Evaluating Gin Rummy Hands Using Opponent Modeling and Myopic Meld Distance


1New York University College of Arts and Science, phoebe.goldman@nyu.edu
2University of Minnesota Duluth, knuts983@d.umn.edu
3Carnegie Mellon University, rmahtab@andrew.cmu.edu
4University of Minnesota Twin Cities, malon559@umn.edu
5University of Minnesota, muel0053@umn.edu
6SIFT, [rfreedman,jmueller]@sift.net

Abstract

Gin Rummy is a popular two-player card game involving choices to draw and discard cards to form sets of matching cards. Unlike other popular games such as Chess, Poker, and Go, there is little formal artificial intelligence research about how to make good decisions when playing Gin Rummy. In this paper, we develop an agent that plays Gin Rummy through a combination of known and expected card values, modeling the opponent to predict their cards of interest, and a conservative approach to assessing when to end the hand. In addition to discussing our observations about Gin Rummy that inspired our agent’s design and how the agent works, we evaluate the relative importance of various features employed by our agent by competing agents which implement various subsets of those features.

1 Introduction

Gin Rummy is a two-player card game that, despite having simple rules, introduces the opportunity for complex strategic decision-making based on both hidden information and randomness. Unlike other incomplete information games such as Poker, there has been little scholarly study about playing Gin Rummy. Although Poker has a betting element that is absent in Gin Rummy, it contains other interesting challenges not found in Poker such as a sequence of actions per turn and gambling whether to terminate the round in confidence of having a better score than the opponent. The absence of existing strategies in the literature, the simplicity of the game rules, and the potential to implement complex reasoning makes Gin Rummy an intriguing problem space for artificial intelligence research.

Using the limited information available to a player, our approach uses approximate statistical analysis of potential hand outcomes and analysis of the opponent’s strategy given observations of their actions each turn. In addition, we applied domain knowledge from our experience playing Gin Rummy to introduce a conservative, but surprisingly simple and effective, strategy that decides to end the round only when the player has the best possible score (because the objective is to minimize a non-negative score, this is 0). The former elements synergize with this strategy to prepare for the case that the opponent ends the game first because there is a penalty for ending the game while having a greater score than the other player.

In this paper, we present our Gin Rummy-playing agent, its components, and further details on the above features. Section 3 first reviews related research about gameplaying agents, and then we present how to play Gin Rummy with our insights in Section 2. Following this background, Section 4 introduces each component of our agent in more depth. To study how our agent performs, we perform experiments involving play against variations of itself in Section 5. Based on the results, we provide discussions and propose future work in Section 6.

2 Game Rules

Gin Rummy is a two-player card game played in a series of rounds. At the beginning of each round, each player is dealt a ten-card hand from a standard 52-card deck. One additional card is placed face up as the initial discard pile. The players alternate turns making two or three decisions in sequence:

1. whether to draw the top card from the discard pile (face-up) or from the deck (face-down),
2. which card to discard from their hand, and
3. whether to knock and end the round (if applicable).

Each card has a point value, with aces worth one point, pip cards worth their number, and face cards worth ten points. Cards are worth no points if they are grouped into melds of at least three cards, either all matching rank (sets) or forming a straight within the same suit (runs). The sum of the cards’ point values in a player’s hand is called their deadwood. If a player has at most ten deadwood points at the end of a their turn, they may knock to reveal their hand and declare their melds. Their opponent takes one final turn, reveals their hand and declares their melds; the opponent is then given the opportunity to lay off unmelded cards into the knocking player’s melds. After laying off, the player with fewer
deadwood points wins the round and scores points equal to the difference between the two players’ deadwood points. The knocking player loses ties. Under the American rules, a player who knocks with zero deadwood points goes gin and scores an additional 25 points; if the winner was not the player who knocked, then they undercut their opponent and score an additional 25 points. At the end of the round, a player wins if they have accrued at least 100 points throughout the game—if no player has reached 100 points, then they play another round.

## 2.1 Observable Information

The information available to each player is asymmetric because the hand of cards reveals their contents exclusively to the hand’s owner. Figure 1 displays what information is available to each player during a game of Gin Rummy. In general, a player can observe the ten cards in their personal hand as well as all the cards (and their order) in the discard pile. Neither player knows the order of the cards that remain in the deck, and the degree to which a player knows which cards are in the deck depends on how much they know (and remember) about the cards in their opponent’s hand.

On a player’s turn, the new information that they observe is limited. Drawing the top card of the discard pile does not reveal any new information to that turn’s player, but it is their final opportunity to add the card to their hand before it is gone for the remainder of the game. On the other hand, drawing the top card from deck reveals another card that was not yet seen; this tells the player that it was not a card that is hidden in the opponent’s hand. The player then knows that such a card is no longer in the deck as well.

During the opponent’s turn, a player observes many new pieces of information: whether the opponent chose to draw or ignore the top face-up card in the discard pile, which card the opponent chose to discard, and whether the opponent chose to knock. In this case, the opponent’s choice to do something is as informative as the opponent’s choice to not do something because either action implies information about their unique knowledge/observations.

### Figure 1: Observable and known information to each player during a game of Gin Rummy. “Turn player” and “Opponent” are relative to whoever is acting on the current turn.

<table>
<thead>
<tr>
<th>Turn Player</th>
<th>Both</th>
<th>Opponent</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Cards in turn player’s hand</td>
<td>• Cards in discard pile&lt;br&gt; • Order of cards in discard pile&lt;br&gt; • Who discarded each card&lt;br&gt; • Who chose to take which card from discard pile&lt;br&gt; • Cards in each player’s hand that were drawn from discard pile and not discarded again&lt;br&gt; • Who chose to not take which card from discard pile&lt;br&gt; • Number of cards remaining in deck&lt;br&gt; • No player knocked yet (if hand is ongoing)</td>
<td>• Cards in turn player’s strategy/preferences&lt;br&gt; • Order of cards in deck&lt;br&gt; • Other player’s strategy/preferences</td>
</tr>
</tbody>
</table>
the opponent’s state (strategy in this case, but could also be the cards in their hand) (Bard and Bowling 2007). Because Gin Rummy has a greater focus on matching cards than out-ranking an opponent (or bluffing) to win, we adjust our uses of opponent modeling and hand evaluation accordingly.

The latter approaches are similar to an application of probabilistic plan recognition to hypothesize opponent strategies in the videogame StarCraft (Kabanza et al. 2013). In general, plan, activity, and intent recognition research provide opportunities to model the other agents based on their actions. These models can identify the opponents’ way-points of interest in first-person shooter videogames (Tastan, Chang, and Sukthankar 2012) as well as infer sets of goals over which the observing agent may compute responsive plans for interaction (Freedman and Zilberstein 2017). Although we do not use recognition algorithms, our agent similarly uses the opponent model to affect its reasoning over actions to take while playing Gin Rummy.

### 4 Methods

Our Gin Rummy-playing agent consists of three main components, each used for at least one decision-making task: Myopic Meld Distance estimation to identify potentially valuable cards for the agent’s hand, Opponent Modeling to predict potentially valuable cards for the opponent’s hand, and a conservative knocking strategy. We describe how each component works below. For the majority of the in-game decisions, our agent uses these components to compute a hand evaluation function \( r_t \) based on the current game state \( S \).

**Definition 1** A game state \( S \) is a tuple

\[
(H_t, H_o, D, t, D_i, U)
\]

where \( H_t \) is the set of cards in the agent’s hand, \( H_o \) the set of known cards in the opponent’s hand, \( D \) the set of cards in the discard pile, \( t \in D \) the face-up card on top of the discard pile, \( D_i \subseteq (D \cup H_o) \) the set of cards the agent has discarded, and \( U \) the set of unrevealed cards either remaining in the face-down pile or hidden in the opponent’s hand.

**Definition 2** A hand evaluation function \( r_t \) maps \((H, S)\) tuples to positive real numbers, where \( H \) is a hand to evaluate in game state \( S \). \( r_t(H_1, S) < r_t(H_2, S) \) denotes that \( H_1 \) is more apt to win than \( H_2 \) from game-state \( S \).

We note that, for any game state \( S \), \( H_t \cup H_o \cup D \cup U = \{A\bullet, 2\bullet, \ldots, K\bullet\} \) is the set of all 52 cards in the game.

#### 4.1 Agent Assumptions

We assume that our opponent will never discard cards which they have chosen to draw face up; that is, we assume that all known cards in \( H_o \) are permanently inaccessible to our agent, and thus that we can never achieve any melds containing those cards. A survey of games by human and computer players could potentially validate or reject this assumption, were such data available, and future research might seek to demonstrate using tests whether the removal of this assumption decreases or improves an agent’s win rate.

We also assume that all cards \( c \in U \) are equally likely to be drawn; for random variable \( \Gamma \) as the top card of the deck,

\[
P(\Gamma = c) = \frac{1}{|U|}
\]

A more sophisticated agent might alter its expectation of drawing particular cards based on inferences about the contents of its opponent’s hand based on their behavior; for example, noticing that an opponent has picked up both \( 2\bullet \) and \( 2\bullet \) from the face-up pile could reduce the projected probability of drawing either \( 2\bullet \) or \( 2\bullet \) as the opponent is likely to already have one or both of those cards.

When counting the deadwood in a hand, there might be multiple sets of melds that yield the same deadwood points for the hand. However, of all the meld sets that derive some hand’s minimum deadwood points, we choose one arbitrarily for computation assuming no loss of generality. We discuss the implications of this assumption in Section 6.

Whenever an opponent discards a card, we assume that they have no intention of forming any melds which would be improved by that card. Likewise, if the opponent decides to draw the top card from the deck instead of \( t \), we assume that the opponent has no intention of forming a meld which would be improved by \( t \). These assumptions allow us to quickly prune the potential melds that our opponent could make, leading to more reasonable predictions when the assumption is accurate. We decided this improvement is worth the trade-off of inaccurate predictions when our opponent passes or discards cards that would fit in a meld they are pursuing—we believe such moments are rare. A more conservative agent would need only to prune those melds which are inaccessible as a result of excluding such cards, leaving as possibilities those which are improved by those cards but do not depend on them. Future research might seek to determine whether such an agent performs better or worse than one which makes this assumption.

#### 4.2 Choosing What to Draw and Discard

Our agent considers whether to draw the face-up or face-down card at the same time that it chooses which card to discard using an evaluation function \( r_m \), which computes a positive real number representing the expected deadwood value of a given ten-card hand in a particular game state. \( r_m \) includes the weighted sum of two functions \( r_m \) and \( r_o \), which Sections 4.3 and 4.4 respectively describe in detail. \( d_{\min}(h) \) is the minimum amount of deadwood in hand \( h \) after removing its valid melds, and \( w_m \) and \( w_o \) are tunable weights.

\[
r_m(h) = d_{\min}(h) - w_mr_m(h) - w_or_o(h)
\]

We first explain how the agent determines the value of drawing the face-up card in the discard pile. Adding the face-up card to the agent’s hand yields the set \( H_t \cup \{t\} \) of eleven cards. Because the taken face-up card cannot be discarded on the same turn, the agent only considers the \( |H_t| = 10 \) choices for which card to discard—our agent ignores the case of drawing and discarding the deck’s top face-down card during this stage of the computation. For each of the ten possible ten-card hands resulting from discarding a card, our agent runs the hand evaluation function in Equation 2 above to score them. This takes into account the deadwood of the hand, the potential for a hand to create
new melds, and the potential for a hand to disrupt the opponent’s play or be laid off onto opponents melds. The agent then takes the minimum of these scores and assigns it as the value for the face-up card $t$. This is because the agent will choose to discard the card that minimizes the hand evaluation function $r_c(H_t)$ after taking $t$. Mathematically, $R(c, H_t)$ is the value of $H_t$ after drawing $c$:

$$R(c, H_t) = \min_{x \in U} \{ r_c((H_t \cup \{ c \}) \setminus \{ x \}) \}. \quad (3)$$

The calculation of the value for the face-down card is similar, except the drawn card may be discarded:

$$R'(c, H_t) = \min_{x \in \{ (H_t \cup \{ c \}) \setminus \{ x \} \}} \{ r_c((H_t \cup \{ c \}) \setminus \{ x \}) \}. \quad (4)$$

We then take the expected value of this modified function $R'(c, H_t)$ over all the cards in $U$:

$$R_f = \sum_{c \in U} P(\Gamma = c) R'(c, H_t) = \frac{1}{|U|} \sum_{c \in U} R'(c, H_t). \quad (5)$$

The simplification is true because of our assumption in Equation 1 that $P(\Gamma = c)$ is uniformly distributed.

The agent finally compares $R_f = R(t, H_t)$ and $R_f$, choosing the actions that lead to the lower score. If the face-up card is chosen, the agent draws it, then discards the precomputed card associated with the best hand containing the face-up card. If the face-down card is chosen, the agent decides which card to discard by evaluating the eleven possible result hands as in equation 4. As deciding whether to draw the face-up or face-down card has already computed the best discard associated with each possible draw in the face-down pile, our agent’s performance might be improved by caching those best discards. Instead, we rerun that computation for the drawn card; as yet, such performance optimizations have not been a serious concern.

### 4.3 Myopic Meld Distance

Many hands have the same naïve deadwood value, but given a particular game state, some hands are worth more than others. Consider the following two hands.

1. $2\spadesuit \ 6\spadesuit \ 7\spadesuit \ 9\spadesuit \ J\spadesuit \ K\spadesuit \ 3\spadesuit \ 8\spadesuit \ 9\spadesuit$ (6)
2. $8\spadesuit \ Q\spadesuit \ 6\clubsuit \ 2\spadesuit \ 9\spadesuit \ 7\spadesuit \ 6\spadesuit \ 7\spadesuit \ J\spadesuit$ (7)

Both of these hands have 74 deadwood. However, hand (6) is better because it has many sets of cards that are *almost-melds* while hand (7) has almost no almost-melds. We propose a myopic algorithm to choose our draw and discard cards based on sets in our own hand that are at most one card away from becoming a meld.

**Definition 3** Let $H_t$ be the set of cards that are in the agent’s hand. Then an almost-meld is a set $\Lambda \cup \{ a \}$, where $\Lambda \subseteq H_t$ and $a \in U$. We call $\Lambda$ and $a$ the root and affix of the almost-meld, respectively.

With this definition, assuming $D = H_0 = \emptyset$, we can see that hand (6) has the following ten almost-melds:

- $\{ 6\spadesuit, 7\spadesuit, 8\spadesuit \}, \{ 6\spadesuit, 7\spadesuit, 8\spadesuit, 9\spadesuit \}, \{ 5\spadesuit, 6\spadesuit, 7\spadesuit \}, \{ J\spadesuit, J\spadesuit, J\spadesuit \}, \{ J\spadesuit, J\spadesuit, J\spadesuit \}, \{ 8\spadesuit, 9\spadesuit, 10\spadesuit \}, \{ 7\spadesuit, 8\spadesuit, 9\spadesuit \}, \{ 9\spadesuit, 9\spadesuit, 9\spadesuit \}, \{ J\spadesuit, Q\spadesuit, K\spadesuit \}$

and hand (7) has only two almost-melds:

$$\{ 5\spadesuit, 6\spadesuit, 7\spadesuit \}, \{ 7\spadesuit, 8\spadesuit, 9\spadesuit \}. \quad (9)$$

Let us inspect one of the almost-melds from hand (6): $\{ J\spadesuit, J\spadesuit, J\spadesuit \}$. We can partition this almost-meld into the root $\Lambda = \{ J\spadesuit, J\spadesuit \} \subset H_t$ and the affix $a = J\spadesuit \in U$.

Using this fact, we can now evaluate each hand to better represent its potential to improve. Our evaluation function begins with the deadwood of a hand, and then it reduces this value for hands with better potential. We construct a map where the keys are the cards in the agent’s hand $c \in H_t$, and the values $V(c, H_t)$ are the set of possible affixes for the set of almost-melds in $H_t$ that contain $c$.

Continuing our example, $J\spadesuit$ maps to value $V(K\spadesuit, 2\spadesuit, 6\spadesuit, 7\spadesuit, 9\spadesuit, J\spadesuit, J\spadesuit, K\spadesuit, 3\spadesuit, 8\spadesuit, 9\spadesuit) = \{ J\spadesuit, J\spadesuit, Q\spadesuit \}$ because the three almost-melds in (8) that contain $J\spadesuit$ are $\{ J\spadesuit, J\spadesuit, J\spadesuit \}, \{ J\spadesuit, J\spadesuit, J\spadesuit \}, \{ J\spadesuit, J\spadesuit, Q\spadesuit \}$. Thus the affixes of these almost-melds are $J\spadesuit$, $J\spadesuit$, and $Q\spadesuit$, respectively.

Let $W_{H_t}$ be the set of cards already in a complete optimal meld. We can compute the expected deadwood reduction using this map via the following formula:

$$r_m = \sum_{c \in \{ H_t \setminus W_{H_t} \}} \sum_{a \in V(c, H_t)} P(\Gamma = a) d(c) \quad (10)$$

where $P(\Gamma = a)$ is the probability of drawing a next turn and $d(c)$ is the deadwood value of $c$. Using our assumption that $P(a)$ is uniformly distributed, we can rewrite Equation 10 as

$$r_m = \sum_{c \in \{ H_t \setminus W_{H_t} \}} \frac{|V(c, H_t)|}{|U|} d(c). \quad (11)$$

$r_m$ naïvely computes the reduction in deadwood that we would expect to have after drawing one card. Drawing any card in $V(c, H_t)$ forms a new meld in the player’s hand, which would reduce $d(c)$ to 0 because cards in a meld are no longer scored as part of the hand’s deadwood. This ignores the fact that certain almost-melds for a single hand may not be mutually exclusive, introducing double-counting during expected deadwood reduction.

### 4.4 Opponent Modeling

Knowing what cards are in the opponent’s hand can be key to winning a game of Gin Rummy. If we know the melds in the opponent’s hand, then we can decide which of our cards to discard that would benefit our opponent the least. The agent should only discard cards from their hand that will not go into the opponent’s melds for two reasons:

1. they are very likely to pick up the card, and
2. if the agent is confident that the card will go into an opponent’s meld, then that card is effectively worth zero deadwood if they knock because the agent can lay off the card.

While it is impossible to know all the cards an opponent has in their hand, our agent can make an informed decision based on what it knows in the current state of the game.
First, we want to find the set of all possible cards an opponent could have, including the cards in the agent’s hand since they must discard one card at the end of their turn:

$$C_p = H_i \cup H_o \cup U.$$  

(12)

The agent does not want to discard a card that would potentially meld with the opponent’s hand. Next, we calculate the set of all possible meld sets \(M_p\) with respect to \(C_p\).

To reduce the search space, we take the subset of melds in \(M_p\) that contain at least one card from the agent’s hand:

$$M_1 = \{M \mid M \in M_p \land (M \cap H_i \neq \emptyset)\}.$$  

(13)

We also reduce the search space to only include melds that contain at least one card that we know the opponent has:

$$M_2 = \{M \mid M \in M_1 \land (M \cap H_o \neq \emptyset)\}.$$  

(14)

The reductions in Equations 13 and 14 allow us to filter out melds that the agent cannot affect as well as the melds that the agent has no evidence to support.

We reduce the search space even further by removing melds that we assume our opponent will not try to attain, due to our assumption that an opponent will not try to make any melds that contain a card they either discarded or chose not to take from the discard pile. In particular, this additional pruning applies to three-card sets because all cards have the same rank; thus the opponent would never forfeit a card of a desired set’s rank. For example, if the opponent discards \(J\heartsuit\), then the agent should assume that the opponent is not interested in meld \(\{J\spadesuit, J\clubsuit, J\heartsuit\}\) because \(J\heartsuit\) could join any pair of cards to form yet another set-type meld.

$$M_3 = \{M \mid M \in M_2 \land \text{issSet}(M) \rightarrow (\text{ranks}(M) \cap \text{ranks}(D_o \cup P_o = \emptyset))\}$$

(15)

In the above equation, \(\text{issSet}(M)\) evaluates to true if and only if the set of cards \(M\) is a set-type of meld, and \(\text{ranks}(M)\) abstracts \(M\) to the set of ranks for each card in \(M\). Equation 15 does not prune any melds of the run type because the set of melds \(M_p\) already removed all such melds related to the above assumption. Discarding \(J\spadesuit\) means \(J\heartsuit \notin C_p\), and run-type melds in \(M_p\) without \(J\heartsuit\) are distinct from run-type melds with the card.

With our final set of melds \(M_3\), we compute the deadwood reduction our opponent would gain for each possible card that we could discard from \(H_i\):

$$r_o = \sum_{c \in H_i} \sum_{M \in M_3 \land c \in M} P(M \in M_o) \cdot d(c)$$

(16)

where \(M_o\) is a random variable for the set of melds in the opponent’s hand and \(d(c)\) calculates the deadwood value of \(c\). The probability that a meld is in the opponent’s hand is

$$P(M \in M_o) = \prod_{c \in M} \left\{ \begin{array}{ll} 1 & c \in H_o \\ \frac{1}{|M|} & c \notin H_o. \end{array} \right.$$  

(17)

By design, \(r_o\) represents the expected score by which the opponent’s hand would decrease with respect to the ten cards in the agent’s hand \(H_i\).

### 4.5 Knocking Strategy

Under the American rules, the gin and undercut bonuses are both worth 25 points. To us, the penalty for knocking too early is severe and the reward is small; correctly predicting that it is safe to knock without gin will usually reward a handful of points, but predicting wrong and being undercut gives the opponent one quarter of the threshold to win. Similarly, the reward for going gin is large enough that, even when a player is confident that they would win the hand, it is often worth the risk to play additional turns for the chance to form melds with last few cards and achieve gin. Specifically, the risk is the possibility that their opponent will recover and win first during these additional turns.

For these reasons, we believe that the number of instances when it is correct to knock without gin forms a negligible edge case. As such, our agent only knocks when it has gin. We call this strategy stubborn knocking and compare it to a strategy that knocks as soon as the player has fewer than ten deadwood, which we term quick knocking. The latter greedily knocks as soon as possible while the former lazily knocks once it is not worth waiting any longer.

### 5 Experiments

To assess the performance of specific features of our agent, we devised an experimental setup of five agents. Each agent played in a tournament of 500 games against every agent, including itself. We chose \(n=500\) as it yields a sufficient sample size for hypothesis testing. Each agent has a different combination of the following components: Stubborn Knocking (SK, Section 4.5), Myopic Meld Distance (MMD, Section 4.3), and Opponent Modeling (OM, Section 4.4). Eight different agents are possible by combining these three components, but due to time constraints, only five were tested. Future work could test all eight combinations. Table 1 compares each agent by their features.

### 5.1 SIGRA, The SIFT Gin Rummy Agent

The SIFT Gin Rummy Agent (SIGRA) incorporates all three components: Stubborn Knocking, Myopic Meld Distance, and Opponent Modeling. The agent calculates the MMD for \(H_i\) as well as the Opponent-Modeled deadwood reduction for \(H_o\), and then weights them by 0.85 and 1.0 respectively. These weights were determined by an iterative test, but future work could refine these values. The current deadwood value of the agent’s hand subtracts the sum of these weighted scores, and the hand with the lowest value is chosen. This combination of components results in a strategy that favors cards that can potentially become part of the opponent’s meld, yet also works to reduce the agent’s own overall deadwood by making melds if possible. If the agent cannot, then it chooses a lower-deadwood card. The agent has a stubborn knocking policy, which means it only knocks once it has gin in order to try and earn the extra 25 point knocking bonus, otherwise undercutting the opponent for an additional 25 point bonus.
Table 1: Comparison of Agent Features

<table>
<thead>
<tr>
<th>Agent</th>
<th>SK</th>
<th>MMD</th>
<th>OM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIGRA</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>StubbornMmd</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>MmdOpModel</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Stubborn</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Simple</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

Table 2: Significance of Incrementally Adding Components

<table>
<thead>
<tr>
<th>Component</th>
<th>SE</th>
<th>Z-Score</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SK</td>
<td>0.031</td>
<td>4.972</td>
<td>6.613e-7</td>
</tr>
<tr>
<td>SK + MMD</td>
<td>0.029</td>
<td>0.477</td>
<td>0.634</td>
</tr>
<tr>
<td>SK + MMD + OM</td>
<td>0.028</td>
<td>1.900</td>
<td>0.0575</td>
</tr>
</tbody>
</table>

5.2 StubbornMmd

StubbornMmd is SIGRA without the Opponent Modeling component. The resulting strategy favors cards that will make melds with the agent’s hand, or else they choose a lower-deadwood card. Once again, the agent only knocks once they have gin in order to secure the 25-point knocking bonus or the 25-point undercut bonus.

5.3 MmdOpModel

MmdOpModel is SIGRA with a quick knocking policy instead of a stubborn knocking policy. This combination of components pick cards that could potentially go into an opponent’s hand, while prioritizing cards in their own hand second. The agent knocks as soon as possible in order to get only the point difference between each player’s hand.

5.4 Stubborn

Stubborn only incorporates the stubborn knocking policy. They only draw the face-up card if it immediately becomes part of a meld or creates a new meld. Otherwise, the agent draws the face-down card. To discard, the agent selects the highest-ranked unmelded card. These policies create a play style that neither pays attention to future melds nor to what the opponent is doing. The agent’s stubborn knocking policy refrains from knocking until it has gin.

5.5 Simple

Simple is Stubborn with a quick knocking policy instead of a stubborn knocking policy. We anticipate this agent to be the least successful in the tournament. The 2020 Gin Rummy EAAI Undergraduate Research Challenge provides Simple as a baseline (Neller 2020).

5.6 Results

Table 3 displays the experimental results of the 15 possible matches between each of the five agents in the tournament. Take the bottom-left cell as an example. We interpret it as, “SIGRA beat Simple 74.6% of the time out of the 500 games they played against each other.” Figure 2 visualizes this information as well. Directed edges $v_1 \rightarrow v_2$ indicate the domination relation such that agent $v_1$ won the majority of the games against agent $v_2$. The bottom-left cell from Table 3 is represented in Figure 2 as the edge originating from the node labeled ‘SIGRA’ pointing towards the node labeled ‘Simple’. This illustrates that SIGRA dominated Simple.

Based on Figure 2, we observe that all four agents with at least one of our three strategic components dominated Simple. Furthermore, SIGRA dominated all other agents, but its win rate over StubbornMmd was only 0.51. This result seems to indicate that both agents are evenly matched such that the domination relation may evenly balance closer to 0.5 or potentially flip with more games.

A question that arises from these results is whether adding certain strategic components significantly changes the win rates of our agents. Using Simple as the baseline, we hypothesize that incrementally including the SK, MMD, and then the OM components each improves the win rate. For a controlled environment, we consider each agent’s 500 matches against Simple that lacks all our components.

To test the inclusion of each component, we want to evaluate whether the agent with that component’s performance against Simple is significantly better than the agent without that component’s performance against Simple. To do so, we used a two-sided, two-proportion z-test of significance. We will treat both instances of 500 games as the two samples and the win rates from each tournament as the two proportions being compared. The assumptions of this hypothesis test are satisfied as each game has a randomly generated deck shuffling, each game starts with a randomly generated starting hand, and the individual games’ shuffled decks are independent of one another. The sample size of $n = 500$ games also greatly improves the likelihood that our win rates will be normally distributed across repeated samples according to the Central Limit Theorem.

For the hypothesis test for the SK component, we evaluate whether Stubborn’s performance against Simple is significantly better than Simple’s performance against itself. Let $P_1 = 0.678$, $P_2 = 0.524$, $n_1 = 500$, and $n_2 = 500$. The hypotheses for this test are as follows: $H_0 : P_1 = P_2$ and $H_1 : P_1 \neq P_2$. The results of this significance test can be found in the first row of Table 2. The p-value of this test is $6.613e-7$, indicating quite a low probability that we would randomly observe such a difference in win rates if the null hypothesis was true. Similarly, we perform significance tests for the MMD and OM components, the results of which can again be found in Table 2.

Based on the results in Table 2, there is no evidence that the MMD component contributes to our agent’s ability to play Gin Rummy effectively if it already uses the stubborn knocking strategy. However, when we combine MMD with OM and SK, there is evidence that our agent is able to play Gin Rummy better than with SK alone. There is also strong evidence that using SK instead of quick knocking improves an agent’s ability to play Gin Rummy.

6 Discussion

In the card game Gin Rummy, there are three main decisions that an agent makes per turn: deciding which card to draw, which card to discard, and when to knock. Our proposed
Table 3: Triangular matrix of agent win rates. Win rate represents proportion of games out of 500 that the agent in the left column won against the agent in the top row.

<table>
<thead>
<tr>
<th></th>
<th>Simple</th>
<th>Stubborn</th>
<th>MmdOpModel</th>
<th>StubbornMmd</th>
<th>SIGRA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple</td>
<td>0.524</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stubborn</td>
<td>0.678</td>
<td>0.508</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MmdOpModel</td>
<td>0.674</td>
<td>0.000</td>
<td>0.512</td>
<td></td>
<td></td>
</tr>
<tr>
<td>StubbornMmd</td>
<td>0.692</td>
<td>0.524</td>
<td>1.000</td>
<td>0.502</td>
<td></td>
</tr>
<tr>
<td>SIGRA</td>
<td>0.746</td>
<td>0.578</td>
<td>0.990</td>
<td>0.510</td>
<td>0.520</td>
</tr>
</tbody>
</table>

The authors would like to thank the anonymous reviewers for their helpful feedback, Nathan Ringo for providing computational resources to run the experiments, Todd Neller for organizing the Gin Rummy EAAI Undergraduate Research Challenge, and everyone at SIFT for their support.
References


Tastan, B.; Chang, Y.; and Sukthankar, G. 2012. Learning to Intercept Opponents in First Person Shooter Games.

In IEEE Conference on Computational Intelligence and Games, 100–107. Granada, Spain: IEEE.