Monte Carlo Analysis of a Puzzle Game

Cameron Browne and Frederic Maire

School of Electrical Engineering and Computer Science,
Science and Engineering Faculty, Queensland University of Technology,
Gardens Point, Brisbane, 4000, Australia
c.browne,f.maire@qut.edu.au
http://www.qut.edu.au

Abstract. When a puzzle game is created, its design parameters must be chosen to allow solvable and interesting challenges to be created for the player. We investigate the use of random sampling as a computationally inexpensive means of automated game analysis, to evaluate the BoxOff family of puzzle games. This analysis reveals useful insights into the game, such as the surprising fact that almost 100% of randomly generated challenges have a solution, but less than 10% will be solved using strictly random play, validating the inventor's design choices. We show the 1D game to be trivial and the 3D game to be viable.

1 Introduction

Any newly designed game must undergo a process of playtesting and refinement to ensure that its equipment and rule set are optimally tuned to realise an interesting playing experience. This can be a painstaking and tedious process that may take years, but can be assisted by mathematical and/or computer modelling of the game in question [2]. However, analyses that rely on full game tree expansions or complete enumerations of the design space can be prohibitively expensive to compute for real-world cases of even modest complexity.

In this paper, we investigate ways in which random sampling can be used instead, to quickly give some insight into a game's inherent nature with less computational effort. We use as our test case a new puzzle game called BoxOff.

Monte Carlo approaches have had spectacular success in game AI over the last decade, especially *Monte Carlo tree search* (MCTS) methods, which now drive the world champion AI players of many games [5]. MCTS approaches have been especially successful in the related field of *general game playing*, i.e. the study of computer programs for playing a range of games well rather than specialising in any one particular game, as they allow the AI to make plausible moves for a given game without any domain-specific strategic or tactical knowledge [6].

However, we use Monte Carlo approaches for a different purpose in this study. We are less concerned with *how to play* the game than with *how well it plays*. This study can be phrased as an optimisation problem – given the basic design of a game, what are the parameters that provide the best experience for the player? – which places it within the remit of *procedural content generation* [13].

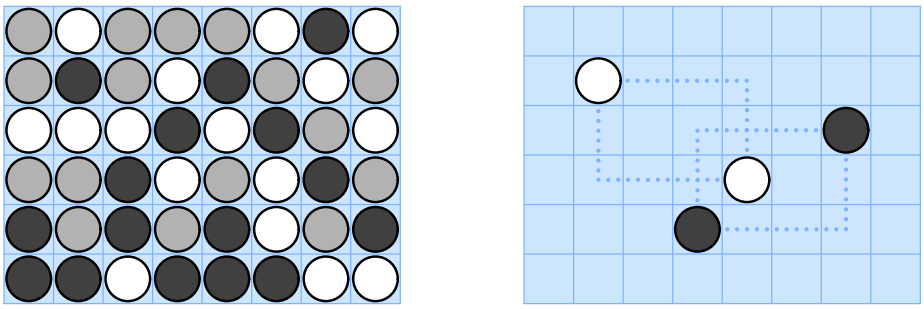

Fig. 1. A 6×8 challenge with three colours, and legal white and illegal black moves.

1.1 BoxOff

BoxOff[1] is a solitaire puzzle game invented in 2013 by American game designer Stephen Meyers [9]. The rules are as follows:

1. *Start*: The game starts with pieces in C colours randomly placed to cover all cells of a regular rectangular grid (the *board*). This defines the *challenge* to be solved. Figure 1 (left) shows a challenge on the standard 6×8 board with $C = 3$ colours.
2. *Play*: The player then makes a series of moves, each involving the removal of a pair of same-coloured pieces that occupy a *box* (rectangle) that includes no other pieces. For example, the two white pieces in Figure 1 (right) can be removed because they occupy a box (dotted) that is otherwise empty, whereas the two black pieces are blocked from removal by the white piece.
3. *End*: The player wins by removing all pieces from the board, else loses if there are no legal moves at any point while pieces remain on the board.

In order to be solvable, each challenge must have an even number N of board cells, and for each colour c the number of pieces P_c must also be even. The standard game is played on a 6×8 board with 16 pieces in each of $C = 3$ colours. For aesthetic reasons, the board dimensions are typically chosen to be as square as possible for a given N, and the piece counts of each colour $P_1 \dots P_C$ are typically chosen to be as similar as possible while summing to N.

1.2 Game Design Goals

The design parameters for BoxOff are therefore the board size N and number of colours C. We define the *design space* to be the set of valid combinations of N and C, the *challenge space* to be the set of possible challenges for a given design, and the *solution space* to be the set of possible solutions for a given challenge. We are interested in whether random sampling of the design space of BoxOff can shed some light on questions such as:

[1] The name "BoxOff" was coined by the first author.

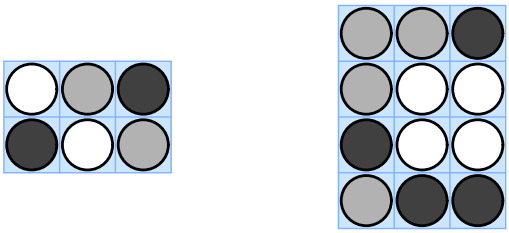

Fig. 2. Unsolvable 2×3 and 4×3 challenges.

- *What are good values for the board size N for this game?*
- *What are good values for the number of piece colours C?*
- *How likely is a randomly generated challenge to be solvable?*
- *How likely is a randomly generated challenge to be interesting?*

Solvability The most important question regarding BoxOff – and indeed most puzzles – is that of *solvability*, namely how likely it is that a given challenge will actually have a solution. There are two relevant probabilities: $P(S_r)$ the probability that randomly sampled challenges will be solvable with random play, and $P(S_p)$ the probability that randomly sampled challenges will be solvable with perfect play.

It is easy to artificially construct unsolvable challenges. For example, Figure 2 shows a 2×3 challenge with no opening moves (left), and a 3×4 challenge that allows several moves but never the removal of the lower left piece (right). Conversely, it is easy to artificially construct challenges that are guaranteed to be solvable, by starting with an empty board then adding same-coloured pairs such that no odd-sized regions of empty cells are created, until the board is full. Greg Schmidt uses this approach in his Axiom AI player for BoxOff [12].

However, a player using a physical version of the game does not want to worry about such niceties. A patient player can construct challenges guaranteed to be solvable if they wish, but most players will just want to place pieces randomly and as quickly as possible, hence the question of solvability becomes important.

Firstly, players must have confidence that the majority of challenges they construct will be solvable with perfect play, otherwise there is little point in playing the game. Knowing the likelihood that challenges are solvable also helps players gauge their progress based on win rate.

Secondly, it would be detrimental if most challenges could be solved by random play. Challenges that can be trivially solved without thought or forward planning will be of little interest to most players. Well designed puzzles tend to display structure or *dependency*, such that certain moves reveal key information that allows further moves to be made, and must be performed in a certain order [4]. Challenges that can be solved by making random moves without any planning indicate a lack of such dependency, and are described as being *susceptible to random play*.

Table 1. Solvability of complete and sampled challenge sets.

		Complete			Sampled
	N	Challenges	Solvable	Ratio	Ratio
3×4	12	1,523	682	**0.4478**	**0.4438**±0.0097
4×4	16	105,561	59,545	**0.5641**	**0.5592**±0.0097
4×5	20	13,098,310	9,036,038	**0.6998**	**0.6934**±0.0090

To answer the game design questions, we consider different complexity measures. *The majority of challenges should ideally be solvable by perfect play but not by random play.* We therefore want to maximise $P(S_p)$ while minimising $P(S_r)$. Section 2 discusses the complexity of BoxOff, Section 3 explores the solvability of randomly generated challenges, Section 4 looks briefly at the interestingness of randomly generated challenges, and Section 5 summarises our results.

2 Complexity

For this analysis, we implemented two types of AI solver for BoxOff:

1. S_r: Random solver that applies a random legal move each turn, until the game is won or lost.
2. S_p: Depth-first backtracking solver that returns the first valid solution found (if any). S_p recursively tries each available action of each state in order, using a *transposition table* to avoid repetition [11].

2.1 Challenge Space Complexity

The *challenge space complexity* of a given board size is the number of distinct challenges that it allows, not counting reflections, rotations and colour permutations.

Table 1 shows the number of distinct challenges found in complete enumerations of smaller board sizes up to 4×5 with $C = 3$ colours, the number of these challenges that are solvable with perfect play S_p, and the ratio of these two numbers in bold. The rightmost column of Table 1 shows the observed solvability ratios of 10,000 randomly generated challenges for the same board sizes, with 95% confidence intervals. Randomly sampled challenges appear to offer a fair representation of the complete set of actual challenges. The number of challenges increases exponentially with board size, for example, there exist approximately 1.355×10^{21} challenges for the standard 6×8 game played with $C = 3$ colours, including reflections, rotations and colour transpositions. This makes exhaustive analysis of even the standard board size impractical.

2.2 Game Tree Complexity

The *game tree complexity* of a challenge is defined as "the number of leaf nodes in the solution search tree of the initial position" [1, p.160]. For example, Figure 3

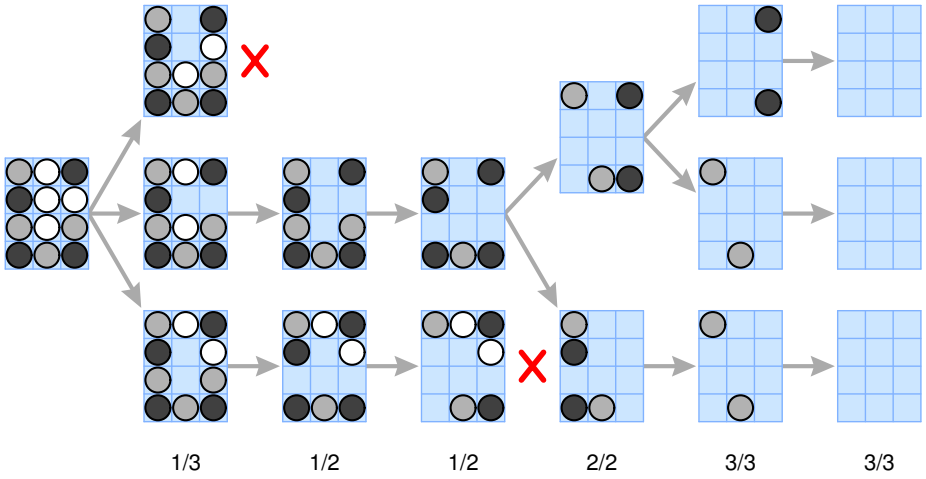

Fig. 3. Full game tree expansion, showing ratios of winning choices.

shows the full game tree expansion of a simple 4×3 challenge with $C = 3$ colours, showing the ratio of winning moves each turn (a statistic used later in Section 4). Obviously, no game of BoxOff on N cells can ever exceed $N/2$ moves, and a solution will always be of length $N/2$. For example, the 6^{th} move wins in all cases on the $N = 12$ board shown in Figure 3. Such complete game tree expansions are infeasible for larger boards, but game tree complexity can be estimated based on a challenge's *branching factor*, i.e. number of legal moves M_t for each turn t.

6×8 challenges tend to start with a branching factor of around $M_0 = 26$ legal opening moves, decreasing almost linearly to zero over the course of the game. The product of means $\prod_{t=1}^{N/2} M_t$ gives an estimated game tree complexity of approximately 2.86×10^{22} non-distinct board positions for the 6×8 board with $C = 3$ colours.

2.3 State Space Complexity

The *state space complexity* of a challenge is defined as "the number of legal game positions reachable from the initial position" [1, p.158]. This is equivalent to the number of distinct board positions stored in the transposition table following a complete traversal of all possible lines of play.

A full BoxOff game tree expansion on a 6×8 board will typically involve less than 1×10^8 distinct board states. As a rough rule of thumb, BoxOff games tend to have state space complexity in the order of $2^{N/2}$. While this is a more manageable number than the game tree complexity, it is still prohibitively time consuming to expand full game trees for even medium sized boards, hence our interest in analysing the game through random sampling alone.

2.4 Computational Complexity

We note that a given challenge can be represented as a directed graph G, in which each potential move corresponds to a vertex, and an arc connects vertex p_i to vertex p_j if move p_j cannot be played before move p_i. The game can then be reduced to the problem of finding the largest subgraph in G that does not contain a directed cycle. For example, Figure 4 shows the reduction of the 4×2 challenge shown to its largest acyclic subgraph.

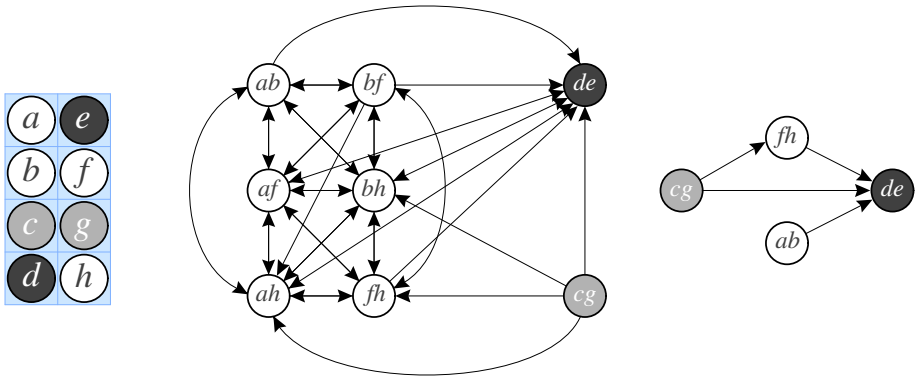

Fig. 4. A challenge, its digraph, and solution subgraph.

A general version of this problem is equivalent to the *feedback vertex set* problem, which was among the first problems shown to be NP-complete by Karp [7]. However, we also note that graphs associated with BoxOff challenges may contain structural properties that allow the design of polynomial time algorithms for their solution. The complexity of the decision problem of the solvability of a given BoxOff puzzle is an open problem.

3 Experiments

As stated in the introduction, we now examine the solvability of 2D, 1D and 3D versions of the game. Recall from Section 1.2 that we especially want to maximise solvability while minimising susceptibility to random play.

3.1 2D Case

The standard 2D version of the game is the case we are most interested in. Figure 5 shows the observed solvability probabilities $P(S_r)$ and $P(S_p)$ for various 2D boards up to size $N = 64$ for $C = 2, 3$ and 4 colours. We only consider board sizes with at least $2C$ cells in each case, to allow at least one pair of each colour, and we only consider board sizes up to $N \leq 64$, in order to allow an efficient

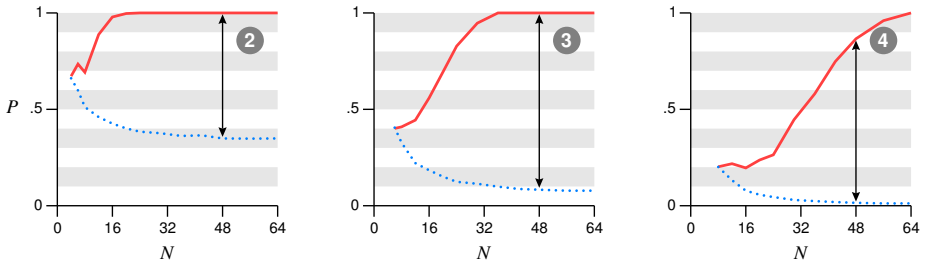

Fig. 5. Observed solvability probabilities $P(S_r)$ (dotted) and $P(S_p)$ (solid) for 2D cases, for $C = 2, 3$ and 4. Arrows show the disparity between $P(S_r)$ and $P(S_p)$ for the default $N = 6 \times 8 = 48$ case, which is the key measurement here.

Table 2. Solvability probabilities for the 6×8 (2D) case for $C = 2 \ldots 6$.

	C				
	2	3	4	5	6
$P(S_r)$	.349±.009	.083±.005	.015±.002	.002±.001	.000±.000
$P(S_p)$	.999±.000	.999±.000	.866±.021	.293±.028	.039±.012

bitboard encoding with 64-bit long integers [3]. Board sizes tested: 2×2, 2×3, 3×4, 4×4, 4×5, 4×6, 5×6, 6×6, 6×7, 6×8, 7×8 and 8×8.

The dotted lines show the observed probabilities of success for the random solver $P(S_r)$ averaged over 10,000 randomly sampled challenges. The solid lines show the observed probabilities of success for the perfect solver $P(S_p)$ averaged over 1,000 randomly sampled challenges (decreasing to 100 for some of the larger board sizes, due to time constraints). The arrows indicate the disparity between $P(S_r)$ and $P(S_p)$ for the standard $N = 48$ (i.e. 6×8) case, which we want to maximise. Note that the random solver S_r solvability curves (dotted) tend to drop sharply, while the perfect solver S_p solvability curves trend upwards to plateau at almost 100% for larger boards.

Table 2 shows the exact solvability probabilities $P(S_r)$ and $P(S_p)$ for the standard 6×8 case, for $C = 2 \ldots 6$ colours. Using two colours, almost 100% of randomly sampled challenges are solvable, although the high random solvability rate of around 35% points to a lack of difficulty. Using three colours, almost 100% of randomly sampled challenges are solvable, with a much lower susceptibility to random play of around 8%. Using four colours, less than 87% of randomly sampled challenges are solvable. Three colours show the greatest difference between $P(S_r)$ and $P(S_p)$ so are the optimal choice here.

In practice, only 1 in around 5,000 randomly sampled challenges prove to be unsolvable on the standard 6×8 board with three colours. Most players could spend their entire lives without constructing a single unsolvable challenge, while still having the luxury of being able to blame bad luck for any failure to solve a particular challenge.

For completeness, Figure 6 shows the observed solvability probabilities $P(S_r)$ and $P(S_p)$ for various 2D boards up to size $N = 64$ for $C = 5$ and 6 colours. It

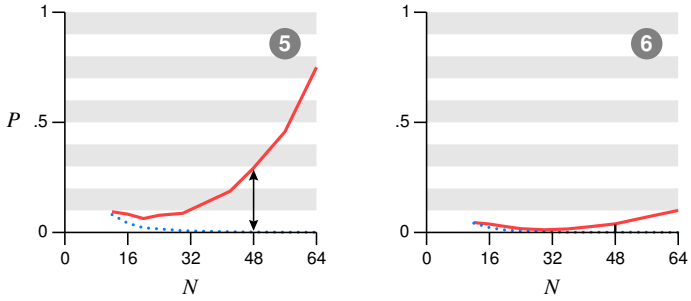

Fig. 6. Observed solvability probabilities $P(S_r)$ (dotted) and $P(S_p)$ (solid) for 2D cases, for $C = 5$ and 6.

can be seen that solvability by random player $P(S_r)$ drops quickly to almost 0% for both $C = 5$ and 6, which is good. However, solvability by perfect play $P(S_p)$ is in general much poorer than when using fewer colours. For example, less than 30% of randomly sampled challenges will be solvable on the standard 6×8 board using five colours, and less than 5% will be solvable using six. This means that most challenges that players set themselves will be unsolvable, which is very undesirable. These findings are consistent with an observation by the game's designer, Steve Meyers, that five and six colours may be suitable for very large boards, e.g. 12×15, but are a poor choice for small or medium sized boards.[2]

Summary: The 6×8 board with three colours seems to be an astute design choice, which allows a good balance between high solvability and low susceptibility to random play.

3.2 1D Case

The simplest version of the game is the 1D case played on a $1 \times n$ board.[3] The pieces start in a line, and the player removes same-coloured pairs in clear line-of-sight of each other. Figure 7 shows the observed solvability probabilities $P(S_r)$ and $P(S_p)$ for various 1D boards up to size $N = 64$ for $C = 2, 3$ and 4 colours. Board sizes tested: 1×4, 1×8, 1×12, 1×16, 1×20, 1×28, 1×36, 1×42, 1×48, 1×56 and 1×64. The first thing to note is that both curves are very similar for each value of C. There is no significant difference between the success rates of S_r and S_p at any point, and these curves would in fact be identical if they were measured on the same sample sets rather than being sampled independently. This is due to an unexpected anomaly that *for any solvable 1D position, no sequence of moves can ever lead to a loss.* This is proven below, and is an example of important knowledge about the game revealed through random sampling.

We characterise the winnable games in terms of a context-free grammar, and show that for such games, any sequence of legal moves wins the game. To

[2] Personal correspondence.
[3] The 1D version of the game might be called "LineOff".

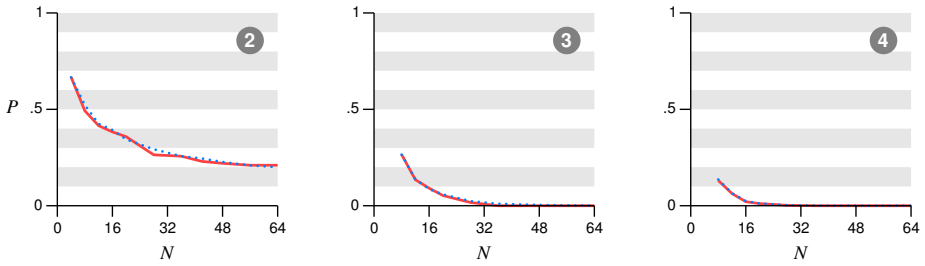

Fig. 7. Observed solvability probabilities $P(S_r)$ (dotted) and $P(S_p)$ (solid) for 1D cases, for $C = 2, 3$ and 4.

Table 3. Average solvability of the 1×48 (1D) case.

	C		
	2	3	4
1×48	0.2268±0.0082	0.0042±0.0013	0.0001±0.0001

compactly represent a challenge, we map a row of coloured pieces to a string of integers. For example, the string "1 2 2 1" codes a challenge where "1" corresponds to a *black* piece, and "2" corresponds to *white* a piece.

We will show that the context-free grammar $\mathcal{G}_1 = (\{S\}, \mathcal{P}, \{1, 2, \ldots, n\}, S)$ defined below generates exactly all winnable challenges. The grammar $\mathcal{G}_1$ has a unique non-terminal S. The set of terminals are the first n integers. The set $\mathcal{P}$ contains the production rules $\{S \to xSx,\ S \to SS,\ S \to \epsilon\}$ where x takes all values in $\{1, 2, \ldots, n\}$, and ϵ denotes the empty string.

We can consider that the rules of the form $S \to xx$ are also part of the grammar as they are obtained by applying the third rule after the first rule. We write $S \xrightarrow{\star} \alpha$, if the string α can be generated by repeated application of the production rules. By abuse of notation, we also write $\alpha \in \mathcal{G}_1$. For example, we have $S \xrightarrow{\star} 322113$, as "3 2 2 1 1 3" can be derived as:

$$S \to \mathbf{3S3} \to 3\mathbf{SS}3 \to 3\mathbf{2S2}S3 \to 32S\mathbf{21S13} \to 322113$$

where the **bold** substrings highlight the latest substitions of S. The grammar $\mathcal{G}_1$ is very similar to a *Balanced Parentheses* grammar.

Lemma 1. *Any string generated by the grammar $\mathcal{G}_1$ is a winnable 1D BoxOff challenge.*

Proof: In the derivation of a string generated by $\mathcal{G}_1$, the sub-sequence of substitions using the first rule of $\mathcal{G}_1$ is of the form $S \to x_1Sx_1$, $S \to x_2Sx_2$, ... , $S \to x_kSx_k$. A winning strategy is to contract x_kx_k, then $x_{k-1}x_{k-1}$, and continue the contractions until x_1x_1. Reciprocally, we have:

Lemma 2. *If a string is a winnable 1D BoxOff challenge, then the string belongs to the language generated by $\mathcal{G}_1$.*

Proof: *Base case*: If the string is of length 2, then the string is of the form xx. Therefore it can be generated by the sequence, $S \to xSx$, and $S \to \epsilon$. *Induction case*: Without loss of generality, we assume that the first character of the string is contracted at step k of a winning sequence of moves. The string is of the form $x_k \alpha x_k \beta$, where β is possibly the empty string. The string α must be contracted before the pair $x_k x_k$. Therefore, by induction on the length of the string, we have $S \overset{\star}{\to} \alpha$. Once the pair $x_k x_k$ is contracted, we are left with the winnable string β. Again, by induction we have $S \overset{\star}{\to} \beta$. In summary, we can derive the initial string as $S \to SS \to x_k S x_k S \overset{\star}{\to} x_k \alpha x_k S \overset{\star}{\to} x_k \alpha x_k \beta$.

The 1D version of BoxOff is uninteresting for a human player because it does not require any forward thinking.

Theorem 1. *A 1D BoxOff challenge is winnable if and only if it belongs to the language generated by the grammar $\mathcal{G}_1$. Moreover, if the challenge is winnable, then any sequence of legal moves is a winning strategy.*

Proof: The first part of the theorem is a direct consequence of the two previous lemmas. We prove the second part of the theorem by induction on the length of the string. *Base case*: The result is trivial for a string of length 2. *Induction case*: Without loss of generality, assume that the winnable string is of the form $\alpha xx\beta$, where xx is a contractible pair that we arbitrarily choose as the first move. If there exists a derivation $S \overset{\star}{\to} \alpha S \beta \to \alpha xx\beta$, then $\alpha\beta$ can be generated by $\mathcal{G}_1$. Indeed, we just have to replace $S \to xx$ with $S \to \epsilon$ as the last step of the derivation. Therefore, by induction, any sequence of legal moves on $\alpha\beta$ is winning. If there exists no derivation $S \overset{\star}{\to} \alpha S \beta$, then the first x following α in $\alpha xx\beta$ must be generated as the right x of a production $S \to xSx$. Similarly, the x in front of β in $\alpha xx\beta$ must be generated as the left x of a production $S \to xSx$. Hence the string $\alpha xx\beta$ must be of the form $\alpha_1 x \alpha_2 x x \beta_1 x \beta_2$, with $\alpha_1 x \alpha_2 x = \alpha x$ and $x \beta_1 x \beta_2 = x\beta$. We therefore have $S \overset{\star}{\to} \alpha_1 SS\beta_2 \overset{\star}{\to} \alpha_1 \mathbf{xSx} S\beta_2 \overset{\star}{\to} \alpha_1 xSx\mathbf{xSx}\beta_2 \overset{\star}{\to} \alpha_1 x\alpha_2 xx\beta_1 x\beta_2$. This shows that $S \overset{\star}{\to} \alpha_1 SS\beta_2$ and $S \overset{\star}{\to} \alpha_2$ and $S \overset{\star}{\to} \beta_1$ with $\alpha = \alpha_1 x \alpha_2$ and $\beta = \beta_1 x \beta_2$.

Recall that we want to prove that $\alpha\beta \in \mathcal{G}_1$. Starting from $S \overset{\star}{\to} \alpha_1 SS\beta_2$ and using $S \to \epsilon$, we derive that $S \overset{\star}{\to} \alpha_1 S \beta_2$. Applying the rules $S \to xSx$ and $S \to SS$, and using the fact that $S \overset{\star}{\to} \alpha_2$ and $S \overset{\star}{\to} \beta_1$, we derive that:

$$S \overset{\star}{\to} \alpha_1 S\beta_2 \to \alpha_1 \mathbf{xSx}\beta_2 \overset{\star}{\to} \alpha_1 x\mathbf{SS}x\beta_2 \overset{\star}{\to} \alpha_1 x\alpha_2\beta_1 x\beta_2$$

We have just shown that $\alpha_1 x \alpha_2 \beta_1 x \beta_2 = \alpha\beta$, therefore $\alpha\beta \in \mathcal{G}_1$.

Solvability in the 1D case drops to around 25% for larger boards with $C = 2$, and around 0% for larger boards with $C = 3$ and $C = 4$. Table 3 shows the average solvability of the special 1×48 case, which has the same number of cells as the standard 6×8 board, with 95% confidence intervals. Players can expect to win around 23% of games on this board with 2 colours, but less than 1% of games with 3 or 4 colours.

Summary: The 1D version is trivial – if a challenge is solvable, then any move is as good as any other – hence is of little interest to players.

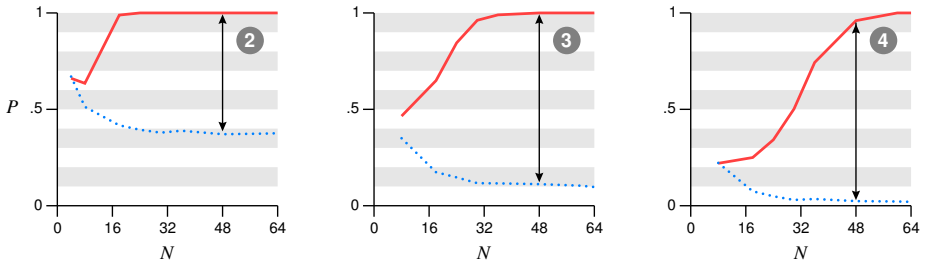

Fig. 8. Observed solvability probabilities $P(S_r)$ (dotted) and $P(S_p)$ (solid) for 3D cases, for $C = 2, 3$ and 4.

Table 4. Solvability probabilities of the $4 \times 4 \times 4$ (3D) case.

		C		
		2	3	4
$4 \times 4 \times 4$	$P(S_r)$	0.376±0.009	0.097±0.005	0.019±0.003
	$P(S_p)$	0.999±0.000	0.999±0.000	0.999±0.000

3.3 3D Case

For completeness, we also consider the 3D version of the game in which piece pairs define 3D boxes that must otherwise be empty. However, such 3D boards would be difficult to make as physical sets so are mostly of academic interest only. Figure 8 shows the observed solvability probabilities $P(S_r)$ and $P(S_p)$ for various 3D boards up to size $N = 64$ for $C = 2, 3$ and 4 colours. Again, the arrows indicate the disparity between $P(S_r)$ and $P(S_p)$ for the $N = 48$ case.

The 3D solvability curves are strikingly similar to those of 2D case, although the solvability rates for both random and perfect play tend to be slightly higher in general. The exact values shown in Table 4 for the target case $N = 48$ ($4 \times 4 \times 4$) indicate that $C = 4$ colours is probably optimal for this board, giving an almost 100% solvability rate with a low susceptibility to random solution of around 2%.

Summary: The 3D case is a viable version of the game.

4 Tension

In this section, we use random sampling to evaluate the potential of BoxOff challenges to interest human players, based on estimated *tension*, i.e. the degree to which the players' decisions affect the outcome of the game. If the player can win by making random choices then the game is not tense, but if every decision is critical to success then the game is very tense. Kramer [8] and Rose [10] observe that well designed games tend to display points of high and low tension.

Tension T is measured as the average ratio of losing moves to total moves at each turn (reduced to 0 if all moves are losing). For example, the first move of the game shown in Figure 3 is relatively tense, as 2 out of the 3 possible moves

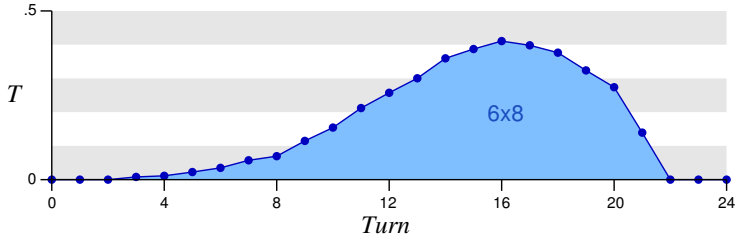

Fig. 9. Average tension probability T for the 6×8 case over 1,000 games.

will lose. However, the remainder of this game lacks tension as every subsequent move leads to a win (or a loss, if the losing path is chosen).

Figure 9 shows relative tension per turn for the 6×8 case, averaged over 1,000 randomly sampled solvable games. Games typically start in a state of low tension that builds to a peak of almost 50% in the mid game, followed by a quick dénouement in the end game. This tension curve is actually a good shape for this game. We want low tension (i.e. fewer losing moves) in the early game, as the repercussions of losing moves may not become obvious until say 20 turns later, which would be frustrating for the player and make such challenges intractable. We want higher tension in the middle-to-end game, where there are fewer move choices and the player can plan ahead with greater certainty, as found.

5 Conclusion

Random sampling yielded useful insights into the BoxOff puzzle game, where analyses through complete game tree expansion would have been impractical. The inventor's default design parameter choices (three colours on a 6×8 board) appear to be optimal, as almost every challenge randomly constructed by the player will be solvable, while few will be susceptible to random play. The 1D version of the the game is trivially solvable and hence of little interest to players, while the 3D version of the game appears to be viable. Our analysis also revealed that BoxOff challenges on the standard board tend to start with a low degree of tension, and build to a climax in the mid-to-late game, allowing them to be tractable but still demanding for players. Monte Carlo analysis proved useful in this case. The general nature of our method, which requires no domain-specific strategic or tactical knowledge, makes it potentially applicable to any domain with discrete actions and computable outcomes.

Acknowledgements

This work was supported by a QUT Vice-Chancellor's Research Fellowship, as part of the project *Games Without Frontiers*.

References

1. Victor Allis. *Searching for Solutions in Games and Artificial Intelligence.* Ph.d. dissertation, University of Limburg, Maastricht, Netherlands, 1994.
2. Ingo Althöfer. Computer-Aided Game Inventing. Technical report, Friedrich-Schiller University, Faculty of Mathematics and Computer Science, Jena, 2003.
3. Cameron Browne. Bitboard Methods for Games. *International Computer Games Association (ICGA) Journal*, 37(2):67–84, 2014.
4. Cameron Browne. The Nature of Puzzles. *Game & Puzzle Design*, 1(1):23–34, 2015.
5. Cameron Browne, Edward Powley, Daniel Whitehouse, Simon Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A Survey of Monte Carlo Tree Search Methods. *IEEE Trans. Comp. Intell. AI Games*, 4:1:1–43, 2012.
6. Hilmar Finnsson and Yngvi Björnsson. Simulation-Based Approach to General Game Playing. In *Proc. Assoc. Adv. Artif. Intell.*, pages 259–264, Chicago, Illinois, 2008.
7. Richard M Karp. *Reducibility among Combinatorial Problems.* Springer, Berlin, 1972.
8. Wolfgang Kramer. What Makes a Game Good? *The Games Journal*, 2000.
9. Steven Meyers. BoxOff: A New Solitaire Board Game. *GAMES*, 37(6):12–13, 2013.
10. John Rose. Addressing Conflict: Tension and Release in Games. Gamasutra. http://www.gamasutra.com, 2015.
11. Stuart J. Russell and Peter Norvig. *Artificial Intelligence - A Modern Approach (3. internat. ed.).* Pearson Education, 2010.
12. Greg Schmidt. The Axiom Universal Game System Project. Mindsports. http://www.mindsports.nl/index.php/axiom, 2012.
13. Julian Togelius, Georgios N. Yannakakis, Kenneth O. Stanley, and Cameron Browne. Search-based Procedural Content Generation: A Taxonomy and Survey. *IEEE Trans. Comp. Intell. AI Games*, 3:172–186, 2011.