Knucklebones

Xiaoya Huang, Yuchen AN

June 28, 2023

1 Introduction

For assignment 6, we want to investigate some strategies in the dice game ***Knucklebones***. This game can be played with two players. Below are the rules of the game:

- Both players have a 3x3 grid on their side, and both of them start the game with a score of zero
- Players take turns rolling a single fair six-sided die and placing it on one empty square of their grid. Each die is worth the equivalent amount of points on its face. Players can place their roll on the same column as the previous roll or place it on a new column.A filled column does not accept any more dice.
- If a player has two or three dice showing the same number in a single column, the value of dices will increase. Their values are added together and multiplied by the number of matches. We call such situations "double" and "triple". For example, if a column contains 2-2-2, the total score of that column is $(2+2+2)\times 3 = 18$. If a column contains 4-1-4, then the score for that column is $(4+4)\times 2+1 = 17$.
- If a player places a dice that matches one or more dice in their opponent's corresponding column, all of their opponent's matching dice are removed.For example, the opponent rolls a 6 and places it on the first column. The player also rolls a 6 and puts it on their third column, which is the first column from the opponent's view. Then, the 6 on the opponent's side will be removed while the player keeps their 6.
- In each round, each player's score will be the sum of all the dice values on their side.
- When one player has filled all nine slots on their board, the game ends. The player with the higher score wins.

Let's simulate the game to better understand it. Here are example scenarios:
Scenario 1

$$Score = 5+2+1 = 8$$

5		
	2	1

Scenario 2

$$Score = (5+5) \times 2 + (2+2) \times 2 + 1 = 29$$

5		
5	2	1
	2	

Scenario 3

Player

5		
5	2	1
	2	

→

5		
5	2	1
2	2	

Opponent

1		
2	4	3
2	5	

→

1		
	4	3
	5	

Since the player places 2 on (3,1), which matches two dice in the opponent's corresponding column. All opponent's matching dice are removed. Therefore, the opponent no longer has the number 2 on (2,1) and (3,1) in his grid.

Based on these basic rules of the game, we first investigate the winning probability of the first mover (referred to as "player" in later sections, and the other player is referred to as "opponent"). Then, we come up with different strategies that may be used in the game to see whether these strategies increase the player's winning probability.

2 Simulations of Game while Playing with Strategies

We've come up with several strategies that can be used while playing the game:

1. Players roll the dice and randomly place the dice on an unoccupied square on their side. Players need to check if they place the dice that match one or more dice in their opponent's corresponding column. If their dices match, all of their opponent's matching dice need to be removed.

2. Make eliminating opponents' dice a priority. Players roll the dice and get a number. Players check if the number they got matches any numbers on their opponent's board. If their dices match, players remember the positions of the matching dice on their opponent's board. Next, players check whether the corresponding positions on their own board are occupied or not. If one corresponding position is unoccupied, players place the number on that position. If all corresponding positions are occupied, players check the columns where the corresponding positions on have empty square or not. If there is an empty square, players put dice on that square. If not, players randomly place the dice. If the number they got matches no numbers on their opponent's board, they randomly place the dice.

3. Players should avoid stacking large numbers in the same column early on. For example, If a player rolls a 6 in the first round and then rolls another 6 in the second turn, they cannot place the second 6 in the same column as the previous one. If the opponent has open spots in a parallel column, the stacked number can be easily erased. Instead, players should spread out big numbers, as not every column needs to be stacked with the same die to win. If players have to stack large numbers in the same column, they can randomly place that dice. So one strategy we have can be: We set numbers ≥ 3 are large numbers. If we roll numbers ≥ 3, we put the those numbers in different columns. If each column already has a number ≥ 3 and we roll a number ≥ 3 again, we randomly place this number.

4. Combine Strategy 2 and 3. In each turn, the player always tries to eliminate the opponent's dice first. If no matching dice on the opponent's board, the play avoids stacking large numbers in the same column.

2.1 Strategy 1: Randomly placing the dice

As we hope to learn whether some strategies are effective in increasing the winning probability of the player, we first estimate the player's winning probabilities when both player and opponent use no strategies but randomly put dice into the empty squares on their grid.

We do 1000 runs of 1000 simulated games which yield 1000 estimates of the winning probability and record the probability from each run. Figure 1 shows the density of the player's winning probabilities if no other strategy is used but only randomly placing dice(plotted by Sage).

Since the histogram in Fig.1 is approximately symmetric and bell-shaped, by using the **Central Limited Theorem**, we assume that the paler's winning probability is approximately normally distributed around the mean ($\bar{x} = 0.44$).

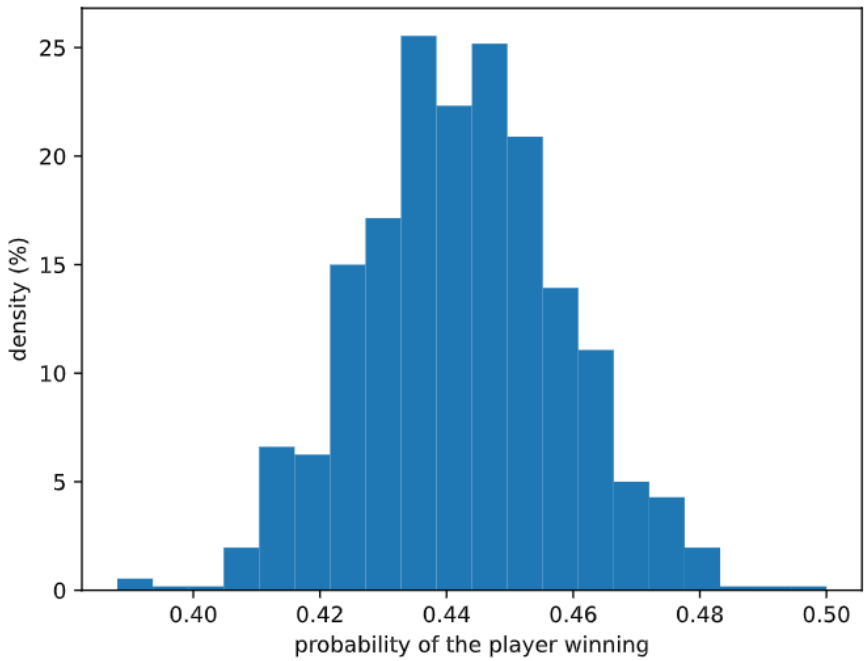

Figure 1: Histogram of player's winning probability in 1000 runs of 1000 simulated games if only randomly placing dice

To get more precise estimates, we do 10 runs of 100000 simulated games, and record the 10 runs' data. The simulation runs of 100000 are long enough since it is clear that the graph becomes increasingly convergent after 200 simulated games, and looks convergent at 1000 simulated games. We found that the line of the winning probability appears to be a horizontal line after 1000 times. Then, we the 99.8% confidence interval of the player's winning probability and convergence in the process of iteration (see Appendix for the reason why we are doing 99.8% CI).

With the calculation of Python, we have the 99.8% CI of the probability that the player wins while both players place the dice randomly in games: $(0.4410, 0.4437)$

2.2 Strategy 2: Make eliminating a priority.

One strategy we come up with is always to try to eliminate the opponent's existing dice on the board. If can't eliminate any opponent's dice, place the dice randomly on an empty square on the player's side of the board.

To estimate the mean probability of winning while both players are using this strategy, we again do 1000 runs of 1000 simulated games as we did in Strategy 1. By using the Central Limited Theorem, we could assume from the obtained data that the player's winning probability is approximately normally distributed with the mean probability to be around 0.420

Also to obtain more precise estimates, we do 10 runs of 100000 simulated games and plot the 10 runs' data to see whether the player's winning probability will converge to a certain value within the long-term estimation (Fig. 2, plotted by Python). The simulation runs of 100000 are long enough

since it is clear that the graph becomes increasingly convergent after 200 simulated games.

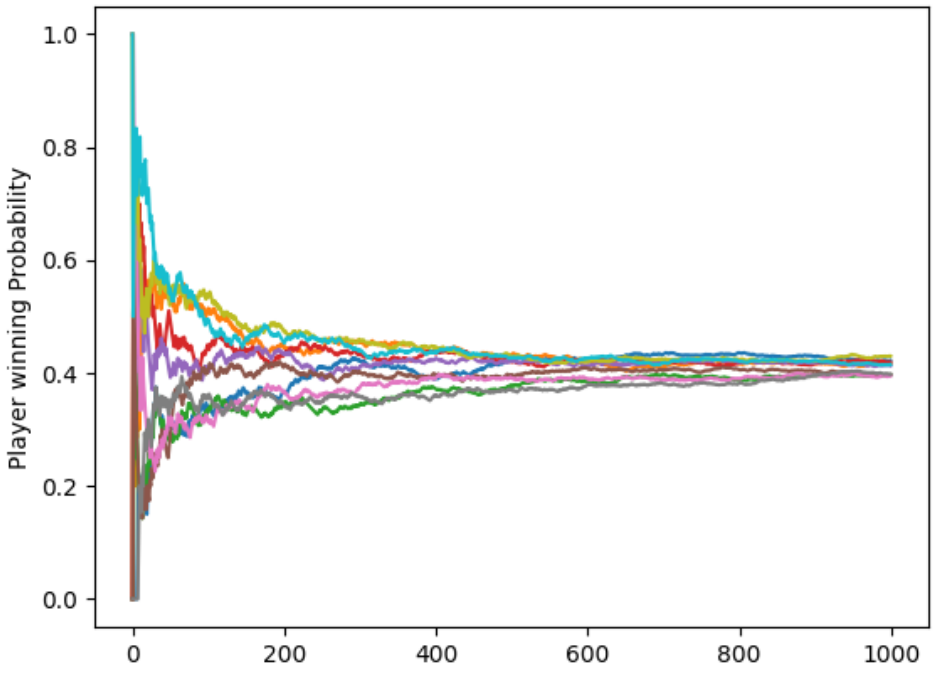

Figure 2: the result of 10 runs of 10^5 games, with both player and opponent using Strategy 2

In Figure 2, we only draw the number of estimated games to up be 1000 in each run, as at the 1000 estimated game, lines already look convergent. Based on Figure 2, we found the results suggest strongly that the player will win over 40% of the time.

We calculated 99.8% CI of the 10 runs of 100000 estimated games when both players used Strategy 2 to see whether this strategy has significantly increased or decreased the player's winning probability, compared to Strategy 1 which both randomly placed the dice. With the calculation by Python, we found the 99.8% CI to be $(0.4182, 0.4206)$. Since the 99.8% CI does not have any intersection with Strategy 1's 99.8% CI (0.4410, 0.4437), we conclude that Strategy 2 reduces the player's winning probability compared to Strategy 1.

2.3 Strategy 3: Avoid stacking large numbers in the same column

Another strategy we come up with is avoiding stacking large numbers in the same column. If all columns already have this large number, place the dice randomly on an empty square on the player's side of the board and check whether any elimination would happen.

To estimate the mean probability of winning while both players are using this strategy, we again do 1000 runs of 1000 simulated games. We assumed that both player and opponent set large numbers to be large or equal to 3, meaning that if they rolled $3, 4, 5$ or 6, they will need to check whether any column already has such number. By using the Central Limited Theorem, we could assume

from the obtained data that the player's winning probability is approximately normally distributed with the mean being around 0.46.

We then use Python to do 10 runs of 100000 simulated games and calculate the 99.8% CI to obtain precise estimates of the winning probability. With calculation, the 99.8% CI is (0.4599, 0.4631), which is larger than both Strategy 1 and 2's confidence intervals. It indicates that the player's winning probability increases when both player and opponent use this strategy and both set the large number as 3.

To further check whether this strategy increases the player's winning probability when both player and opponent use this strategy and set larger numbers differently, we continue to calculate the 99.8% CI of the player's winning probability (Table. 1) from 10 runs of 100000 simulated games using Python

Table 1: Probability of the player winning with various large numbers against the opponent with another large number

Player	Opponent			
	3	4	5	6
3	(0.4599, 0.4631)	(0.1166, 0.1190)	(0.0209, 0.0218)	(0.0042, 0.0047)
4	(0.8332, 0.8354)	(0.4504, 0.4540)	(0.1692, 0.1716)	(0.0600, 0.0618)
5	(0.9620, 0.9629)	(0.7489, 0.7520)	(0.4395, 0.4429)	(0.2283, 0.2306)
6	(0.9901, 0.9906)	(0.8944, 0.8970)	(0.6636, 0.6652)	(0.4334, 0.4368)

Based on the table, we found that if the large number the opponent chooses is larger than the large number the player chooses, the player's winning probability will decrease significantly, compared to playing the game by randomly placing the dice. When the large number chosen by the opponent is smaller than that chosen by the player, the player's winning probability will increase significantly, compared to playing by randomly placing the dice.

2.4 Strategy 4: Combine Strategy 2 and 3

We suspect how the player's winning probability will change if both player and opponent use Strategy 2 and Strategy 3 together. It means that both players will need to check if any chance to remove dice on the other's board first when they rolled the dice. If there is a dice to remove and they have an empty square on the corresponding column on their own side, they will choose to remove the other's dice. If no empty square or no same dice on the other's board, the player will avoid stacking large numbers in the same column. If can't randomly place the dice.

We calculate the 99.8% CI of the player's winning probability from 10 runs of 100000 simulated games using Python. Table. 2 shows different winning probabilities of the player when the player and opponent choose different values as large numbers:

Table 2: When using both strategies, the probability of the player winning with various values as big numbers

Player	Opponent			
	3	4	5	6
3	(0.3842, 0.3864)	(0.3084, 0.3114)	(0.2405, 0.2425)	(0.1880, 0.1903)
4	(0.4647, 0.4665)	(0.3903, 0.3922)	(0.3094, 0.3123)	(0.2420, 0.2451)
5	(0.5572, 0.5591)	(0.4807, 0.4833)	(0.3992, 0.4019)	(0.3230, 0.3251)
6	(0.6392, 0.6429)	(0.5682, 0.5721)	(0.4856, 0.4903)	(0.4089, 0.4120)

Compared to the confidence interval of the player's winning probability that both the player and opponent play the games randomly, we found that when the player chooses a large number that is smaller or equal to the opponent's large number, the player's winning probability will significantly decrease. If the player chooses a large number that is larger than the opponent's, the player's winning probability will significantly increase.

3 Comparison between Each Strategy

To compare which strategy is better, we hope to investigate the player's winning probability when the player and opponent are using different strategies while playing the game. For each condition, we calculate the 99.8% CI of the player's winning probability from 10 runs of 100000 simulated games using Python as well, and compare obtained confidence intervals with each other.

Table 3: Compare different strategies

Player	Opponent		
	Strategy 1	Strategy 2	Strategy 3
Strategy 1	(0.44099, 0.44369)	(0.26334, 0.26466)	(0.99508, 0.99552)
Strategy 2	(0.59839, 0.60079)	(0.41822, 0.42056)	(0.99701, 0.99732)
Strategy 3	(0.00197,0.00225)	(0.00110, 0.00127)	(0.45985, 0.46305)

Due to the long running time, we exclude the comparison between Strategy 4 and other strategies. We found that when player uses strategy 3 against the opponent who employs strategies 1 or 2, the winning probability for player is significantly low.In the similar situations, the winning probability for player is significantly higher if the opponent uses strategy 3.

4 Appendix

4.1 Why using 99.8% CI

According to the central limit theorem, the estimated winning probability should be close to normally distributed. We find the histogram we got is approximately normal. Then, we can use the fact that the normal distribution implies symmetric the mean to estimate the probability that all 10 of our estimates are on one side of the true probability. The probability is

$$\frac{2}{2^{10}} = \approx 0.001953125.$$

This is the probability that all of the estimates are greater, or all of the estimates are less than the true probability. Therefore, there is a 99.8% probability that the true probability is between the highest estimate and the lowest estimate.

4.2 Code for this project

We've uploaded all Python and Sage codes on GitHib:
https://github.com/Chensss0816/Math381