LOVE LETTER

An Implementation using Artificial Intelligence

Nicholas Walters 22243339

Shikai Wang 21938451

Literature Review

Introduction

Love Letter is a game of risk, deduction, and luck for 2–4 players. The goal is to get your love letter into the Princess's hands while deflecting the letters from competing suitors. From a deck of only **16 cards,** each player starts with a state of only one card in the hand. On each turn, you are supposed to draw one card and then remove/play one card, using the effects of that card played. Each card has different effects and values when played and will affect the other opponents depending on each card played. The goal is to expose others and knock them out of the game by knowing their cards in hand and playing moves accordingly.

Love letter is a card came with **incomplete information**. Each player is able to view their own hand, but not others (some exceptions). In order for a player/agent to win a game, the player needs to know or guess the hidden cards of the other opponents first, and then knock them out. There is some reasoning made when forming a guess of the other players card.

As there are only 16 cards in a deck, a realistic probability of how many of each specific card that each player may have can be calculated (card counting).

However, with incomplete information about the game state and opponents' cards, there is some uncertainty applied and there could be many possible game worlds.

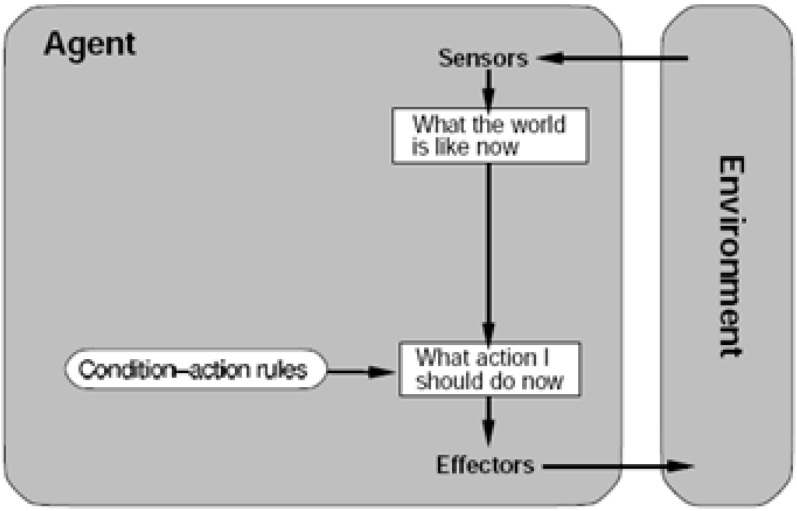
- At the start of the game, everything is based on luck, because there is no knowledge of other players cards and their values. There are too many different game scenarios to be played extrapolating from the start.

However, we can assume that we start from a 16-card deck.

- As more cards are played, some data can be recorded as to what cards each member may have. Then, more of an idea of the game states and cards held can be formed giving more information to agent.

Rule Based Approach

Makes pre-defined decisions based on a set of rules to follow, depending on the game state/situation (current environment). The environment directly contributes to the move played, and there is no global optimum. If the agent can guess another player's unseen card with high probability, thanks to the discarded deck, it will execute that decision. This is a simple reflex-based approach, as shown in (Jeffrey Bradshaw, 1997):



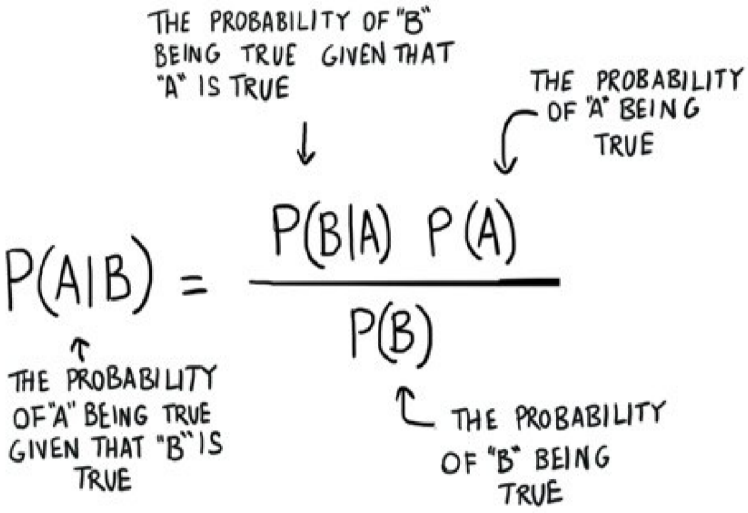
When playing 'Love Letter' there are a number of rules/strategies that people unconsciously follow when playing the game:

- By default, Discard/Play the lower value cards, and keep the higher value ones until the end of the round
- Play the Baron only if you have another relatively high-ranking card, and the probability that the opponent has a higher card is low (by looking at unseen deck). Especially when you have Princess then target the highest scorer.
- Play the King only if there are more Unseen cards which are ranked higher than your current card in hand (i.e. Don't play the King if the probability of getting a worse card is high)
- If the player has seen another person's card and knows their card value, you can play a guard or baron and try to eliminate them

- When taking an uninformed guess using a guard, count the discarded guard deck, and find the card which has been played least and has the greatest number of cards left (i.e. greatest number of unseen cards).
- If you know that another player has a high value card, use prince on them to make them discard theirs. If you have a low value hand, then discard your deck.
- If a player has played a countess, then most likely (not always) they have a prince or king.
- Always Target a player who has the highest number of rounds won so far (except yourself).

The above strategies were used in the implementation of this rule based, reflex agent. Some more strategies can be applied such as Bayes Theorem, which is an extension to the rule-based agent:

This agents extension will have some similarities to the Bayes theorem, as it figures out the probabilities that an opponent will have a card, where the models of opponent strategies have been provided already. With reference to Cadwallader Olsker (2011) bayes theorem will guess the probability of a value occuring based on what has been played previously.



In this selection strategy, the agent memorises the deck of unseen cards. The agent considers two scenarios to guess a player's card. Firstly, it considers both the number of unseen cards, and secondly it acknowledges that the target player will likely keep the higher valued cards over the smaller ones. For example, if there is 1 unseen Baron and 1 unseen King in the deck, the selection criteria would be skewed towards the King (higher value selection). These scenarios help the agent to determine the probability of what the target may have in his hand and play a card accordingly.

If there is a choice between A (Baron) and B (King) based on values in the unseen deck, the opponent is likely to keep the higher value card B and play the lower one A (Baron).

However, this strategy will not work on a Random agent, because it discards anything regardless of value and has no strategy.

Monte-Carlo Tree Search (MCTS)

MCTS is a heuristic based algorithm which expands the nodes which have the highest probability of returning a good outcome first (but also explores other possibilities that don't look favourable). MCTS involves Selection, Expansion and Simulation (and Backpropagation).

In the case of love letter, we would use a (Single Observer) SO-MCTS algorithm, as the agent doesn't know the values of other players cards. MCTS works the best if it knows the entire state of the game, and possible plays of the opponent's cards are known, but in love letter, this is not the case (uncertain).

According to (Tamirlan Omarov 2018) in MCTS, a node is traversed from the root node to a leaf node, the child (move in Game) that maximises the UCB1 formula is selected.

$$UCB1(S_i) = \underline{V_i} + c\sqrt{\frac{lnN}{n_i}}$$

$$\underline{V_i} = average\ value\ of\ the\ current\ state$$

$$N = number\ of\ visits\ of\ the\ parent\ node$$

$$n = number\ of\ visits\ of\ the\ current\ node$$

$$c = exploration\ parameter$$

When a leaf node is reached, the expansion stage adds all the legal and valid actions as children. Simulation stage will play out a round of the game and

determine its probable result. This result is used to update information along all nodes from node to the root (backpropagation).

MCTS is efficient in being able to understand the current state of the game in 'Love Letter'. If you can run MCTS for a long period of time, then it will give greater reliability, but it can still be stopped at any time and return its best move so far.

As shown in Figure 3, determinations of all possible moves need to be made, and a simulation is done (0/1) to show the agent whether that branch/move is a good play.

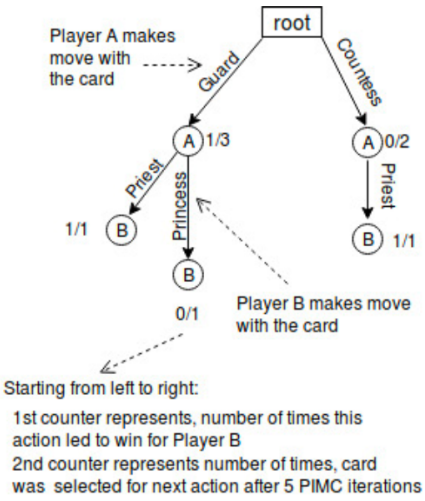


Figure 3: Game Tree after 5 MCTS iterations on Determinized game state

Rationale, Selection and Implementation of Agents

The probability-based goal agent was finally selected after much testing of viable techniques.

It is easy to see that the technique of bayes probability agent was chosen because there are not many other ways to have an edge/advantage a game which involves too much uncertainty. The only advantages gained may be in the case where you can remember the cards already played/discarded (card counting), keeping higher value cards and assuming others do the same, and finally guess using the seen cards (priest and king).

The probability-based agent will know other opponents last discarded card and make an assumption as to what card they have, while also using other variables such as the discarded deck. It will also be a 'Conservative Agent' as its technique is very defensive in nature. In this case, the agent will only play the risky move of baron (or King) if the probability of winning the comparison is greater than 60%. We found that 60% probability is a good trade-off, as we don't want to discard higher value cards too often. It will not play the baron especially if its chances of losing the comparison are low.

P (higherCard | lowerCard) = P(higherCard / lowerCard) P(UnseenCard count of Higher Card) ÷ P(lowerCard)

Assuming the heuristic of the opponents always playing the higher value card by default *P (higherCard | lowerCard)*, we can calculate the probability of the opponent holding {higherCard} based on their play of the {lowerCard} . Unseen card count is calculated using the number of unseen cards and discarded cards, implemented in the function unseenCards() and further extended by guess() and other functions in the heuristicAgent.

separate probability is also made by counting the deck:

Guard Guess:

Guess (C) = max(unseenCardCount(C))

- if multiple max card counts are the same, choose the highest value card (opponent will likely keep higher value cards)
- This function will also be influenced by the opponents last discarded card. Their discarded card will most likely be lower than their current card. this influence comes from the method applySelectionBias() in heuristicAgent, which incentivises the agent to choose cards higher than its last discard.

Baron and King Guess:

*playBaron_or_King(Y | N) : 60 > (numberOfUnseenCardsBelow(myCardValue) / totalUnseenCards * 100)*

(I.E, count the number of unseenCards above the agent's current hand, if there are too many cards above the agent (>40%), then don't play Baron or King)

In less risky moves such as playing the guard, or prince the agent will also make an educated guess based on what has been already played/discarded by everyone.

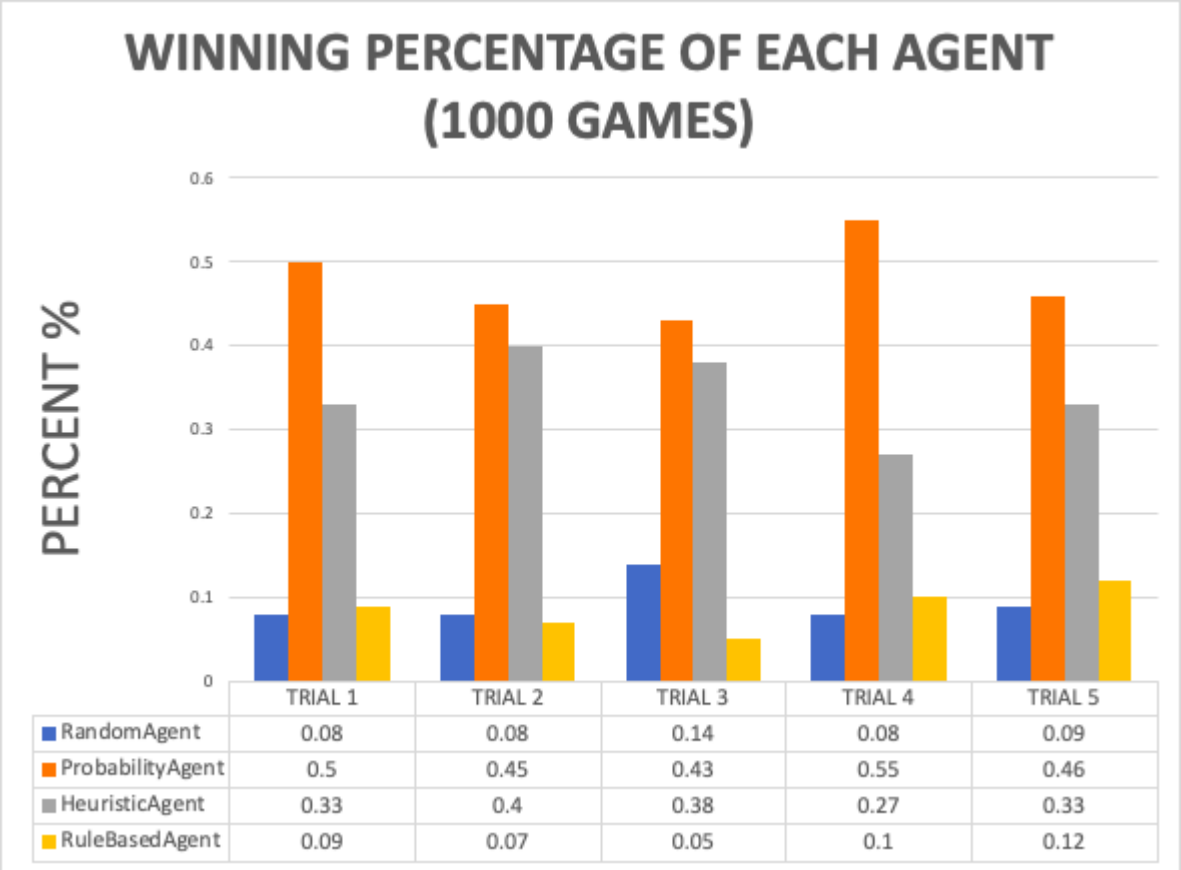
While the Monte Carlo Tree Search Algorithm (MCTS) is a viable technique for this game, it was dropped after some prototyping and experimentation. MCTS method requires simulation the actions other players will take, but without knowledge of the opponent's hand it cannot simulate a game properly. MCTS did not yield any noticeable results for the effort involved. The MCTS algorithm did not perform above our expectations after experimentation.

Simulation in MCTS requires little or absolutely no uncertainty. As there is so much uncertainty (randomness) around what a player may hold, it was not viable in our approach.

The Single Observer MCTS (SO-MCTS) considers the uncertainty of a game, and is designed to accommodate unknown variables, but making incomplete simulations were questionable at best, as it is no better than selecting random moves.

Validation

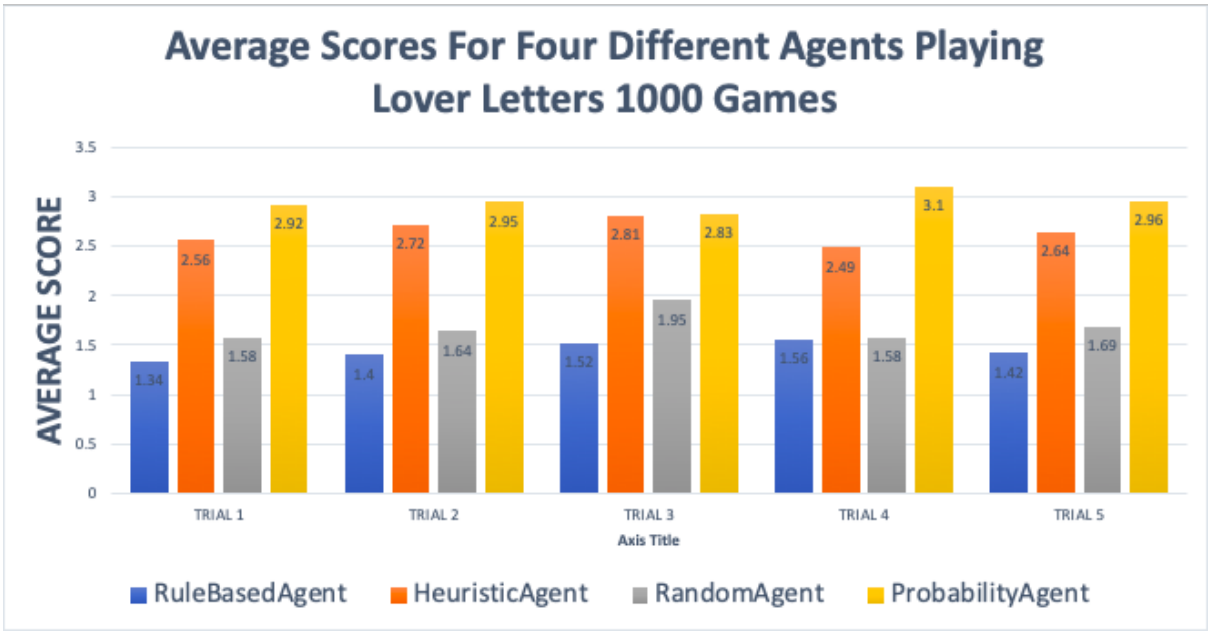
Analysis of the Probability Agents Performance against 3 different agents (RandomAgent, HeuristicAgent, and RuleBasedAgent), over 1000 games (4 Players Total)



	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
RandomAgent	0.08	0.08	0.14	0.08	0.09
ProbabilityAgent	0.5	0.45	0.43	0.55	0.46
HeuristicAgent	0.33	0.4	0.38	0.27	0.33
RuleBasedAgent	0.09	0.07	0.05	0.1	0.12

Figure: Winning percentage four different agents 1000 games

After tested performance of four different agents, heuristic Agent and probability Agent performed significantly better than the Random Agent and Rule Based Agent.This can be contributed to the fact that the random agent has no concept of its environment and relies on randomness for its next move. Rule Based Agent is slightly better than the Random Agent due to a number of rules to follow. While the rule based agent is somewhat aware of its surroundings/environment our implementation of rules don't give it a significant advantage over the random agent.



Both the Heuristic Agent and Probability Based agents perform much better than the other agents, as they are similar in structure (but different strategies). However the strategy of the Bayes Probability agent is superior to the Heuristic agent which guesses based on the last discard.

Based on the results Guessing the probability of another opponent having a card based on the current discards is the best strategy to employ, as its performance has significantly improved.

References:

Rule Based Agents, Compliance, and Intention – Antonino Rotolo

https://page-one.springer.com/pdf/preview/10.1007/978-3-642-22546-8_7

Monte Carlo Tree Search for Love Letter – Tamirlan Omarov, Hamna Aslam, etc.

Available from: https://www.researchgate.net/profile/Joseph_Brown7/publication/327679828_Monte_Carlo_Tree_Search_for_Love_Letter/links/5b9e5bbba6fdccd3cb5c9c8a/Monte-Carlo-Tree-Search-for-Love-Letter.pdf

Cadwallader Olsker, T. D. (2011). when 95% accurate isn't: Exploring Bayes's Theorem. *The Mathematics Teacher, 104*(6), 426-431. Retrieved from http://www.jstor.org.ezproxy.library.uwa.edu.au/stable/20876909