Rook in the Menagerie

Kai Maffucci
Ed Lamagna (mentor)

Abstract

Rook is a card game in the Bridge-Whist family. The principal version uses a special deck of 41 playing cards. We examine bidding and playing strategies through computer simulations. These are based upon a colorful cast of amusing characters appearing in Victor Mollo's Bridge in the Menagerie book series.

Introduction to Rook

Rook is a trick-taking card game in the Bridge-Whist family. In four-player variants, two teams of two face off against each other. Rook employs a unique 57-card deck, unlike a traditional 52-card deck of playing cards. A Rook deck has no face cards, and instead has numbers 1-14 as its ranks. Instead of suits, it employs colors, typically red, yellow, green, and black. It also has an additional Rook ("bird") card, which acts sort of as a wildcard. The Rook deck was constructed in this way because traditional playing cards were seen as "un-Christian" due to associations with gambling and card magic, spurring Rook's creation. It is popular primarily in Appalachia, but it also has footholds in Michigan and upstate New York.

For this project, we looked specifically at Tournament Rook, also known as Kentucky Discard. This is the standard ruleset that comes with the instructions of the game from Hasbro. In Kentucky Discard, card numbers 1-4 are removed from the deck entirely, creating a new 41-card deck. 9-card hands are dealt to each player, and the remaining 5 cards are set aside in a "nest," or "kitty" as it is sometimes called.

With each player seated (and partnerships decided), one player begins bidding at 70 points. The bidding goes around, and players must bid in multiples of 5. The player with the winning bid gets to take the "nest" cards into their hand and set aside a new nest of 5 cards. They also get to decide on what the "trump" color is (the color/suit that always wins). The bidding team also must earn a combined number of points during play that totals or exceeds their bid (or contract, such as in Bridge). Points are earned by taking "tricks" (like Bridge) that have "counter" cards in them. Counters are 5s (worth 5 points each), 10s and 14s (worth 10 points each), and the Rook (worth 20 points, and wins the trick no matter what). One player begins a trick, and players must "follow suit," playing cards of the same color as the leading player, unless they are unable to do so (at which point they may play any card, including a rook or trump). The highest card of the suit led, or the highest trump/rook if played, wins the trick, and that player takes those cards and sets them aside. At the end of all 9 tricks, all the tricks taken are counted up, and if the bidding team fails to make their bid, they lose a negative amount of points equal to their bid, no matter how many points they scored. Otherwise, they earn the total amount of points they took. Then another round is repeated, until a single player reaches 300 points.

Bridge in the Menagerie

I wanted to explore strategies in Rook, and my mentor, Dr. Ed Lamagn, recommended I read the book *Bridge in the Menagerie*, the namesake of the project. This is a book (which spawned a series) about another card game, Bridge (of course), and it features a colorful cast of characters. Among these are Karapet, a cautious and weary player; Papa, a careful calculator who still can fall on bad luck; the Hideous Hog, who is aggressive and not-so-humble; and the Rueful Rabbit, a timid and unpredictable character who occasionally lucks into good moves. To use these characters to explore Rook strategies, I wrote a simulation to automatically run Rook games and implemented these players into the simulation. Their strategies are developed by me, but inspired by how I believe each player may play Rook.

Player Strategies

Each player has a bidding strategy where they analyze their hands and determine a maximum bid. All players add +20 points if they have the rook and additional points based on partner prediction, so they are omitted from the descriptions of the individual bidding strategies. Here is how they determine how many points their partners may be able to score:

1. *trumps_in_hand* = number of trump cards in hand
2. *trumps_elsewhere* = 9 - *trumps_in_hand*
3. For the first *n* whose probability of having *n* trumps in a starting hand with *trumps_elsewhere* trumps in the rest of the deck is ≥50%, add *5n* to maximum bid

They also have strategies for determining what cards they put in the nest. Finally, they have strategies for which cards they play during each trick. There are two separate strategies in this case: one for if they are leading the trick, and one for if they are not. We tested their strategies over 100,00 games, and they were as follows.

Karapet

Bidding: +10 pts each 10, 14; +5 pts each 11-13
Nest: 5 lowest non-trump
If leading:

1. Lowest non-counter, non-trump
2. Lowest non-counter trump
3. Lowest trump
4. Lowest card overall (can be Rook)

Else:

1. Lowest non-counter in lead suit
2. Lowest counter in lead suit
3. Lowest non-counter, non-trump
4. Lowest non-counter trump
5. Lowest trump counter
6. Lowest card overall (can be Rook)

Papa

Bidding: 10 pts for 10s, 14s; 5 pts for 11-13s; +5 pts for each 2 of 5-9s
Nest: 5 lowest non-trump
If leading: highest of all cards (regardless of suit)
Else:

1. Lowest card in hand in lead suit still higher than highest card in current trick
2. Lowest card in suit overall
3. Highest trump
4. Rook
5. Lowest non-counter
6. Lowest card overall

Hideous Hog
Bidding strategy: +10 pts for 10s/14s; 5 pts for 11-13s; +5 for 5-9s
Nest: 5s, then lowest cards
If leading: highest of all cards (regardless of suit)
Else:

1. Rook
2. Highest lead
3. Highest trump
4. Lowest non-counter
5. Lowest counter

Rueful Rabbit
Bidding strategy: like Karapet, then +0 to 25 pts
Nest: 5 random cards
If leading: random card
Else: randomly picks another player's strategy

Results

After running 100,000 games, these were the results. An important note is that while Rook is a partnership game, the player strategies are extremely simple. Consequently, they do not cooperate based on who their partner is: they all play for themselves.

Individual Players	**Games**
Karapet	54.90%
Papa	54.82%
Hideous Hog	53.11%
Rueful Rabbit	37.16%

Individual Players	**Tricks**
Papa	29.46%
Hideous Hog	26.24%
Rueful Rabbit	22.47%
Karapet	21.84%

Partnerships	**Games**
Karapet-Papa	21.42%
Papa-HH	20.72%
Karapet-HH	20.70%
Papa-RR	12.76%
Karapet-RR	12.71%
HH-RR	11.70%

Partnerships	**Tricks**
Papa-HH	18.26%
Papa–RR	17.20%
Karapet–Papa	16.93%
HH–RR	16.37%
Karapet–HH	16.24%
Karapet–RR	14.98%

Trimester 1	**Win Rate**	**Trimester 2**	**Win Rate**	**Trimester 3**	**Win Rate**
HH	44.30%	**Papa**	35.50%	**Karapet**	43.14%
Papa	34.10%	**RR**	24.30%	**RR**	26.52%
RR	16.59%	**HH**	22.84%	**Papa**	18.77%
Karapet	5.01%	**Karapet**	17.36%	**HH**	11.57%

Something very interesting happens here: a player's overall win rate does not directly correlate to when tricks are won, or even how many tricks they won in the first place.

Analysis - Part 1

The results of the simulation seem accurate, but they are the result of code, after all. To verify that the results of my simulation are accurate, I performed some combinatorial analysis. My thought process was as follows:

1. What scenarios are arising?
2. What conditions are required for those scenarios to occur?

A few things are obvious: Karapet wins late but loses early, and the Hog vice versa. Papa and the Rabbit's trick win rates seem to be relatively high, but Papa doesn't start to win more until trimester 2. Now I look at how likely it is that each player is to win a trick during each trimester. Each player is set to play one kind of card first if they have it: Karapet will play a low card, the Hog will play a high card, and Papa will sometimes play a high card. How likely is it that each player has these cards and thus can play them?

This changes throughout the game. For instance, if Karapet is to play a low card in the first trick, he only needs 1 in his starting hand. If he is to play a low card on trick 4, the beginning of trimester 2, then he needs 4 low cards to keep playing them. This same logic applies to all players and their strategies. Therefore, I devised a formula for determining each player's scenario's likelihood.

I decided to make great use of the hypergeometric distribution formula, since it made putting things together simpler. One may be able to simplify the math with some algebra, but since I would be writing a script to perform these computations, it was unnecessary. More important was that I could understand it.

If I'm determining how many cards of a specific type (or "target cards"), such as low cards, are in a player's hand, I first need to determine the chance that a specific number of target cards are removed from the deck via the nest.

Initial deck size (population size): d_0
Total target cards (number of successes in population): t_0
Total cards removed from deck (sample size): d_r
Target cards taken out in sample (observed successes): t_r

$$\frac{\binom{t_0}{t_r}\binom{d_0-t_0}{d_r-t_r}}{\binom{d_0}{d_r}} = hyper(d_0, t_0, d_r, t_r)$$

Only then may I determine the chance that a certain number of target cards appear in a player's hand, having accounted for the cards removed via the nest.

$$\begin{aligned}&\text{Population size: new deck size, } d_0 - d_r = d_n\\&\text{Successes in pop: remaining target cards, } t_0 - t_r = t_n\\&\text{Sample size: hand size, } h\\&\text{Successes in sample: target cards in hand, } t_h\\&hyper(d_0 - d_r, t_0 - t_r, h, t_h) = hyper(d_n, t_n, h, t_h)\end{aligned}$$

Putting it together, we have the precise probability that t_h target cards will appear in a player's hand if exactly t_r target cards are removed from the deck when the nest is drawn and set aside.

$$hyper(d_0, t_0, d_r, t_r) \times hyper(d_n, t_n, h, t_h)$$

However, this is for an extremely specific scenario, and not very useful on its own. To be more useful, I sum over the possible number of target cards that could be removed from the nest. Say there are 21 target cards, and the nest is 5 cards. There could be anywhere from 0-5 target cards removed from the deck via the nest, and I need to sum all of these probabilities to get the total probability. Although if there are fewer than 5 total target cards (say, 4 10s), only 0-4 can be removed via the nest.

$$\sum_{t_{ri}=0}^{min(d_r, t_0)} hyper(d_0, t_0, d_r, t_{ri}) \times hyper(d_n, t_0 - t_{ri}, h, t_h)$$

Now we have the probability that exactly t_h cards appear in a player's hand, regardless of what happens with the nest. This could still be more useful, though. Karapet needs 1 low card in his starting hand to play one on trick 1, but in this scenario, it doesn't matter what the other cards are. My formula, as it stands above, only accounts for *exactly* t_h cards appearing, but I want it to account for t_h cards *or more*. To do that, I sum again over the probabilities for each number of cards t_h or higher appearing in a player's starting hand. Of course, no more target cards can appear in a starting hand than there are in the deck, so we only sum up to the lesser of the two.

$$\sum_{t_{hi}=t_h}^{min(h, t_n)} \left(\sum_{t_{ri}=0}^{min(d_r, t_0)} hyper(d_0, t_0, d_r, t_{ri}) \times hyper(d_n, t_0 - t_{ri}, h, t_{hi}) \right)$$

This final formula gives the probability that $\geq t_h$ target cards appear in a player's starting hand, regardless of what happens with the nest.

Analysis - Part 2

With my newfound tools, I can now provide a proper analysis of who is most likely to win the trick each trimester.

The probability of a player having ≥1 low card(s) (5-9s) in their starting hand is 99.92%, and 99.95% for high cards (10-14s + rook). Karapet will prioritize a low card if he has it, and HH will prioritize a high card. They are both extremely likely to be able to satisfy their strategies, which explains why HH wins often in T1 and Karapet loses often in T1, as the low cards will always lose to the high cards. Papa is just as likely to have a high card, giving him a 50/50 shot of beating the Hog in a vacuum; however, he will only play that card if he goes after the Hog and sees his card. Remember, Papa optimizes: he'll only play a card higher than the other cards in the current trick. This gives Papa realistically a *quarter* of the chance of beating the Hog. This makes sense, as Papa falls between the Hog and Karapet in the simulation. The Rabbit's randomness complicates things, although it serves as a good baseline/mean performance metric between the players. Karapet falling behind him in the T1 shows just how poorly Karapet performs early in the game.

For Karapet and HH to keep playing at the start of T2 the way they prioritize according to their strategies, Karapet needs 4 low cards and HH 4 high cards. They will have played one in each of the previous tricks, plus they need one for the current trick. The probability of Karapet having ≥4 low cards is 74.79%, and the probability that HH got ≥4 high cards is 79.84%. Since Papa doesn't play high cards as often as the Hog does, I estimated that he would need half as many high cards as the Hog to keep playing the way he does. From this, I glean that the chance Papa got ≥2 high cards is 99.20%. Of course, this is pretty high, and explains why Papa starts beating the Hog: the Hog has begun to run out of high cards at a quicker rate than Papa has.

The facts of the matter are fully exacerbated in T3. The chance Karapet got ≥7 low cards is 5.45%, and the chance that HH got ≥7 high cards is 7.55%. Neither player can play how they normally would, so they play oppositely: Karapet is playing high cards, and the Hog is playing low cards. The chance that Papa had ≥4 high cards is 79.84%, so while this is good, he stands no chance against Karapet. The Rabbit's surprisingly high trick win rates in T2 and T3 suggest that a mix of strategies may be best, but his actual overall win rate is terrible.

Conclusion

This project proved a few universal factors about playing Rook: that playing conservatively leads to early losses and later winnings, playing aggressively leads to early wins and later losses, but that taking a moderate method (perhaps leaning in one direction or the other) is likely the best approach. While these player strategies were extremely simple, building the simulation provides a framework for potential future work.

If I can work further on the subject in the future, there are a few things I would look at. First, I would determine how players may cooperate with their partners more. Second, I would devise more complicated player strategies, potentially involving card-counting or complex orders of play. Finally, given the resources, I would investigate how machine learning may be able to generate an AI that is able to play the game better than any hard-implemented strategy.

There is a lot to explore, since the only other mathematical work I could personally find is Steve Huddleston's guide, listed in the bibliography. Hopefully, this was of interest: any further inquiries can be sent to kai.maffucci@gmail.com. The code for the project can be found at https://github.com/KaiMaffucci/rooksim, but I make no promises with respect to how good a condition it is in.

Bibliography

Mollo, Victor, and Bill Buttle. Bridge in the Menagerie. Master Point, 2013.

Mollo, Victor. Bridge in the Fourth Dimension. Batsford, 2014.

"Winning at Rook." stevehuddleston.tripod, stevehuddleston.tripod.com/sitebuildercontent/sitebuilderfiles/warebook.pdf. Accessed 1 Apr. 2025.

Special thanks to Camille Woodring for helping with some code

Rook card image on cover slide: Bruckner, Jared. Index of Rook Cards, jaredbruckner.com/NewestRookWebsite/Rooks/IndexOfRooks.asp. Accessed 31 Mar. 2025.