**CITS 3001 Algorithms, Agents and Artificial Intelligence**Wesley Coleman - 21485619  
Wen Tan - 21503781

**Literature Review**

To build an artificial intelligence agent for The Resistance, firstly the game must be studied. General strategies and assumptions on the game state that have been deduced by players of the card game are researched. A basic strategy for both resistance and spies alike would be to propose themselves to be part of a mission [1], this allows for a resistance player to reduce the likelihood of a betrayal on the mission, and allows for a spy player to betray the mission themselves. An overall strategy for a spy is to create confusion in the game [1], making it difficult for the resistance members to ascertain which players are spies and which are resistance. A general assumption that players can make is that if everyone voted yes for a mission, it is highly likely that there is at least one spy in the mission [2]. This assumes that the agents are intelligent, and that if two spies were not on a mission, they would reject it and try to include one of the spies, and if one of the spies are on the mission, the spies would vote yes for the mission to proceed with a spy.

Prior implementations of Artificial Intelligence(AI) for *The Resistance* and the methods used have been studied. In a paper by D.P. Taylor, AI for *The Resistance* were constructed based on an expert rules model and opponent modelling techniques [3]. Taylor found that expert-rule models, that is, AI with hard coded conditions would we effective against simple opponents, but against more complex opponents, the existence of the rules themselves mean that the AI might behave optimally since the AI assumes that other agents are at least as intelligent or more intelligent than itself, playing safe when the other agents may not necessarily track every action [4]. Taylor describes one of the biggest challenges in *The Resistance* being the fact that the AI is allocating suspicion and trust levels for each player which themselves may have incomplete knowledge of the game’s current state, assuming the competence of other players led to AI with the tendency to be suspicious of most resistance members [4]. While Taylor [3] implements AI based upon opponent modelling, it would not be relevant in this case as the opponents’ strategies are not known and thus there cannot be AI based upon that.

https://s0.wp.com/latex.php?latex=P%28A%7CB%29+%3D+%5Cdfrac%7BP%28A%29P%28B%7CA%29%7D%7BP%28B%29%7D&bg=ffffff&fg=333333&s=0Bayesian analysis is statistical method using Bayes’ theorem to update the probability of hypotheses as new data emerges [5]. Bayes' theorem states that:

P(A) and P(B) refer to the probability of the events A and B occurring, P(A|B) refers to the probability of event A occurring if event B occurs and P(B|A) is the probability that event B occurs given that A occurs [5]. Bayesian analysis can be implemented to *The Resistance* by constructing an agent which records the probability of each of the other players being spies, all starting at a baseline, the agent would increase an agent’s probability if they participate in spy-like behaviours and reduce the probability if they participate in resistance-like behaviours. The agent can then act based upon each player’s probability with voting and mission proposals. Pike [6] describes Bayesian analysis as a great fit for *The Resistance* as partial information is learnt about each player as the game progresses, round by round. Suppose that there are four players (a,b,c,d) playing the game, initially, each player has the same probability of being a spy calculated as the number of spies divided by the number of players, in this case, there are two spies, and thus each player has a 0.5 base probability of being a spy. If three players (a,b,c) go on a mission and (Success,Fail,Fail), it means that the two spies are either a, b or c and that d is definitely not a spy. Using Bayes’ theorem, we calculate the increased probability of a,b and c being spies. Given that ‘A’ is the event of a player being a spy, and ‘B’ being the event of the observed mission, we can compute P(A|B) using Bayes’ Theorem. To calculate P(B), we determine every possible assignment of the three players as spies, being (a,b), (a,c), (b,c). For each possible assignment, the probabilities are multiplied, thus for each case for the two players in the possible assignment being a spy, with the player not in the set being resistance. P(B) is thus calculated as the sum of the three combinations, . Thus, the Bayes algorithm can be applied with each of the players a, b and c having their probability of being an agent increased to . These calculations occur for each mission and thus can inform a resistance agent on which players to trust and which players to be suspicious of.

**Rationale**

Bayesian analysis is to be implemented into the agent, calculating and storing each player’s probability of being a spy based upon their actions. For example, if a mission fails, the probability of the players which participated in the mission increase as the agent becomes more suspicious of them. The resistance agent should then propose missions with themselves and another player with the lowest probability of being a spy, and subsequently vote no for proposed missions with players with a high probability of being a spy.

While a resistance agent clearly benefits using this model a spy agent can too. A spy agent can use it to their advantage, assuming competence of other players, they can propose missions with spies who have low recorded probabilities and accuse resistance players which high recorded probabilities of being spies. The spy agent can thus manipulate other players into believing falsehoods to win the game.

A spy agent which creates as much noise as possible was considered, creating noisy data with wild accusations, it would assume that other players were tracking and factoring in the amount of accusations other players were receiving, while not growing suspicious of our agent who was doing the accusing. While this concept was scrapped however, it conceived the idea of an agent which factors in the voting and accusation process into its probability calculation. Thus, the two proposed agents would differ in this case to examine the difference that factoring voting and accusations would do to the probability calculation that the agent conducts. Thus, it will be expected that the agent that factors in additional Bayesian calculations will perform better than the base agent.

**Implementation**

The implementation of the two agents will be of an agent who will conduct Bayesian analysis based upon the missions that succeed and fail, and an agent who will conduct additional Bayesian analysis based upon other player’s voting and accusations. The two agents will thus be named Agent Watcher and Agent Listener respectively.

An explanation of the methods are as follows:

**Get\_status:**

This method initialises the spyState hashmap storing the probability of each player being a spy, with the player’s name being the key of the hashmap. It also remembers the names of the other spies if the agent is a spy.

**Do\_Nominate:**

This method will always nominate itself for the mission, as studied self-nomination is the best strategy for both spies and resistance agents [1].

**LikelySpies:**

This method simply returns the name of the player with the highest probability of being a spy.

**LikeyInnocent:**

This method simply returns the name of the player with the lowest probability of being a spy.

**GetLowestProbability:**

This method returns the current lowest probability of being a spy being held in the hash map.

**GetHighestProbability:**

This method returns the current highest probability of being a spy being held in the hash map.

**GetProposedMission:**

For Agent Listener, this stores the last proposed mission for probability calculation.

**Do\_Vote:**

This method simply votes upon the last proposed mission based on if the agent is a spy or resistance.

**spyVote:**

This method is called upon in do\_vote if the agent is a spy. It votes true if the mission has a spy participating in it

**ResistanceVote:**

This method is used in do\_vote if the agent is a resistance member. It votes false if the mission has a confirmed spy or one of the most likely spies, and true otherwise.

**Get\_Votes:**

This method is used in agent Listener to increase the probability of a player being a spy if they were in a mission proposed by a leader who is considered a spy.

**Do\_Betray:**

This method for a spy agent determines when to betray and when not to betray. It will betray if the spies are one mission failure away from winning. This method also accounts for a mission with 3 participants, only choosing to betray of another spy is on the mission.

**Get\_Traitors:**

While this method returns the traitors, it also uses the assumption that if the number of betrayals is equal to the number of players on the mission, then all the players on the mission are spies.

**Do\_Accuse:**

If the agent is a resistance member, it will only accuse players who have been confirmed to be spies. If the agent is a spy, it will accuse the players with the top two probabilities of being spies, but who are resistance members.

**Get\_Accusation:**

This method is used for Agent Listener, increasing the probability of a player being a spy if they are accusing someone with a lower probability of being a spy than themselves.

**UpdateWentAgents:**

This method updates the probabilities of players who participated in the mission

**GetCombinations:**

This method is used in the Bayesian calculation to get all combinations of players who participated on the mission.

**FindPB:**

This method is used in the Bayesian calculation to calculate P(B).

**pBGivenA:**

This method uses Bayes’ Theorem to calculate each player’s probability of being spies.

**updateNonParticiapnts:**

This method updates the probabilities of the players who did not participate on the mission.

**Validation**

**Validation**

There were two components to the agent validation. The agents themselves can be split into two sections: The Bayesian Probability component, and the surrounding logic. In order to validate the Bayesian Probability component, the spyState values were checked across multiple games using Eclipse’s debugger tool. The values for each player were checked every time they were updated, and this test was repeated until the values accurately suggested which players were the spies. A sample consisting of the values of two players in a single game consisting of 5 Watcher agents is included below.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Player Watched: A | | | | | | | |
| Status: Spy | | | | | | | |
| Round | A | B | C | D | E | Mission | Traitors |
| 0 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |  |  |
| 1 | 0.25 | 0.5833 | 0.583 | 0.25 | 0.583 | DA | 0 |
| 2 | 0.161 | 0.875 | 0.875 | 0.18 | 0.827 | EAD | 1 |
| 3 | 0.466 | 0.583 | 0.586 | 0.336 | 0.552 | AD | 1 |
| 4 | 1 | 1 | 0 | 1 | 0 | BDA | 2 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Player Watched: C | | | | | | | |
| Status: Innocent | | | | | | | |
| Round | A | B | C | D | E | Mission | Traitors |
| 0 | 0.5 | 0.5 | 0 | 0.5 | 0.5 |  |  |
| 1 | 0.25 | 0.583 | 0 | 0.25 | 0.583 | DA | 0 |
| 2 | 0.161 | 0.875 | 0 | 0.18 | 0.827 | EAD | 1 |
| 3 | 0.466 | 0.583 | 0 | 0.3395 | 0.5519 | AD | 1 |
| 4 | 1 | 1 | 0 | 1 | 0 | BDA | 2 |

The spies in this game were players A and B, and the game ended with a spy victory in 4 round. It can be noted that player B was consistently considered to be the most spy-like agent by both Player A and Player C. However, by not betraying in the first round, Player A was able to keep a low profile throughout the game. Additionally, it can be seen that in the last round, both spies chose to betray the mission. This gave away their identities, but this is immaterial since the game ended with a government victory as a result of this round.

Since the Bayesian Probability component of both agents is identical, validation of one implies validation of the other.

The second part of the validation process involved testing the two agents against the RandomAgent AI, and against the Watcher AI. By testing first against RandomAgent, and then Watcher, the strengths of the two agents can be seen. It would be expected that Watcher would perform better against the extremely dumb RandomAgent, whereas Listener should perform better against the enhanced ability of Watcher. Watcher was chosen as a baseline since its performance is much more independent of the quality of the other players present. The results of these tests can be seen below:

Ideally, tests would have been performed against other quality agents in order to properly optimise the two agents. It is hard to evaluate the performance of an agent when its only competition is itself; if all players have the same AI, then one subset will always win, and another will always loose. What it can indicate however is whether the spies or the resistance are favoured. Since there are always fewer spies than resistance members, a win percentage below 50% indicates that the spies are favoured. But further evaluation is difficult without an external point of comparison. The only alternative agents available were the completely random RandomAgent, and an interface allowing human play. Access to other, competitive, agents would have allowed for much better optimisation; but the competitive and assessed nature of this project made obtaining other agents an awkward proposition. Once the tournament is complete, and a large variety of competitive agents would be available, Watcher and Listener would be able to be further optimised, but this was impossible to perform before the tournament itself.

**References**

[1] P.Migas (2012, Oct 5). "Looking for some play tips", *Board Game Geek*. [Online]. Available: https://boardgamegeek.com/thread/865343/looking-some-play-tips. [Accessed: 31- Oct- 2016]

[2] Andrey (2012, Oct 4). "Looking for some play tips", *Board Game Geek*. [Online]. Available: https://boardgamegeek.com/thread/865343/looking-some-play-tips. [Accessed: 31- Oct- 2016]

[3] D. P. Taylor, “Investigating Approaches to AI for Trust-based, Multi-agent Board Games with Imperfect Information; With Don Eskridge's ``The Resistance'',” BscH. dissertation, Dept. Comp. Sci., Univ of Derby., Derby, United Kingdom, 2014.

[4] D. P. Taylor, "AI for The Resistance", *Confect's Codex*. 2014 [Online]. Available: https://dtconfect.wordpress.com/projects/year-4/resistance/. [Accessed: 31- Oct- 2016]

[5] K. Cowles, R. Kass and T. O'Hagan, "What is Bayesian Analysis? | International Society for Bayesian Analysis", *International Society for Bayesian Analysis*, 2016. [Online]. Available: https://bayesian.org/Bayes-Explained. [Accessed: 31- Oct- 2016]

[6] L. Pike, "Viva La Resistance! A Resistance Game Solver", *A Critical Systems Blog*. 2016 [Online]. Available: https://leepike.wordpress.com/2016/02/08/viva-la-resistance-a-resistance-game-solver/. [Accessed: 31- Oct- 2016]