

Toward Solving “EinStein würfelt nicht!”

François Bonnet^{1(✉)} and Simon Viennot^{2(✉)}

¹ Graduate School of Engineering, Osaka University, Suita, Japan
francois@cy2sec.comm.eng.osaka-u.ac.jp

² Graduate School of Advanced Science and Technology, JAIST, Nomi, Japan
sviennot@jaist.ac.jp

Abstract. “EinStein würfelt nicht!” is a simple board game, played usually on a 5×5 board with 6 stones per player and a die. Many computer programs have been developed for this game, but in this research, we compute for the first time an exact solution to some instances of the game, with fewer stones on smaller (or larger) boards. When the rules allow the players to choose their initial configuration, a solution consists in computing the exact optimal winning chances of the players for any initial configuration, and then computing the resulting Nash Equilibrium between the two players. Our most difficult result is the solution for a 4×4 board with 6 stones per player.

1 Introduction

EinStein würfelt nicht! (EWN) is a simple two-player board game designed by Althöfer in 2004 [1]. The simplicity of rules, the short length of a game (a few minutes), the randomness aspect from the die, and the not-so-trivial strategies make this game interesting to play for everyone.

This introduction largely overlaps with that of [2], in these same proceedings. It is organized like this in order to keep both papers self-contained.

Rules. The game is played on a 5×5 board with 6 stones per player, numbered 1 to 6. Initially the first (or, alternatively, second) player places his¹ stones in the top-left corner of the board and the second player in the bottom-right corner. Depending on the rules, the placements are random or chosen by the players. One possible initial configuration is depicted in Fig. 1, with red disks for the first player and blue squares for the second one.

Players play alternately: each turn consists of throwing a 6-sided die and then moving a stone. If the stone corresponding to the die value still exists on board, the player must move it; otherwise he can choose to move either his stone with the highest number below the die value, or his stone with the lowest number above the die value.² The first player can move his stones only to the right,

¹ For brevity, we use “he” and “his” whenever “he or she” and “his or her” are meant.

² For example, let us assume that a player still has stones 1, 5, and 6 on the board. Given a die value of 1, 5, or 6, the player must move the corresponding stone. With a die value of 2, 3, or 4, he can choose to move either stone 1 or stone 5.

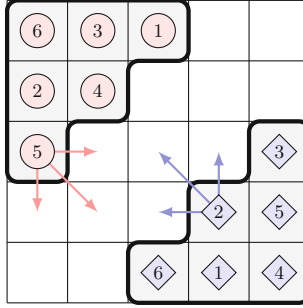


Fig. 1. Possible initial configuration (Color figure online)

down, or down-right directions; while the second player can move only to the left, top, or top-left directions (see examples in Fig. 1).

The players can capture both their own stones and the opponent's stones by moving another stone on the same cell (for example, in Fig. 1, the first player can capture stone 4 with his stone 2). The game ends (1) when a player moves one of his stones to the opposite corner (as seen from his starting corner); then he wins or (2) when a player does not have any remaining stones; then he loses.

Remark. We summarized here the rules for the usual 5×5 board and 6 stones per player. In the paper, we also consider generalizations to other board sizes, and fewer stones per player, namely 3, 4, or 5 stones. When playing with d stones per player, we consider a d -sided die.

Contributions. We are interested in solving the game. As always, there are multiple interpretations of the word *solving*. For a game such as EWN, randomness implies that there is (generally) no strategy that ensures a win. A player can only maximize his winning chances. In this context, we decided to compute what is the optimal winning chance of each player for any possible initial board. Put differently, assuming both players play optimally, what is the chance of winning the game for each of them?

Then, based on the rule to select the initial position, we compute the winning chance for a given instance of the game (board size and number of stones per player):

- Random position: If the players randomly select their initial positions, it simply consists in averaging the winning chances over all initial boards.
- Chosen position: If the players choose their initial positions, we need to compute a Nash Equilibrium and find the optimal probability distribution of initial placements. Surprisingly, the best placements are not always identical for the two players.

Outline. Section 2 presents related work on EWN and other games. Section 3 describes how to compute the winning chance of the first player for a given initial board. In Sect. 4, we summarize the computed values and comment on some unexpected results. Finally, Sect. 5 concludes the paper.

2 Related Work

Just after the publication of the game, a first program based on Monte-Carlo sampling was described in 2005 by Schäfer [3]. Since then, a tournament is organized in the Computer Olympiad every year during the ICGA conference. In 2012, Lorentz reported that most programs were using a min-max approach, and that Monte-Carlo Tree Search could reach a close performance or better [4]. In the 2012 tournament, Turner used a retrograde analysis of the game. He had built an endgame database up to seven pieces [5]. Three years later (in 2015) Karl’s Race, a game closely related to EWN, was solved by Hartisch with a retrograde analysis [6].

In this paper, we are interested in obtaining optimal exact solutions of the game. One of the first complete analytical solutions to a game is probably the solution to the game of Nim, by Bouton, in 1901 [7]. Since then, many games have been solved, either analytically, or with the use of computers, some milestones being the solution to Connect-Four by Allis in 1988 [8], and the solution to Checkers by Schaeffer et al. in 2007 [9]. In 2002, Van den Herik et al. gave an overview of the games already solved, and speculated on games which might be solved in the future, with a classification of games based on the state-space and game-tree complexities [10].

EWN is interesting in that there is (1) a chance factor coming from the die, and (2) some hidden information element when the players choose the initial configuration, so that the concept of Nash Equilibrium [11] is required to define the optimal strategy. Our interest in EWN comes from our work on computing the Nash Equilibrium for the game of Mastermind in 2016 [12].

We have also obtained exact analytical results on the game of EWN when the players start with only one stone on an arbitrary board size. These analytical results are presented in a separate paper [2].

3 Computing Winning Chances

In this section, we describe the computation of the maximum winning chances of the players for a given initial board.

General Idea. Given an initial board, it is possible to compute the optimal winning chance of each player using the expectiminimax algorithm [13], which is an extended version of minimax that includes *chance nodes*. In the game tree, between each layer of min and max nodes, there is an additional layer of chance nodes. Instead of applying a min or max function, the value of a chance

node is computed as the weighted average of the children’s values according to the probability distribution of the random event (here we see the influence of the die).

Pruning Optimization. In practice, minimax is never used and people rely on the $\alpha\beta$ -algorithm to compute the value of (deterministic) games. Similarly, it is possible to prune branches while executing expectiminimax. The corresponding algorithm is called *-Minimax [14, 15]. It includes both α -cuts and β -cuts, but we (re)discovered them partially and implemented this pruning before finding the relevant literature, so that in effect, our current program includes only β -cuts.

Move Ordering. As for the $\alpha\beta$ -algorithm, the efficiency of the branch pruning is directly affected by the order in which the moves are explored. Moves with a higher winning probability should be explored first, so that their values can be used to prune inferior moves as soon as we can prove that they are inferior. Our current heuristic consists mainly in trying first the moves that capture a stone, then the diagonal moves, and then the moves that make the stone closer to the board diagonal.

Transposition Table. We use a transposition table to store already known results, and we have observed that the size of the transposition table is a determinant factor of the computation time. In order to achieve the highest possible number of entries in the table, we optimized both the implementation of the transposition table, by using no pointer at all, and also the encoding of the elements stored in the table. Our final implementation uses only 24 bytes per entry in the table, 8 bytes for the board configuration, and 16 bytes for the value encoding the optimal winning chance of that configuration.

For some of our results, the transposition table cannot store all the intermediary results of the computation, so that a replacement scheme is needed. Roughly, we permute elements in the transposition table when they are added or accessed, so that when needed, the oldest elements are replaced with newer ones.

Symmetry Optimizations. To improve the efficiency of the transposition table, it is important to take into account all possible board symmetries. We found three types of symmetry that can be used to canonize configurations. Note that the last one is specific to this game.

1. 180° Symmetry: The two players have the same set of moves, hence exchanging the stones of the players and rotating the board by 180° do not change the winning chances (see example of Fig. 2).
2. Diagonal Symmetry: For any board, applying a TopLeft-BottomRight diagonal symmetry does not change the winning chance (see example of Fig. 3).



Fig. 2. Equivalent winning chances; for all $(x, y) \in \{1, 2\} \times \{1, 2\}$ and $\bar{x} = 3 - x$. Left: original — winning chance of Player x when Player y is next to play; right: with 180° Sym — winning chance of Player \bar{x} when Player \bar{y} is next to play.



Fig. 3. Equivalent winning chances for Player x when Player y is next to play; for all $(x, y) \in \{1, 2\} \times \{1, 2\}$. Left: original; right: with diagonal Sym

3. Stone Symmetry: Let us assume d stones per player. For either player, applying the function $x \mapsto d + 1 - x$ to all stones, does not change the winning chances. There are two independent Stone Symmetries; SS1 for the first player and SS2 for the second player (see examples of Fig. 4).

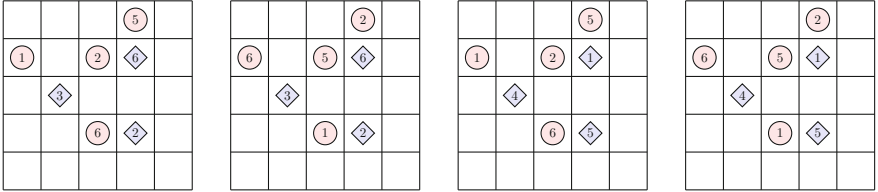
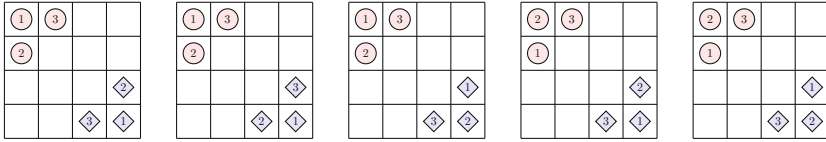


Fig. 4. Equivalent winning chances for Player x when Player y is next to play; for all $(x, y) \in \{1, 2\} \times \{1, 2\}$. From left to right: original, with SS1, with SS2 and with SS1&2.

Taking into account the diagonal-based and both stone-based symmetries, the number of configurations is roughly divided by $2^3 = 8$. Considering only the initial configurations, we obtain the numbers of Table 1. For example, Fig. 5 gives the five initial configurations with three stones per player. Note that the number of initial configurations depends only on the number of stones, not on the size of the board.

**Fig. 5.** All five initial configurations with three stones per player**Table 1.** Initial configurations with/without symmetries for d stones per player

d	Total number	Reduced number
3	$3!^2 = 36$	5
4	$4!^2 = 576$	72
5	$5!^2 = 14\,400$	1 808
6	$6!^2 = 518\,400$	64 800

Table 2. Maximal length L of a game and required bits storage for numerators with d stones on $n \times n$ boards

n	d	L	$\log_2(d^L)$
4	5	39	≈ 90.56
4	6	45	≈ 116.32
5	4	49	98
5	5	59	≈ 136.99
5	6	69	≈ 178.36
8	3	75	≈ 118.87
9	3	87	≈ 137.89

Memory Consumption Optimizations. We describe here how we could achieve an encoding on only 24 bytes of the board configuration and its associated value (the winning chance of the next player to play).

For simplicity, let us consider only square boards, of size n . Let d (for die) be the number of stones per player. Each stone is either located on a board cell or already captured, so the number of configurations is bounded by $(n^2 + 1)^{2d}$. It implies that we can encode a configuration by using only $2d \log_2(n^2 + 1)$ bits. For the usual instance of the game $(n, d) = (5, 6)$, it gives approximately $56.4 < 64 = 8$ bytes (the typical size of a `long int`).

We are interested in computing exact values, thus we use rational numbers instead of floating-point numbers. In order to minimize the memory footprint, we implemented our own (small) fraction library. Only some operations are needed to compute the winning chances: addition, complement to one, and division by d . The last statement comes from the fact that all chance nodes correspond to throwing an unbiased d -sided die. It means that all fractions encountered during a computation will be of the form $\frac{x}{d^e}$; it is then sufficient to store the pair (e, x) .

Since stones have limited movements, it is possible to give a bound L to the Length of a game. For the usual game $(n, d) = (5, 6)$, the number of moves is bounded by $2(7 + 2 \cdot 6 + 3 \cdot 5) + 1 = 69$. Since $69 < 2^8$, we can use only 1 byte to store the denominator of any fraction. Since all stored values are probabilities, the numerator of the fractions is at most 6^{69} , which requires $\log_2(6^{69}) \approx 178.4$ bits, i.e., 3 bytes. However, for all instances solved up to now, 2 bytes are sufficient (see Table 2).

Based on these observations, it is natural to use bitfields to store fractions. We use either 8 or 16 bytes (one or two `long int`) for each rational number as depicted in Fig. 6.

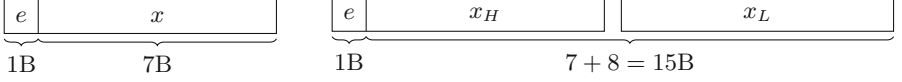


Fig. 6. Our ad-hoc encoding for rational numbers

Parallelization. Since we need to compute the winning chance for all possible initial boards, it is natural to execute multiple computations simultaneously. First, we have parallelized with the use of multiple threads. Each thread computes a different initial board, but all threads share the same transposition table. We have implemented locks in the transposition table to allow multi-threaded access. Different initial board configurations lead to many identical positions so that this straight-forward parallelization works efficiently.

Moreover, for our most difficult result (4×4 board with 6 stones per player), we have distributed the computation on multiple nodes of a super-computer with the standard Message Passing Interface (MPI) [16]. A list of initial board configurations is attributed with MPI to each node of the super-computer, and then each node uses multiple threads (typically 12) and one transposition table (typically 128 GB of memory) to compute the winning chance of the attributed boards. Finally, all results are collected with MPI to a master node.

Our current parallelization proved quite efficient for computing the exact value of the 4×4 board with 6 stones per player, mainly because the computation of one initial board is reasonably fast (less than an hour with one thread) in this case. However, it is not efficient (and even useless) when the computation of a single board is very long, which is the case for the usual game on a 5×5 board with 6 stones per player. A more sophisticated parallelization “inside” the game tree itself will be required in the future.

4 Results

We present here the results obtained in our computations. Section 4.1 summarizes the results for three stones per player with details for the 3×3 board. Section 4.2 gives the results obtained for four stones. Finally, Sect. 4.3 gives some preliminary results for six stones, but only on the 4×4 board.

4.1 Three Stones per Player

Three Stones on 3×3 Board. The winning chances of Player 1 assuming optimal play of both players are presented in Fig. 7. This matrix indicates the

winning chance of the first player for all possible initial boards. The winning chances vary from $\frac{143}{243} \approx 0.59$ to $\frac{170}{243} \approx 0.70$ with an average value of $\frac{1459}{2187} \approx 0.67$, which corresponds to the winning chance assuming a random initial board.













						
	$\frac{170}{243} \approx 0.70$	$\frac{56}{81} \approx 0.69$	$\frac{170}{243} \approx 0.70$	$\frac{56}{81} \approx 0.69$	$\frac{170}{243} \approx 0.70$	$\frac{170}{243} \approx 0.70$
	$\frac{56}{81} \approx 0.69$	$\frac{170}{243} \approx 0.70$	$\frac{170}{243} \approx 0.70$	$\frac{170}{243} \approx 0.70$	$\frac{170}{243} \approx 0.70$	$\frac{56}{81} \approx 0.69$
	$\frac{50}{81} \approx 0.62$	$\frac{50}{81} \approx 0.62$	$\frac{143}{243} \approx 0.59$	$\frac{50}{81} \approx 0.62$	$\frac{143}{243} \approx 0.59$	$\frac{50}{81} \approx 0.62$
	$\frac{56}{81} \approx 0.69$	$\frac{170}{243} \approx 0.70$	$\frac{170}{243} \approx 0.70$	$\frac{170}{243} \approx 0.70$	$\frac{170}{243} \approx 0.70$	$\frac{56}{81} \approx 0.69$
	$\frac{50}{81} \approx 0.62$	$\frac{50}{81} \approx 0.62$	$\frac{143}{243} \approx 0.59$	$\frac{50}{81} \approx 0.62$	$\frac{143}{243} \approx 0.59$	$\frac{50}{81} \approx 0.62$
	$\frac{170}{243} \approx 0.70$	$\frac{56}{81} \approx 0.69$	$\frac{170}{243} \approx 0.70$	$\frac{56}{81} \approx 0.69$	$\frac{170}{243} \approx 0.70$	$\frac{170}{243} \approx 0.70$

Fig. 7. Winning chance of player 1 for all possible initial configurations

Assuming that the players can choose their initial positions, we need to compute the Nash Equilibrium (NE) of this matrix. One can observe that four initial positions are dominated, namely the positions with stone 2 in the corner (lines 3 and 5, columns 3 and 5). As already mentioned in Sect. 3, a number of configurations are equivalent with respect to winning chances due to diagonal-based or stones-based symmetries. Thus it is sufficient to consider a reduced matrix to compute the NE (see Fig. 8).

The NE consists of playing each option with probability 0.5 for both players. At the Equilibrium, the winning chance of Player 1 is $\frac{169}{243} \approx 0.695473$. This is significantly higher than the average value. The optimal strategies of both players are depicted in Fig. 9. In order to represent succinctly mixed strategies, we use the following notation $p_1 C_1 + p_2 C_2 + \dots$ to denote the strategy in which configuration C_1 is played with probability p_1 , configuration C_2 with probability p_2 , and so forth.

	$\frac{170}{243} \approx 0.70$	$\frac{56}{81} \approx 0.69$
	$\frac{56}{81} \approx 0.69$	$\frac{170}{243} \approx 0.70$

Fig. 8. Reduced matrix

In the specific case of Fig. 9, there are infinitely many optimal strategies. The players can choose any value between 0 and $\frac{1}{2}$ for the parameters. In this instance, the strategies are identical for both players.

$$\begin{aligned}
 \alpha & \begin{array}{|c|c|c|} \hline \textcircled{1} & \textcircled{3} & \\ \hline \textcircled{2} & & \\ \hline & & \\ \hline \end{array} + \left(\frac{1}{2} - \alpha\right) \begin{array}{|c|c|c|} \hline \textcircled{3} & \textcircled{1} & \\ \hline \textcircled{2} & & \\ \hline & & \\ \hline \end{array} + \beta \begin{array}{|c|c|c|} \hline \textcircled{1} & \textcircled{2} & \\ \hline \textcircled{3} & & \\ \hline & & \\ \hline \end{array} + \left(\frac{1}{2} - \beta\right) \begin{array}{|c|c|c|} \hline \textcircled{3} & \textcircled{2} & \\ \hline \textcircled{1} & & \\ \hline & & \\ \hline \end{array} \\
 \gamma & \begin{array}{|c|c|c|} \hline & & \\ \hline & & \diamond 2 \\ \hline & \diamond 3 & \diamond 1 \\ \hline \end{array} + \left(\frac{1}{2} - \gamma\right) \begin{array}{|c|c|c|} \hline & & \\ \hline & & \diamond 2 \\ \hline & \diamond 1 & \diamond 3 \\ \hline \end{array} + \delta \begin{array}{|c|c|c|} \hline & & \\ \hline & & \diamond 3 \\ \hline & \diamond 2 & \diamond 1 \\ \hline \end{array} + \left(\frac{1}{2} - \delta\right) \begin{array}{|c|c|c|} \hline & & \\ \hline & & \diamond 1 \\ \hline & \diamond 2 & \diamond 3 \\ \hline \end{array}
 \end{aligned}$$

Fig. 9. Optimal strategies with 3 stones per player. Top: for the first player (for any $\alpha, \beta \in [0, \frac{1}{2}]$); bottom: for the second player (for any $\gamma, \delta \in [0, \frac{1}{2}]$).

Three Stones on $n \times n$ Board for $n \in \{4, 5, 6, 7, 8, 9\}$. For larger boards up to a size of $n = 9$, it appears that the optimal strategies (i.e., choice of initial position) are the same for both players. We could not prove it formally, but we conjecture that it is true for any larger board size.

Indeed, for three stones, there are only two kinds of initial positions, either with stone 2 in the corner, or not. When a player places his stone 2 in the corner, it is (obviously) impossible to capture this stone in the first move. While, when stone 2 is not in the corner, the player can (if he wants) capture the stone with a probability $\frac{1}{3}$ (when the die equals the stone located in the corner). It is generally more interesting for a player to capture stone 2 (compared to capturing another stone), since it gives him more later options.³

³ After capturing the stone 2, a die value of 2 lets the player decide to move either stone 1 or stone 3. Such choice does not exist if stone 1 or 3 is captured.

4.2 Four Stones Per Player

Due to space limitations, we highlight here only the surprising results. We study the game with four stones on the 4×4 and 5×5 boards. The main observation is that optimal placements for the first and the second players are different on the 4×4 board. In order to maximize their winning chances, the players should play according to the optimal strategies depicted in Fig. 10. At the NE, the first player wins with probability $\frac{148461956373}{274877906944} \approx 54.0\%$.

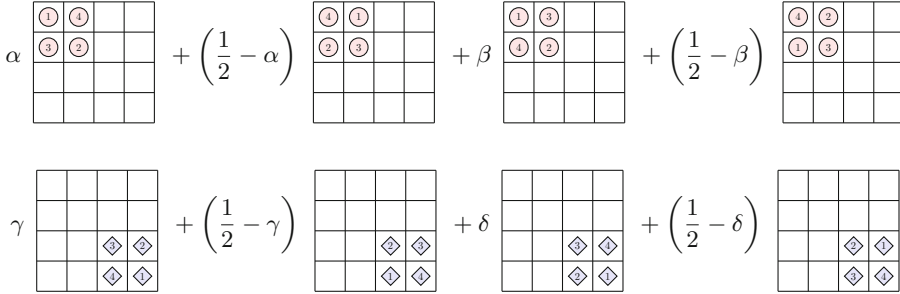


Fig. 10. Optimal strategies with 4 stones per player on the 4×4 board. Top: for first player (for any $\alpha, \beta \in [0, \frac{1}{2}]$); bottom: for second player (for any $\gamma, \delta \in [0, \frac{1}{2}]$).

4.3 Six Stones Per Player

For six stones, we were not able to compute the winning probabilities for the real 5×5 board. Our computations were done only for the 4×4 board. As we already observed for four stones, the two players should place their stones in different initial positions. In order to maximize their winning chances, the players should play according to the optimal strategies depicted in Fig. 11. At the NE, the first player wins with probability $\frac{427783129292781527705}{767617776808101937152} \approx 55.7\%$.

It is interesting to observe that the optimal placements match the best starting positions found by Schäfer [3] using random simulations. Note that his study deals with the 5×5 board and does not distinguish the two players.

The above result on the 4×4 board with six stones (see Fig. 11) was most difficult to obtain. We used 64 nodes of a super-computer for around 24 h, with each node using 128 GB of memory and 12 threads. This computation would have required roughly 2 years of computation with only one thread on a single machine.

4.4 Estimated Time to Solve EWN

The main question about EWN is whether it is possible to solve the real game: 6 stones per player on the 5×5 board. We started some computation with 5 stones

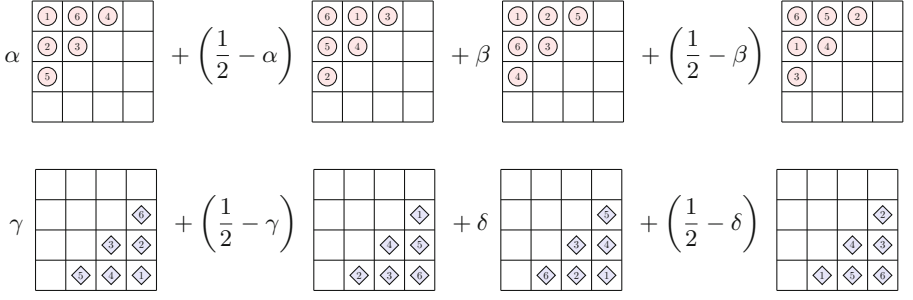


Fig. 11. Optimal strategies with 6 stones per player on the 4×4 board. Top: for first player (for any $\alpha, \beta \in [0, \frac{1}{2}]$); bottom: for second player (for any $\gamma, \delta \in [0, \frac{1}{2}]$).

per player. Our current estimate is that 6 months of computation with a single thread are needed for each initial board, so that the total time needed to solve the game with 5 stones per player would be around 2 years on a super-computer (with 64 nodes, 12 threads per node). It seems feasible in a near future.

A number of similar computation tests is needed to provide an estimate for the real game with 6 stones per player, but here we can note that the number of possible board states with 2 more stones is roughly multiplied by $26^2 = 676$. If the same factor applies on the computation time, it seems reasonable to conjecture that an exact solution to EWN will be computed in the future.

5 Conclusion

Table 3 summarizes all the results computed so far. For each instance of the game, we give the minimal/maximal winning probability, the average winning

Table 3. Summary of all computed results (d stones per player on $n \times n$ board)

n	d	NE value	Average	P1-then-P2	P2-then-P1	Min	Max
3	3	0.695	0.667	0.691	0.700	0.588	0.700
4	3	0.551	0.552	0.550	0.551	0.517	0.592
5	3	0.571	0.570	0.568	0.574	0.514	0.628
6	3	0.557	0.558	0.556	0.558	0.464	0.652
7	3	0.570	0.570	0.569	0.571	0.456	0.682
8	3	0.564	0.573	0.564	0.565	0.473	0.677
9	3	0.590	0.607	0.590	0.590	0.527	0.711
4	4	0.540	0.544	0.538	0.541	0.448	0.646
5	4	0.543	0.546	0.542	0.543	0.447	0.639
4	5	0.5668	0.5664	0.5665	0.5670	0.4984	0.6329
4	6	0.5573	0.5589	0.5560	0.5586	0.4534	0.6625

probability (random initial board), the NE winning probability (initial board freely chosen by each player). The two additional columns (P1-then-P2 and P2-then-P1) are the optimal winning probability. P1-then-P2 means that the first player places his stones first, then the second player places his stones while knowing the choice of the opponent. P2-then-P1 means that the second player places his stones first, then the first player places his stones knowing the choice of the opponent.

As future work, we (obviously) would like to tackle the real instance of the game, i.e., 6 stones per player on the 5×5 board. A great number of computation resources are needed, but there is room for improvement in our current algorithm, especially in the pruning method, the move ordering and the parallelization. When we started this research, a complete and exact solution to the game of EWN on a 5×5 board with 6 stones seemed completely out of reach, but now, we believe that an answer is possible.

Acknowledgments. This work is partially supported by JSPS KAKENHI Grant (C)(JP15K00183) and (JP15K00189) and Japan Science and Technology Agency, CREST (JPMJCR1404) and Infrastructure Development for Promoting International S&T Cooperation and Project for Establishing a Nationwide Practical Education Network for IT Human Resources Development, Education Network for Practical Information Technologies. We would also like to thank the anonymous reviewers for their comments that helped us improve the paper.

References

1. Althöfer, I.: On the origins of “EinStein würfelt nicht!” (2011). <http://www.althofer.de/origins-of-ewn.html>
2. Bonnet, F., Viennot, S.: Analytical solution for “EinStein würfelt nicht!” with one stone. In: Winands, M., et al. (eds.) ACG 2017. LNCS, vol. 10664, pp. 1–12. Springer, Cham (2017)
3. Schäfer, A.: Rock’n’Roll, A Cross-Platform Engine for the Board Game “EinStein würfelt nicht!”. Student Research Project, Friedrich Schiller University Jena (2005)
4. Lorentz, R.J.: An MCTS program to play EinStein Würfelt Nicht!. In: van den Herik, H.J., Plaat, A. (eds.) ACG 2011. LNCS, vol. 7168, pp. 52–59. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31866-5_5
5. Turner, W.: Einstein würfelt nicht – an analysis of endgame play. ICGA J. **35**, 94–102 (2012)
6. Hartisch, M.: Impact of rounding during retrograde analysis for a game with chance nodes: Karl’s race as a test case. ICGA J. **38**, 81–93 (2015)
7. Bouton, C.L.: Nim, a game with a complete mathematical theory. Ann. Math. **3**, 35–39 (1901)
8. Allis, V.: A knowledge-based approach of connect-four. Master’s thesis, Vrije Universiteit (1988)
9. Schaeffer, J., Burch, N., Björnsson, Y., Kishimoto, A., Müller, M., Lake, R., Lu, P., Steve, S.: Checkers is solved. Science **317**, 1518–1522 (2007)
10. van den Herik, H., Uiterwijk, J.W., van Rijswijck, J.: Games solved: now and in the future. Artif. Intell. **134**, 277–311 (2002)
11. Nash, J.F.: Non-cooperative games. Ph.D. thesis, Princeton University (1950)

12. Bonnet, F., Viennot, S.: Nash equilibrium in mastermind. In: Plaat, A., Kusters, W., van den Herik, J. (eds.) CG 2016. LNCS, vol. 10068, pp. 115–128. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-50935-8_11
13. Michie, D.: Game-playing and game-learning automata. In: *Advances in Programming and Non-Numerical Computation*, pp. 183–200. Pergamon Press Ltd. (1966)
14. Ballard, B.W.: The *-minimax search procedure for trees containing chance nodes. *Artif. Intell.* **21**, 327–350 (1983)
15. Hauk, T., Buro, M., Schaeffer, J.: Rediscovering *-MINIMAX search. In: van den Herik, H.J., Björnsson, Y., Netanyahu, N.S. (eds.) CG 2004. LNCS, vol. 3846, pp. 35–50. Springer, Heidelberg (2006). https://doi.org/10.1007/11674399_3
16. MPI: A Message-Passing Interface Standard. <http://mpi-forum.org/>

Advances in Computer Games

15th International Conferences, ACG 2017, Leiden, The

Netherlands, July 3-5, 2017, Revised Selected Papers

Winands, M.; van den Herik, J.; Kusters, W. (Eds.)

2017, XX, 235 p. 117 illus., Softcover

ISBN: 978-3-319-71648-0