# CISC 204 Modelling Project Report

Group 8: Aleeza Balbas, Ripley Visentin, Kylie Wong

# Project Summary

The objective of Gin Rummy is to earn points by collecting a hand where the majority or all the cards can be combined into different sets and runs, trying to group up the remaining unmatched cards on hand before the opponent chooses to end the round, checking to see who has the lowest point value. There are three ways to end the game: knock, gin, and big gin, where knock is has a 50/50 win-rate depending on the opponent’s remaining card point value. If one knocks and their point value is higher or equal to the opponent, it is the opponent’s victory, and vice versa. Whoever has a gin, a big gin or the lowest point value between you and the opponent wins the round.

In this project, we intend to model the ideal game strategy again the opponent’s sets of predetermined cards. We determine which melds are possible based on the opponent’s decision to either pick up a new card or one that we dropped, as well as analyze what the opponent chooses what card to drop. As we determine the optimal strategy, we strategize what the player’s next move should be, defining when it is the best time to pick up a new card or take the top card from the discarded pile.

# Propositions

*List of the propositions used in the model, and their (English) interpretation.*

|  |  |
| --- | --- |
| player\_a\_b | The player has a card with “a” rank and “b” suit |
| player\_run\_r\_v\_u | The player has a run of 3 or more consecutive cards of the same suit, r = rank, v= lower bound of the run, u = upper bound of the run, y<z (e.g. [player\_1\_A, player\_2\_A, player\_3\_A]) |
| player\_set\_k\_n | Considered true if the player has a set of 3 or 4 cards of the same rank k = rank, n = the rank that the set does NOT include, with Z indicates that the set include all four ranks (e.g. player\_set\_1\_A= [player\_1\_B, player\_1\_C, player\_1\_D]) |
| pl\_want\_a\_b | Considered true if obtaining a card of “a” rank and “b” suit would create a meld |
| opp\_a\_b | The opponent has a card with “a” rank and “b” suit |
| opp\_pick\_a\_b | The opponent picks a card with “a” rank and “b” suit from the discarding pile |
| opp\_discard\_a\_b | The opponent discards a card with “a” rank and “b” suit |
| deck\_a\_b | A card with “a” rank and “b” is in the deck |

# Constraints

*List of constraint types used in the model and their (English) interpretation. You only need to provide one example for each constraint type: e.g., if you have constraints saying “cars have one colour assigned” in a car configuration setting, then you only need to show the constraints for a single car. Essentially, we want to see the pattern for all of the types of constraints, and not every constraint enumerated.*

|  |  |
| --- | --- |
| MODEL | CONSTRAINT |
| player\_a\_b opp\_a\_b | If player have card with “a” rank and “b” suit, i.e. card(a,b), then opponent does not the same card |
| player\_a\_b player\_a\_b+1 player\_a\_b+2 … player\_run\_r\_v\_b | Definition of a run; the player has 3 or more consecutive cards of the same suit. r = lower bound of the rank, v = upper bound, b = suit |
| (player\_a\_j player\_a\_k player\_a\_m) (player\_a\_A player\_a\_B player\_a\_C ∧ player\_a\_D) player\_set\_a\_n | Definition of a set; the player has 3 or 4 cards of the same rank. a = rank, n = the excluded suit with ‘Z’ = none |
| opp\_pick\_a\_b (opp\_a-1\_b ∧ opp\_a+1\_b) ∨ (opp\_a+1\_b ∧ opp\_a+2\_b) ∨ (opp\_a-1\_b ∧ opp\_a-2\_b) ∨ (opp\_a\_j ∧ opp\_a\_k) ∨ (opp\_a\_j ∧ opp\_a\_k ∧ opp\_a\_m) | If the opponent picks card(a,b), that card must create a meld or contribute to an existing meld. |
| opp\_discard\_a\_b (¬opp\_a-1\_b ∧ ¬opp\_a+1\_b) ∨ (¬opp\_a+1\_b ∧ ¬opp\_a+2\_b) ∨ (¬opp\_a-1\_b ∧ ¬opp\_a-2\_b) ∨ (¬opp\_a\_j ∧ ¬opp\_a\_k) ∨ (¬opp\_a\_j ∧ ¬opp\_a\_k ∧ ¬opp\_a\_m) | If the opponent discards card(a,b), the card would not have created a meld or contributed to an existing meld, so we know which cards the opponent does not have |
| (player\_a-1\_b ∧ player\_a+1\_b) ∨ (player\_a-1\_b ∧ player\_a-2\_b) ∨ (player\_a+1\_b ∧ player\_a+2\_b) ∨ (player\_a\_j ∧ player\_a\_k) ∨ (player\_a\_j ∧ player\_a\_k ∧ player\_a\_m) → pl\_want\_a\_b | The player wants a card(a,b) if it would create a meld or contribute to an existing one |
| player\_a\_b opp\_a\_b deck\_a\_b | If the player doesn’t have card(a,b), the card must either be in the deck or in the opponent’s hand |
| opp\_a\_b pl\_want\_a\_b | If the opponent has card(a,b), we assume they will not discard it, so the player does not want that card |
| opp\_pick\_a\_ b opp\_a\_b | If opponent does not pick card (a,b), then for sure they do not have card(a,b) |
| opp\_discard\_a\_ b opp\_a\_b | If opponent discard card (a,b), then for sure they do not have card(a,b) |
| opp\_pick\_a\_ b ∧ opp\_discard\_a\_ b (opp\_a-1\_b ∧ opp\_a+1\_b) ∨ (opp\_a+1\_b ∧ opp\_a+2\_b) ∨ (opp\_a-1\_b ∧ opp\_a-2\_b) ∨ (opp\_a\_j ∧ opp\_a\_k) ∨ (opp\_a\_j ∧ opp\_a\_k ∧ opp\_a\_m) | If the opponent does not want card (a,b), i.e., opponent does not pick or discard it, then they do not have related card that makes card (a,b) into a meld. |

# Model Exploration

*List all the ways that you have explored your model – not only the final version, but intermediate versions as well. See (C3) in the project description for ideas.*

We simplify the game and focus on dealing with total 36 cards with ranks = {1, 2, 3, 4, 5, 6, 7, 8, 9} and suits = {A, B, C, D}. We have made up some rules of how the player behave: the player only picks up the card from the discarding pile if and only if the card can be added into an existing meld OR the card can make into a meld.

First, we explore the possibility of cards that opponent is holding based on how the opponent behave in the game. The only two inputs we have from the opponent is whether they pick the card facing up from the discard pile or not and the card that they discarded. For instance, if the opponent picks up the initial card facing up, then the initial card makes into a meld or adds into existing meld that the opponent is holding. Same logic goes to the card that opponent discards, if a card is not in a meld or it does not have potential being a meld, the opponent will discard that card. Given we know the cards the player is holding, we can conclude the cards that opponent potentially have by finding all possible meld combination that the opponent has.

The intermediate version of the model is determining whether the player should pick the card from the discarding pile or draw a card. This is done by counting and comparing the solutions of all choices that the player can make and the choice with the most solutions would be the best move to make in that round. Furthermore, based on the opponent cards that we logically assumed and the discarded cards as the game progress, we can decide whether a potential meld in the player’s hand is still possible by analysing if there exist a model that satisfy the conditions.

# Jape Proof Ideas

*List the ideas you have to build sequents & proofs that relate to your project.*

1. Player P does not have card(a,b) implies that either opponent T has the card, or the card is in the card deck D. Given that opponent does not have the card, therefore, the card is in the deck.

A screenshot of a computer

Description automatically generated

# Requested Feedback

*Provide 2-3 questions you’d like the TA’s and other students to comment on.*

1. Since most of our proposition is based on variables, we are having hard time building jape proof with propositional logic. Any suggestion on what jape proofs we could explore?
2. For the model, we have modelled the opponent so that they only pick up the initial card if that card would go into a meld which the opponent have less than 50% of picking the card up. Below is a comparison of two cases.

The sample model below shows when opponent picks up the initial card, with a 0.4 of chance of having the correct card that opponent has.

A black background with white letters

Description automatically generated

The sample model below shows when opponent does not pick up the initial card, with a 0.8 of chance of having the correct cards that opponent has. *A black screen with white text

Description automatically generated*

We found that if the opponent does not pick up the card, the accuracy of the analysis of opponent’s potential cards decrease significantly. Since we want to keep the opponent’s potential card absolute, it is ideal to have a high accuracy of the actual cards that the opponent has. How should we model it so that the accuracy of the opponent’s card would be ideal or what would you suggest?

# First-Order Extension

*Describe how you might extend your model to a predicate logic setting, including how both the propositions and constraints would be updated.* ***There is no need to implement this extension!***

Redefining our propositions with predicate logic

|  |  |
| --- | --- |
| PL(a, b) | The player has a card with “a” rank and “b” suit |
| PL\_R(r, v, u) | The player has a run of 3 or more consecutive cards of the same suit, r = rank, v= lower bound of the run, u = upper bound of the run, y<z (e.g. [player\_1\_A, player\_2\_A, player\_3\_A]) |
| PL\_S(k, n) | Considered true if the player has a set of 3 or 4 cards of the same rank k = rank, n = the rank that the set does NOT include, with Z indicates that the set include all four ranks (e.g. player\_set\_1\_A= [player\_1\_B, player\_1\_C, player\_1\_D]) |
| PL\_W(a, b) | Considered true if obtaining a card of “a” rank and “b” suit would create a meld |
| O(a, b) | The opponent has a card with “a” rank and “b” suit |
| O\_P(a, b) | The opponent picks a card with “a” rank and “b” suit from the discarding pile |
| O\_D(a, b) | The opponent discards a card with “a” rank and “b” suit |
| D(a, b) | A card with “a” rank and “b” is in the deck |

|  |  |
| --- | --- |
| MODEL | CONSTRAINT |
| PL(a, b) O(a, b) | If player have card with “a” rank and “b” suit, i.e. card(a,b), then opponent does not the same card |
| PL(a, b) PL(a, b+1) PL(a, b+2) … PL\_R(r, v, b) | Definition of a run; the player has 3 or more consecutive cards of the same suit. r = lower bound of the rank, v = upper bound, b = suit |
| (PL(a, b) PL(a, k) PL(a, m)) (PL(a, A) PL(a, B) PL(a, C) ∧ PL(a, D)) PL\_S(a, n) | Definition of a set; the player has 3 or 4 cards of the same rank. a = rank, n = the excluded suit with ‘Z’ = none |
| O\_P(a, b) (O(a-1, b) ∧ O(a+1, b)) ∨ (O(a+1, b)∧ O(a+2, b)∨ (O(a-1, b) ∧ O(a-2, b) ∨ (O(a, j) ∧ O(a, k)) ∨ (O(a, j) ∧ O(a, k) ∧ O(a, m)) | If the opponent picks card(a,b), that card must create a meld or contribute to an existing meld. |
| O\_D(a, b) (¬O(a-1, b) ∧ ¬O(a+1, b)) ∨ (¬O(a+1, b) ∧ ¬O(a+2, b)) ∨ (¬O(a-1, b) ∧ ¬O(a-2, b)) ∨ (¬O(a, j) ∧ ¬O(a, k)) ∨ (¬O(a, j) ∧ ¬O(a, k) ∧ ¬O(a, m)) | If the opponent discards card(a,b), the card would not have created a meld or contributed to an existing meld, so we know which cards the opponent does not have |
| (PL(a-1, b) ∧ PL(a+1, b)) ∨ (PL(a-1, b) ∧ PL(a-2, b)) ∨ (PL(a+1, b) ∧ PL(a+2, b)) ∨ (PL(a, j) ∧ PL(a, k)) ∨ (PL(a, j) ∧ PL(a, k) ∧ PL(a, m)) → PL\_W(a, b) | The player wants a card(a,b) if it would create a meld or contribute to an existing one |
| PL(a, b) O(a, b) D(a, b) | If the player doesn’t have card(a,b), the card must either be in the deck or in the opponent’s hand |
| O(a, b) | If the opponent has card(a,b), we assume they will not discard it, so the player does not want that card |
| O\_P(a, b) O(a, b) | If opponent does not pick card (a,b), then for sure they do not have card(a,b) |
| O\_D(a, b) O(a, b) | If opponent discard card (a,b), then for sure they do not have card(a,b) |
| O\_P(a, b) b ∧ O\_D(a, b) (a-1, b) ∧ a+1, b)) ∨ (a+1, b) ∧ a+2, b)) ∨ (a-1, b) ∧ a-2, b)) ∨ (a, j) ∧ a, k)) ∨ (a, j) ∧ a, k) ∧ a, m)) | If the opponent does not want card (a,b), i.e., opponent does not pick or discard it, then they do not have related card that makes card (a,b) into a meld. |

# Useful Notation

*Feel free to copy/paste the symbols here and remove this section before submitting.*