

Game AI: Assignment 2

Ajay Kumar Malik (am21932@essex.ac.uk)

1. Introduction: -

This project deals with the creation a new agent for the game “Hanabi”. Hanabi is cooperative game that can be played between 2 or up to 5 players.

The agent was created using Genetic algorithm to generate a sequence of rules, known as chromosome. The agent uses the rules to decide on what action should be taken in the game when its turn to play. The agent plays the game and tries to improve the win percentage over numerous consecutive games.

The motivation behind choosing Genetic Algorithm for the selection of rules is that the algorithm is renowned for producing superior solutions for various problems. Since our agent plays the game using a pre-defined set of rules, genetic algorithm can help in generating an order that can provide us with high win percentage.

2. Literature Review: -

Hanabi [1] is cooperative game that is played anywhere between 2 to 5 players [1]. If the game has 3 or less players, each player is dealt with 5 cards. The size goes down to 4 for 4 or more players. Every player can see the cards of other players but cannot see his/her own set of cards.

Each set of cards has three 1s, two 2s, two 3s, two 4s, and one 5. The sets must be completed in rank order from lowest to highest. Each player has a full hand size when the game starts. The team has eight information tokens and three mistake tokens. The game ends when the deck runs out and each player has one final turn, or when the players have zero mistake tokens left.

A player can perform either of the three actions as mentioned below: -

- Play a card: -
The player can decide to play a card when it is his turn but playing an incorrect card will result in a loss of mistake token.
- Give a hint: -
The player can decide to hint anyone of the other players about the cards they hold. But the player can only hint about either the colour or the rank of the card at any given turn.
If the player decides to hint about colour, he needs to hint the other player about all the cards of the colour he is to be hinted for. For example, if the

active player decides to hint the player about yellow cards, former will have to point to all the yellow cards that the latter player holds. The same rule applies when hinting the player of the rank of the cards.

Hinting other players of the cards results in the loss of an information token.

- Discard a card: -

The player can discard a card when it is his turn and gain an information token. The discarded card goes into the discard pile and cannot be used again in the game.

A team can only win if they can place all the cards ranked from 1 to 5 from for all the colours in the fireworks pile as shown below: -

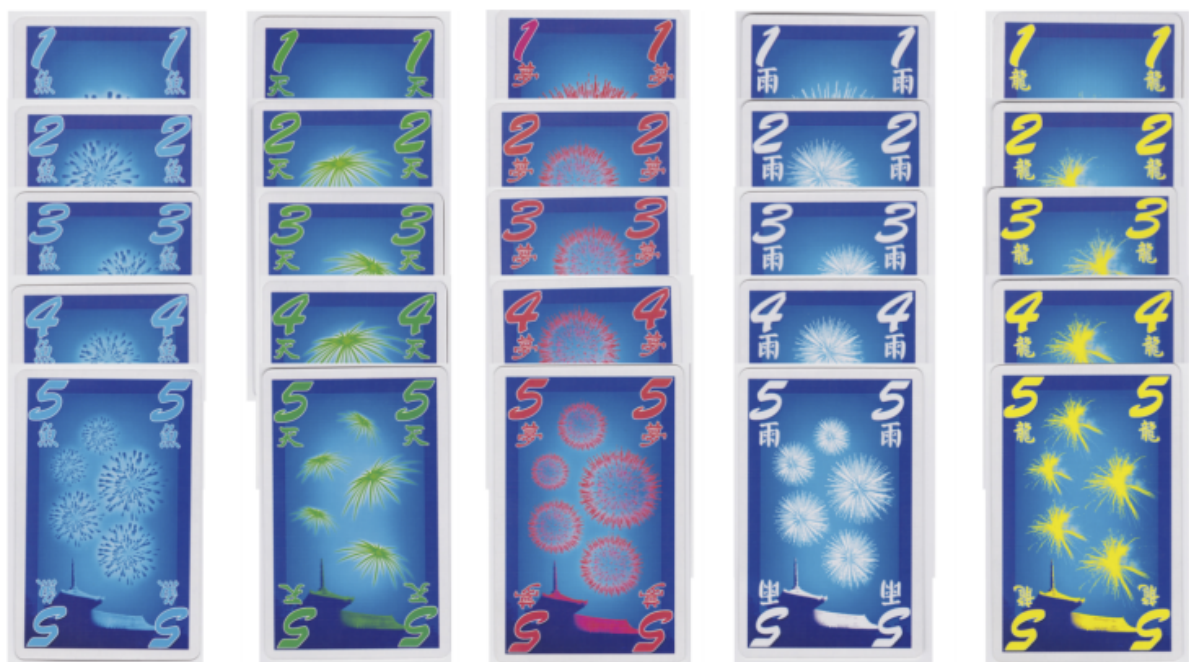


Image Credit: R&R games

The fun and the most challenging part of the game is the communication that is required between the players of the game. The game requires a greater understanding of how the other players play than the game itself, both for the humans and the AI. If the player does not understand the other players, the team will end up scoring low often.

3. Background: -

Hanabi can be played using a set of rules that represents the actions that can be taken to determine the next step during the course of the game. To determine the order of the rules to we can use Genetic Algorithm.

The Genetic Algorithm was developed by John Holland [2] and his collaborators in the 1960s and 1970s and is based on Charles Darwin's theory of natural selection.

The implementation of Genetic algorithm involves the below mentioned 4 basic functions.

1. Selection
2. Crossover
3. Mutation
4. Elitism

The two algorithms that were tried to get a high performing chromosome are as following: -

1. Population-Based EA method – This algorithm tries to work a better performing chromosome using a population of individuals.
 - a. For each generation, the algorithm first evaluates all the solutions and promotes the best performing individuals to the next generation if using Elitism.
 - b. For the remaining population, two parents are chosen to create a new individual.
 - c. The new created individual is then mutated and moved to the next generation.
2. Stochastic Hill Climber – This algorithm only tries to improve one trial solution.
 - a. First a random solution is created.
 - b. The individual is then mutated to generate a new solution and its performance is evaluated.
 - c. If the newly generated chromosome performs better than the previously best chromosome, it replaces the latter and is then further mutated and evaluated.
 - d. This cycle continues until we get a desired chromosome.

4. Techniques Implemented: -

A rule-based agent was created using two evolution-based algorithms – Population-based genetic algorithm and Stochastic Hill Climber genetic algorithm.

The reason for choosing two algorithms is that the Population based GA algorithm takes a considerable amount of time to figure out a better solution. This restricts the number of iterations that we can run on a diverse population of the chromosomes. There could be chance that the returned chromosome could have been further altered to get an even better chromosome. So, to be sure, the returned chromosome is then fed into SHC agent that tries to get a better chromosome.

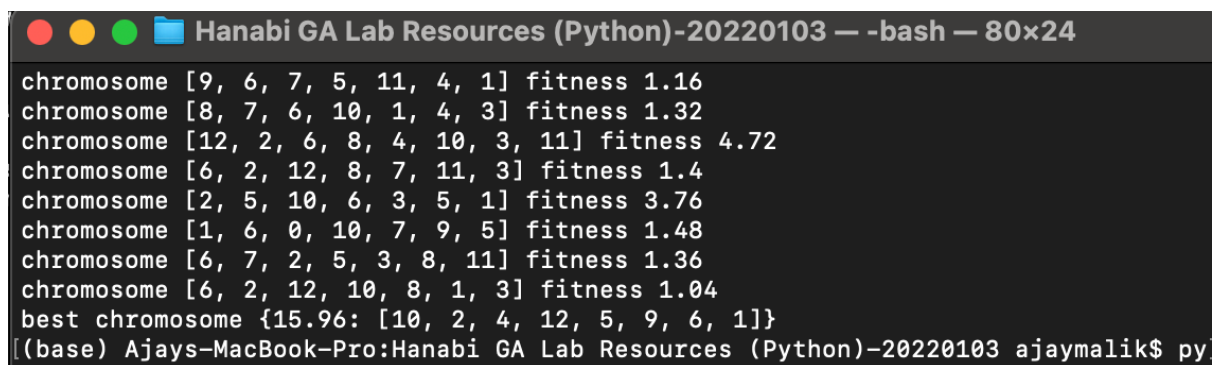
In addition to the existing set of rules, following new rules were added: -

- 1) Rule 7 – This rule identifies a useless card that a user can discard. A useless card for this rule is described as the card that has been played in the fireworks pile. For example, a ranked 1 Red card becomes useless if such a card has already been played on the fireworks pile. This rule helps the players to identify cards that can be safely discarded.
- 2) Rule 8 – This rule looks for the cards that are playable and can be straightaway played onto the fireworks pile. For this rule, a playable card is one that has both the colour and the rank hinted and it can be played onto the fireworks pile. For example, if the highest ranked card on the green fireworks pile is 2, then a green card ranked 3 is termed as playable.
- 3) Rule 9 – This rule provides the remaining hint for cards that have either rank or colour hinted before. After this rule gets executed, a player has complete set of information for his card, which later helps in identifying if the card is playable or is useless.
- 4) Rule 10 – This rule combines two different rules into one – Rule 8 that identifies and plays a card that is directly playable onto the fireworks pile and Rule 0 which identifies and plays the most playable card from the hand if the first condition is not met.
- 5) Rule 11 – As per this rule, we try to give hint to the next player for the cards that will give the maximum information to the player about his/her hand. For example, if a player has 3 green and 1 red non-hinted cards and 2 ones and 1 three non-hinted cards, then we hint the player about the 3 green cards. This way the next player gets the maximum information about the cards in his hands.
- 6) Rule 12 – As per this rule, we discard the oldest card that we have in our hand, if the count of information tokens goes below 3. This way we try to ensure that we have enough information tokens available to continue providing other players with hints.

5. Experimental Study: -

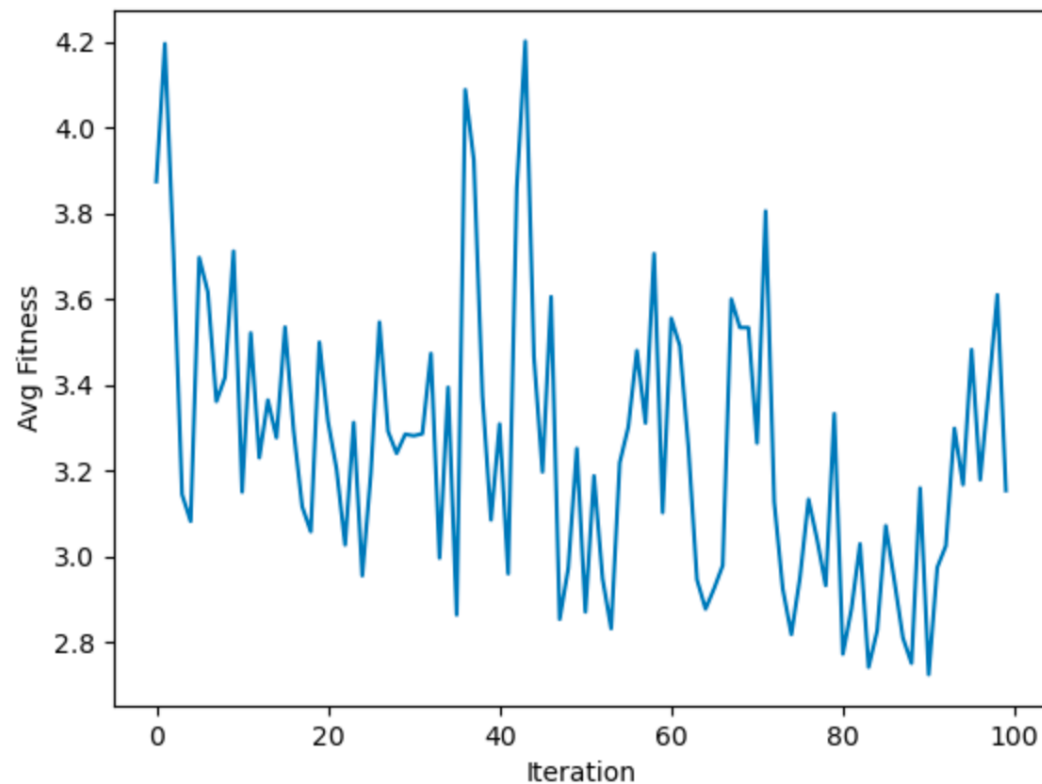
After the 13 rules ruleset was created (numbered 0 to 12), a bot (population_based.py) was used to evaluate the ruleset by trying and create a best performing chromosome by following the below mentioned steps: -

1. The bot first creates a population of 200 randomly generated chromosomes each of length 9. The length of the chromosomes was kept on the higher side to include majority of the rules.
2. The bot then evaluates the fitness of all these chromosomes by playing 25 games for each of the chromosomes. The average result of the all the games is called the fitness of the chromosome.
3. The best performing top 5 chromosomes were then directly moved to the next population and rest of the chromosomes are used as parents for crossover.
4. After the crossover, the resulting chromosomes are mutated, either using swap mutation or delete mutation.
5. Once mutated, the child chromosome is then added to the next generation for evaluation.
6. This cycle continues for 100 generations.
7. Also, during the generations, we keep a track of the best performing chromosome.
8. The best chromosome that came up attained the fitness score of 15.96. The chromosome was **[10, 2, 4, 12, 5, 9, 6, 1]** as shown in the below screen grab: -

A terminal window titled "Hanabi GA Lab Resources (Python)-20220103 — -bash — 80x24" displays the output of a Python script. It lists several chromosomes with their fitness scores. The highest fitness score shown is 4.72 for the chromosome [12, 2, 6, 8, 4, 10, 3, 11]. At the bottom, it indicates the best chromosome found is {15.96: [10, 2, 4, 12, 5, 9, 6, 1]}. The prompt shows the user is in a directory named "Ajays-MacBook-Pro:Hanabi GA Lab Resources (Python)-20220103" and has run a command starting with "ajaymalik\$ py".

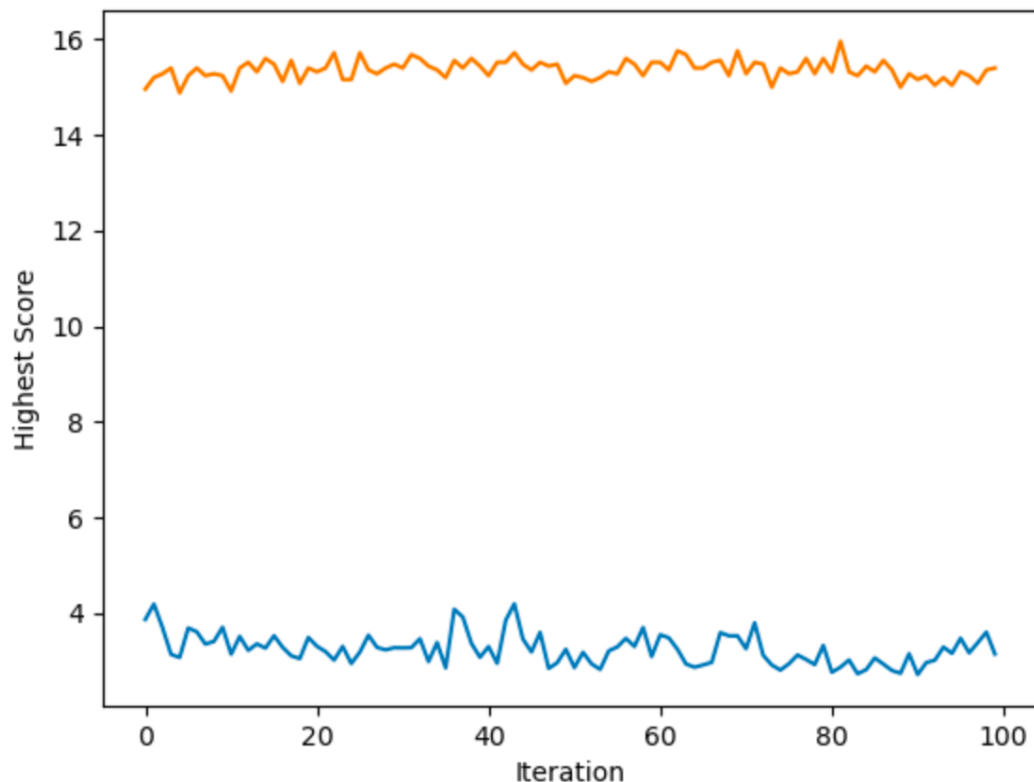
```
Hanabi GA Lab Resources (Python)-20220103 — -bash — 80x24
chromosome [9, 6, 7, 5, 11, 4, 1] fitness 1.16
chromosome [8, 7, 6, 10, 1, 4, 3] fitness 1.32
chromosome [12, 2, 6, 8, 4, 10, 3, 11] fitness 4.72
chromosome [6, 2, 12, 8, 7, 11, 3] fitness 1.4
chromosome [2, 5, 10, 6, 3, 5, 1] fitness 3.76
chromosome [1, 6, 0, 10, 7, 9, 5] fitness 1.48
chromosome [6, 7, 2, 5, 3, 8, 11] fitness 1.36
chromosome [6, 2, 12, 10, 8, 1, 3] fitness 1.04
best chromosome {15.96: [10, 2, 4, 12, 5, 9, 6, 1]}
[(base) Ajays-MacBook-Pro:Hanabi GA Lab Resources (Python)-20220103 ajaymalik$ py
```

The average fitness attained in each generation is as shown below: -



It can be observed that the average fitness goes down initially and takes big leap between the 30-50 generations before going down again. It starts to slowly pick up again after the 80 iterations are done.

The below mentioned graph shows the highest fitness achieved across all the generations.

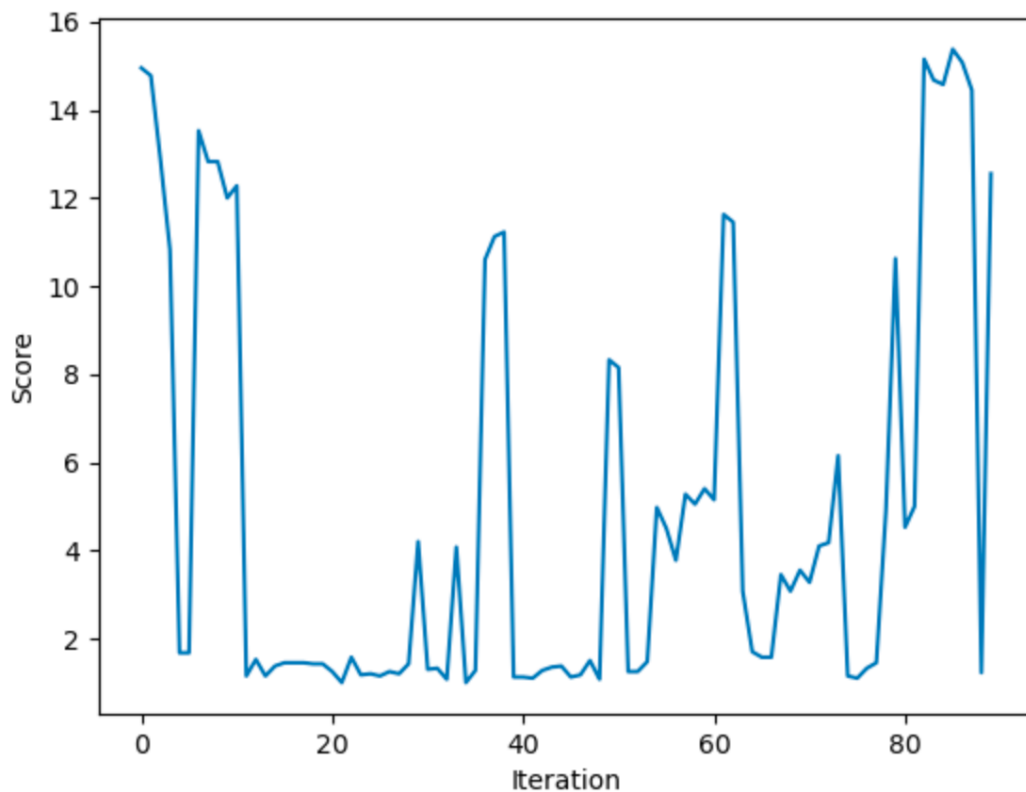


It can be observed that the highest score hovers around 15 across all the generations. This is so because the best performing chromosomes were moved to the next generation after evaluation without any crossover or mutation.

After getting the best performing chromosome, it was also passed through another bot (SHC_evaluator.py) to see if this chromosome can be further improved. This bot uses **Stochastic Hill Climber** algorithm for evolution.

1. The bot is first initialized with the chromosome [10, 2, 4, 12, 5, 9, 6, 1].
2. The chromosome is then evaluated by playing over 40 games. The average of the scores at the end is called the fitness of the chromosome.
3. Once evaluated, the chromosome is mutated using swap mutation and then evaluated again.
4. This cycle continues for 100 iterations and the best performing chromosome is chosen as the chromosome for the agent to play.

The following graph shows the score of each mutated chromosome.



It was observed that none of the mutated chromosomes performed better than the original chromosome.

Hence the chromosome selected to be used in the agent is [10, 2, 4, 12, 5, 9, 6, 1].

6. Overall Conclusions and future work: -

It was observed that genetic algorithms can evaluate and come up with better performing chromosomes over several generations. But with the increase in length of the chromosome, the evaluation starts getting slower, thereby forcing to cut down on the number of generations to look for optimal solutions. With just 13 rules and length 9 of the chromosome, the algorithm took around 8-10 minutes to run for 100 generations.

With further increase in the number of rules and size of the chromosomes, genetic algorithms may not be the most efficient algorithms to work out an optimal solution.

As part of the future work, we could try out other algorithms like Neural networks, MCTS/IS-MCTS hybrids and evaluate their performance against that of the genetic algorithms.

For Assignment 2, Your Hanabi agent scored an average of 5.3 points per game.

Part #	Outcome	
Class MyAgent is defined	Yes	✓
Agent Score (for Assignment 2)	5.3/25	✓

7. References: -

1. Hanabi card game - [https://en.wikipedia.org/wiki/Hanabi_\(card_game\)](https://en.wikipedia.org/wiki/Hanabi_(card_game))
2. John Holland – American scientist and Professor of electrical engineering and computer science - https://en.wikipedia.org/wiki/John_Henry_Holland