Hanabi Game Player

Qian Lin, Shanshan Xu

1 Introduction

Hanabi is a cooperative card game, where players, aware of other players' cards but not their own, attempt to play a series of cards in a specific order to reach the success state. During the game, players are limited in the types of information they may give to other players, and in the total amount of information that can be given.

In a standard Hanabi game, the cards are divided into five colors (pink, yellow, green, blue, white). For each color, there are three 1s, two 2s, two 3s, two 4s, and one 5. In addition, the group collectedly have seven information tokens. The goal is to reach a state where cards played on the table are ordered by number from 1 to 5, in all five colors. During the game, each player will have 4 cards in his hand. All players take terms to act, and at each term he has three options:

(1) play a card on his hand, in which case the predecessor card of the same color must have been played. Otherwise the card is discarded and a information token is taken as punishment.

(2) discard a card on his hand, which earns back a information token.

A new card is drawn after both option (1) and (2).

(3) tell one another player which of his cards are of a particular color or number but not both. A information token is taken after this action.

After the first attempt to draw from an empty deck, each player gets another term to act, after which the result is evaluated to see if the goal state is reached.

2 Task definition

In this project, we design and analysis various strategies that try to maximize the chance to achieve the success state. We build a simulator of the hanabi game and evaluates the performances of these strategies by applying them on the simulator.

3 Simulator

Each card is characterized by a tuple (c, n), where $c = 0, \cdots, N_c - 1$ labels its color with N_c be the number of colors, and n represents its number. Each player takes a action sequentially.

State: As shown in Fig.1, each state contains the following information

- All the cards remaining in the deck.
- All the cards played in the table.

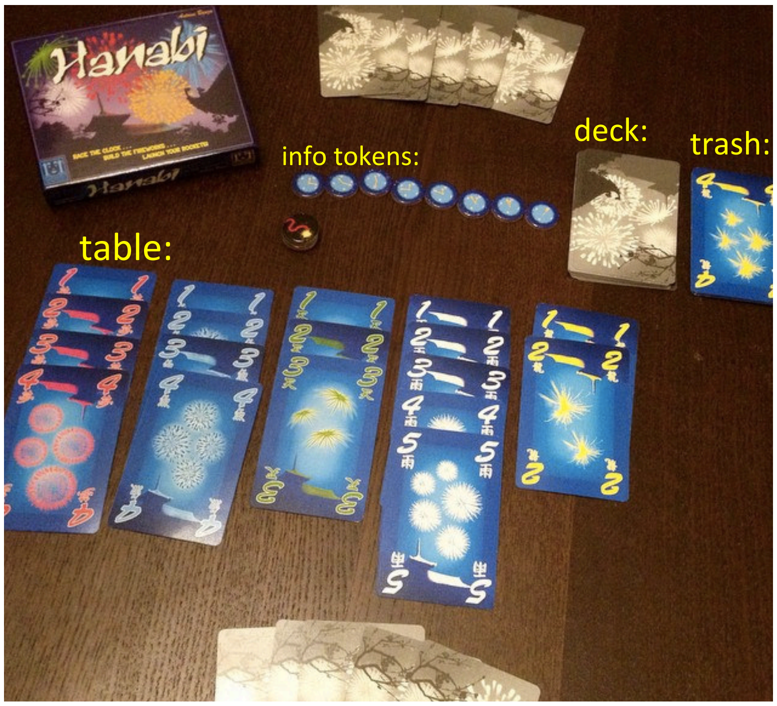

Figure 1: a state of the hanabi game.

- All the discarded cards in the trash.
- The cards in each player's hand. Each card is attached additional information wether its color and number are known to its owner.
- The number of information token remained.
- The number of additional terms left. 'inf' means the mode of the additional rounds haven't been activated.

IsEnd(s): All the distinct cards appear in the table. Or the number of additional terms is zero.

Actions: Each player can takes one of the following actions unless the condition is not satisfied. At the first time when a player tries to draw a new card from the empty deck, the mode of the additional rounds is activated and the number of additional terms is set to be the number of players in the game. After the turn that the mode of the additional rounds is activated, each taken action makes the number of additional terms decreased by one.

- ('play', i): The player chooses his/her i-th card. If there is some card in the table with the same color but has the number closely before this i-th card, the card is added in the table. Otherwise, the card is discarded into trash and the number of information tokens decreases by one if larger than zero. Finally, the player draws a new card from the deck if the deck is not empty.

- ('discard', i): The player discard his/her i-th card into the trash. The number of information tokens increases by one if smaller than the initial value. Finally, the player draws a new card from the deck if the deck is not empty.
- ('color', p, c): The player tells the p-th player which cards in the p-th player's hand have the color c. This action is allowed only when the number of information tokens is larger than zero. After this action, the number of information tokens decreases by one.
- ('number', p, n): The player tells the p-th player which cards in the p-th player's hand have the number n. This action is allowed only when the number of information tokens is larger than zero. After this action, the number of information tokens decreases by one.

Succ(s,a): return a new state from s following the description of the action a in the above. Note that For $a[0]$ = 'play' or 'discard', Succ(s,a) is not unique because a new card is drawn from the randomly shuffled deck.

Utility(s): The sum of all the cards' numbers on the table.

As for the **AgentState**, it contains the following variables:

- *cards*: a list of cards from oldest to newest. This is known only to the other players.
- *know*: a list of tuple of logic binaries indicating whether a property (color, number) of a particular card is known. This is known by all players (both the owner and the other players).
- *infer*: a list of inference about card made by the owner about his own cards. The type of information stored in the infer list is agent-type-specific. This is known only to the owner.

Besides the custom *getAction()* method, each agent also has a *inference()* method, which takes the *gameState*, *agentIndex* and *action* as the input, and updates the list *infer*.

4 Strategies

The unique feature of Hanabi is that it is a cooperative multi-agent game with incomplete information. Although it is deterministic (in that its successor state is unique) and can technically be described by a game tree, each player may need to have a probabilistic model describing its own viewpoint. The mainly challenges we try to address in designing our game agents as summarized as follows:

First of all, what's the appropriate amount of hard-coded heuristics, and how to optimize the set of heuristics. Human Hanabi players use simple heuristics like always discarding the oldest card first or conveying information that implies either play or keep, as well as complicated heuristics such as if only one 1 card is played, conveying information that a card is 1 means play. We will see that some of this complicated heuristics will emerge from our coded agents using only simple heuristics and statistical inference.

Secondly, how to perform effective inference to estimate viewpoints, and how to incorporate recursive inference, previous run clue and multi-agents. This is still undergoing effort.

As we will see below, the random player (no heuristics, no inference) scores 6 out of a 25/50 stack , while the oracle (complete information) scores close to 25 out of 25/50 stack. 25/50 means the game has total 50 cards and the maximal score on the table is 25. Reasonably experienced human players can also score an average 24 out of 25/50, and our goal is to match that score.

Before jumping into each specific strategy, we introduce some terminologies that will bring convenience to the later description. A card is said to be **playable** if there has been enough information to infer that it can be added on the table when played. For example, if no cards have been played, cards with the smallest numbers is playable regardless of its color. A card is **discardable** if it there has been enough information to infer it as a duplicate of a card on the table. Finally a card is **dangerous** if there has been enough information to infer it as the only card of its kind left and has not been played. Note that 'discardable' has no intersection set with 'playable' or 'dangerous'. However, a card can be simultaneously playable and dangerous. For practical purpose, the playable property takes priority.

4.1 Random Player (Baseline)

Each players takes action randomly. The player first randomly chooses the first element of the action 'play', 'discard', 'color', 'number' with probabilities 1/3, 1/3, 1/6, 1/6, respectively. Then the left elements in the action are sampled by a uniform distribution. We expect that this strategy gives the lower bound of the utility in this game. By simulation, this scores on average 6 out of a 25/50 card stack.

4.2 Oracle

If each player knows its own card, they can play optimally. But this does not guarantee a win, depending on end-game situations. Each player chooses a legal action in the following order:

- play the oldest playable card
- if information token is available, spend it
- discard the oldest discardable card
- discard the oldest non-dangerous card
- discard a random card

This scores very close to 25 out of a 25/50 card stack.

4.3 Brute Force

In class, we talked about minimax or expectiminimax algorithm for competitive games. Here, as a collaborative game, Hanabi game only requires pure maximization rather than expectiminimax. For every player, the recurrence relation can be formulated as

$$V_{\text{opt}}(s) = \max_{a \in \text{Actions}(s)} \sum_{s'} T(s, a, s') V_{\text{opt}}(s', a)\,, \tag{1}$$

where $T(s, a, s') = 1/r$ when $a[0] =$ 'play' or 'discard'. r is the number of remaining cards in the deck before drawing. For $a[0] =$ 'color' or 'number', s' is unique and $T(s, a, s') = 1$.

A elementary analysis shows that above brute force is computationally expensive. Suppose there are three players in the game. At the very beginning of the game, the first player could choose any of his four cards to play or discard. The card he then draws from the deck has 34 possibilities. The first player could also tell any one of the other two players the information of one color or number. Therefore, consider the search tree, for the root , the number of children nodes is at least $4 \times 34 + 4 \times 34 + 2 \times 1 + 2 \times 2 = 278$. Even we introduce a evaluation function and limit the maximal depth of search to be $d = 1$, to generate a action for the root, we need to explore at least $278 \times 270 \times 262 \approx 2 \times 10^7$ number of nodes. As a result, we have to design strategies instead of brute-force searching.

4.4 Encoding Agent

This is a class of deterministic strategy due to a friend at Stanford, Daniel P. Z. Whalen. Its essence is a commonly-known strategy for assigning values to a hand, and communicate that value to other players by discarding a specific card, or providing information about a specific card. The interpretation of player action is unique.

In these set of strategies, the action content is not relevant. The form of action encodes all the information. For example, if a player want to commute a value of 1, he discard his first card. This action has nothing to do with his knowledge about what his first card is, but merely to encode the value 1 in his action.

This a set of strategy uncommon for human players but easy for compute algorithms. We implemented two examples of this class:

- The first one is informationless Agent, in which no information token is allowed (so this is a harder game than the standard game). Information about other players' hand is encoded in the discard action.

- The second one allows use of information token. Information about other players' hand is encoded in the inform action. More specifically, which player to inform and whether the information is about color or number is mapped to a value.

The latter agent achieves the best average score, 19.2, of all implemented agents (except for the oracle). It was able to score 25 with small probability.

4.4.1 Informationless Agent

Start with Hanabi with no information token.

Assign to each player's hand a value from 1 to 4 in the following way (in this agent only card slot in a hand are labeled 1 to 4 from oldest to newest):

- 1: if card 1 is playable
- 2: if card 2 is playable and card 1 is not
- 3: if card 4 is playable and 1 and 2 are not
- 0: if cards 1, 2, 4 are unplayable

Players keep track of the value of their hand if they know it. On a players turn, do the following:

- if they know their hand has value of 1, 2 or 3, play card 1, 2 or 4 respectively
- if they know their hand has value 0, or they don't know the value of their hand, discard card n where n=(sum of values of the other players' hand) mod 4. This is enough information for all other players to determine the value of their own hand (by looking around)

An improvement is also implemented. When a player is discarding and evaluating n, if its next player has a playable card that some other players also have, set the next player's hand value as 0.

This scores on average 16 out of a 25/50 card stack.

4.4.2 Information Agent

Allow the use of information token. Calculate the value of each player's hand in the same way as before. On a players' turn

- if they know their hand has value of 1, 2 or 3, play card 1, 2 or 4 respectively
- if they know their hand has value 0, or they don't know the value of their hand, do the following:
- if information token is available,
- if information token is not available, discard card n where n=(sum of values of the other players' hand) mod 4.

This is enough information for all other players to determine the value of their own hand (by looking around).

This scores on average 19.2 out of a 25/50 card stack.

4.4.3 Analysis

Under the assumption of these set of agents, the number of states for each player's hand is only four, and there is only four types of actions. Thus the actions can uniquely encode information about player's hand. This is a significant reduction (abstraction) of the states, thus will fail to capture some important information about the state, for example certain cards being dangerous. However, the benefit gained from having unambiguous interpretation of player actions is significantly.

4.5 State Strategy

We also implement another set of agents that partially simulates what human players will do. They operate based on a simple set of heuristics. Due to the heuristic and the type of information allowed by the game, sometimes interpretation of an action has unresolved ambiguity, for example player A may want to signal to player B that its first card Y2 is playable, but player B has multiple cards of 2 and of yellow respectively, thus whatever information A can provide about the first card in player B's hand, multiple cards are informed, and player B needs to decide which card player A intend to be playable.

4.5.1 Internal State Strategy

This is a naive heuristic-based agent. Each player chooses a legal action in the following order:

- play the newest card inferred to be playable
- if information token is available, and next player has playable cards, inform next playable about the oldest playable card with incomplete information (number first, then color)
- if information token is available, and next player has dangerous cards inform next playable about the oldest dangerous card with incomplete information (number first, then color)
- discard the oldest card inferred to be discardable
- discard oldest card with no information
- discard oldest card not inferred to be dangerous
- discard a random card

Inference of each card's being dangerous, playable and discardable is performed by an agent after its previous agent has taken an action. Inference is based purely on what's know about the agent's card, and does not depend on the action taken by the previous agent.

This scores on average 14 out of 25/50 stack.

4.5.2 State-Guess Agent

To improve on the previous heuristic agent, this State-Guess agent plays by a set of similar rule at its term; at inference, however, its takes into account both the local knowledge, and the previous agent's action. If the previous agent choose to use an information token, this itself means that the agent has either playable or dangerous card.

An obstacle towards making use of previous agent's action is the ambiguity of an inform action indicating either dangerous or playable, and which card it is if multiple cards have the informed property. To partially resolve this, for playable card signal number first then color; for dangerous card signal only color.

This current strategy simply make use of a set of heuristic: if number is signal, set the newest signaled card playable. If color is signal, and the newest signaled card's number is known, set it as playable; otherwise set the oldest signaled card as dangerous.

This scores on average 17.5 out of 25/50.

4.5.3 Analysis

In terms of state base, this strategy has $2^4 = 16$ states for each hand (4 means one of the following properties for each card: playable, discardable, dangerous, and not-known), which is still a significant reduction from the full state-representation, but is more detailed than the encoding agent class.

5 Summary

A comparison of the previous five types of agents is plotted in Fig.2

6 Acknowledgement

We thanks to Daniel P. Z. Whalen for sharing his strategy.

References

[1] Osawa, Hirotaka. "Solving Hanabi: Estimating Hands by Opponent's Actions in Cooperative Game with Incomplete Information." Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence. 2015.

[2] CS221 pacman assignment.

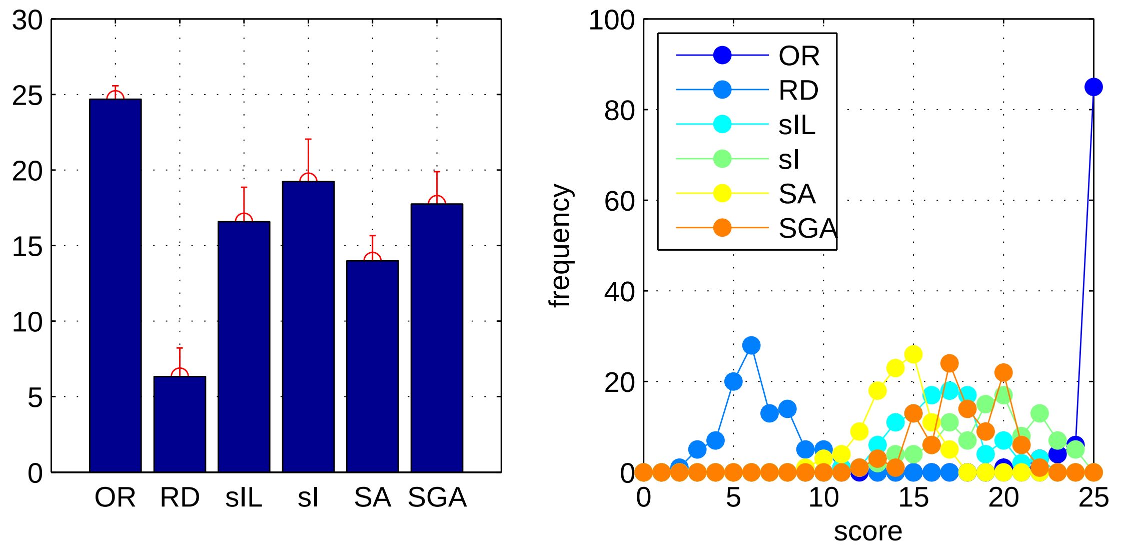

Figure 2: Left: mean and stander deviation of scores over 100 trails. Right: histogram of scores over data used in the left plot.