JOURNAL OF
EMERGING INVESTIGATORS

Article

Comparative analysis of player ability, game size, and ideal starting positions in Nim games

Rohan Sinha[1], Rhun Henderson[2], William Gasarch[3]
[1] Richard Montgomery High School, Rockville, Maryland
[2] Montgomery Blair High School, Silver Spring, Maryland
[3] Department of Computer Science, University of Maryland, College Park, Maryland

SUMMARY

Nim is an impartial subtraction game consisting of a pile or piles of sticks, where two players alternate taking turns to remove sticks, and the player to remove the last stick wins. Nim has a mathematically proven solution, but the outcomes of games with imperfect, flawed play still remain unclear. We analyzed imperfect play in the game of Nim. More specifically, we questioned whether player ability or the starting positions are more important in influencing winning game outcomes. We also researched the required ability differential needed for a player to win from a losing position. We hypothesized that when the pile size is large, the initial position has minimal impact on win probability and player ability becomes the primary determinant under imperfect play. We tested several computational game simulations and derived a mathematical model for win percentages, both of which support our hypothesis. Our results showed that a favorable starting position only benefits a player until a certain point during imperfect play, then has negligible effect. This research holds significance because, according to the Sprague-Grundy Theorem, any impartial game is equivalent to a single-pile game of Nim. Additionally, artificial intelligence reinforcement learning algorithms, which learn through trial and feedback, face significant challenges with impartial games like Nim due to their highly abstract mathematical nature. Considering these limitations, our research provides insights on impartial game outcomes under imperfect play, which is essential to train robust decision-making agents that are capable of accurately making moves in uncertain, complex game or game-like environments.

INTRODUCTION

The game of Nim is a well-studied example of a combinatorial impartial game, with origins tracing back to early 20th century mathematical analysis (1). There is still active research for the standard game and its variants (1-6). Many variants notably incorporate economic elements like taxation, and systems utilizing bidding for turn rights (3-6). Most players are unlikely to play perfectly, but a common foundational assumption uniting existing research on Nim is perfect play (3-6).

Nim consists of two players taking turns removing sticks from a pile containing n number of sticks (1). Players can remove a certain number of sticks from the pile per turn, and the player who removes the last stick wins. Nim games can consist of multiple piles with varying amounts of sticks per pile. There are many variations of Nim (2). For this paper, we used a common variant of the game: $|A| = 3$ and $1 \in A$ where A is a finite set of natural numbers representing the allowed moves, how many sticks can be removed, in a turn. The starting position is the number of sticks in each pile at the beginning of the game.

To illustrate the nature of the game, consider $A = \{1, 2, 3\}$ as an example. It is apparent that the player with four sticks at the start of their turn is always going to lose. If they remove one, their opponent can remove three and win; if they remove two, their opponent can remove two and win; if they remove three, their opponent can remove one and win. Hence, leaving the opponent with four sticks left is ideal. This applies to any multiple of four as well. Each position (remaining number of sticks in the pile) is said to be winning for a certain player if they play perfectly. Our paper focuses on when both players play imperfectly.

To calculate the winner between two players at any given position for any value of A (allowed moves in a turn), Grundy numbers are utilized when simulating perfect play (7). The Grundy numbers for the game of Nim are defined as:

$$G(n) = \begin{cases} 0 & n = 0 \\ mex\,\{G(n-a) : a \in A, n-a \geq 0\} & n \geq 1 \end{cases} \quad \textbf{(1)}$$

where $G(n)$ is recursively defined for each position n and *mex* returns the smallest whole number that is not in the set (8). It indicates which player the position is winning for through a recursive piecewise function that ultimately returns to either 0, 1, 2 or 3. If $G(n) = 0$, n is a winning position for the player who just made their move. Additionally, if $G(n) = 0$ at the start of the game, the position is winning for Player 2. The opposites apply respectively if $G(n) \neq 0$, the position is winning for Player 1. Each starting position and value for A affects whether the position is winning for Player 1 or 2. For example, if $A = \{1, 2, 3\}$ and $G(1) = 1$ for Player 1, then this starting position is winning for Player 1 since $G(1) \neq 0$. In the same scenario, if $G(4) = 0$, this starting position is winning for Player 2.

Grundy numbers can be expanded to multi-pile Nim (9). Let $A_1, \ldots, A_k$ be k finite sets of naturals representing the valid moves in pile i where $1 \leq i \leq k$. Let $G_i(n_i)$ be the Grundy function for Nim-A_i. Then the Grundy function for the Nim-($A_1, \ldots, A_k$) is

$$G(n_1, \ldots, n_k) = \oplus_{i=1}^{k} G_i(n_i) \quad \textbf{(2)}$$

Where $\oplus$ is the bitwise XOR function. $G_i(n_i)$ is represented in base 2 (9). The Sprague-Grundy Theorem explains that

https://doi.org/10.59720/25-114

Player 1 wins the game starting f rom position $(n_1, n_2, \ldots, n_k)$ if and only if $G(n_1, n_2, \ldots, n_k) \neq 0$ (10).

To illustrate the Sprague-Grundy Theorem, consider a game with two piles. Let $A_1 = \{1, 2, 3\}$ and $n_1 = 17$. Let $A_2 = \{1, 3, 4\}$ and $n_2 = 5$. The Grundy numbers for each pile are:

$$G_1(n_1) = mex\,\{G(17 - a) : a \in A_1, 17 - a \geq 0\} \quad (3)$$

$$G_2(n_2) = mex\,\{G(5 - a) : a \in A_2, 5 - a \geq 0\} \quad (4)$$

Hence, $G_1(n_1) = 1$ and $G_2(n_2) = 3$, then

$$G(n_1, n_2) = G_1(n_1) \oplus G_2(n_2) = 1 \oplus 3 = 2 \neq 0 \quad (5)$$

Hence Player 1 wins in this configuration. We use a simple algorithm that plays the game perfectly given a winning position for both single and multi-pile Nim. When both players play optimally, the player in the initial winning position at the start will always win (e.g., with $A = \{1, 2, 3\}$, Player 1 will always win if n is not a multiple of 4) (**Table 1**). Our research focused on the game's outcomes when players make imperfect moves.

We hypothesized that when the pile size is large, the initial position has minimal impact on win probability and player ability becomes the primary determinant under imperfect play. Our methodology consisted of computationally simulating Nim games with varying levels of imperfect play, number of sticks, and starting positions to draw empirical conclusions from the results. Specifically, we analyzed the percentage of games each player wins at losing or winning starting positions with different probabilities of making the optimal move. The optimal moves were calculated using Grundy numbers (Equations 1 and 2). Additionally, we studied how much better a player must be for them to win starting from a losing position. We also found that when the number of sticks increases the impact of the initial position on win probability decreases and player ability becomes the primary determinant under imperfect play.

RESULTS

We simulated 10^6 games in Java with player ability represented as the probability of a player making the optimal move (P_1 for Player 1 and P_2 for Player 2), and the set of allowed moves (A). This program then computed the percentage of games each player won (W_1 for Player 1 and W_2 for Player 2) at each starting position n, where $n \in \{1, \ldots, 300\}$. We completed a Monte Carlo Simulation as it involves many simulations with varying game values (11).

Conjecture 1.1: Player Ability vs Win Percentage in One-Pile Nim

Using moving averages, we computed the stabilization points (n_s) at the value (n) where the win percentages largely stopped changing. The data confirmed that in games where Player 1 played at a higher ability than Player 2, W_1 always stabilized to a greater value than W_2 and vice versa, regardless of whether n was a favorable position for them or not (given that $n > n_s$) (**Figure 1**). This is represented as:

If $P_1, P_2 \rightarrow$ (0%, 100%) and $P_1 > P_2$, then as $n \rightarrow \infty$, $W_1 > W_2$.

Conversely, if $P_2 > P_1$, then as n $\rightarrow \infty$, $W_2 > W_1$.

Our empirical evidence from simulations of all game configurations strongly supported this conjecture as the player with the higher ability won a greater number of games as n increased, independent of winning and losing positions. These stabilization values remained consistent within each game type. For example, in one-pile Nim, regardless of the value of A, if P_1 was greater than P_2 (P_1 = 90% and P_2 = 80%), and $n \geq n_s$, then W_1 was higher than W_2 ($W_1 \approx$ 70% and $W_2 \approx$ 30%) (**Figures 1A and 1B**).

Conjecture 1.2: Equal Player Ability vs Win Percentage in One-Pile Nim

If Player 1 and Player 2 played at equivalent ability, then W_1 and W_2 both stabilized at 50%. This means that as $n \rightarrow \infty$ both players would win approximately an equal number of games regardless of whichever values of n were favorable for each player (**Figures 1C and 1D**). This is represented as:

If $P_1, P_2 \rightarrow$ (0%, 100%) and $P_1 = P_2$,
then as $n \rightarrow \infty$, $W_2 = W_1$.

Even though the values of A do not influence the fact that stabilization occurs, they do affect the value of n_s. For example, if $P_1 = 90\%$ and $P_2 = 80\%$ in (1, 2, 3)-Nim, then n_s = 58. However, in (1, 4, 5)-Nim, $n_s = 123$ (**Figures 1A and 1B**).

	One-Pile Nim Games																
Number of sticks	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Winning player	2	1	1	1	2	1	1	1	2	1	1	1	2	1	1	1	2
Grundy number	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0

Table 1: Analysis of one-pile Nim games with 0 − 16 sticks. A = {1, 2, 3}. Player 2 is winning if the number of sticks is a multiple of 4: if Player 1 removes 1, 2, or 3, Player 2 can remove 3, 2, or 1, respectively, bringing the number of sticks to 0 or another multiple of 4. The Grundy number is the minimum whole number not in the set of all reachable positions' Grundy numbers (**Equation 1**). The Grundy number of 0 sticks is 0 (**Equation 1**). Although both can be derived independently, there is a connection between the winning player and the Grundy number: if the Grundy number is 0, Player 2 is winning (10). Hence, Player 2 is winning when there are 0 sticks.

https://doi.org/10.59720/25-114

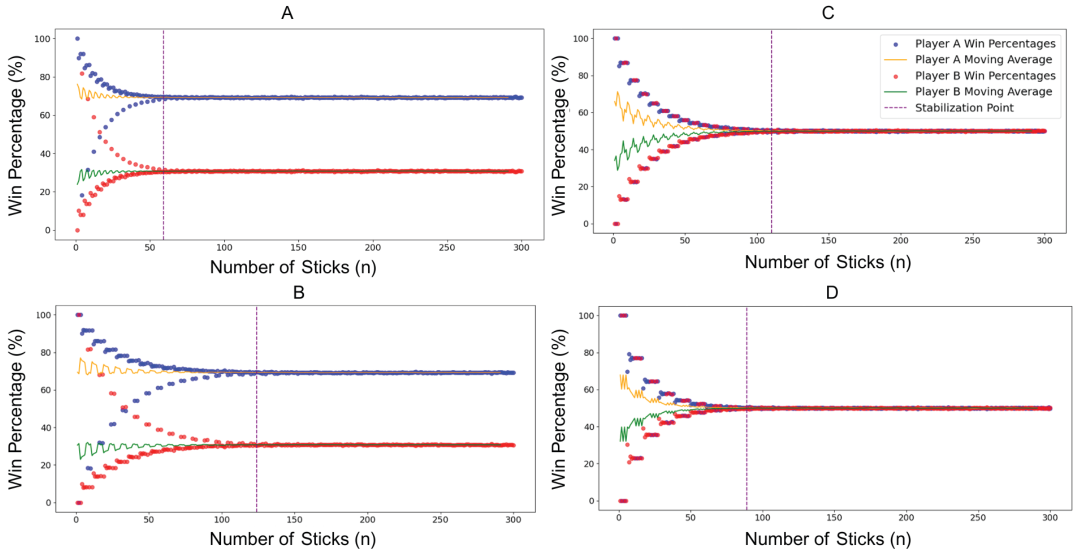

Figure 1: Win percentages of Player 1 (W_1) and Player 2 (W_2) vs. starting number of sticks (n). Scatter plot, moving averages, and stabilization points for win percentages in different game configurations: **A)** A = {1, 2, 3}, P_1 = 90%, and P_2 = 80% (n_s = 58); **B)** A = {1, 5, 6}, P_1 = 70%, and P_2 = 70% (n_s = 88); **C)** A = {1, 3, 4}, P_1 = 85%, and P_2 = 85% (n_s = 109); **D)** A = {1, 4, 5}, P_1 = 90%, and P_2 = 80% (n_s = 123). Each red or blue data point represents the percentage of 10^6 Monte Carlo simulations won by a player at each n. Games were simulated using a Java algorithm that determined the optimal move, if any, using Grundy numbers, and used them with a probability of P_1 or P_2 accordingly. The yellow and green moving averages (window = 10) smooth short-term fluctuations in the data. An algorithm was used to compute the stabilization point (n_s) as the 5^{th} value of n in the first moving average window whose range of derivatives < 0.06 (**Equation 7)**.

Conjecture 1.3: Player Ability vs Stabilization Point in One-Pile Nim

Incidentally, we discovered that there was a strong direct relationship between the ability of both players (P_1 and P_2) and n_s (**Figure 2**). The value of n_s if minimized, but not necessarily to 0 when $P_1 + P_2 \approx 100\%$. Furthermore, combinations where both P_1 and P_2 are either high or low result in higher values of n_s, i.e., n_s is maximized when $P_1, P_2 \rightarrow 0\%$ or $P_1, P_2 \rightarrow 100\%$.

Minimum Player Ability vs Losing Starting Position

We investigated the ability difference required for Player 1 to have a higher win rate ($W_1 > W_2$) in losing starting positions for Player 1 where G(n) = 0. To achieve this, we developed a program that simulated 10^6 games for each combination of both P_1 and P_2 in 1% increments and determined the minimum difference $\min(P_1 - P_2)$ for which $W_1 > W_2$. We replicated this for several different starting positions that were losing for Player 1. The results revealed that as n increased there was a decrease in the minimum ability difference to win from a losing position and the ideal starting position's influence over the game diminished (**Figure 3**). It additionally revealed a roughly parabolic relationship between P_1 and $\min(P_1 - P_2)$, where low or high values of P_1 required greater differences in ability to achieve $W_1 > W_2$ (**Figure 3**).

Two-Pile Nim

We simulated 10^5 games given the ability of players, P_1 and P_2, and the sets of allowed moves, A_1 and A_2. This program then computed the percentage of games each player won at each combination of starting positions n_1, n_2, where $n_1, n_2 \in \{1, . . ., 100\}$. As with one-pile Nim, two-pile Nim stabilized as n_1, n_2 increased (**Figure 4**). The stabilization points identified in the graphs represent the coordinates where W_1 and W_2 stop changing significantly. This implies that conjecture 1.1 can be extended to games with multiple piles.

DISCUSSION

We hypothesized that when the pile size is large, the initial position has minimal impact on win probability and player ability becomes the primary determinant under imperfect play. We researched the comparative importance of player ability versus ideal starting positions and the number of sticks in single and multi-pile Nim games. Our computations showed that as the number of sticks increases, the significance of having an ideal starting position diminished (given that both players play imperfectly). Notably, beyond a certain number of starting sticks, W_1 and W_2 stabilized to specific values. In particular, when $P_1 = P_2$, both W_1 and W_2 stabilized to 50%. Moreover, our analysis revealed that the player with a higher ability will always stabilize to a higher win rate, despite that player starting in losing positions. This was anticipated because as n increases, the number of moves made increases. As the number of moves increases, the probability of a player making a mistake increases. Hence, before the stabilization point, ideal positions had a greater influence on win percentages due to fewer opportunities for mistakes. This highlighted the importance of player ability over positional advantage in certain scenarios, especially after n_s. Additionally, our results showed that the greater the values of A, the greater the value of n_s. This is because as a greater number of sticks were removed each turn, the n_s increased to accumulate enough optimal and sub-optimal moves for a

https://doi.org/10.59720/25-114

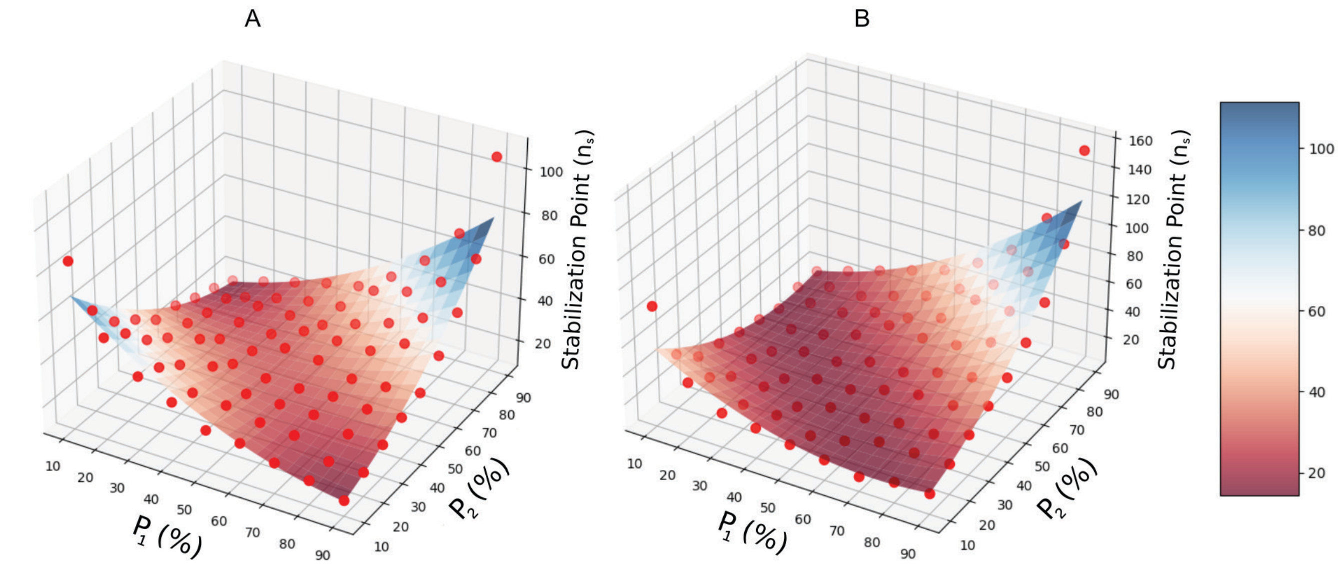

Figure 2: Player 1 ability (P_1) and Player 2 ability (P_2) vs. number of sticks needed for stabilization (n_s). Scatterplots and 2nd order 3-dimensional best-fit surfaces for n_s for two game configurations: **A)** A = {1, 2, 3}; **B)** A = {1, 3, 4}. Each data point represents the stabilization point (n_s) for every combination of (P_1, P_2) ϵ {1%, …, 100%} for 10% increments. A Python algorithm computes the stabilization point (n_s) as the 5th value of n in the first moving average window whose range of derivatives < 0.06 (**Equation 7**).

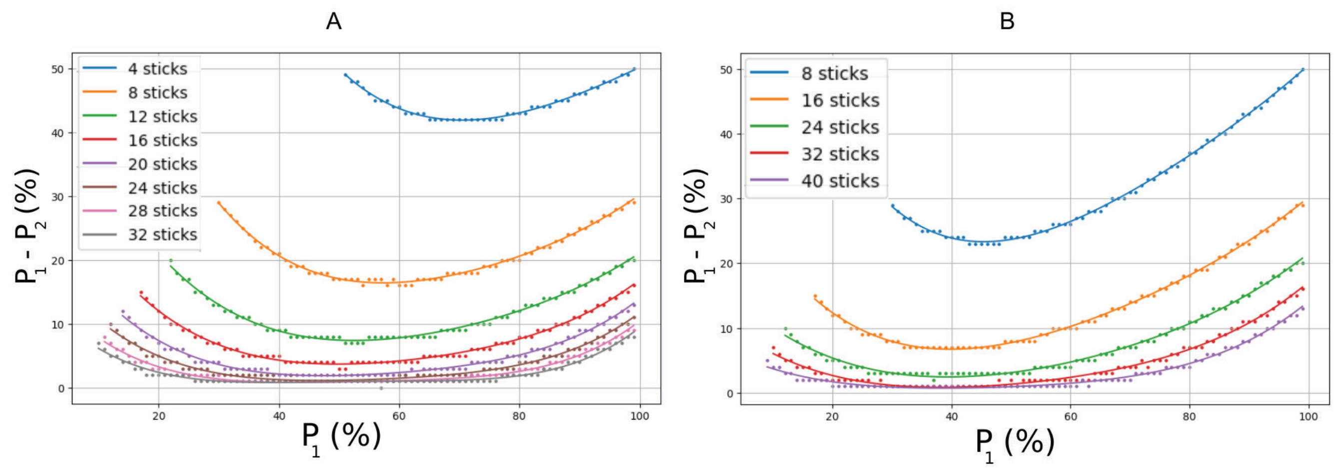

Figure 3: Minimum ability difference (P_1 − P_2) where W_1 > W_2 vs. Player 1 ability (P_1) in ideal positions for Player 2. Scatter plots and parabolic regressions for two game configurations: **A)** A = {1, 2, 3} (R^2 = 0.986); **B)** A = {1, 3, 4} (R^2 = 0.994). All values of n are losing starting positions for Player 1. The minimum ability difference was found by simulating games for every combination of P_1 – P_2 where P_1 and P_2 ϵ {1%, …, 100%} in 1% increments, identifying the smallest (P_1 − P_2) where Player 1 outperformed Player 2. Games were simulated using a Java algorithm that determined the optimal move, if any, using Grundy numbers, and used them with a probability of P_1 or P_2 accordingly. Win percentages were then calculated by dividing the number of games won by each player by 10^6 for each game configuration.

full game, which are needed to negate the advantage of ideal starting positions.

Examining stabilization points further, we found that n_s are minimized when P_1 + P_2 ≈ 100% and maximized when both abilities are either high or low. This is because if both players play with greater ability, a greater number of moves are required for imperfect moves to accumulate. Conversely, if both players play less accurately, a greater number of moves are required for optimal moves to accumulate.

Our data showed that the minimum ability differential required for a player in a losing position to win more than 50% games is highly dependent on their ability and is not constant. Furthermore, our data revealed that with each incremental increase in sticks, with the games starting in non-ideal positions for Player 1, the minimum ability difference for Player 1 to win decreased. This is because since there are a greater number of moves to be made it is more likely that the winning player will make a mistake, hence a lower difference in ability. Additionally, there were visible differences between games with different values of *A*. The minimum difference at the lowest value of P_1 in (1, 2, 3)-Nim was roughly equivalent to the minimum difference at the greatest value of P_1 (**Figure**

https://doi.org/10.59720/25-114

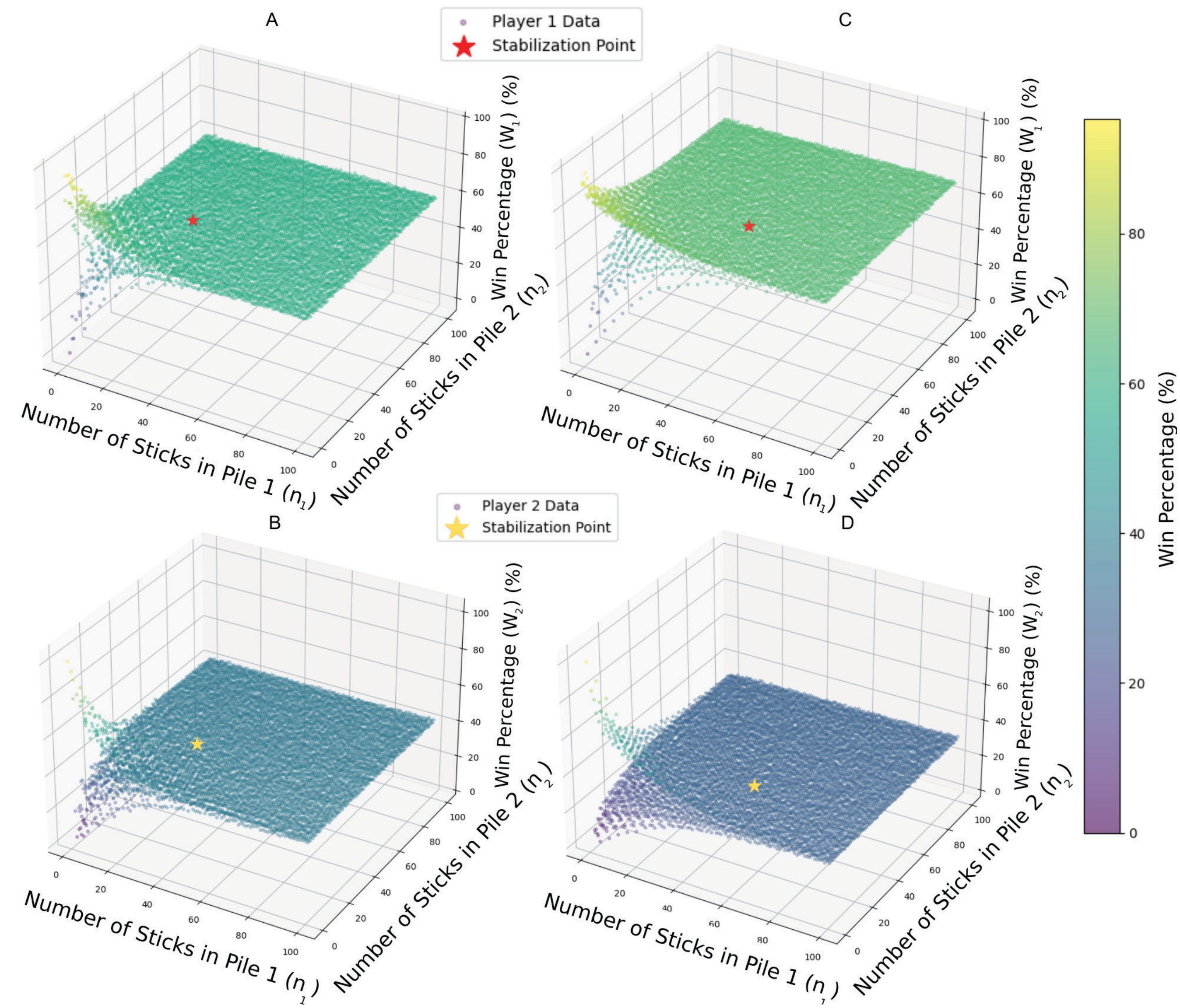

Figure 4: Number of sticks in pile 1 (n_1) and number of sticks in pile 2 (n_2) vs. Player 1 and Player 2 win percentages (W_1, W_2). Scatterplots and stabilization points for two game configurations in 10^5 simulations: **A)** and **B)** A_1 = {1, 3, 4}, A_2 = {1, 4, 5}, P_1 = 80%, P_2 = 70% (n_s = (31, 40)), though **A)** represents W_1 while **B)** represents W_2. Graphs **C)** and **D)** A_1, A_2 = {1, 2, 3}, P_1 = 90%, P_2 = 80% (n_s = (51, 33)), though **C)** represents W_1 while **D)** represents W_2. Games were simulated using a Java algorithm that brute-forced every allowed move in both piles until $G_1(n_1) \oplus G_2(n_2) = 0$ (if possible) and used the optimal move, if any, with a probability of P_1 or P_2 accordingly. Win percentages were then calculated by dividing the number of games won by each player by 10^5 for both game configurations. A Python algorithm computes the stabilization point n_s as the value of (n_1, n_2) in the first moving average window whose range of gradient magnitudes < 1 (**Equation 8**).

3A). In (1, 3, 4)-Nim, the lowest value of P_1's minimum difference was significantly lower than that of the greatest value of P_1). Also, while the curves' vertices in (1, 2, 3)-Nim were around the center of their domain, the curves' vertices in (1, 3, 4)-Nim were left of the center.

While our experiments effectively demonstrated various properties inherent to Nim, we encountered several limitations that warrant further consideration. Due to computational constraints, we were only able to simulate 10^5 (two-pile) and 10^6 (one-pile) games for each game configuration. A greater number of simulations would give more accurate data, with little to no significant anomalies. In addition, we were unable to compute data related to stabilization points for more than two piles, since it resulted in 10^{2p+5} game simulations for p number of piles. Access to more powerful computers would have allowed us to compute data for more piles. Our playing algorithm also used a specific strategy for situations with no optimal moves or multiple optimal moves. It tended to result in shorter games, decreasing the probability that a player in an ideal starting position would make a mistake, increasing their win percentages.

For future studies, we can use a probability equation to estimate the stabilized win percentage (win percentage after n_s) for both players as n approaches ∞:

$$\frac{a}{a+b} = \frac{P_1(100-P_2)}{P_1(100-P_2)+P_2(100-P_1)} \cdot 100 \quad (6)$$

For example, the stabilized win percentage for (1,2,3)-Nim (P_1 =90% and P_2 = 80%) was computed to be approximately 69.29% and the equation's estimate was 69.23%. Notably, the probability equation does not involve any variables unique to Nim, so we can apply it in future experiments to explore the similarities between imperfect play in Nim and other impartial games. However, this equation is not applicable for multi-pile Nim; hence, we defaulted to computational simulations in all our experiments for standardization.

Our work provides a framework for understanding how player ability, number of sticks, and starting positions influence win probabilities in impartial games. We computationally and mathematically showed that an increase in the number of sticks reduced the effect of an ideal starting position, and player ability became the primary driver of performance in Nim games. While our research is grounded in Nim, the findings may offer broader insights into decision-making. According to the Sprague-Grundy theorem, all impartial games are equivalent to single-pile Nim (10). Hence, these results may extend to games like Sprouts, Kayles, Quarto, and Chomp (9). This impartial game logic is also relevant in reinforcement learning (RL), where agents must learn strategies in structured, turn-based environments (12). RL agents still face significant challenges with impartial games like Nim due to their highly abstract mathematical nature (12). By modeling how imperfect decisions accumulate over repeated interactions, this work may offer a basis for exploring learning behavior in systems where agents are not guaranteed to act optimally (12-14). This is essential to train robust decision-making agents that are capable of accurately making moves in uncertain, complex game or game-like environments, including impartial games (13,14). Additionally, for Nim variations intersecting economic theory, such enhancements in efficient RL reasoning could also be applicable to the respective economic interaction models such as resource allocation and bidding systems.

MATERIALS AND METHODS

Stabilization Point of One-Pile Nim

To find the stabilization point in a one-pile game, the derivative of a moving average *MA,* a statistical method that smooths data by averaging values in a window, was used, specifically a window size of 10. The stabilization point was defined using a threshold of 0.06:

$$n_s = \arg\min_{i \in [i, i+10)} (\max(MA') - \min(MA') < 0.06) + 5 \quad (7)$$

Argmin is a mathematical function that returns the minimum value that satisfies a given condition. The condition is that the range of the derivatives < 0.06 in a 10-stick window where *i* is the number of sticks at the start of the window and *i+10* is the number of sticks at the end of the window. The range was calculated by subtracting the lowest derivative *(min(MA'))* from the greatest in the window *(max(MA'))*.

Stabilization Point of Two-Pile Nim

To find the stabilization point in a two-pile game, a Gaussian smoothing with Σ = 0.1 is applied and then gradient magnitudes (‖▽‖) are computed:

$$n_s = \arg\min_{(i,j) \in W_{i,j}} (\max \| \nabla Z \| - \min \| \nabla Z \| < 1) \quad (8)$$

Argmin returned the lowest combination of number of sticks that satisfy the condition: the range of the gradient magnitudes < 1 in a window where *W* is the region from the combination of starting number of sticks *(i, j)* to the point where the starting number of sticks in both piles is 100.

Game Simulation

A Java program calculated the optimal move by testing every single possible move in every single pile until the Grundy number computed to 0 (winning). If there were no such moves, it chose the smallest move possible in the first pile by default. If there was more than one optimal move, it chose the biggest move. It tallied up the total number of wins for each player and divided them by the total number of games played. When playing imperfectly, the probability of Player 1 choosing an optimal move was represented as P_1 and Player 2 probability as P_2.

Software and Packages

The Java Programming Language (version 1.8.0 51) was used for the Monte Carlo simulations and game calculations (Grundy numbers and optimal move calculation). Java's Util Package was utilized for hash maps, hash sets, sets, scanners, and random number methods (15). Java IO was used for file writing functionality (16). For stabilization point calculations and graphing, the Python Programming Language (version 3.10) was used. The NumPy package (version 1.26.4) was used for mathematical calculations (17). Mathplotlib (version 3.7.1) was used for graphing (18). The SciPy package (version 1.13.1) was used for Gaussian filters and linear algebra functions (19). All code used in this study can be found in our Github repository (20).

ACKNOWLEDGMENTS

We thank Rena Katz for proofreading the paper, providing significant feedback, and assisting with the references.

Received: March 31, 2025
Accepted: October 7, 2025
Published: August 2, 2026

REFERENCES

1. Bouton, Charles. "Nim, A game with a Complete Mathematical Theory." *Annals of Mathematics*, vol. 3, 1901, pp. 35–39, https://doi.org/10.2307/1967631.
2. Berlekamp, Elwyn R., et al. "Winning Ways for Your Mathematical Plays. Vol. 1: Games in General." Academic Press, 1982, pp. 55, 310, 381, 393.
3. Berlekamp, Elwyn R. "The economist's view of combinatorial games." *Games of No Chance*, edited by Richard J Nowakowski, Cambridge University Press, 1996, pp. 365–405.
4. Kant, Prem, and Urban Larsson. "Survey of Richman bidding combinatorial games." *Games of No Chance 6,* edited by Urban Larsson, Cambridge University Press, May 2025, pp. 43–54.

https://doi.org/10.59720/25-114

5. Lazarus, Andrew J., et al. "Richman games." *Games of No Chance*, edited by Richard J. Nowakowski, Cambridge University Press, 1996 pp. 439–450.
6. Davies, Alfie M., et al. "Combinatorial game theory monoids and their absolute restrictions: a survey." *Games of No Chance 6*, edited by Urban Larsson, Cambridge University Press, May 2025, pp. 1–24.
7. Grundy, Patrick M. “Mathematics and Games.” *Eureka*, vol. 2, 1939, pp. 6–8.
8. Kogler, Jakob, and Malek Sware. “MEX (Minimal Excluded) of a Sequence.” *Algorithms for Competitive Programming*, 2024, https://cp-algorithms.com/sequences/mex.html. Accessed 5 Aug. 2024.
9. “Theory of Impartial Games.” *Massachusetts Institute of Technology*, 2009, https://web.mit.edu/sp.268/www/nim.pdf. Accessed 4 Aug. 2024.
10. Sprague, Roland. “Über Mathematische Kampfspiele.” *Tohoku Mathematical Journal*, vol. 41, 1 Jan. 1935, pp. 438–444.
11. Kroese, Dirk P., et al. “Why the Monte Carlo Method Is so Important Today.” *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 6, no. 6, 20 June 2014, pp. 386–392, 1314, https://doi.org/10.1002/wics.1314.
12. Zhou, Bei, and Søren Riis. “Impartial Games: A Challenge for Reinforcement Learning.” *ArXiv*, 25 May 2022, https://doi.org/10.48550/arxiv.2205.12787.
13. Gleave, Adam, et al. "Adversarial policies: Attacking deep reinforcement learning." *ArXiv*, Jan 2021, https://doi.org/10.48550/arXiv.1905.10615.
14. Pinto, Lerrel, et al. “Robust Adversarial Reinforcement Learning.” *Proceedings of the 34th International Conference on Machine Learning*, vol. 70, 6 Aug. 2017, pp. 2817–26, https://dl.acm.org/doi/10.5555/3305890.3305972
15. “Package Java.util.” *Oracle*, 6 Jan. 2020, https://docs.oracle.com/javase/8/docs/api/java/util/package-summary.html. Accessed 5 Aug. 2024.
16. “Package Java.io.” *Oracle*, https://docs.oracle.com/javase/8/docs/api/java/io/package-summary.html. Accessed 5 Aug. 2024.
17. “NumPy User Guide.” *NumPy*, https://numpy.org/doc/stable/user/index.html. Accessed 5 Aug. 2024.
18. “Users Guide.” *Mathplotlib*, https://matplotlib.org/stable/users/index.html. Accessed 5 Aug. 2024.
19. “SciPy User Guide.” *SciPy*. https://docs.scipy.org/doc/scipy/tutorial/index.html. Accessed 5 Aug. 2024.
20. Sinha, Rohan, et al. *NIM*. GitHub, 2026, https://github.com/newrohansinha/NIM.