

Developing Strategies for the Bidding Card Game 'Diamonds' with GenAI

Prathama Pratihar

March 26, 2024

Contents

| | | |
|----------|---------------------------------|----------|
| 1 | Introduction | 1 |
| 2 | Problem Statement | 1 |
| 3 | Rules | 2 |
| 4 | Teaching Gen AI the game | 3 |
| 5 | Iterating upon Strategy | 4 |
| 6 | Code | 4 |
| 7 | Conclusion | 5 |

1 Introduction

This report explores the outcomes of prompting GenAI to develop a computer program utilizing optimization strategies to compete in the Diamonds card game against human players. Diamonds, known for its strategic depth and bid-based gameplay, serves as an ideal testing ground for AI-driven decision-making. Through this simulation, we aim to assess the effectiveness of AI-powered strategies in navigating the complexities of Diamonds gameplay.

2 Problem Statement

To write a report or an essay with the result of prompting GenAI to create a computer program that uses some optimizing strategy to play the game of Diamonds against you.

3 Rules

The basic rules for playing Diamonds:

- **Setup:**
 - Use a standard 52-card deck.
 - Deal 10 cards to each player for games with 2-3 players, or 7 cards for games with 4-6 players.
 - Place the remaining cards in the draw pile.
- **Objective:**
 - The goal is to earn the most points by collecting diamond cards and utilizing diamond tokens effectively.
- **Bidding Phase:**
 - Players bid for the number of diamond cards they think they can collect in tricks. Bids must be between 0 and the number of cards in their hand.
 - The player with the highest bid becomes the "declarer" for the round.
- **Gameplay:**
 - The player to the left of the declarer leads the first trick.
 - Players must follow suit if possible. If not, they can play any card.
 - If a player cannot follow suit and plays a diamond, they earn a diamond token.
 - The player who wins a trick containing at least one diamond collects those diamonds and places them in front of them, along with any diamond tokens they've earned.
 - The winner of each trick leads to the next trick.
- **Scoring:**
 - At the end of the round, players score points based on:
 - * The number of diamond cards collected in tricks.
 - * The number of diamond tokens earned.
 - * Additional bonuses provided by certain cards.
 - Unused diamond tokens can be converted into points.
- **End of Round:**
 - After all tricks have been played, the next player becomes the declarer for the next round.

- **End of Game:**
 - The game typically consists of multiple rounds, with the player accumulating the most points over all rounds declared the winner.
- **Special Cards:**
 - Some cards have special abilities, such as allowing players to exchange cards with the draw pile or steal diamonds from opponents.

4 Teaching Gen AI the game

Bidding Phase:

- Player 1 assesses their hand and decides to bid 3 diamond cards.
- Player 2 believes they have a strong hand and confidently bids 5 diamond cards.
- Player 3 (the AI Player) carefully evaluates their cards and cautiously bids 4 diamond cards.
- In the bidding phase, each player strategically evaluates their hand and predicts how many diamond cards they can collect during the round. Player 2 makes a bold bid, while Player 3 (the AI Player) chooses a more conservative approach.

Gameplay:

- Player 1 leads the first trick by playing the 8 of Diamonds.
- Player 2 follows suit and plays the Jack of Diamonds.
- Player 3 (the AI Player) decides to play the 6 of Hearts, as they lack diamonds and want to save their higher cards for later tricks.
- Player 1 plays the 4 of Hearts.

Scoring:

- At the end of the round, Player 2 successfully collects 3 diamond cards, earning 3 points.
- Player 3 (the AI Player) manages to collect 2 diamond cards, earning 2 points.
- Player 1, unfortunately, collects only 1 diamond card, scoring 1 point.
- Players score points based on the number of diamond cards they collect during tricks. In this round, Player 2 earns the most points, followed by Player 3 (the AI Player) and Player 1.

5 Iterating upon Strategy

On asking ChatGPT to suggest some strategies to consider while playing Diamonds, it gave the following response: Certainly! Here are some strategies to consider when playing Diamonds:

- Evaluate Hands: Assess your hand's strength before bidding.
- Bid Wisely: Make cautious bids based on hand strength and potential.
- Focus on Diamonds: Prioritize winning diamond tricks for points.
- Acquire Tokens: Strategically earn diamond tokens for scoring advantages.
- Utilize Special Cards: Time the use of special cards for maximum benefit.
- Adapt to Opponents: Adjust strategy based on opponents' moves and bids.
- Manage Risk: Balance risk in trick-taking to avoid penalties.
- Exploit Bonuses: Seize opportunities for bonus points through cards and tokens.
- Seek Consistency: Strive for consistent scoring across rounds.
- Remain Flexible: Stay adaptable to evolving game situations and opponents' strategies.

6 Code

Prompt Given - Can you write a basic Python code that simulates a third player implementing these strategies? ChatGPT:

```
import random
```

```
class Player:
```

```
    def __init__(self, name):  
        self.name = name  
        self.hand = []
```

```
    def add_to_hand(self, card):  
        self.hand.append(card)
```

```
    def play_card(self, valid_cards):  
        # Simulate the decision-making process of the player  
        # Implement the strategies here  
        # For simplicity, this code just randomly selects a valid card from the  
        return random.choice(valid_cards)
```

```

def main():
    # Example initialization of the game
    player1 = Player("Player 1")
    player2 = Player("Player 2")
    player3 = Player("AI Player") # AI Player implementing the strategies

    players = [player1, player2, player3]

    # Example dealing cards to players
    deck = ["2D", "3D", "4D", "5D", "6D", "7D", "8D", "9D", "10D", "JD", "QD"]
    random.shuffle(deck)
    for i, card in enumerate(deck):
        players[i % 3].add_to_hand(card)

    # Example trick-taking phase
    tricks = 5
    for trick in range(tricks):
        trick_cards = []

        for player in players:
            # Simulate valid cards the player can play (just for demonstration)
            valid_cards = player.hand[:] # Assuming all cards are valid for
            selected_card = player.play_card(valid_cards)
            player.hand.remove(selected_card)
            trick_cards.append(selected_card)

        print("Trick", trick + 1, ":", trick_cards)

if __name__ == "__main__":
    main()

```

7 Conclusion

Some parts of the code generated by ChatGPT weren't up to the mark and needed some corrections. Overall I was surprised by how much it knew about the game without me needing to tell the rules.