fitness robots saved at generations 500 and 5,000.

The method used to choose 500 and 5,000 generations for sampling intervals was frankly *ad-hoc.* The lack of a tight statistical method for document-

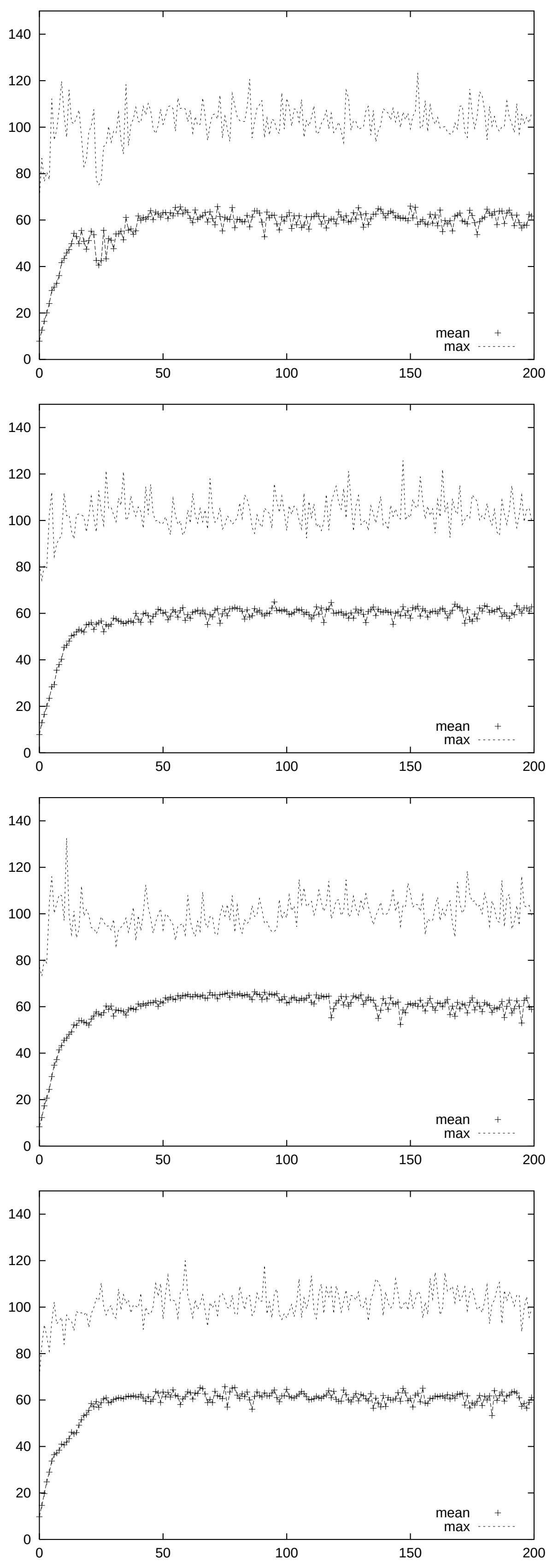

Figure 6: The mean and best fitness for four of the populations used to estimate the settle-down time of the competitive painting system.

ing that the fitness had effectively settled down by generation 50 motivates the spacing of samples at intervals orders of magnitude beyond our estimate. Our results also do not depend on correctly computing the settle down point, but rather on being well past the settle down point.

5 Results

A collection of 400 evolutionary runs was performed. The best painting robots in each of the populations were saved at generations 500 and 5,000. The resulting robots were placed in competition, computing the fitness of each in 400 random fitness cases. The more evolved robots moved first in half of these fitness cases; the less evolved moved first in the other half. All fitness cases pitted a more evolved robot against a less evolved robot and the competitors were assigned uniformly at random for each of the fitness cases. A 95% confidence interval for the mean fitness of the more evolved and less evolved robots was computed and is given in Table 1 and is displayed in Figure 7.

Evolution time	Mean fitness	95% confidence interval
5000 Gen.	67.4	(66.8,68.1)
500 Gen.	61.5	(60.9,62.0)

Table 1: Tabular 95% confidence interval for mean fitnesses of more evolved and less evolved robots. These intervals are based on comparison of 2 sets of 400 paint bots.

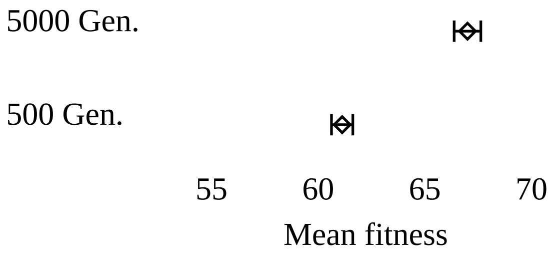

Figure 7: Graphical 95% confidence interval for mean fitnesses of more evolved and less evolved robots. These intervals are based on comparison of 2 sets of 400 paint bots.

While the separation of mean values is only 5.8 out of a maximum possible fitness of 143, the separation is over seven standard deviations. There is little chance that the results are accidental: the more evolved robots are better than the less evolved robots. This is true in spite of relatively little change in the mean fitness of the competing populations and in spite of the fact that $\frac{399}{400}th$ of testing was between agents from distinct populations and hence between agents with no kinship.

6 Discussion and Conclusions

The desired form of general adaptation to the painting task is exhibited by the painting robots. This implies that the game represented by trying to paint the floor, and to paint over the opponent trail, is sufficiently complex to permit new strategies to continue to appear over a relatively large span of simulated evolutionary time. The task of having one robot paint the floor has exceedingly simple solutions, e.g.: advance 11, turn left, advance one, turn right, repeat. Most of the complexity in the task must result from the competitive character of the task.

If the task of competitive painting is in fact complex, then a diversity of strategies should be observed. In an attempt to estimate the diversity of strategies the fraction of actions of each type executed in each generation were recorded. Examining these graphs showed the robots exhibiting a multiplicity of behaviors. Four different combinations of actions plotted for 200 generations are shown in Figure 8. The top plot in Figure 8 shows two distinct strategic regimes distinguished by the exchange in the preeminence of think and advance actions. All four plots exhibit distinct uses of turning. The top plot has the lowest level of turning, the second the highest. The third plot exhibits the most handedness in turning behavior while the bottom one shows two exchanges of handedness.

The four sets of action plots shown form a sample. A set of 100 evolutionary runs were examined over their first 200 generations and many distinct relative levels of handedness and of advance actions over turning actions were observed. This suggests that the painting robots were not simply selecting from a relatively small effective set of strategies. If the set of strategies for competitive painting is not small then there is room for general adaptation. A 12-state GP-Automaton can encode fairly complex behavior. It is still somewhat remarkable that such a simple structure can encode generally useful strategies.

The effect observed in this study, using painting robots, was of about the same order of statistical significance as that observed in the earlier IPD study of general adaptation [3]. This replication in a different evolutionary system of the observation of general adaptation suggests the phenomenon is worthy of additional examination. One direction al-

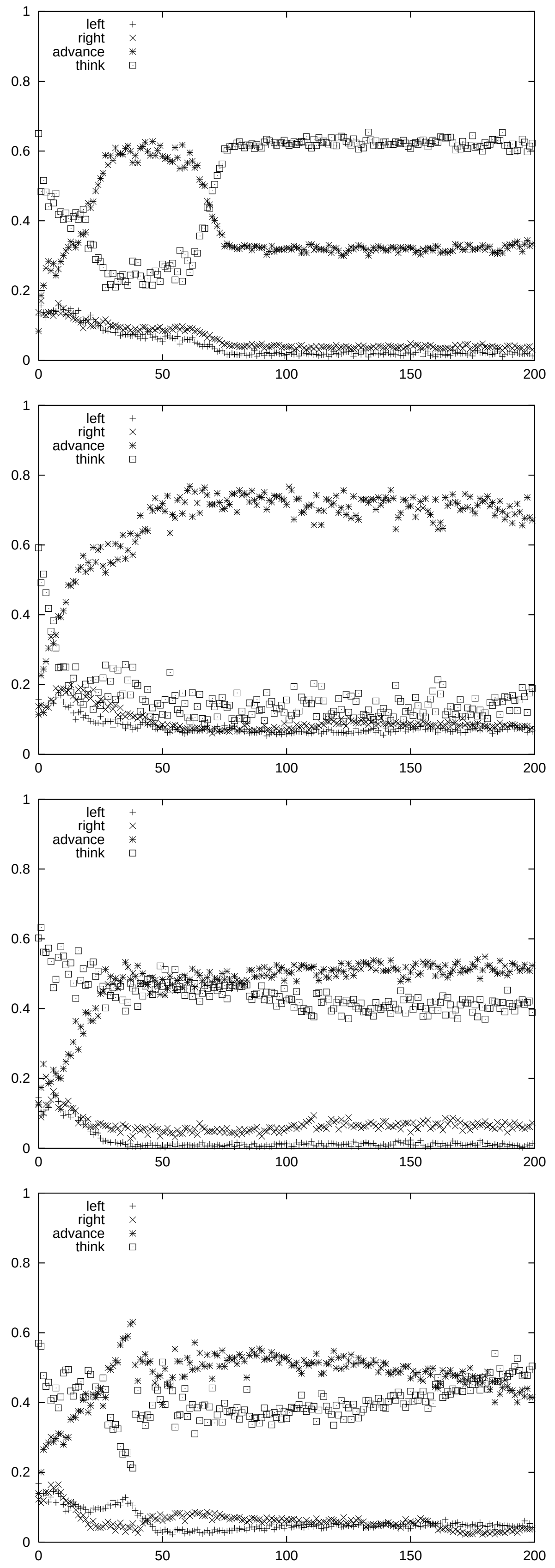

Figure 8: Fraction of actions of each type for four different evolutionary runs spanning 200 generations.

ready being explored by the authors is to attempt to document the phenomenon in additional systems and also to locate systems in which the phenomenon does *not* occur. We conjecture that the competitive task must have some minimal amount of complexity before general adaptation to the task can be detected. A second line of research is to attempt to ascertain the scale of the general adaptation phenomenon within the context of a single system. For IPD, competitive painting, or other tasks under study, additional intermediate populations like the generation 500 population in this study could be saved. By comparing these intermediate populations to one another and the final population, the amount of time required for superior general adaptation to arise could be estimated. It is likely that the time required for a significant performance gap to form would increase as a function of the age of the younger sample. The improvement from generation 500 to 5000 is quite likely more than that from 5000 to 9500. Mathematically, we conjecture that the shape of the function measuring the degree of general adaptation to a task is convex.

The painting task used in this study is essentially a competitive one. A robot receives no fitness whatsoever for parts of the floor painted by its opponent. The IPD is a task in which software agents may either cooperate or compete with these two broad categories of behavior displacing one another as evolution progresses. We have observed general adaptation in both of these tasks. Would general adaptation take place in roughly the same manner in an entirely cooperative task? The identification of such a task may be difficult. In ongoing work with virtual robots similar to those used in this study the authors observe it is often the case that the agents will degrade in overall performance while enhancing their own relative fitness even when the task is nominally cooperative. Actually writing a fitness function that rewards cooperation and no form of competition has proven difficult.

This technique for detecting general adaptation is called a *Mayfield assay* after John Mayfield of the Iowa State University Zoology and Genetic Department who inspired it. At present we use the assay to document a form of general adaptation. The assay might also be used as a means of classifying evolutionary systems. The least complex systems will exhibit no general adaptation. More complex systems should have longer periods during which additional evolution increases general adaptation as measured by the Mayfield assay. This notion is preliminary and requires more thought to lend it a useful degree of rigor. There is a need for a means of taxonimizing the complexity of problems where a problem is not just a fitness function but an entire evolutionary system including representation, variation operators, and parameter settings.

The earlier studies that demonstrated the Mayfield effect in the IPD elicited the questions "does this effect occur in biological systems?" It is not too difficult to imagine an experiment using bacteria that might help to answer this question. It seems unlikely that such an experiment would detect any effect, however, because all available bacteria have evolutionary histories measured in billions of years. The "less evolved" bacteria are essentially unavailable for experimentation. We do not raise this point only to discourage those that wish to perform a biological version of this experiment but also to suggest a reason that the sort of general adaptation observed here is at odds with current biological dogma. While general adaptation of the sort exhibited by our painting robots may occur in nature, it is not visible on the exceedingly short time scales available to biologists. Life on earth can be argued to exhibit an extraordinary degree of general adaptation, particularly on the molecular level. This general adaptation grants little competitive advantage because it is enjoyed by all competitors.

7 Acknowledgments

The authors would like to thank the Iowa State University Department of Mathematics for its support of the second and third authors during the period that this research was performed. They would also like to thank Andrew Kirpalani and Sam Schmidt who invented the competitive painting task. We would also like to thank the members of the Iowa State Complex Adaptive Systems Program for many helpful discussions.

References

[1] Daniel Ashlock. Gp-automata for dividing the dollar. In *GP97, Proceedings of the 1997 conference on genetic programming*, pages 18–26. MIT Press, 1997.

[2] Daniel Ashlock. Data crawlers for optical character recognition. In *Proceedings of the 2000 Congress on Evolutionary Computation*, pages 706–713, 2000.

[3] Daniel Ashlock and John Mayfield. Aquisition of general adaptive features by evolution. In *EP VII, Proceedings of the Seventh Annual Conference on Evolutionary Programming*, pages 75–84, 1998.

[4] Daniel Ashlock and Charles Richter. The effect of splitting populations on bidding strategies. In *GP97, Proceedings of the 1997 confer-*

ence on genetic programming, pages 27–34. MIT Press, 1997.

[5] Robert Axelrod. *The Evolution of Cooperation*. Basic Books, New York, 1984.

[6] Kenneth Kinnear. *Advances in Genetic Programming*. The MIT Press, Cambridge, MA, 1994.

[7] Kenneth Kinnear and Peter Angeline. *Advances in Genetic Programming, Volume 2*. The MIT Press, Cambridge, MA, 1996.

[8] John R. Koza. *Genetic Programming*. The MIT Press, Cambridge, MA, 1992.

[9] John R. Koza. *Genetic Programming II*. The MIT Press, Cambridge, MA, 1994.