Creativity and Competitiveness in Polyomino-Developing Game Playing Agents

Daniel Ashlock and Jeremy Gilbert

***Abstract*—This study proposes a new mathematical game called *Polyomination* which involves the competitive placement of polyominoes to capture area. The game playing agents used are able to encode both their strategy and the game pieces they will play with. Strategy is encoded in a finite state representation called a *binary decision automata* which has access to a variety of pieces of information abstracted from the game state. Playing pieces are encoded by a developmental representation. An extensive parameter study is performed. The elite-fraction used by the evolutionary algorithm that trains the agents is found to be relatively unimportant. The number of states in the automata and the maximum number of squares used to build polyominoes are found to have a significant impact on competitive ability. The polyomino playing pieces are found to evolve in a strategic manner with playing pieces specializing for area-occupation, area-denial, and cleanup in which small pieces can fill in small remaining areas. This study serves as an initial study of Polyomination, intended to serve as a springboard for the design of simpler, related games.**

I. Introduction

THIS study introduces a new game, *Polyomination*, that trains intelligent agents to lay down polyominoes on a rectangular board. The goal is to maximize area covered by a player's own polyominoes while minimizing the opponent's opportunities. Agents have three polyomino types they may lay down, with an unlimited supply of each type available. The agent representation is a type of finite state device called a *Binary Decision Automata* (BDA) augmented with a linear gene specifying its three polyominoes. An example of a BDA appears in Figure 1. Many examples exist [6], [7], [9], [3], [2], [5], [1] of using finite state agents to play mathematical games. This study introduces a two-player, sequential move game that permits each player to select the pieces they use to play. In essence they are selecting their available set of moves. Each play in Polyomination modifies the game state. This means that the agents are playing a game that is far more complex than prisoner's dilemma, but still an abstract mathematical game. Polyominoes are used in many existing games, the web site *Board Game Geek* is a good source.

The ability of agents to evolve their own playing pieces makes the game intrinsically asymmetric. Since any polyomino one player picks is also available to others the game is not unfair. Agents in the game generate three polyominoes with the maximum number of squares available being an experimental parameter. The reason for giving agents three polyominoes is that there are three natural roles: a large polyomino to grab area, a sprawling area-denial polyomino, and a small polyomino meant for filling in small gaps late in the game. The agent shown in Figure 1 was evolved. Its polyominoes exemplify this principle. The 4×5 polyomino can cover large areas. An absolute exclusion principle is in force during play - you may not partially cover another player's polyomino when you play - so the 4×5 polyomino is also easy to block. The irregular polyomino is an example of a simple area denial move. The 2×1 polyomino is very likely to fit even in a mostly full board. An earlier study [11] studied a version of this game without an evolvable strategy component in order to model robustness as a feature of evolution.

The remainder of this study is structured as follows. Section II describes the game of Polyomination. Section III gives the design of the experiments performed, including a careful description of the agent representation, fitness evaluation, and a description of the parameters of the experiments actually performed. Section IV reports and discusses results. Section V draws conclusions and outlines next steps.

II. The Polyomination Game

This game has two players that take turns placing polyominoes on a rectangular board. The board is 40×70 in this study. A polyomino may only be placed in a position where it does not overlap other placed polyominoes at all. Play continues for 100 turns (100 plays for each player), though if there is no place a player may move then they forfeit their turn. In addition, since we are training initially stupid populations of simple AIs with evolution, a player's turn is skipped if they try and make an impossible move. A pair of players play twice, averaging their scores, with each player having an opportunity as first player. This eliminates any first-player bias that exists.

The agents each have three types of polyomino that they may play. The maximum size of the polyomino is an experimental parameter and the agents have an unlimited supply of each of these polyominoes. The shapes of the polyominoes are chosen by the player or, in this study, are genetically specified. These polyominoes are chosen without knowledge of what the other player's polyominoes are. This means that there are at least two strategic levels to the game - selecting polyominoes and then taking turns placing them. This layered strategy makes the game more interesting both for players and researchers analyzing agents evolved to play the game.

Daniel Ashlock and Jeremy Gilbert are with the Department of Mathematics and Statistics at the University of Guelph, in Guelph, Ontario, Canada, N1G 2W1. Email: dashlock@uoguelph.ca, jgilbe01@uoguelph.ca

The authors thank the University of Guelph and the National Science and Engineering Research Council of Canada (NSERC) for supporting this work.

978-1-4673-5311-3/13/$31.00 ©2013 IEEE

III. Design of Experiments

This study will use a full factorial design on the elite size of the evolutionary algorithm, which has been found to be highly significant in earlier prisoner's dilemma work, the length of the specifiers for polyominoes which control the maximum number of squares they have, and the number of states in the BDAs. We now describe each of the pieces of the experimental design.

State	Condition	If(T)	If(F)
0	(Fits1 imp Fits3)	Move +(63,16) → 1	Move +(54,20) → 14
1	(Fits3 imp Fits1)	Move +(44,31) → 12	Drop3 → 5
2	(Fits3 xor Fits2)	Drop2 → 5	Drop2 → 8
3	(Fits2 nor Fits2)	Drop3 → 2	Drop2 → 13
4	(Fits2 xor Fits3)	Drop1 → 13	Drop1 → 12
5	(Fits2 xor Fits2)	Move +(2,17) → 9	Drop1 → 12
6	(Fits3 imp Fits2)	Drop1 → 2	Drop1 → 12
7	(Fits3 or Fits2)	Drop2 → 6	Move +(69,33) → 8
8	(Fits3 nor Fits2)	Drop1 → 2	Drop2 → 7
9	(Fits2 nor Fits1)	Drop3 → 5	Drop1 → 6
10	(Fits3 imp Fits1)	Drop2 → 0	Move +(58,31) → 11
11	(Fits3 and Fits1)	Drop2 → 5	Drop1 → 1
12	(Fits1 and Fits2)	Drop2 → 9	Drop1 → 12
13	(Fits1 nor Fits1)	Drop2 → 11	Drop2 → 2
14	(Fits3 imp Fits2)	Drop2 → 2	Drop2 → 5

Polyominoes:

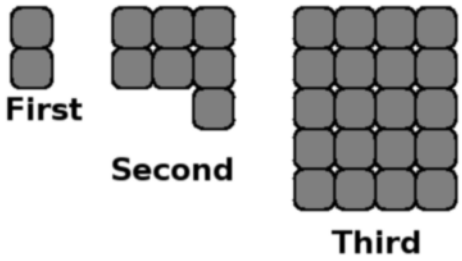

Fig. 1. An example of an evolved binary decision automata and its three evolved polyomino playing pieces.

A. Representation

The BDAs used in this study have two parts: the finite state device and a linear gene that specifies a generative representation for the polyominoes. Refer to the example shown in Figure 1. The finite state device is treated as a linear chromosome of states each of which contains a Boolean test and, for both a true and a false outcome of that test, an action and a transition to another state of the automata. The automata starts with a viewpoint at position (0,0) of the arena. There are three Boolean primitives available to an agent, each of which report the truth value of the statement that one of the agent's three polyominoes will fit at the current viewpoint. The board wraps at the edges for this purpose. These primitives are made into tests by the conjunction of two of the primitives with one of the Boolean operators and, or, nand, nor, impies, or exclusive or. The moves available to agents are either to drop one of its polyominoes or to move its viewpoint. Moves are specified as purely positive translations of the viewpoint along both the horizontal and vertical axes of the board but the totals are interpreted modulo the board direction. Automata caught in a large finite loop (which is used as a surrogate for detecting infinite loops) forfeit the remainder of their turns in a given encounter.

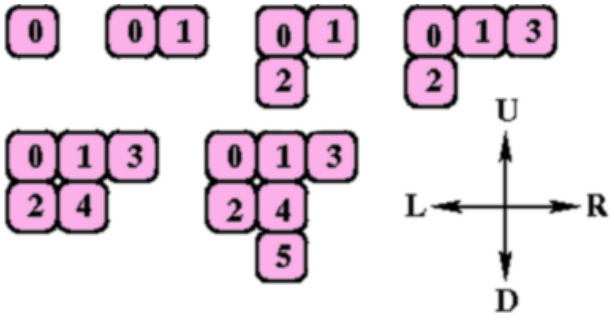

Fig. 2. This figure demonstrates the generative polyomino representation. Developmental path: right at zero, down at zero, right at one, down at one, down at four.

The polyomino specifier consists of an array of integers in the range $0 \leq x < 2500$. A polyomino is grown, starting with a single cell, by first interpreting the integer $(mod\ 5)$ as a direction or stop code (up, down, left, right, stop) and then the remainder of the integer, divided by 5, modulo the number of squares developed so far, as the grid square from which to attempt growth. Stop codes are needed to permit the encoding of small polyominoes. An example of the developmental path of a small polyomino is shown in Figure 2. The polyomino specifier genes consist of three arrays of integers of the correct length concatenated together. These three arrays are used, independently, to specify the three polyominoes. Figure 3 shows the difference of sizes of a large collection of polyominoes, with a maximum size of 20, generated uniformly at random without stop codes. This histogram shows that the size two polyomino that evolved in the example agent shown in Figure 1 would have been almost impossible without some form of stop instruction.

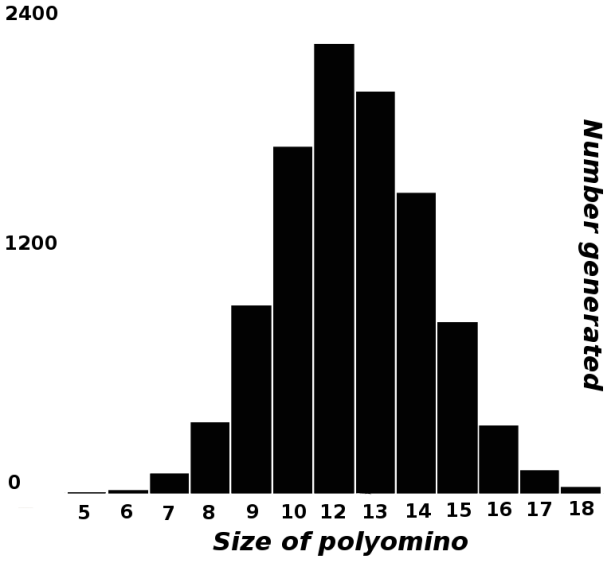

Fig. 3. Histogram of the number of squares in 10,000 randomly generated polyominoes with a maximum size of 20.

The variation operators used for the finite state and polyomino specifier parts of the agent's chromosome are as follows. Two point crossover of the array of states and the array of integers used to specify polyominoes were each used at a rate of 50% with the use being checked independently. Crossover of the polyomino specifier treats the array of three construction plans as a single object, granting boundaries between polyominoes no special status. A new automata was also subjected to 1-3 mutations with the number of mutations chosen uniformly at random. A mutation has 50% chance of replacing an integer in the polyomino specifier with a new one and a 50% chance of choosing one discrete object in the finite state device (Boolean test, transition, action) and replacing it with a new one generated uniformly at random.

B. Evolutionary Algorithm

The evolutionary algorithm operated on a population of 120 Polyomination players. Fitness evaluation is performed by shuffling the population 30 times to pair up the automata. They then play one time as first player and one as second. The resulting scores are then averaged to obtain the player's fitness. Once the fitness of a population is known, an elite whose size is an experimental parameter is copied into the next generation. The remainder of the population is replaced by selecting pairs of parents, without replacement, by fitness proportional selection from the elite. These parents are copied onto the new population members and then undergo crossover and mutation as described in the previous section.

Because fitness evaluation is stochastic (because of the random shuffling of the population to establish pairs of opponents) the evolutionary algorithm is generational. The algorithm is run for 250 generations and the final elite population is saved. The non-elite population often contains inviable players damaged by the variation operators and so saving the elite helps to ensure that only trained players undergo analysis.

C. Experiments Performed

A full factorial design was performed for the parameter choices 15, 45, or 135 states, elite size 40, 60, 80, and maximum number of squares in a polyomino 20, 22, 24, 26, and 28. The mutation rate and population size parameters were not studied due to lack of space and time and that remains a goal for future research. The total number of experiments performed was 45. Each experiment consists of 150 runs of the individual algorithm. The experiments are numbered for reference and the numbering, together with parameters, are given in Table I.

Two types of assessments were performed. In the first, players were permitted to play without an opponent and their score was assessed. In the second, groups of agents were put through a round robin tournament of the game recording the victor in each pairing. The probability of victory of the first group was estimated using a normal approximation to the binomial. The former assessment is called *single fitness* assessment, the latter is called *competitiveness analysis.*

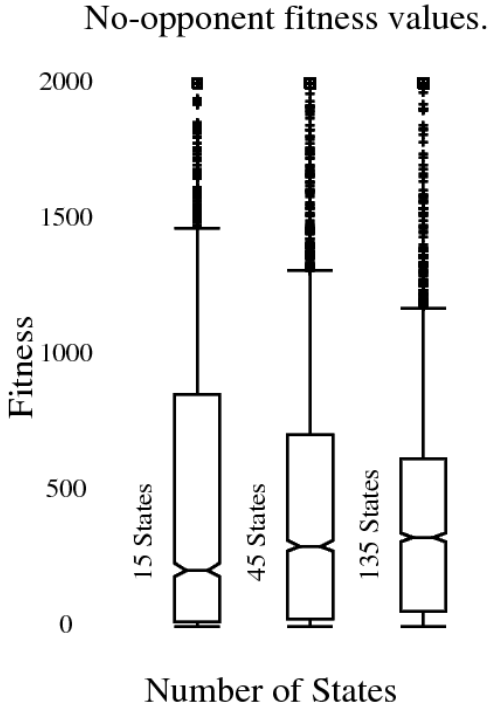

Fig. 4. Impact, aggregating all simulations, of changing the number of states in the automata on single fitness.

IV. Results and Discussion

Figure 4 gives box-plots for agents with particular numbers of states, including data from all 45 experiments. The best performance declines with number of states made available to an agent, while the average performance improves. The improvement in average performance is significant. Since all agents were permitted the same amount of training time, the behavior of the maximum may simply represent slower convergence of the populations of agents with more states.

Figure 5 shows the impact of changing the maximum size of polyominoes available to the agents. The differences are small - most agents don't use the full size available in any of their polyominoes - but a maximum size of 26 turned in a performance that was significantly better than any of the other sizes. Examination of individual polyominoes suggest that creating 4×5 or 4×6 rectangles is easier for agents with an upper bound of 26 on the size of their polyominoes. Using up all available space is quite difficult (see Figure 3), so a size limit slightly above multiple large rectangles is probably beneficial.

Figure 6 shows the impact of varying elite size. While elite size 60 was the worst and 80 was the best none of the effects were statistically significant and the elite size seems to matter substantially less than in studies on the iterated prisoner's dilemma. This robustness to choice of elite size was not expected. Because elite size seems to have little impact, no competitiveness studies were performed based on the elite size.

An alert reader will have noticed that the fitness dis-

TABLE I
EXPERIMENTAL PARAMETERS AND INDEX NUMBERS.

#	Elite	States	Max Size	#	Elite	States	Max Size	#	Elite	States	Max Size
1	40	15	20	16	60	15	20	31	80	15	20
2	40	15	22	17	60	15	22	32	80	15	22
3	40	15	24	18	60	15	24	33	80	15	24
4	40	15	26	19	60	15	26	34	80	15	26
5	40	15	28	20	60	15	28	35	80	15	28
6	40	45	20	21	60	45	20	36	80	45	20
7	40	45	22	22	60	45	22	37	80	45	22
8	40	45	24	23	60	45	24	38	80	45	24
9	40	45	26	24	60	45	26	39	80	45	26
10	40	45	28	25	60	45	28	40	80	45	28
11	40	135	20	26	60	135	20	41	80	135	20
12	40	135	22	27	60	135	22	42	80	135	22
13	40	135	24	28	60	135	24	43	80	135	24
14	40	135	26	29	60	135	26	44	80	135	26
15	40	135	28	30	60	135	28	45	80	135	28

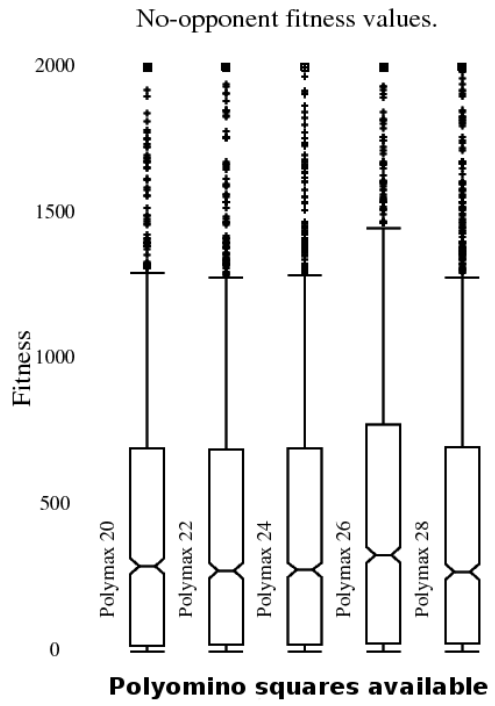

Fig. 5. Impact, aggregating all simulations, of changing the length of the genes specifying polyominoes on single fitness.

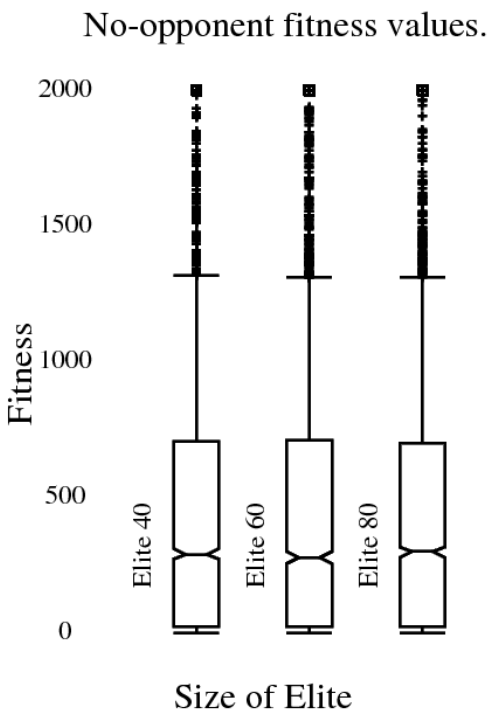

Fig. 6. Impact, aggregating all simulations, of changing the elite size of the evolutionary algorithm on single fitness.

tributions for single fitness, in which an agent is allowed to place polyominoes without opposition, has a distribution with many upward outliers and a relatively low mean value. Examining individual agent scores, it is apparent that this results from many agents (but not all) getting lower scores in unopposed play. Since the agents never encountered unopposed play during training, this is not surprising, but it does mean that many of the agents have evolved to use area-denial strategies or otherwise react to their opponents. This suggests both a potential change to the training regime, either also assessing or pre-training on unopposed ability to fill area. It also means that interfering with your opponents ability to gain fitness is a substantial feature in many of the strategies that arise.

Figures 7, 8, and 9 show results of competitiveness testing for agents with different numbers of states, holding all other experimental parameters constant. Only three of the confidence intervals include $p = 0.5$ and so 42 of the Competitiveness comparisons show a statistically significant difference in competitive ability.

Figure 10 shows the competitiveness results pitting agents with different maximum polyomino sizes against one another while leaving other factors constant. Most of the comparisons are significant, but with no obvious pattern to the results. This

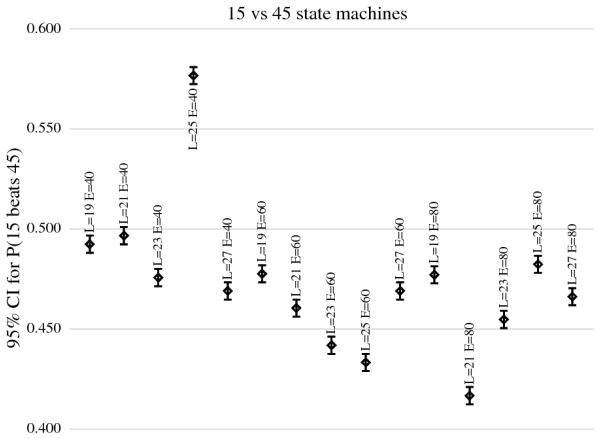

Fig. 7. Competitiveness study for 15 vs 45 state agents.

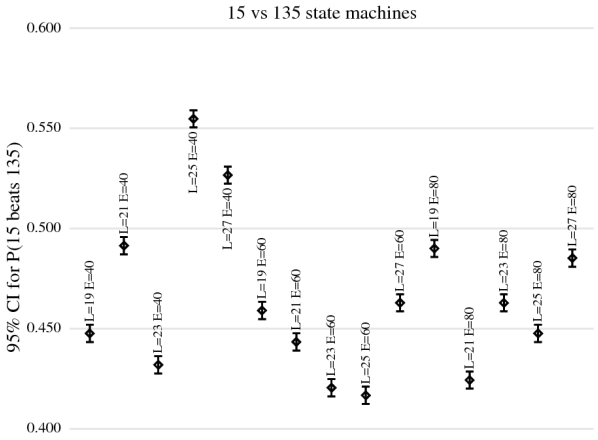

Fig. 8. Competitiveness study for 15 vs 135 state agents.

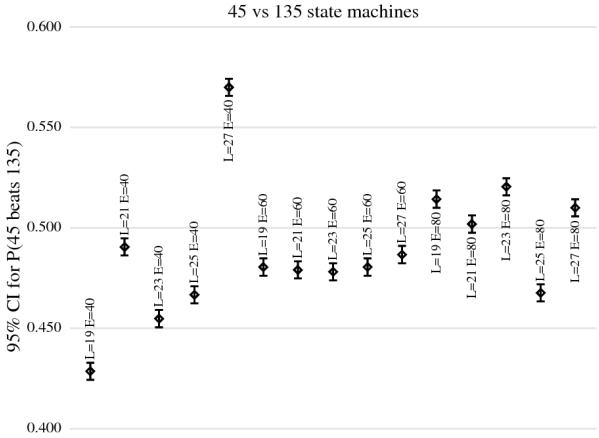

Fig. 9. Competitiveness study for 45 vs 135 state agents.

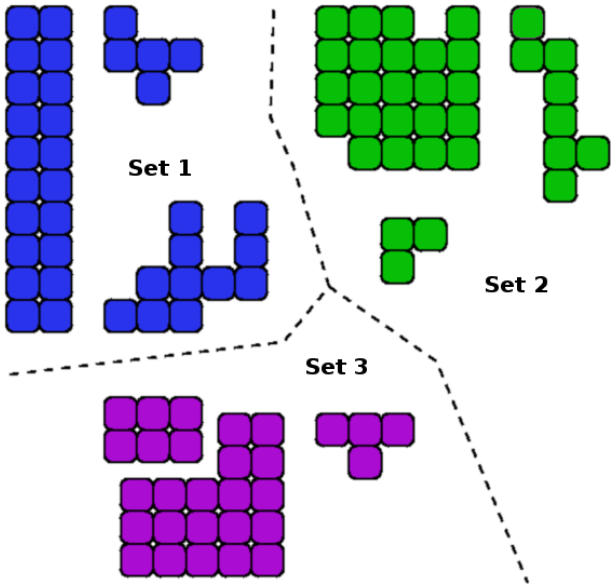

Fig. 11. Three sets of evolved polyominoes from high fitness results of Experiment 4, which exhibited strong competitive superiority. This experiment tended to eschew small area polyominoes for multiple area denial weapons.

suggests a nontrivial interaction between the three parameters that will require additional experiments and finer grained analysis to uncover. One clear result is that experiment 4 with 15 states, elite size 40, and polyomino maximum size 26 (specifier length 25) is highly superior. A subsequent check showed that the agents evolved with those parameters significantly outperformed all others in competitiveness comparison.

Strategies adopted by the agents include both rapid area coverage and blocking stratgies. Figure 12 shows examples of final board states for various pairs of agents.

V. Conclusions and Next Steps

The study introduces the game Polyomination and demonstrates that agents can learn to create sensible sets of polyominoes and strategies for playing them. At the level of polyomino selection, a number of strategies emerge. All agents have a large polyomino that they try to drop quite a lot on an initial, empty board. They then also have crack fillers, area obstruction polyominoes with several protruberences, and smaller area fillers. Figure 11 gives an example of sets of co-evolved polyominoes. The large-small area filling pair in set three of this figure is relatively rare. Since this is clearly an advantageous development, it seems likely that permitting evolution to run for substantially longer periods of time may yield strategically more effective sets of polyominoes.

The principle set of data in this study are a large parameter study with 45 experiments. The most surprising and least interesting result was that the elite size of the algorithm was nearly irrelevant to the quality of agents found. Both the maximum size of polyominoes and the number of states in the agent turned out to make significant differences in their ability to capture area without opposition and in their ability to compete against other agents. Other than “more states are better” there was little pattern to where competitive advantage occurred. This suggests that there is a complex interaction of number of states and maximum polyomino size that will require additional study to understand.

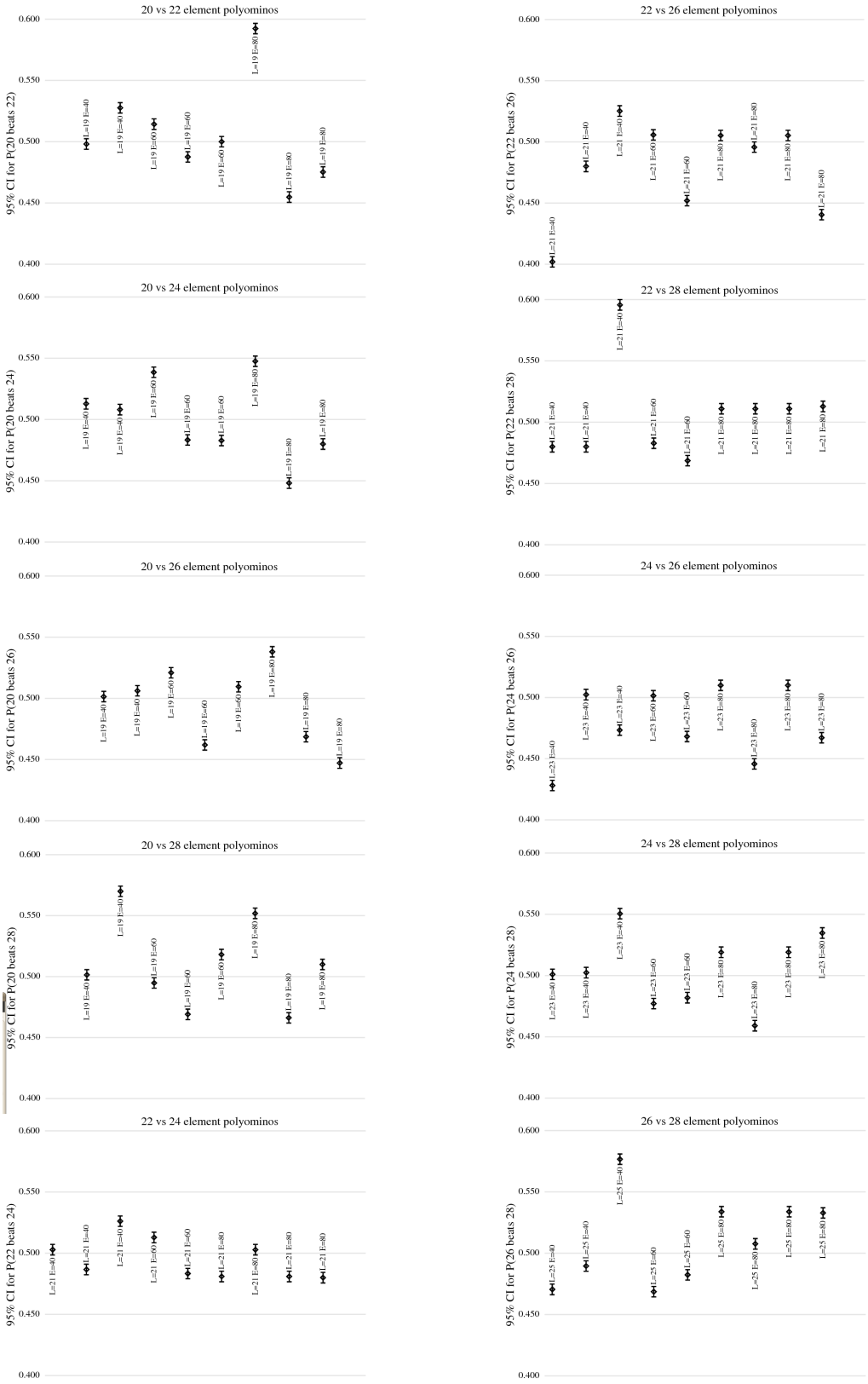

Fig. 10. Competitiveness comparisons for polyomino size.

The game playing agents here, while they co-evolve their polyominoes and finite state strategy box, do not directly incorporate polyomino shape information into their strategy box. This means that the finite state strategy boxes have the potential to be trained as general game-playing agents for polyomino games. An interesting first step in this direction would be to study how well the agents play with their polyominoes changed. It is very likely that the gaps traversed when an agent moves their viewpoint are strongly adapted to their particular polyominoes. This, in turn, suggests that the size of the bounding box of the polyominoes might profitably be made available to the finite state devices as an alternative method of encoding viewpoint movements.

An early goal for additional research is to use the tournament software used to assess overall competitiveness of large groups of agents to sort sheep from goats and find a small number of highly competitive agents for additional study. While initial work that has been done analyzing individual agents thus far has concentrated on their choice of polyominoes, there is a good deal of room to visualize their play and attempt to perform analysis of their placement strategies.

This study is the first on the game polyomination and it sets the stage for a number of subsequent studies.

A. Simpler Derived Games

Much of the complexity of polyomination comes from the fact that players select, or – in the case of the software agents used here – evolved, their playing pieces. Fixing the identity of the playing pieces would yield a far simpler game environment. The experiment in this study has yielded the beginning of a strategic lexicon for polyomino selection. These include larger, near-rectangular *area covering* polyominoes, very small *crack filling* polyominoes, oddly shaped *area denial* polyominoes. In addition there is the issue of having polyominoes that fit well together, *co-evolved* or *synergistic* polyominoes. The results of this study will be quite useful for a followup in which fixed sets of polyominoes, both symmetric and asymmetric, are used. This line of research has the eventual goal of finding sets of polyominoes that are interesting for mini-games or board games played by humans.

B. A Controlled Laboratory for Studying Game Balance and Interest

Automatically balancing games is the focus of much active research [12], [8], [10]. The simpler games with fixed sets of polyominoes form a potential laboratory for studying game balance. Questions to study might include:

- Can we find distinct sets of polyominoes for the first and second player that nevertheless result in a balanced game?
- Given a balanced game with a given set of polyomino playing pieces, which additional pieces most unbalance the game?
- Can we locate interesting sets of polyominoes?

For questions like the last one, a surrogate for interestingness is required. We propose that the lack of a dominant strategy combined with a failure of any strategy to last more then a few generations is a surrogate for interesting games. The former means there is not a single good way to play, the latter checks for the property that not all effective strategies have obvious counter-strategies.

C. Deck Based Games

In [4] a version of iterated prisoner's dilemma was studied in which players were required to play from a deck of cooperate and defect moves. This completely changed the character of the game - to either a coordination or anti-coordination game. In this case the deck of cards would be replaced with a finite allowance of polyominoes that are all that a player may place. These games are ones that might make good board games. A number of variations are possible including games in which a fixed pool of pieces is shuffled or ones in which players receive specific allotments. This latter type of game leaves room for handicapping and dynamic difficulty.

REFERENCES

[1] A. Ashlock, C. Kuusela, and N. Rogers. Hormonal systems for prisoners dilemma agents. In *Proceedings of CIG 2011*, pages 63–70, 2011.

[2] D. Ashlock. Training function stacks to play iterated prisoner's dilemma. In *Proceedings of the 2006 IEEE Symposium on Computational Intelligence in Games*, pages 111–118, 2006.

[3] D. Ashlock, W. Ashlock, and G. Umphry. An exploration of differential utility in iterated prisoner's dilemma. In *Proceedings of the 2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 271–278, 2006.

[4] D. Ashlock and E. Knowles. Deck-based prisoner's dilemma. In *Computational Intelligence and Games (CIG), 2012 IEEE Conference on*, pages 17–24, 2012.

[5] W. Ashlock. Why some representations are more cooperative than others for prisoner's dilemma. In *2007 IEEE Symposium on the Foundations of Computational Intelligence*, pages 314–321, Piscataway, NJ, 2007. IEEE Press.

[6] D.B. Fogel. Evolving behaviors in the iterated prisoners dilemma. *Evolutionary Computation*, 1(1):77–97, 1993.

[7] Lindgren K. and Nordahl M. G. Evolutionary dynamics of spatial games. In *Proceedings of the Oji international seminar on Complex systems : from complex dynamical systems to sciences of artificial reality*, pages 292–309, New York, NY, USA, 1994. Elsevier North-Holland, Inc.

[8] T. Mahlmann, J. Togelius, and G.N. Yannakakis. Evolving card sets towards balancing dominion. In *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pages 1–8, 2012.

[9] J. H. Miller. The coevolution of automata in the repeated prisoner's dilemma. *Journal of Economic Behavior and Organization*, 29(1):87–112, January 1996.

[10] S. Samothrakis, S. Lucas, T.P. Runarsson, and D. Robles. Coevolving game-playing agents: Measuring performance and intransitivities. *Evolutionary Computation, IEEE Transactions on*, 17(2):213–226, 2013.

[11] Justin Schonfeld and Daniel Ashlock. A study of evolutionary robustness in stochastically tiled polyominos. In *Proceedings of the 2005 Genetic and Evolutionary Computation Conference*, pages 19–26, 2005.

[12] J. Togelius and J. Schmidhuber. An experiment in automatic game design. In *Computational Intelligence and Games, 2008. CIG '08. IEEE Symposium On*, pages 111–118, 2008.

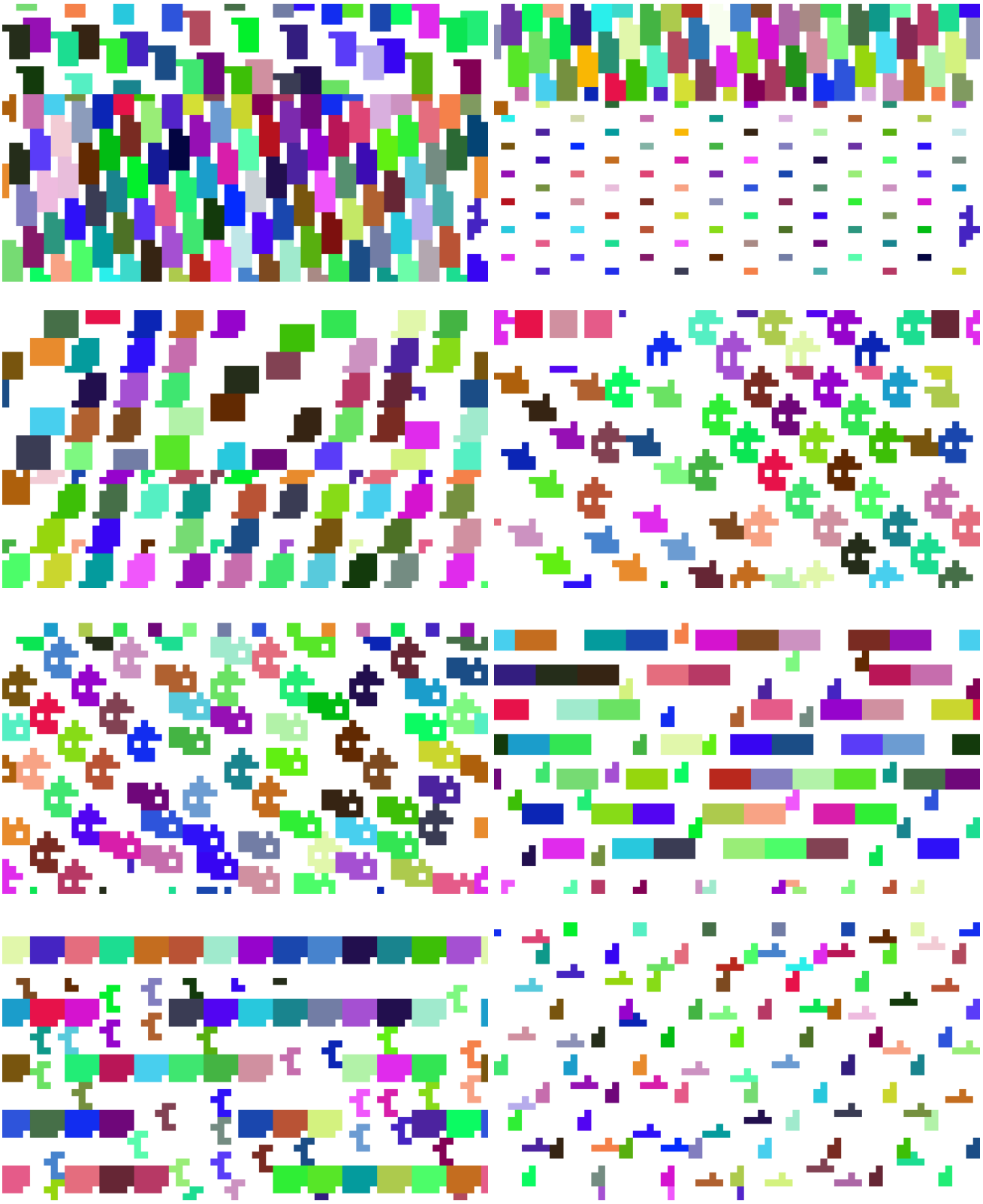

Fig. 12. Example of final boards states for selected pairs of automata. Each polyomino played is assigned its own random color.