An Analysis of Snakes and Ladders as a Markov Chain and through Simulation

Alan Guo (The Flash), Guy Thampakkul (Batman),
Christopher Zhang (Aquaman), Alan Zhou (Green Lantern)

April 30, 2023

Abstract

We explore the game of Snakes and Ladders, first in a single-player setting and then in a multi-player setting. We create a sample board with 16 squares, and analyze its behavior mathematically and through simulation. Our findings show that game duration is highly variable on a small board, and decreases sharply as number of players increase. Further, in a two-player game, the player first to act has approximately a 3% edge keeps a slight (but decreasing) edge as we increase the handicap of the second player, until we start the second player on square 7. At this point, the second player is more likely to win. In multi-player settings, the player first to act retains approximately 4% above average win up to 5-players. Code for producing random Snakes and Ladder game boards, multi-player simulation, and calculation of transitions and long run probabilities and matrices for any Snakes and Ladders game board are provided in the paper's code repository.

1 Executive Summary

We are interested in examining games of chance, and we do so through the lens of Snakes and Ladders. Snakes and Ladders is a children's board game in which players, racing toward a destination at the end of the board, take turns rolling a standard die to determine how far they move that turn. If they land on the head of a snake, they are instead moved to its tail, and if they land on the bottom of a ladder, they are moved to its top. Snakes and Ladders provides a simplistic setting, absent factors like idiosyncratic player strategy and decision-making, for us to ask questions such as "What is the edge of the player that goes first?"—equivalently, "How much would you pay to play a two-player game of Snakes and Ladders where you start?"—or "How confident are you that a one-player game of Snakes and Ladders will finish within 10 turns?" Our report is divided into two parts.

First, we create a sample board with 16 squares and explore the game of Snakes and Ladder in a one-player setting, learning characteristics about the

expected duration of the game. It is important to note that our results are highly specific to our particular game board and its placement of snake and ladders. We find analytically that a single-player game conclude in 10.38 turns on average, with standard deviation 7.90, suggesting that game duration is quite variable. These values are confirmed in simulation. Interestingly, the expected number of turns to reach the final square is the same if the player is at squares 11, 12, 13, or 14. This is because the player must roll exactly the right number to reach square 16—otherwise, they will stay put. Square 15 has different behavior because it contains the head of a snake.

Then, we extend our analysis to two-player games, looking at the probability that Player 1 wins. We find that Player 1 has a 52.7% chance of winning in a standard game where both player start on the first square, decreasing all the way to 38.2% when Player 2 starts on squares 12, 13, or 14. Player 1 retains an edge over Player 2 when Player 2 starts on squares 1-6, and actually has a higher win rate when Player 2 starts on squares 3 or 5 instead of 1. These results are confirmed in simulation. Finally, we simulate 3, 4, and 5 player games and find that game duration decreases sharply as number of players increases.

2 Introduction

Throughout the annals of time, human civilization has had an intimate relationship with games, particularly those of chance. Games of chance have been enjoyed by humans for thousands of years, tracing back to ancient civilizations such as the Greeks, Romans, and Chinese [5]. A crude predecessor to the die, an animal heel bone called the astragalus or knucklebone, dates to 3600 BCE, a history that birthed the gambling colloquialism of "the bones" that we use to this day [4]. For those civilizations of the past, these games of chance were a way to 'divine' fate—randomness was seen as a way of divination, ascertaining the will of the Gods [5].

It wasn't until the 16th century when people formally attempted to analyze chance via mathematical theory. Gerolamo Cardano's book, *Liber de ludo aleae* ("Book on Games of Chance"), was one of the first accounts of mathematical probability and calculating odds, written in the 1560s [2]. Furthermore, the canonical birthplace of mathematics in chance, probability, dates to a correspondence between the French mathematicians Pierre de Fermat and Blaise Pascal in 1654, arguing over the proper division of stakes when a game of chance is interrupted [3]. It's inarguable that the history of probability is deeply rooted within, and intertwined with, these games of randomness, its evolution spurred on in efforts to find greater success at the gambling table. This can be seen throughout each probabilistic innovation—take Monte Carlo methods, a revolutionary sampling technique named after the famed casino in Monaco.

As an homage to this storied relationship between games of chance and probability, we've chosen to model and analyze two dice games via stochastic processes (Markov chains). The first is Snakes and Ladders, a simple game with a history dating all the way back to the tombs of Egypt (where it was known

as Hounds and Jackals [1]). The second is a more modern game known as Cosmic Wimpout, a twist on previous dice-rolling strategy games like Yahtzee, Generala, and Kniffel. Through our analysis, we seek to 1) find expected values and variance of game length and scoring, and 2) use insights of expected values and probabilities to ascertain the edge certain players have.

3 Snakes and Ladders

In Snakes and Ladders, players race from the start square to the end square by rolling a standard die each turn and moving that many squares. If the square a player lands on contains the bottom of a ladder, they instantly climb to the top of the ladder. Similarly, if the square contains the bottom of a snake, they are consumed by the snake and fall to the bottom of the snake. The game is typically played on an 8x8, 10x10, or 12x12 board.

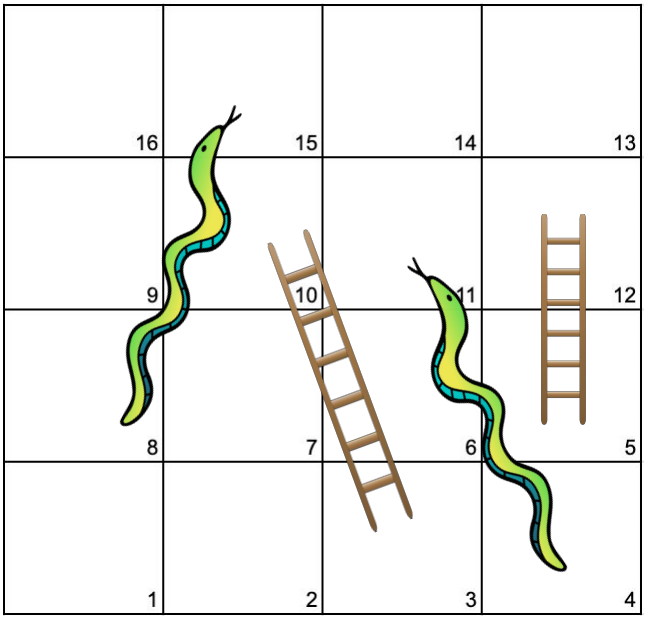

Figure 1: Example Snakes and Ladders board

We first discuss Snakes and Ladders in the one-player setting.

3.1 Absorbing Markov Chain

We will model Snakes and Ladders in the one-player setting using an Absorbing Markov chain; the game satisfies the Markov property because the probability P_{ij} of moving from state i to j depends only on the current state i. Each state represents a different square. Let L be the set of ladders where $(L_s, L_e)_i \in L$ represents the start and end squares of a ladder. Similarly let S be the set of snakes with $(S_s, S_e)_i \in S$. Thus, the set R of states reachable from i includes all states j where $i + 1 \leq j \leq i + 6$, unless $(j, L_e) \in L$ or $(j, S_e) \in S$, in which case L_e or S_e, respectively, is included in R instead. In general, we have

$$P_{ij} = \begin{cases} 1/6 & \text{for } j \in R \\ 0 & \text{otherwise} \end{cases}$$

One additional rule states that, to win, a player must land exactly on the end square. If the player's roll would take them past the end square, they instead do not move.

Snakes and Ladders, modeled as a Markov Chain, is an absorbing chain because the final state is an absorbing state—a state that once entered, cannot be left, and it is possible to reach the absorbing (final) state from any state. In context, once we reach the final square of the board, we do not move anywhere else, and we are able to reach the final square of the board from any other square on the board.

3.1.1 Mathematics

First, we introduce some notation.

The matrix, P, above represents a transition matrix, where rows of P represent sources, while columns represent destinations. The elements in the matrix is the probability of transitioning from the source to the destination.

Let an absorbing Markov chain with transition matrix P have t transient states and r absorbing states.

The transition matrix P can be written as

$$P = \begin{bmatrix} Q & R \\ \mathbf{0} & I_r \end{bmatrix}$$

where Q is a t-by-t, R is a non-zero t-by-r matrix, $\mathbf{0}$ is an r-by-t zero matrix, and I_r is the r-by-r identity matrix. Q describes the probability of transitioning between transient states, while R describes the probability of transition from some transient state to some absorbing state. I_r is the identity matrix as once in an absorbing state, the probability of remaining there is 1. In some cases, there is a set of absorbing states, in which case I_r would be replaced by the transition matrix between the absorbing states.

A property of absorbing Markov chains is the expected number of visits to a transient state j starting in transient state i before exiting the set of transient states and being absorbed. This information is stored in the *fundamental matrix*, N, which is computed by summing Q^k for all non-negative integers, i.e.

$$N = \sum_{k=0}^{\infty} Q^k.$$

The matrix N can be be computed by doing $(I_t - Q)^{-1}$, where t is the number of transient state, as stated above. This computation follows from the calculation of the sum of infinite geometric series of scalars:

$$\sum_{k=0}^{\infty} q^k = \tfrac{1}{1-q} = (1-q)^{-1},\ q \in [0,1].$$

The expected number of steps before being absorbed into absorbing states when starting from any transient state can be computed with the sum over the transient states. The expected number of steps before reaching the absorbing state from transient state i is given by the i^{th} entry in the vector, $\mathbf{t} = N\mathbf{1}$.

The variance on the number of steps before being absorbed by an absorbing state is the i^{th} entry in the vector, $(2N - I_t)\mathbf{t} - \mathbf{t}_{sq}$, where $\mathbf{t}_{sq}$ is $\mathbf{t}$ squared, element-wise.

3.1.2 Example

For the example 4x4 board in Figure 1, the transition probability matrix is given by

$$P = \begin{bmatrix}
0 & 1/6 & 0 & 1/6 & 0 & 1/6 & 1/6 & 0 & 0 & 1/6 & 0 & 1/6 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 0 & 1/6 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 0 & 1/6 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 1/6 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 0 & 1/6 & 1/6 & 0 & 1/6 & 1/6 & 1/6 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 1/6 & 0 & 1/6 & 0 & 1/6 & 1/6 & 1/6 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 1/6 & 1/6 & 1/6 & 0 & 1/6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/6 & 0 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 1/6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 1/3 & 1/6 & 1/6 & 0 & 1/6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 0 & 0 & 1/6 & 1/2 & 1/6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 0 & 0 & 2/3 & 0 & 1/6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5/6 & 1/6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

Note that the end state (16) is the only absorbing state in our chain. We are interested in the expected and variance of time spent in the transitive states. In context, we are interested in estimating the number of die rolls a player is expected to perform before winning the game.

$$Q = \begin{bmatrix}
0 & 1/6 & 0 & 1/6 & 0 & 1/6 & 1/6 & 0 & 0 & 1/6 & 0 & 1/6 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 0 & 1/6 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 0 & 1/6 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 1/6 & 1/6 & 1/6 & 0 & 1/6 & 1/6 & 0 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 0 & 1/6 & 1/6 & 0 & 1/6 & 1/6 & 1/6 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 1/6 & 0 & 1/6 & 0 & 1/6 & 1/6 & 1/6 & 0 \\
0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 1/6 & 1/6 & 1/6 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/6 & 0 & 0 & 1/6 & 1/6 & 1/6 & 1/6 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 1/3 & 1/6 & 1/6 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 0 & 1/2 & 1/6 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/6 & 0 & 0 & 0 & 0 & 0 & 2/3 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5/6
\end{bmatrix}$$

$$N = (I-Q)^{-1} \begin{bmatrix}
1 & 0.17 & 0 & 0.73 & 0 & 0.32 & 0.37 & 1.34 & 0.46 & 0.73 & 0 & 1.28 & 1.39 & 2.6 & 0 \\
0 & 1 & 0 & 0.71 & 0 & 0.28 & 0.33 & 1.47 & 0.47 & 0.71 & 0 & 1.24 & 1.4 & 2.64 & 0 \\
0 & 0 & 1 & 0.72 & 0 & 0.29 & 0.34 & 1.5 & 0.64 & 0.58 & 0 & 1.27 & 1.44 & 2.71 & 0 \\
0 & 0 & 0 & 1.55 & 0 & 0.26 & 0.3 & 1.45 & 0.59 & 0.69 & 0 & 1.21 & 1.42 & 2.68 & 0 \\
0 & 0 & 0 & 0.76 & 1 & 0.29 & 0.34 & 1.51 & 0.65 & 0.76 & 0 & 1.08 & 1.45 & 2.72 & 0 \\
0 & 0 & 0 & 0.69 & 0 & 1.12 & 0.3 & 1.45 & 0.59 & 0.69 & 0 & 1.21 & 1.42 & 2.68 & 0 \\
0 & 0 & 0 & 0.64 & 0 & 0.11 & 1.12 & 1.4 & 0.54 & 0.64 & 0 & 1.11 & 1.61 & 2.65 & 0 \\
0 & 0 & 0 & 0.59 & 0 & 0.1 & 0.11 & 2.22 & 0.5 & 0.59 & 0 & 1.03 & 1.48 & 2.91 & 0 \\
0 & 0 & 0 & 0.59 & 0 & 0.1 & 0.11 & 1.36 & 1.36 & 0.59 & 0 & 1.03 & 1.48 & 2.91 & 0 \\
0 & 0 & 0 & 0.5 & 0 & 0.08 & 0.1 & 1.17 & 0.31 & 1.36 & 0 & 0.88 & 1.27 & 2.49 & 0 \\
0 & 0 & 0 & 0.29 & 0 & 0.05 & 0.06 & 1.11 & 0.25 & 0.29 & 1.2 & 0.81 & 1.24 & 2.45 & 0 \\
0 & 0 & 0 & 0.29 & 0 & 0.05 & 0.06 & 1.11 & 0.25 & 0.29 & 0 & 2.01 & 1.24 & 2.45 & 0 \\
0 & 0 & 0 & 0.29 & 0 & 0.05 & 0.06 & 1.11 & 0.25 & 0.29 & 0 & 0.51 & 2.74 & 2.45 & 0 \\
0 & 0 & 0 & 0.29 & 0 & 0.05 & 0.06 & 1.11 & 0.25 & 0.29 & 0 & 0.51 & 0.74 & 4.45 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6
\end{bmatrix}$$

The expected number of rolls before reaching the absorbing state from each state is:

$$E = \mathbf{t} = N\mathbf{1} = \begin{bmatrix} 10.38 \\ 10.26 \\ 10.48 \\ 10.15 \\ 10.55 \\ 10.15 \\ 9.81 \\ 9.52 \\ 9.52 \\ 8.16 \\ 7.76 \\ 7.76 \\ 7.76 \\ 7.76 \\ 6 \end{bmatrix}$$

The variance number of rolls before reaching the absorbing state from each state is:

$$V = (2N - I_t)\mathbf{t} - \mathbf{t}_{sq} = (2N - I_t) \begin{bmatrix} 10.38 \\ 10.26 \\ 10.48 \\ 10.15 \\ 10.55 \\ 10.15 \\ 9.81 \\ 9.52 \\ 9.52 \\ 8.16 \\ 7.76 \\ 7.76 \\ 7.76 \\ 7.76 \\ 6 \end{bmatrix} - \begin{bmatrix} 107.74 \\ 105.27 \\ 109.83 \\ 103.02 \\ 111.3 \\ 103.02 \\ 96.24 \\ 90.63 \\ 90.63 \\ 66.59 \\ 60.22 \\ 60.22 \\ 60.22 \\ 60.22 \\ 36 \end{bmatrix} = \begin{bmatrix} 62.45 \\ 62.17 \\ 61.93 \\ 61.94 \\ 62.06 \\ 61.94 \\ 61.68 \\ 61.36 \\ 61.36 \\ 61.35 \\ 59.32 \\ 59.32 \\ 59.32 \\ 59.32 \\ 30 \end{bmatrix}$$

Therefore, from our analysis, in the Snakes and Ladders game as described in Figure 1, the expected number of rolls for a player from the start to winning is 10.38 rolls with a variance of 62.45 (standard deviation of 7.90).

3.2 Simulation

Below are histograms of number of the of turns required to reach the final state (state 16) from all states i. The mean and variances are shown on each plot. They match the expected value and variances computed in the previous section.

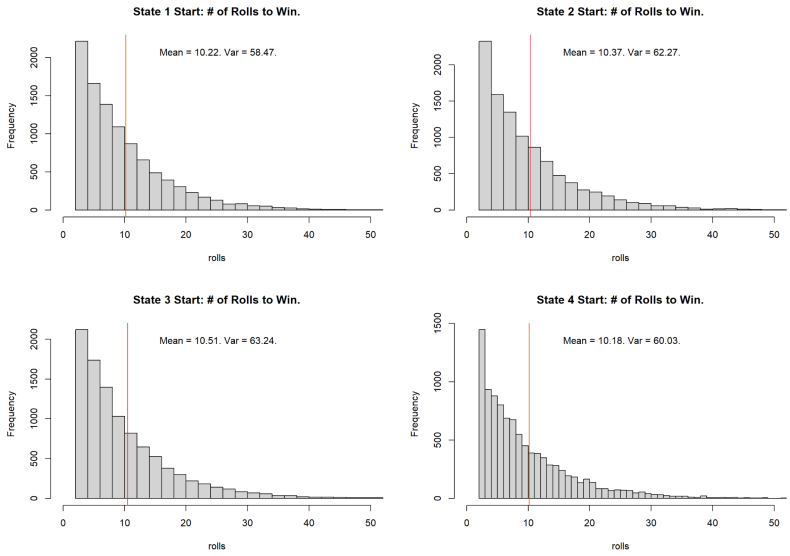

Figure 2: Histograms for Expected Number of Rolls Starting in States 1-4

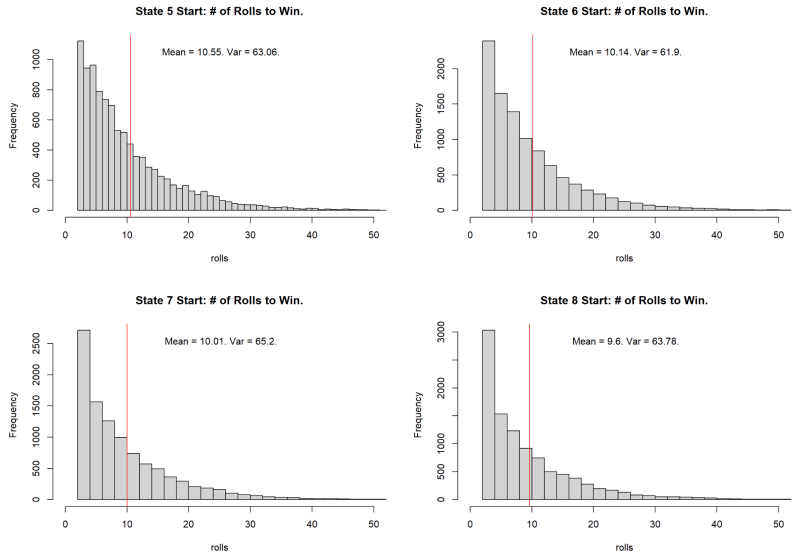

Figure 3: Histograms for Expected Number of Rolls Starting in States 5-8

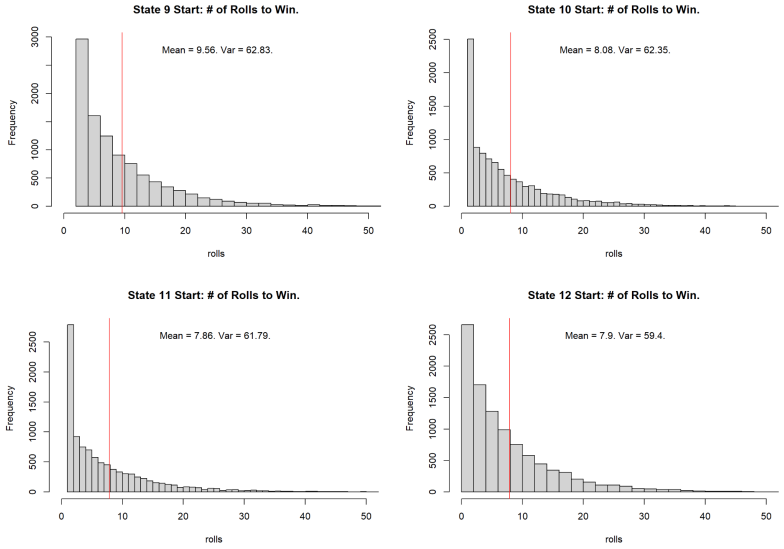

Figure 4: Histograms for Expected Number of Rolls Starting in States 9-12

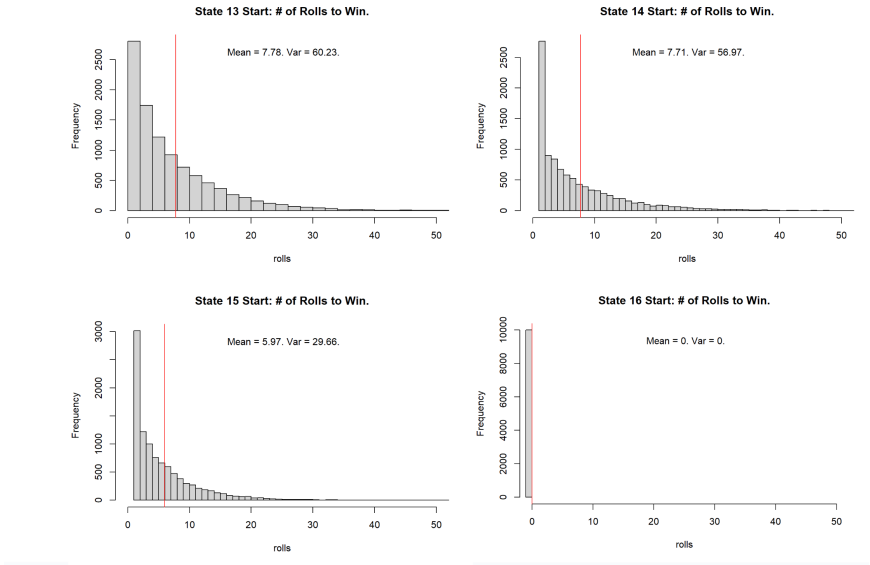

Figure 5: Histograms for Expected Number of Rolls Starting in States 13-16

3.3 Multi-player Snakes and Ladders

The one-player setting gives us insight into the behavior of gameplay in Snakes and Ladders, but the outcome is pre-determined; there is only one possible winner. With two or more players, we can begin to analyze the likelihood of any particular player winning the game. However, our model has to keep track of the square that each player is on. That is, with n players, each state is an n-tuple of the current squares of each player. Thus, our state space grows exponentially in n and quickly becomes analytically intractable. We first examine the two-player setting analytically, and then we show simulation results for more players.

3.3.1 Two-player Markov Chain Set-up

Let B be the set of squares on the Snakes and Ladders board. Then, the state space Ω of the two-player game is given by $\Omega = B \times B$, with cardinality $|\Omega| = |B|^2$. In the 256-by-256 transition matrix induced by the board in Figure 1, Each state represents the probability $P_{(i,j),(l,k)}$ of transitioning from state (i, j) to state (l, k) in one step, where $i, j, l, k \in B$. Every state $(i, 16) \forall i$ and $(16, j) \forall j$ is an absorbing state; $(i, 16)$ for $i \in 1, 2, \ldots, 15$ correspond to states where Player 2 wins and $(16, j)$ for $i \in 1, 2, \ldots, 16$ correspond to states where

Player 1 wins. (Since Player 1 goes first, the state $(16, 16)$ corresponds to Player 1 winning.) The transition probabilities follow directly from those of the one-player game by updating both players' states simultaneously and independently. That is, $P_{(i,j),(l,k)} = P_{il}P_{jk}$ by independence. For states where $i, j \leq 11$, we have

$$P_{(i,j),(l,k)} = \begin{cases} 1/36 & \text{for } l \in R_i \text{ and } k \in R_j \\ 0 & \text{otherwise} \end{cases}$$

where R_i and R_j are the states reachable from i and j, respectively.

For states where $i > 11$ or $l > 11$: Without loss of generality, let $i > 11$, and $j \leq 11$. Define $d = 6 - (16 - i)$. We have

$$P_{(i,j),(l,k)} = \begin{cases} 1/36 & \text{for } l \in R_i \neq i \text{ and } k \in R_j \\ d/36 & \text{for } l = i \text{ and } k \in R_j \\ 0 & \text{otherwise} \end{cases}$$

This extends to its symmetric case, $i \leq 11, j > 11$. For the final case, $i, j > 11$, let $d_1 = 6 - (16 - i)$ and $d_2 = 6 - (16 - j)$. We have

$$P_{(i,j),(l,k)} = \begin{cases} 1/36 & \text{for } l \in R_i \neq i \text{ and } k \in R_j \neq j \\ d_1/36 & \text{for } l = i \text{ and } k \in R_j \neq j \\ d_2/36 & \text{for } l \in R_i \neq i \text{ and } k = j \\ d_1 d_2/36 & \text{for } l = i \text{ and } k = j \\ 0 & \text{otherwise} \end{cases}$$

3.3.2 Analysis of Two-Player Game

Similarly to the one-player setting, we reorder the rows and columns of our transition matrix to group the absorbing states together, writing the transition matrix as

$$P = \begin{bmatrix} Q & R \\ \mathbf{0} & I_r \end{bmatrix}$$

where Q is a t-by-t matrix, R is a non-zero t-by-r matrix, $\mathbf{0}$ is an r-by-t zero matrix, and I_r is the r-by-r identity matrix.

Then, as in the one-player game, we compute the fundamental matrix $N = (I - Q)^{-1}$. The NR matrix represents the absorption probabilities, i.e. $(NR)_{(i,j),(l,k)}$ is the probability of being absorbed into state (l, k) having started in state (i, j). We are interested in the starting states $(1, 1), (1, 2), \ldots, (1, 16)$, i.e. the standard starting state $(1, 1)$ along with the starting states where Player 2 has an initial

advantage. Since we had access to enough computational power, we instead computed NR by raising the transition matrix to a very large exponent and taking the appropriate rows and columns. For visual clarity, we only show the absorbing states $(16,1),(16,2),\ldots,(16,16)$ where Player 1 wins:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
(1,1)	0	0	0	0.033	0	0.009	0.012	0.075	0.027	0.033	0	0.067	0.081	0.136	0	0.055
(1,2)	0	0	0	0.032	0	0.007	0.01	0.073	0.027	0.032	0	0.064	0.082	0.14	0	0.055
(1,3)	0	0	0	0.032	0	0.007	0.01	0.075	0.027	0.034	0	0.065	0.084	0.143	0	0.055
(1,4)	0	0	0	0.032	0	0.006	0.008	0.072	0.025	0.03	0	0.062	0.083	0.143	0	0.056
(1,5)	0	0	0	0.035	0	0.008	0.011	0.075	0.028	0.035	0	0.064	0.084	0.143	0	0.055
(1,6)	0	0	0	0.03	0	0.007	0.008	0.072	0.025	0.03	0	0.062	0.083	0.143	0	0.056
(1,7)	0	0	0	0.027	0	0.006	0.007	0.07	0.022	0.027	0	0.055	0.084	0.143	0	0.056
(1,8)	0	0	0	0.024	0	0.006	0.007	0.069	0.019	0.024	0	0.05	0.076	0.151	0	0.056
(1,9)	0	0	0	0.024	0	0.006	0.007	0.067	0.021	0.024	0	0.05	0.076	0.151	0	0.056
(1,10)	0	0	0	0.019	0	0.005	0.006	0.056	0.018	0.021	0	0.041	0.064	0.128	0	0.047
(1,11)	0	0	0	0.019	0	0.005	0.006	0.056	0.018	0.021	0	0.041	0.064	0.128	0	0.047
(1,12)	0	0	0	0.016	0	0.003	0.003	0.053	0.014	0.016	0	0.039	0.062	0.128	0	0.047
(1,13)	0	0	0	0.016	0	0.003	0.003	0.053	0.014	0.016	0	0.028	0.073	0.128	0	0.047
(1,14)	0	0	0	0.016	0	0.003	0.003	0.053	0.014	0.016	0	0.028	0.039	0.162	0	0.047
(1,15)	0	0	0	0.028	0	0.005	0.006	0.058	0.024	0.028	0	0.05	0.072	0.131	0	0.047
(1,16)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Summing across the columns, we get the probability vector $\mathbf{p_1}$ of Player 1 winning conditioned on the game's starting state and the simulation results $\hat{\mathbf{p_1}}$ ($n = 10,000$ trials) for comparison:

$$\mathbf{p_1} = \begin{array}{c} \\ (1,1) \\ (1,2) \\ (1,3) \\ (1,4) \\ (1,5) \\ (1,6) \\ (1,7) \\ (1,8) \\ (1,9) \\ (1,10) \\ (1,11) \\ (1,12) \\ (1,13) \\ (1,14) \\ (1,15) \\ (1,16) \end{array} \begin{array}{c} p \\ \left(\begin{array}{c} 0.5274 \\ 0.5211 \\ 0.5339 \\ 0.5156 \\ 0.5373 \\ 0.5156 \\ 0.4969 \\ 0.4807 \\ 0.4807 \\ 0.4041 \\ 0.4041 \\ 0.3824 \\ 0.3824 \\ 0.3824 \\ 0.4471 \\ 0 \end{array} \right) \end{array} \qquad \text{Simulation} = \begin{array}{c} \\ (1,1) \\ (1,2) \\ (1,3) \\ (1,4) \\ (1,5) \\ (1,6) \\ (1,7) \\ (1,8) \\ (1,9) \\ (1,10) \\ (1,11) \\ (1,12) \\ (1,13) \\ (1,14) \\ (1,15) \\ (1,16) \end{array} \begin{array}{c} p \\ \left(\begin{array}{c} 0.529 \\ 0.561 \\ 0.52 \\ 0.528 \\ 0.529 \\ 0.483 \\ 0.483 \\ 0.474 \\ 0.5 \\ 0.375 \\ 0.413 \\ 0.386 \\ 0.362 \\ 0.355 \\ 0.445 \\ 0 \end{array} \right) \end{array}$$

From this probability vector, we can note some interesting findings: 1) Player 2 is less likely to win if starting on square 15 than 10, 11, 12, 13, 14. This is because of the snake on square 15–if player 2 doesn't immediately roll a 1, they remain on square 15, which then causes them to take the snake down to square 8. 2) Player 1 actually has a higher win rate if Player 2 starts on squares 3 or 5 instead of square 1. This is because squares 3 and 5 contain ladders–if Player 2 starts on squares 3 and 5, they are unable to take those ladders up, while Player 1 still can, which gives Player 1 a slight edge. Player 2's winning probabilities are simply the complement of Player 1's.

3.3.3 Simulation for {1, 2, 3, 4, 5} players

For more than two players, the transition matrix quickly becomes computationally intractable. Luckily, we can simulate multi-player games of Snakes and Ladders by simulating each player's turn independently and taking the first player to reach the final square as the winner. Ties are broken by player order, i.e. Player 2 would win over Player 3 if both arrived at square 16 on the same simulated turn. In k-player games ($n = 10000$ trials), we find average game lengths of

$$\begin{array}{cc} k & \text{turns} \\ 1 & \\ 2 & \\ 3 & \\ 4 & \\ 5 & \end{array} \begin{pmatrix} 10.3023 \\ 6.4012 \\ 5.0224 \\ 4.3372 \\ 3.8934 \end{pmatrix}$$

.

The proportion of Player 1 wins was

$$\begin{array}{cc} k & p \\ 1 & \\ 2 & \\ 3 & \\ 4 & \\ 5 & \end{array} \begin{pmatrix} 1 \\ 0.5206 \\ 0.3648 \\ 0.2830 \\ 0.2368 \end{pmatrix}$$

.

The proportion of Player 2 wins was

$$\begin{array}{cc} k & p \\ 1 & \\ 2 & \\ 3 & \\ 4 & \\ 5 & \end{array} \begin{pmatrix} 1 \\ 0.4794 \\ 0.3413 \\ 0.2661 \\ 0.2244 \end{pmatrix}$$

.

The proportion of Player 3 wins was

$$\begin{array}{cc} k & p \\ 1 & \left(\begin{array}{c} 0 \\ 0.5384 \\ 0.3788 \\ 0.2958 \\ 0.2446 \end{array} \right. \\ 2 & \\ 3 & \\ 4 & \\ 5 & \left. \begin{array}{c} \\ \\ \\ \\ \\ \end{array} \right) \end{array}$$

.

The proportion of Player 4 wins was

$$\begin{array}{cc} k & p \\ 1 & \left(\begin{array}{c} 0 \\ 0 \\ 0.2939 \\ 0.2314 \\ 0.1936 \end{array} \right. \\ 2 & \\ 3 & \\ 4 & \\ 5 & \left. \begin{array}{c} \\ \\ \\ \\ \\ \end{array} \right) \end{array}$$

.

The proportion of Player 5 wins was

$$\begin{array}{cc} k & p \\ 1 & \left(\begin{array}{c} 0 \\ 0 \\ 0 \\ 0.2195 \\ 0.1863 \end{array} \right. \\ 2 & \\ 3 & \\ 4 & \\ 5 & \left. \begin{array}{c} \\ \\ \\ \\ \\ \end{array} \right) \end{array}$$

.

Figure 6 shows the distribution of game length. Since the game ends when the first player reaches the final square, game length decreases sharply as the number of players increases. Similarly, with more players, Player 1 becomes less and less likely to win (although they always retain an edge, having observed winning probability greater than $\frac{1}{k}$).

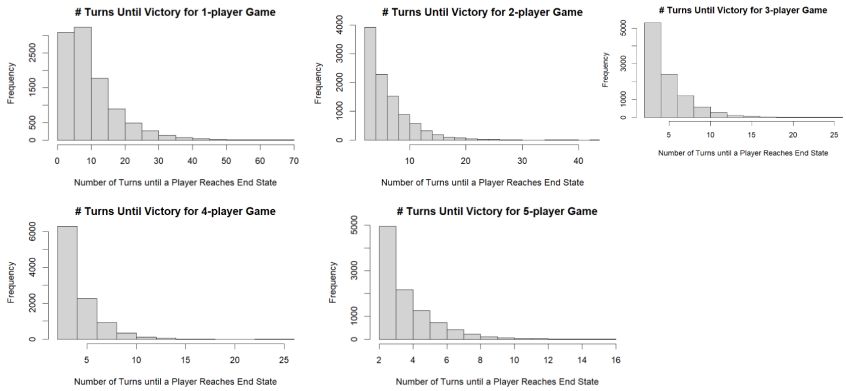

Figure 6: Histograms for number of turns played before one player reaches end state ($n = 10000$ games)

4 Appendices

4.1 Snakes and Ladders R Code

1. Set Up

   - `initialize.board(dim, n.ladders, n.snakes)`
     - Input(s): Desired dimensions of the square board, Desired number of ladders on the board, Desired number of Snakes on the board
     - Output(s): Board object (dimensions of the square board, list of ladder starts and ends, list of snake starts and ends). Snake and ladder starts and ends are never in the same square.

2. Mathematics

   - `trans.mat(board)`
     - Input(s): Snakes and Ladders Board Object, as outputted by the
       `initialize.board()`
       function.
     - Output(s): Transition probability matrix, P.
   - `print.tm(tm)`
     - Input(s): A matrix
     - Output(s): None. (Prints out the matrix as a string in LaTeX, to be inserted between
       `\begin{matrix}`
       and
       `\end{matrix}`
       .)
   - `E.V.turns.vector(board)`
     - Input(s): Snakes and Ladders Board Object, as outputted by the
       `initialize.board()`
       function.
     - Output(s): A list of two vectors, the first being the expected number of rolls vector, and the second being the variance of number of rolls vector. The i^{th} element of each vector corresponds to the expected number and variance of rolls starting from state i respectively.
   - `trans.mat.2players(board)`

- Input(s): Snakes and Ladders Board Object, as outputted by the

  `initialize.board()`

  function.
- Output(s): Transition probability matrix, $P \in M_{(n\times n)\times(n\times n)}$, where n is the number of spaces on the board.

3. Simulation

- `roll.die()`
  - Input(s): None
  - Output(s): A number between 1-6 (the result of a standard die roll).
- `play.turn(state,board)`
  - Input(s): State number to play from and Snakes and Ladders Board Object, as outputted by the

    `initialize.board()`

    function.
  - Output(s): The new state number.
- `play(board,state=0)`
  - Input(s): Snakes and Ladders Board Object, as outputted by the

    `initialize.board()`

    function and state number to start play from (default to 1 i.e., start playing from the start state (the beginning)).
  - Output(s): The number of turns/roll until final state was reached.

References

[1] Hounds and Jackals: A Game from an Ancient Egyptian Tomb.

[2] Liber de ludo aleae - noscemus.

[3] Probability and statistics | History, Examples, & Facts | Britannica.

[4] 13.1: Introduction to Games of Chance, May 2020.

[5] Gambling - History | Britannica, April 2023.