Applying an Adaptive Generative Representation to the Investigation of Affordances in Puzzles

Daniel Ashlock
Department of Mathematics and Statistics
University of Guelph
Guelph, Canada
dashlock@uoguelph.ca

James Montgomery
School of Technology, Environments and Design
University of Tasmania
Hobart, Australia
james.montgomery@utas.edu.au

***Abstract*—This study uses a real-coded representation to encode discrete strategies for playing a simple grid based game. This representation is able to adapt itself on the fly to the local situation in the game. This adaptability makes the representation particularly suitable for finding good play strategies which, in turn, permit us to explore biases in the play strategies and even to compare instances of the game for difficulty. Variability in the best results achieved by the solver can be used to gauge difficulty, while the shape of the distribution of best results can indicate how interesting an instance is. Results indicate that the variety and combination of affordances produce instances of the game with varying degrees of anticipated difficulty and interestingness, and confirm that the solver can be used to evaluate the quality of different affordance combinations for producing good game instances. Design principles may also be discovered through post hoc analysis of instances with high or low average best score.**

I. Introduction

Puzzle games like Minesweeper™ [12] or Same Game [11] have the property that there are a huge number of different instances of the game. The random game generator can create easy games, hard games, games that are impossible, or games which turn on purely random choices. Deductive reasoning in minesweeper sometimes reveals that there are two possible moves, one of which must result in death. In this study we create a family of simple puzzles, played on a grid by an agent that moves on the grid, that can be used for research into instances of puzzle-games and to tune versions of puzzle games. A recently developed representation [1] with properties that make it appropriate for playing the game in this study is used in an evolutionary solver for instances of the game.

The family of games used in and invented for this study are called, in aggregate, *The Affordance Sandbox*. These games are characterized by an exclusion law. The player scores for entering squares they have not visited before, but may not enter a square they have visited before. As the player moves, they create barriers to their own movement. The player is given a ration of moves and seeks to maximize their score before they runs out of moves. A variety of tokens are placed on the grid. Moving into a square with a token grants the player an *affordance* [10]. Affordances are also called *power-ups* in video games [9]. The adaptive representation used in this paper can rapidly solve instances of the game; the variability of outcomes located by the evolutionary solver is a measure of the difficulty of an instance of the game.

The goal of the study is to demonstrate the ability to sort instances of the game by difficulty using both the maximum score and the variability of performance of the evolutionary solver. The research question is determining how including or excluding affordances will change this measurement of game difficulty. Both different instances of the game with the same affordances and instances of the game with different affordances are compared. This permits the generation of games with predictable difficulty and allows some estimation of the degree to which a game instance is interesting.

There is growing interest in evaluating the affordances of games so that appropriate rule sets [4] and engaging non-player characters [5] can be developed with little or no manual definition. While some of these efforts involve user studies (e.g., [7]) many attempt to completely automate the process.

Evolving engaging instances of a game is an example of *search-based procedural content generation* (SBPCG), a variant of procedural content generation that incorporates search rather than generating acceptable content in a single pass. SBPCG is typically used when a single pass will not suffice to locate content with the desired qualities. A survey and the beginnings of a taxonomy of SBPCG can be found in [13]. We now turn to the representation used by the game solver.

A *generative representation* [6], in evolutionary computation, is one that does not directly encode a solution to the problem of interest. Rather, it gives directions for constructing a solution. This study examines an application of a general *adaptive* generative representation (AGR), first defined in [1], to instances of our simple family of games. In the earlier publication, the AGR was used to solve the self-avoiding walk problem [2], a standard evolutionary computation problem on which our simple game is based. It was shown that the AGR could solve instances of the self-avoiding walk (SAW) problem of unprecedented size and also could be used to bias the walk to move in particular directions.

The remainder of this study is structured as follows. Section II gives the game and the available affordances. Section

III discusses the details of the AGRs used in this study. Section IV gives the experimental design, Section V presents and discusses results, and Section VI draws conclusions and outlines potential next steps.

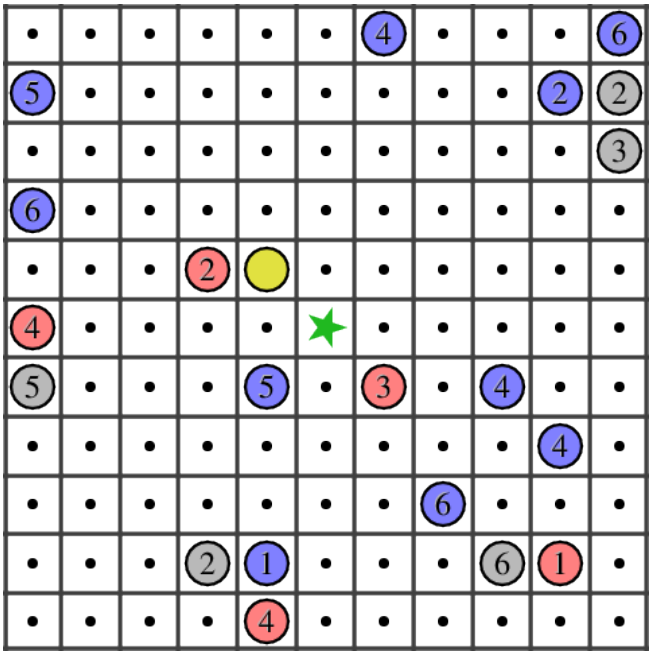


Fig. 1. An 11×11 example board using all four affordances. The green star marks the agent's starting position.

II. A Grid Game with Affordances

This game is played on a square grid with an odd side length $N = 2S + 1$. The agent and tokens representing affordances occupy squares of the grid. The agent begins in the middle of the board. The agent may move up, down, left, or right and its strategy consists of the sequence of moves made. The agent is initially given $3S$ moves it can make. The agent executes its moves in the order specified by its gene. Moves that attempt to leave the grid or enter a square that has been entered before are not executed and so wasted. When the agent enters a square it has not visited before it scores one point. This is the basic game. The game is modified by adding *affordances* which act as power-ups or modify the games rules in some way. Tokens are distinguished by color and, if they have variable effect, have numbers on them. Figure 1 gives an example of an instance of a game with 10 blue tokens, five red ones, a gold token, and five black tokens. The effects of the tokens are as follows.

- A **blue** token adds its number to the agent's number of available moves. This increases the possible score and creates the simple strategic goal of reaching all the blue tokens without running out of moves or getting stuck.
- A **red** token clears a number of filled squares equal to its number, or all currently filled squares if there are fewer than the token's number. Squares are cleared in the order in which they were filled. Clearing a square does not restore any token that used to be in that spot.
- A **gold** token activates a reward for changing direction, one point for each change of direction. This is an irreversible state change that potentially increases subsequent scoring substantially and so early acquisition of the gold token is an important strategic goal.
- A **black** token increases the reward for visiting a new square for a number of moves equal to its number. This is a *matching* effect. The black tokens charge a pool of points used to match the score for entering new squares as it is earned. Black tokens increase the reserve in the matching pool; the pool decreases as it pays out to match points for entering squares.

Instances of the game may use some, all, or none of these affordances and one of the goals of the study is to examine the degree to which the presence of various affordances affects play. A *game specification* gives the number of each of the affordances placed on the board. An *instance* of the game gives the precise placement of all affordances, given a specification.

The initial ration of $3S$ moves is enough to reach any point on the board and so the agent can always avail itself of at least some of the affordances. Blue affordances extend the number of moves, red ones free up space to be moved into, the gold affordance increases the strategic complexity while creating more ways to score, and the black affordance increases the score, but in a way that requires strategic play.

The gold token, as we will see in the experiments, creates the largest change in the game: paths the same length have different worth to the agent's score once the gold token is captured.

III. Specification and Justification of the Adaptive Generative Representation

The representation used is an adaptation of that used in [1]. The representation consists of a list of numbers drawn from the unit interval. Moves are generated by moving through the chromosome, with the number at each loci generating a single move. The directions in which the agent is permitted to move by the rules of the game are enumerated in the order of the squares above, below, left, and right of the agent's current position. The unit interval is then partitioned into k equal intervals based on the number k of permissible moves. The move corresponding to the interval in which the current genetic loci lies is the move selected by the agent. If there are no admissible moves, then the agent's play ends.

The game stops when the agent runs out of moves or reaches the end of its genome. The advantage of using the AGR is that it cannot encode a move that is not allowed by the rules – it always chooses a direction to move that takes it to an empty position, if one exists. In comparison to a direct encoding, one that simply specifies which way to move in each time step, the AGR encoding removes from consideration an exponentially large fraction of the space of possible sequences of moves. This was demonstrated in [1] in explaining the ability of the AGR to solve large instances of the SAW problem.

The grid game in this study is a modification of the SAW, both because the basic game does not allow enough moves to solve the full SAW problem, and because the affordances change the game to be substantially more complex than the original SAW problem. Despite this increased complexity, the AGR is a highly effective basis for an evolutionary solver for the grid games used in this study.

An example of the AGR in operation is shown in Example 1. Notice the number of "no choice" moves and moves with less then three choices. This suggests that this representation can radically reduce the size of the effective search space.

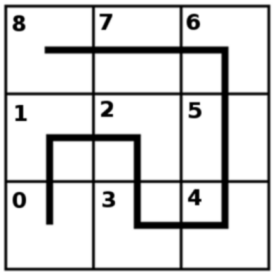
Example 1: Suppose we have the chromosome

$$(0.23, 0.56, 0.81, 0.95, 0.17, 0.22, 0.32, 0.41)$$

and we are traversing a 3×3 grid, starting in the lower left corner. Assuming the order we examine the moves in is UDLR, this chromosome would specify the walk as follows:

Position	Step	Move	Choices	Value
(0,0)	0	Start	2	n/a
(0,1)	1	UP	2	0.23 (bottom half)
(1,1)	2	RIGHT	3	0.56 (top half)
(1,0)	3	DOWN	1	0.81 (top third)
(2,0)	4	RIGHT	1	0.95 (no choice)
(2,1)	5	UP	1	0.17 (no choice)
(2,2)	6	UP	1	0.22 (no choice)
(1,2)	7	LEFT	1	0.32 (no choice)
(0,2)	8	LEFT	0	0.41 (no choice)

Yielding the walk:



IV. Experimental Design

The evolutionary solver used adopts parameters from earlier studies of the SAW problem [1], [8]. Given an instance of the game the solver operates on a population of 1000 AGR chromosomes of length 98 (this should be a good deal longer than needed). Fitness consists of playing the game and recording the score, which the algorithm tries to maximize. Selection and replacement in the algorithm are performed with size seven single tournament selection [2]. A run of the evolutionary algorithm is continued for 10,000 updatings (instances of tournament selection). In an instance of tournament selection, seven members of the population are selected. The two most fit, breaking ties uniformly at random, are copied to replace the two worst. The new chromosomes undergo two point crossover and then have 1–3 loci replaced with new numbers drawn uniformly from the unit interval. The number of loci replaced is selected uniformly at random.

An experiment is performed on a single game specification, like "11×11 board with 10 blue and five red tokens". In the experiment 100 different instances of the specification are generated, placing tokens uniformly at random in distinct grids and avoiding the center grid where the agent starts. When a token has a numerical effect, like increasing the number of moves the agent can make, the number is printed on the token. These numbers are selected uniformly at random in the range 1–6. For each instance of a game specification, the evolutionary solver is used for 30 independent runs using the AGR. The best scores for the 30 runs are saved to characterize the difficulty of the instance.

Initially two experiments using 7×7 boards with blue and red affordances were performed. These showed almost no variation in best scores – the distribution of best scores over all 100 instances for the more variable experiment is shown in Figure 2. The distribution of average best fitness values *across* instances is due to the way the values for move extending tokens are generated, which essentially imposes an upper bound on the achievable score for a given instance: $3S$ plus the sum of blue token values. While the consistency across trials demonstrates the power of the AGR on these games, it leaves little traction to demonstrate the utility of the AGR-based solver. For that reason, four experiments were performed with 11×11 boards and the affordances shown in Table I. These values are chosen to put a fairly large number of tokens on the board without completely obstructing it. As this is an initial exploration of the affordance sandbox, these values are exploratory and somewhat arbitrary.

TABLE I
NUMBER OF AFFORDANCES IN EXPERIMENTS WITH 11×11 BOARDS.

	Affordances			
Experiment	Blue	Red	Gold	Black
3	10	5	0	0
4	10	5	1	0
5	10	5	0	5
6	10	5	1	5

The experimental design presumes that the use of red and blue affordances creates a puzzle of low to intermediate difficulty. If blue tokens do not lead a player into a trap then they are an unmixed benefit. Red tokens can be wasted, if the number of grids they re-open is larger than the number of currently obstructed grids, but they are also not hard to use strategically. For that reason all four 11×11 experiments use a generous ration of blue and red tokens.

The most problematic token is the gold one. This token activates a second mechanism for gaining score and makes the nature of the path taken important. This increases the complexity of the strategy of the game, save for the obvious goal of reaching a gold token as soon as possible, and the strategy after activating the turn reward with the gold token

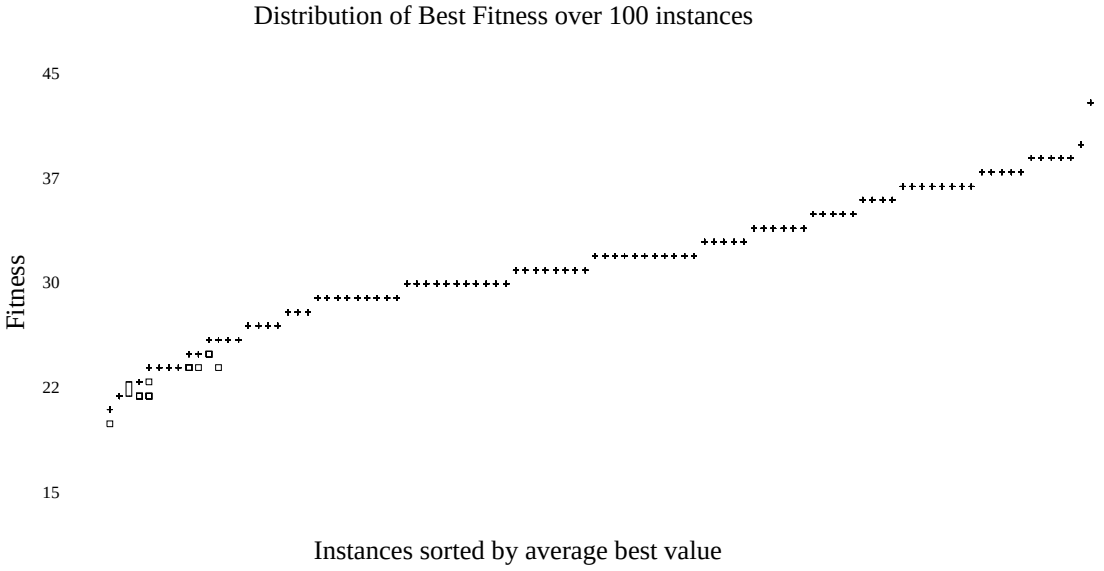


Fig. 2. Shown are the distributions of best final fitness values of 30 runs of the AGR-based algorithm on 100 randomly generated instances of a game with six blue and three red tokens on a 7×7 board. The objects above are box plots, often truncated by having a distribution composed of a single value.

is to turn as much as possible, but doing this increases the number of locally high-scoring paths that end in a trap.

The black tokens represent additional score that must be earned, in particular black tokens must be captured while enough space remains (or red token to clear space). This means that black tokens represent a moderately difficult resource-management problem. Each sort of token increases strategic difficulty – and this increase is cumulative. We expect to see both an increase in average best score and in variability of the best score as we increase the number of types of affordance available to the agent playing the game.

V. Results and Discussion

Figure 3 shows the experiments with no affordances beyond the blue and red in uniform use, and those with a gold and five black tokens added. Figure 4 shows the experiment that uses all four affordances.

The top panel of Figure 3 shows the basic game specification and the 100 instances of this specification vary in high score from the mid-thirties to the low sixties, which is to be expected given the initial budget of $3S = 15$ moves and the expected value of 10 discrete uniform random variables in $[1, 6]$ afforded by the blue tokens. This is the least top-to-bottom variation of the best fitness located by the evolutionary solvers in any of the 11×11 experiments. Some individual instances, including the two with the highest score, exhibit *no* variation of best fitness found within the instance. Only a few of the instances in this set of experiment show a high level of variability. The variation in fitness between instances is almost exclusively due to the variation in the number on the (blue) tokens.

When we add the gold token to the mix, shown in the center panel of Figure 3, the range of fitness values almost doubles in comparison to the red-blue only experiment. The variability of individual runs also increases sharply; the instances with no variation are absent in the 100 sampled instances. The gold token dramatically increases the variability of the game. Since getting to the gold token is not difficult, this suggests that the increase in difficulty is substantially due to the side effects of increasing the agent's score by turning as often as possible, which is otherwise a contraindicated strategy for filling a grid without becoming trapped [8].

The third panel of Figure 3 shows the result of adding five black tokens to the red and blue baseline. These instances are more variable than the pure red-blue and less variable than those with the gold token (note that the vertical scales of the gold and black experiments are different). The black-token experiments also lack instances with no variation. While the black token experiments have a lower maximum than the gold token experiments (the gold token doubles the score for all moves after it is obtained, the black tokens can add at most 5–30 points, depending on their face values), they also have a higher minimum. This suggests that, at the low end, the added score induced by the black tokens is easier to obtain, perhaps because it doesn't require the riskier turning behaviour.

Figure 4 gives the score-variability for the experiments with all four affordances present. Having all four affordances present increases the minimum and maximum best scores as well as the variability of scores. This suggests that the additional complexity caused by the gold and black tokens is additive.

A. Fitness landscapes of instances

The evolutionary solver uses a large population, crossover, and a moderate mutation rate – one that earlier work suggests is a good balance between exploration and preserving extant good results [1]. This means that high variability of the

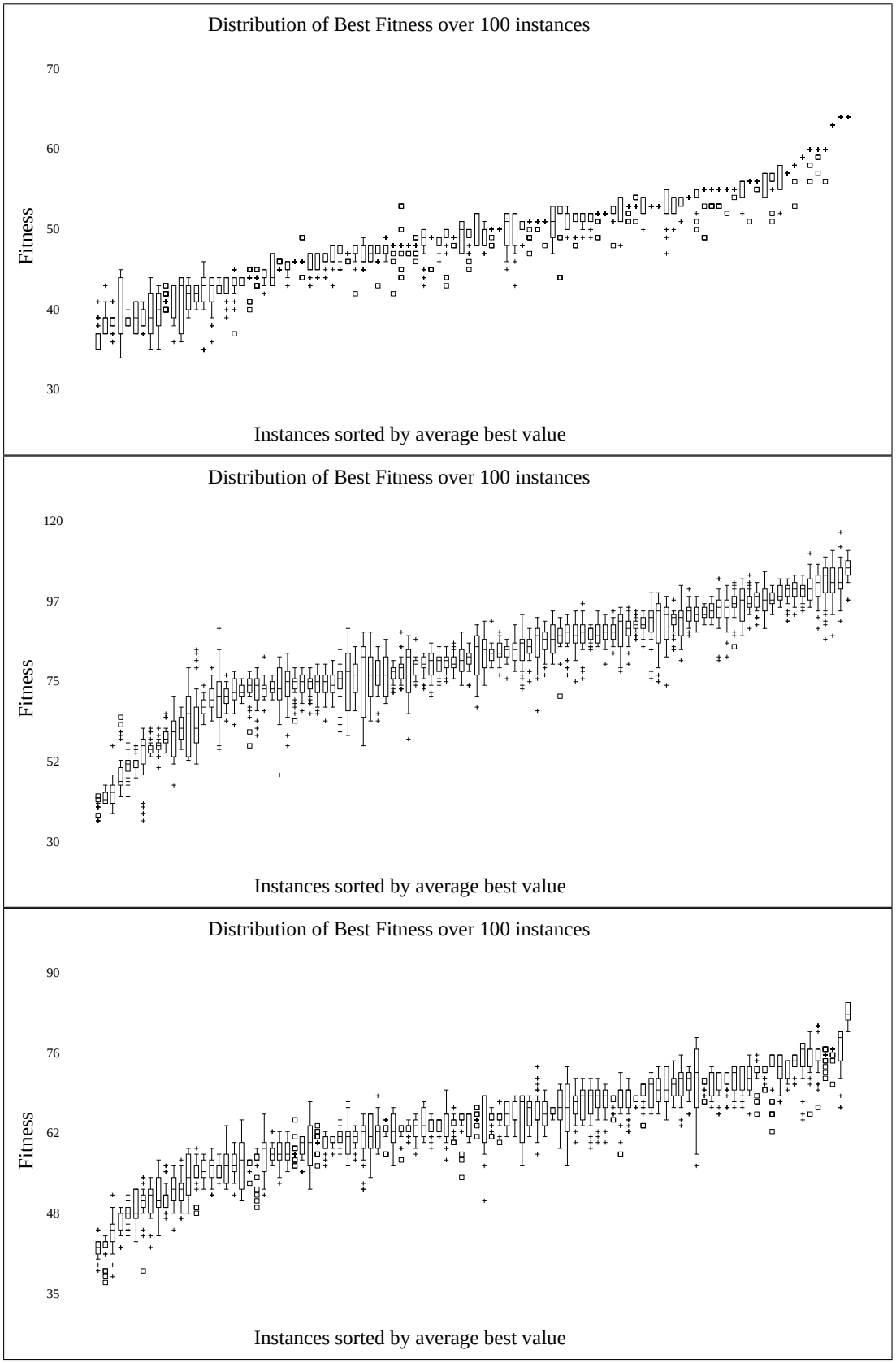


Fig. 3. Shown are the distributions of best final fitness values of 30 runs of the AGR-based algorithm on 100 randomly generated instances of three different games on an 11×11 board. All three game have five blue tokens and five red tokens. The middle game adds a gold token, the bottom adds five black tokens. Note the differences in the vertical scale of the plots.

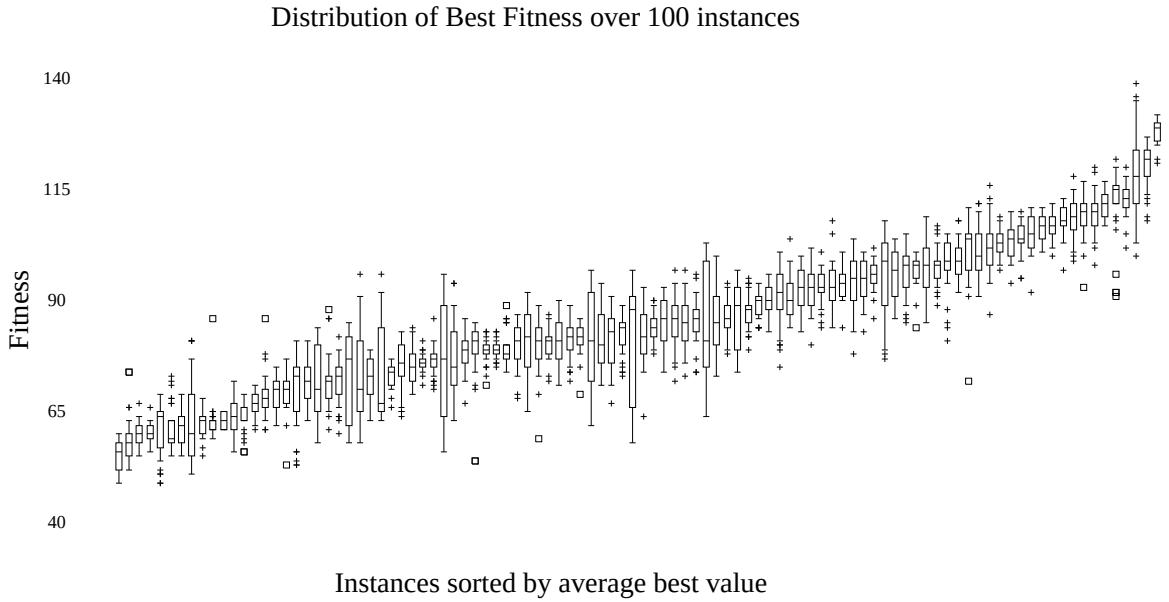


Fig. 4. Shown are the distributions of best final fitness values of 30 runs of the AGR-based algorithm on 100 randomly generated instances of a game with 10 blue, five red, one gold, and five black tokens on an 11×11 board.

best score located represents a fitness landscape with several distinct optima. Unless a human player has and exercises a good analytic faculty, they will be prone to fall into the same traps that AGR-based evolving agents do. This causes us to conjecture that we have the basis for both a hardness and interestingness measure for the grid games.

Hardness is easier: it is proportional to the variability of the best scores found. Interest would also need to take top-to-bottom variation of the best scores as well as relative frequency of the highest scores in comparison to others. Rarely found high scores are indicative of a challenging puzzle.

Another factor, not visible in the current reporting structure for these experiments, is the diversity of solutions that achieve a given score. Blue tokens can be captured in a flexible order and with flexible timing; red tokens become needed as an agent gets stuck, gold tokens should be captured as soon as possible, and black tokens have their own timing issues, but are also flexible. If there are very few paths to a maximum score, but many to lower scores, then a puzzle has clever design.

Figure 5 shows the boards with the highest and lowest average scores of the 100 instances examined in Experiments 4, 5, and 6. Experiments 4 and 6 use a gold token. The two high scoring boards both have a cluster of blue tokens near the gold token; the two low scoring boards do not. The blue tokens are simple ways of enabling a higher score and they extend play. This means that solutions picking up blue tokens are likely to be discovered rapidly by the evolutionary solver. If the gold token is near several blue tokens it means that solutions that find the gold token early, and so maximize the score from the turning reward, are easier to discover. This gives us a design principle.

The low and high boards for Experiment 5, without a gold token, seem to show the pattern that the low scoring board groups the blue tokens more than the high scoring board. Having blue tokens scattered would cause the evolutionary solver to range around more and, as a side effect, may pick up the black tokens in a well-spaced out manner. It is also worth noting that the total value of the blue tokens is much higher in the high-scoring board from Experiment 5 than in the low one. This means that more moves will be made, if the solution does not involve a trapping cul-de-sac, and so more of the black tokens will collected and exploited. Beyond the apparent patterns with the blue tokens, there is no really obvious difference between the high and low boards.

VI. Conclusions and Next Steps

This study demonstrates that the family of grid games proposed can be solved well by the AGR; the lack of score variability in the simpler version of the game speaks to this. This, in turn, suggests that good progress has been made toward the goal of automatically sorting game instances by difficulty. In the future, additional solvers such as Monte-Carlo Tree search [3] could be used to enhance analysis.

The next logical step is to use the evolutionary solver as a fitness function to design instances of the game with desired levels of hardness and challenge. Statistical abstractions like the fraction of times the highest score was discovered or the value of the highest score relative to known instances of a game specification can be employed to search for game instances that are challenging.

Another topic of interest is gauging if a game has a good number of types of affordances. Too few types makes a game easy for a player to master while too many can cause the game to be to difficult to learn, saturating a player's ability to

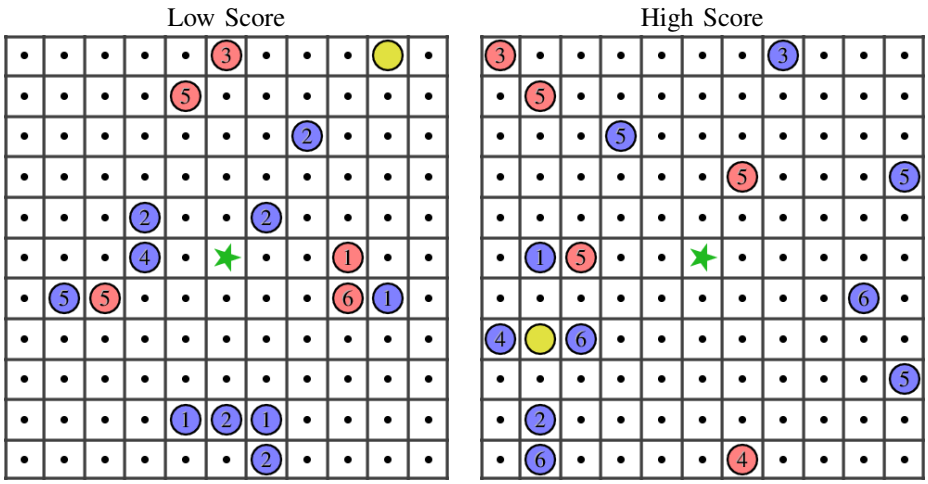


Fig. 5. Boards with the highest and lowest average score for the 4th, 5th, and 6th experiment.

learn the game. This latter effect can be ameliorated, to some degree, by introducing affordances serially to the player, but this still leaves the question of a good starting number of types of affordances to use.

Similarly, given the types of affordances used, the number and numerical range of each type of affordance needs to be set. This problem is tractable for solution by the evolutionary solver presented here, and represents an early goal for future work. The affordances in this study appear to have additive complexity, but the potential for non-additive interaction is another interesting issue for game design and automatic content generation [4], [5], [13].

A. Other possible affordances

Performing research on the useful or engaging number and number of types of affordances would be enhanced by adding interesting additional affordances to the family of grid games. Here are some potential extension of the set of affordances, which we propose for future work.

- A purple token has two effects. It adds its value to the agent's score immediately and, if the added score for turning associated with the gold token is active the purple token deactivates it. This increases the utility of having multiple gold tokens and also creates a strategic dilemma in which purple tokens are best captured last.
- White tokens permit the agent to enter full squares for a number of moves equal to the token's value. This represents a strategic escape hatch that could play out in a large number of ways if much of the board is obstructed.
- A green token adds new tokens to the board. These tokens should be generated before play in fixed locations to prevent the introduction of stocasticity into game instances. The green token uncovers previously hidden tokens – possibly in obstructed squares that would need to be uncovered by the action of red tokens or made accessible by the action of white tokens.
- A dark-green token requires a number of moves equal to its value to traverse, it is a swamp or quagmire token that would enhance strategic complexity – a "partial" barrier.
- A silver token forces the agent to keep moving in the same direction for a number of steps equal to its value. These moves may be impossible and so wasted.
- Arrow tokens force the direction of exit from a cell in the direction the arrow is pointing.
- Bomb tokens could clear an area around the bomb; this creates the strategic issue of having full tokens for the bomb to clear. A bomb might simply clear obstructions *or* it might also destroy tokens. The area cleared is another design issue.
- Brown tokens are numbered, but the number does not indicate a strength, rather entering a brown token teleports the player to the other one. Brown tokens are single use teleporters that vanish after use. Brown tokens increase the connectivity of the board and so change the number of options.

The proposed white and brown tokens increase the connectivity of the board, in different ways. A good hypothesis to test is if boards with a higher connectivity engender a higher variability of score because there are more possible paths. Another possible way to test this is to remove the walls at the boundary of the board, treating to board as torroidal.

Another tool for design of the grid games would be to create walls. These wall are "pre-filled" grids and might or might not be susceptible to the action of white tokens.

The system presented in this study is intended as a sandbox for study of the affordances in simple puzzle games. Adding affordances is relatively simple in the current code design and the evolutionary solver is highly instance specific, meaning it is probably better at not being overwhelmed than a human player.

In follow-up studies it may be profitable to examine more game specifications with a specific set of affordances before increasing the number of affordances in the game. The present system does not track the tokens picked up by the solver. Adding the ability to tally which tokens were picked up will permit more sophisticated analysis.

REFERENCES

[1] D. Ashlock and J. Montgomery. An adaptive generative representation for evolutionary computation. In *Proceedings of the IEEE 2016 Congress on Evolutionary Computation*, pages 1578–1585, Piscataway NJ, 2016. IEEE Press.

[2] Daniel Ashlock. *Evolutionary Computation for Opimization and Modeling*. Springer, New York, 2006.

[3] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. I. Cowlingand P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.

[4] Michael Cook, Simon Colton, and Azalea Raad. Inferring design constraints from game ruleset analysis. In *IEEE Conference on Computatonal Intelligence and Games, CIG*, 2018.

[5] Christian Guckelsberger, Christoph Salge, and Julian Togelius. New and surprising ways to be mean. In *2018 IEEE Conference on Computational Intelligence and Games, CIG'18*, pages 157–164, Maastricht, 2018.

[6] Philip F. Hingston, Luigi C. Barone, and Zbigniew Michalewics. *Design by Evolution*. Natural Computing Series. Springer, New York, NY, 2008.

[7] David Milam, Magy Seif, and El Nasr. Analysis of level design 'push & pull' within 21 games. In *International Conference on the Foundations of Digital Games, FDG '10*, pages 139–146, 2010.

[8] J. Montgomery and D. Ashlock. Applying the biased form of the adaptive generative representation. In *Proccedings of the 2017 IEEE Congress on Evolutionary Computation*, pages 1079–1086, Piscataway NJ, 2017. IEEE Press.

[9] Dan Pinchbeck. Counting barrels in quake 4: affordances and homodiegetic structures in fps worlds. In *DiGRA 2007*, pages 8–14, Tokyo, Japan, 2007.

[10] Dan Pinchbeck. An affordance based model for gameplay. In *DiGRA 2009*, London, UK, 2009.

[11] Maarten P.D. Schadd, Mark H.M. Winands, Mandy J.W. Tak, and Jos W.H.M. Uiterwijk. Single-player monte-carlo tree search for samegame. *Knowledge-Based Systems*, 34:3 – 11, 2012. A Special Issue on Artificial Intelligence in Computer Games: AICG.

[12] A. Suganuma, S. Ohara, H. Inoue, and N. Tetsutani. Feature classification of heart rate variability depending on difficulty levels of a puzzle game. In *2018 International Workshop on Advanced Image Technology (IWAIT)*, pages 1–4, 2018.

[13] J. Togelius, G. Yannakakis, K. Stanley, and C. Browne. Search-based procedural content generation. In *Applications of Evolutionary Computation*, volume 6024 of *Lecture Notes in Computer Science*, pages 141–150. Springer Berlin / Heidelberg, 2010.