Research Report

Agent Implementation for The Resistance game

CITS3001

Semester 2, 2021

Name: Josephine Bienes

Student ID: 22511218

Due Date: 22 October 2021, 5pm

Table of Contents

1.0 Introduction

The unit has given a project to research on methods or techniques to use for an implementation of an agent that plays on the game "The Resistance", created by Don Eskridge (Taylor, 2013-2014). The game revolves around five to ten players where a third (rounded up) of the players, at the start of the game, are chosen as spies with the rest as part of the resistance. The goal of the game is for the resistance to successfully complete three missions, or the spies successfully fails three missions. The players sent to these missions are chosen by an elected team leader that varies each vote time at the start of every round. Game lasts around 15-30 minutes, classic rules (Wikipedia, 2021).

In this research, the only rule considered is the classic and will be used for the two agents that will be implemented as part of this project, in a five-player setting.

It should be known that the game is an imperfect game where, if the player was part of the resistance, initially the role of each player is not known. For each mission round, it is also partially observable as the players would not have the ability to determine directly who failed them; the game only tells the players the total failures. The game is not stochastic as there is no random element involve that directly influences the result of the game, say a dice roll on Backgammon for example.

1.1 Scope of Research

For this report, the research will focus on the performance of an agent implemented using Bayesian Reasoning to get the suspicion values for each player against another agent without the use of it, in a five-player game setting. The agents will be known as Baerule – with the Bayesian Reasoning – and BasicRebel for the other agent. The primary difference between the two agents is that the former will be using some sort of probability modelling that contributes to the identification of which players are highly likely to be spies, thus avoiding them to be voted in the mission as possible. Whilst the latter will only be implemented with the most basic logic of the game where no modelling is applied – no statistics, rather the suspicion values will be incremented or decrement by one whole integer.

The main concept that will be analysed is the ability of both agents – as a resistance player – to determine which players are spies. The game outcome, whether the resistance won or loss, will not be taken into consideration as part of this analysis.

Aside from comparing Baerule's performance of determining spies to BasicRebel, both agents will also be analysed against Random agents in a five-player setting.

2.0 Literature Review

The following sections contains the types of techniques and theorems that could be used to implement an agent for The Resistance, to be used with the research scope. Sections below will be looking into the usage of Opponent Modelling with the use of Bayesian Networks and Monte Carlo Tree Search, and its probability modelling through Bayesian Reasoning and approximation sampling algorithms – Markov Chain Monte Carlo specifically the Gibbs Sampling.

2.1 Opponent Modelling

Based on various sources and searches conducted, Opponent Modelling (OM) in a general sense is used to identify the strategies, policies, and goals of the opponent in a multi-agent and adversarial environment. The key point for this modelling is to use probabilistic and statistical models to determine the said attributes of the opponent and counter them or exploit that said weaknesses (Taylor, 2013-2014).

OM has been used as a technique to implement in AI's for various of board games, specifically for games that involves complex predictions on future actions of opponents in partially observable, and imperfect game. Classic games include Poker, and computer games relating to Real-Time Strategy (RTS), such as Starcraft (A. Torkaman, 2018).

In terms of The Resistance, OM would be able to extract data from each game session to update the models based on the statistical behaviour of the opponents – the frequency of voting for missions, players in missions that failed, or even on the frequency of spies, when in a mission, chooses to sabotage. For the purposes of this research, the hope is for the agent to be able to build a good quality of its model to accurately identify which of the players are spies, with the use of Bayesian Reasoning (see section 2.2.1).

2.1.1 Bayesian Networks

To be able to represent the probabilistic model needed as part of the Opponent Modelling, Bayesian Network (BN) will aide in representing the dependencies among the variables that will be used as part of the analysis in determining the opponent strategies. It can concisely represent any full joint probability distribution (Stuart Russel, 2020).

In general, Bayesian Network is a directed acyclic graph where each node contains quantitative probability information. Each node corresponds to a random variable, which can be continuous or discrete. The links or arrows that have direction, are to connect pairs of nodes. Example would be if node A links to node B, node A is then a parent of B, but do note that the graph itself has no directed cycles. Thus, why it is a directed acyclic graph (DAG). Lastly, each node will have an associated probability information that will somewhat quantify the consequences of the parents on the node using several parameters (Stuart Russel, 2020). An example of a BN can be seen on figure one.

To elaborate the set of nodes and links, specifies the conditional independence relationships that are held in the domain. Typically, the definition of an arrow in the graph is that node A has a direct influence on node B. This suggests that causes are effects of parents. Additionally, the probability information associated to each node takes in the form of what is called as a conditional probability table (CPT) (Stuart Russel, 2020).

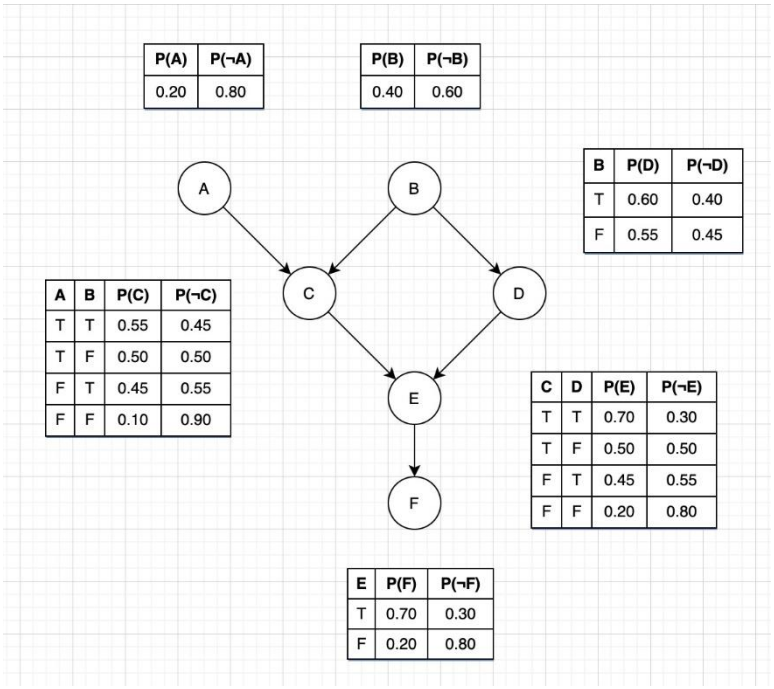

Figure 1, Example of a Bayesian Network (Poppe, 2020)

2.1.2 Monte Carlo Tree Search

Unlike Bayesian Networks, which serves more of a data structure to store the joint probability of nodes, the Monte Carlo Tree Search (MCTS) is a search algorithm that determines decision processes for the possible actions that the agent could choose. Generally, MCTS is used to solve the game tree. By formal definition, it is a heuristic and probabilistic search algorithm that adopts the principles of reinforcement learning along with the classic tree search (Roy, 2019).

The main idea behind this algorithm uses a principle known as “exploration-exploitation trade-off” where it exploits the strategies and actions that is the best at a current state but will continue to explore alternative decisions locally. Through this exploration, the algorithm will try to find if an alternative decision is better than the current decision that was chosen. By doing so, it expands the tree’s breadth to ensure that it can find the optimal path. However, the expansion can cause the algorithm to become inefficient, and so through exploitation, the search will then choose a path with the greatest estimated value. With this, the tree’s depth will be expanded – more of a greedy approach (Roy, 2019). To ensure the trade-off between exploration and exploitation are balanced, the algorithm implements an upper confidence bounds formula (UCB) (Stuart Russel, 2020).

For each iteration, four steps are involved in the process, and these are selection, expansion, simulation, and back-propagation. Selection traverses the current tree from the root node with a specific strategy. An evaluation function will the cause the strategy to select optimally the nodes with the highest value. The UCB formula will be applied as it traverses on the tree. Once child node has been selected, containing the highest estimated value, it will then proceed to the expansion phase. This second phase adds the new child node chosen in the selection process as part of the tree search. The simulation process will then perform a playout from the new child node, alternating actions/moves on that child node for players according to a specific playout policy. Note that these actions are not added in the tree search. The result in the simulation process will update all the search tree nodes up to the root node, thus why the back propagation (Stuart Russel, 2020).

2.2 Probability Modelling

The following sections will involve in acquiring the probability information that is needed for each node as part of the Opponent Modelling. As stated before, the probability will be modelled using Bayesian Reasoning as part of this research report but will cover other types of approximation inferences to calculate the joint probability.

2.2.1 Bayesian Reasoning

The idea behind Bayesian Reasoning or Inference is applying probability theory to inductive and abductive reasoning. It uses the interpretation of probabilities as the expressions of uncertainties within agents. Uncertainties involves the environment, in this case, the game environment amongst the other players' behaviour. For this reasoning, the Bayes' Rule or Theorem can be used to calculate the probability of various degrees of beliefs upon the illustration of a new evidence (nLab, 2021). Equation below presents the equation of Baye's Rule.

$$P(h|e) = P(e|h) \cdot \frac{P(h)}{P(e)}$$

Hypothesis is h and evidence is e in the equation above. General idea is that when an observation is done on e then the degree of belief in h changes from $P(h)$ to $P(h|e)$ – known as conditionalizing (nLab, 2021). In a more formal definition, the Bayesian Reasoning is derived from the posterior probability because of a prior probability and a "likelihood function". In reference to the equation above, $P(h)$ is the prior probability (approximation of the probability of the hypothesis). $P(h|e)$ is considered the posterior probability – the probability of h given e. $P(e|h)$ would be the probability of observing e given h and is termed the likelihood, which involves a function of e with h fixed. $P(e)$ termed as the marginal likelihood (Wikepedia, 2021).

2.2.3 Markov Chain Monte Carlo – Gibbs Sampling

On the other hand, the Markov Chain Monte Carlo (MCMC) generates a sample by creating random changes to the preceding sample; an algorithm that for every variable, specifies a value associated with it, and generates the next state by randomizing some changes to the current state (Stuart Russel, 2020). In general, MCMC draws samples where the next sample depends on the existing sample (Brownlee, 2019). A more common and simple approach would be the use of Gibbs Sampling as an algorithm for MCMC, which considers multivariant distribution.

Gibbs Sampling or Sampler is an algorithm that obtains a sequence of observations that are approximated from a probability distribution, which this sequence can be used to approximate the joint distribution to the Bayesian Network at section 2.1.1 (Wikepedia, 2021). In terms of Bayesian Networks, this sample algorithm initially begins with an arbitrary state and generates the next state by randomising a sampling value towards one of the variables that are not part of the evidence. Additionally, evidence variables, part of the "given by this event", will be fixed at their observed values (Stuart Russel, 2020).

3.0 Methods used in Agents

Each of the agents will use suspicion values to identify which of the players could potentially be a spy. The main difference between the two agents is that the suspicion value gained by BasicRebel is very simplistic – an incremental of a whole integer based on specific parameters of the game. Whereas Baerule uses a probability distribution with Bayes' Rule/Theorem to gain the suspicion value. Again, based on similar parameters.

To have some consistency for the analysis in both agents, the main parameters that contributes to the outcome of the suspicion values are the players sent in the mission, the number of fails in that mission, the leader that proposed the mission, and the outcome of the mission whether it has failed or not. The voting outcomes for each voting sessions in each round will not be considered as part of the analysis. Furthermore, the outcome of the game itself, whether the resistance has won or not, will also not be considered as part of the analysis.

The following sections below gives out more details on the two agents.

3.1 BasicRebel

This agent consists of very logical statements of the game. Due to comparison purposes, this agent will also have spy actions/behaviours implemented.

There are specific rules that this agent will consider which influences the resulting suspicion value towards other members in the game. These rules are in reference to how an amateur player would play the game, in a five-player setting. Rules specified, as a resistance member, are:

- On the very first round, any proposition on the teams will always be voted yes to start the game.
- Agent will keep a taboo list where players proposed in the mission, and the leader who proposed it will add in frequency of uncertain value where the suspicion value is added up.
- The higher the value, the more untrustworthy the players are.
- If the mission fails, then the players in the mission will have their respective suspicion values increment to 1 if the number of fails is 1, otherwise increment to 3. If the leader proposes the mission is also in the mission that failed it, then its suspicion value will increment to a further 2, otherwise an increment of 1.
- When the mission passes, all players in the mission and the leader's suspicion values will decrement by 1.
- During the voting session, if on the fifth vote then the agent doesn't have a choice but to vote yes.
- It will only vote yes on missions with players that all have less than 4 suspicion values (0-3).
- If on the losing side, it will become paranoid and will only vote yes on missions if it was the leader that proposed it or that it's in the mission itself. If the previous conditions are false, then It will only vote yes if the players proposed have a suspicion value less than 2 (0-1). Although, this is irrelevant if it's on the fifth mission vote.
- Proposing a mission will have the players be sorted in an array according to their suspicion values. The array will be sorted in an ascending manner where the first few players have smaller suspicion values. The first ever player added to be proposed in the mission is the agent itself, then adds in the remaining players needed for the mission from the sorted array.

For research purposes, this agent will also implement a basic spy behaviour that also has its own set of rules when it betrays a mission, how it votes, and how it will propose players in the mission. These rules are:

- It will never betray missions on the first round.
- It will highly likely betray rounds 2, 4 and 5.
- If the other spy is in the third round, it will not betray unless the spies are losing at that point.

- It will only vote yes on the mission if it was the leader that proposed it, if itself is in the mission, if another spy is in the mission, the vote count is at 5, or if it was the first round.
- When it proposes players in a mission if the spies are losing it will always add itself. The rest of the players chosen will be at random.
- This goes the same if the number of fails required is greater than one, it will always add itself then add the remaining amount of spy players needed to fail the mission. Again, the rest of the players remaining to be added are chosen at random.

3.2 Baerule

This agent will contain the same set of parameters that influences the suspicion values. It will also have very similar rules with BasicRebel in terms of inferring to the specified events; refer to BasicRebel's set of rules when it is a resistance member. Before looking into the changes made, here are some concepts that are added in this agent.

Since this agent is using Bayes' Rule to get the suspicion value, certain numerical values are needed to calculate the posterior probability. First thing to get is the prior probability for the formula. Gathering some people's thought process, the prior probability of a player being a spy will depend on each round and the number of fails for that round. The table below are the prior probabilities for each round; inferred from logical thought process with other people's thought process.

No. of Fails	Game Rounds				
	1	**2**	**3**	**4**	**5**
1	1/2	1/3	1/2	1/3	1/3
2	1	2/3	1	2/3	2/3

Table 1, Prior probabilities of a player being a spy given the round and the number of fails in that round.

The other numerical value needed is the likelihood of an event happening. In this case the posterior probability is the probability of a player being a spy given that the mission failed, so the likelihood is the probability of a mission failing given that there is a spy. Table 3 shows the likelihood of a spy in the mission failing that said mission. These probabilities are dependent on the rounds the agent is in. Again, these values are inferred logically and from other people's thought process.

Game Rounds				
1	**2**	**3**	**4**	**5**
0.10	0.85	0.30	0.80	0.90

Table 2, Likelihood probabilities of a spy failing the mission given the round.

In addition, to be able to make the formula work well even though resistance players will never fail a mission, it has been given a likelihood of 0.1, which is constant throughout the game.

With these lookup tables, we are then able to calculate the posterior probability. Example, for the second round, player 1 proposed [1,2,4] to be sent in the mission, and it failed with 1 fails. These players currently have these suspicion values 0.0, 0.5, 0.2, respectively. For player 1, prior probability would be 1/3 and since its current suspicion

value is zero, then the numerator (see 2.2.1 equation) will be $^1/_3 \times 0.85 \approx 0.283$. The denominator, marginal likelihood, will then become $0.283 + (^2/_3 \times 0.1) \approx 0.35$. With this, the posterior probability of player 1 is approximately 0.81. If player 1 had a suspicion value greater than zero like player 2 having 0.5 suspicion value before, then the old suspicion value is multiplied in part of the numerator ($^1/_3 \times 0.85 \times 0.5$). Below represents the previous example for player 1 with Bayes' Rule where mission failed, and mission succeeded.

$$P(S|Mission\ Failed) = \frac{(^1/_3 * 0.85)}{(^1/_3 * 0.85) + (^2/_3 * 0.1)} \approx 0.81$$

$$P(S|Mission\ Succeeded) = \frac{(^1/_3 * 0.15)}{(^1/_3 * 0.15) + (^2/_3 * 0.9)} \approx 0.077$$

The leader that proposed the mission will also be affected with the additional parameters of whether the leader was in the mission or not. Given that the mission failed, if the leader was not in the mission, then the leader's prior probability will be 1/5. However, if the round was on round 1 or 3 where the mission size is only 2, and the number of fails is 2, then the leader's suspicion value is -1.0; guaranteed that the leader is not a spy, so no further changes of the suspicion value is needed.

The following are the changes involved to the rules that was implemented in this agent:

- Calculation of the suspicion values is now determined with Bayes' Rule rather than incremental integer values.
- It will now only vote yes for missions with players' suspicion values no greater than 0.60 (0 – 0.6).
- If on the losing side, it will get paranoid and only vote yes if it was the leader or that the suspicion values for the players proposed is no greater than 0.30 (0-0.30).
- If the agent is in the 3 player missions and ended up having 2 fails in any of those missions, then the spies are guaranteed to be the other two players in that mission, which brings their suspicion values to 1.0.

4.0 Performance

Before conducting the experiments, a prediction was made on what the data would reflect in terms of the performance of the agents identifying none of the spies, at least identified one spy, and identifying both spies. The prediction was that the Baerule agent would perform better in terms of identifying spies than BasicRebel due to the probabilistic model that it uses. This prediction relates to all the three environments in the following sections.

The data that are shown in the tables on the following sections will be based on the aspects stated above over the number of games that were simulated. Both sections 4.1 and 4.2, there will be a total of 200 simulation rounds of the game to infer from.

A simulation round in this case will consist of having 2000 games played consecutively, and once the simulation has finished, the data will then be extracted and logged into the respective files. Overall, the experiment will have 200 trials, and in each trial consist of 2000 games played containing both agents in different environments.

A java class for simulating the experiment is provided in the zip file named "Experiment.java". The file will contain the logic in which the experiments were done in and can be run for replication – be wary on how the simulations can be run as it will differ in the different environment settings.

4.1 Amateur Agents environment

Spies Identified	0-spies		1-spies		2-spies	
Agents	**Baerule**	**BasicRebel**	**Baerule**	**BasicRebel**	**Baerule**	**BasicRebel**
Average	448.73	456.71	615.23	607.81	936.05	935.49
Percentage (/2000)	22.44%	22.84%	30.76%	30.39%	46.80%	46.77%

*Table 3, Average number of times Baerule and BasicRebel identified zero spies, one spy, and both spies over 200 trials of 2000 games. Also contains the percentage of the performance with the given aspects, for 2000 games (average/2000 * 100). This is in an amateur environment*

From the data in table 4, Baerule seems to be performing slightly better than BasicRebel in terms of identifying spies. With a 0.03% difference in identifying both spies and 0.37% difference in identifying at least one spy. Even with the slight difference, the prediction still stands that Baerule performs better than Basicrule when identifying spies in an amateur environment.

Although, 46.80% identification is still considered incredibly low considering that the environment the agent was in was simple, and to put it blunt, quite basic. The initial expectation for the accuracy was at least close to 55% in identifying both spies for Baerule. Additionally, it was a surprise for BasicRebel to perform quite well. Having that 46.77% identification of both spies was unexpected as the initial expectation for the accuracy was well below 40%.

For further comparison, if both percentages for the identification of one spy and both spies were combined, Baerule and BasicRebel will then have these following general identification percentages:

$$Baerule: \frac{615.23 + 936.05}{2000} * 100 \approx 77.56\%$$

$$BasicRebel: \frac{607.81 + 935.49}{2000} * 100 \approx 77.17\%$$

As a general comparison of a binary like outcome of whether the agents identified the spies or not, both were able to do so with a decent accuracy rate. Again, Baerule performed slightly better with 0.39% difference.

4.2 Random environment

Spies Identified	0-spies		1-spies		2-spies	
Agents	**Baerule**	**BasicRebel**	**Baerule**	**BasicRebel**	**Baerule**	**BasicRebel**
Average	448.73	456.71	782.51	817.98	705.10	576.46
Percentage (/2000)	22.44%	22.84%	39.13%	40.90%	35.25%	28.82%

*Table 4, Average number of times Baerule and BasicRebel identified zero spies, one spy, and both spies over 200 trials of 2000 games. Also contains the percentage of the performance with the given aspects, for 2000 games (average/2000 * 100). This is in a random environment.*

Considering the events that the agents use to update the suspicion values on each player were now random, the expectation identification percentage for spies were low, where Baerule would have ~30% accuracy identification for both spies, and with the identification of one spy slightly higher, ~35%. On the other hand, BasicRebel's is even lower with ~10% on identifying both spies and ~15% for identifying at least one spy.

From the data in table 5, the initial expectation values were underestimated for both agents. Starting with Baerule, this agent ended up having an accuracy of 35.25% in identifying both spies, and 39.13% in identifying at least one spy; 5.25% and 4.13% higher than the initial estimate. With BasicRebel, its identification percentage accuracy was 18.82% and 25.9% higher than the initial estimate.

Unexpectedly, BasicRebel performed quite well in identifying at least one spy compared to Baerule, having 1.77% difference. Although, Baerule did perform more at identifying both spies than BasicRebel with 6.73% difference, and in reference to the game goal, this is considered better. Below are the total identification percentages for both agents.

$$Baerule: \frac{782.51 + 705.10}{2000} * 100 \approx 74.38\%$$

$$BasicRebel: \frac{817.98 + 576.46}{2000} * 100 \approx 69.72\%$$

Overall, the prediction still stands that Baerule performed better than BasicRebel, and from the results above, there was a difference of 4.66%. The performance of Baerule was significantly better than BasicRebel when it comes to random agents, and still maintained that accuracy above 70%.

5.0 Conclusion

Using probabilistic modelling for changing the suspicion values in identifying spies, deemed to have more accuracy (however slightly) than having suspicion values change based on incrementing or decrementing integer values. Performance is more distinct in the random environment, where the difference in their accuracy was more so. Furthermore, from the data previously, the prediction stands where the agent using Bayesian Reasoning to identify the spies performed better in identifying spies as an overall than the agent without it.

References

A. Torkaman, R. S., 2018. *Robust Opponent Modeling in Real-Time Strategy Games using Bayesian Networks.* [Online]
Available at:
http://jad.shahroodut.ac.ir/article_1266_94b0747ecd9eeccad88d6d89fe74fe8b.pdf#:~:text=We%20use%20a%20Bayesian%20network%20to%20construct%20the,Using%20the%20Bayes%20rule%2C%20the%20player%20can%20infer

Brownlee, J., 2019. *A Gentle Introduction to Markov Chain Monte Carlo for Probability.* [Online]
Available at: https://machinelearningmastery.com/markov-chain-monte-carlo-for-probability/

nLab, 2021. *Bayesian Reasoning.* [Online]
Available at:
https://ncatlab.org/nlab/show/Bayesian+reasoning#:~:text=Bayesian%20reasoning.%20Idea.%20Bayesian%20reasoning%20is%20an%20application,some%20notion%20of%20objective%20chance%20in%20the%20world.

Poppe, M., 2020. *Bayesian Networks.* [Online]
Available at: https://www.mauriciopoppe.com/notes/mathematics/probability/bayesian-networks/

Roy, R., 2019. *ML | Monte Carlo Tree Search (MCTS).* [Online]
Available at: https://www.geeksforgeeks.org/ml-monte-carlo-tree-search-mcts/

Stuart Russel, P. N., 2020. *Artificial Intelligence: A Modern Approach, eBook, Global Edition.* 4th ed. s.l.:Pearson.

Taylor, D. P., 2013-2014. *Investigating Approaches To AI For Trust-Based Multi-Agent Board Games With Imperfect Information With Don Eskridge's "The Resistance".* [Online]
Available at: https://teaching.csse.uwa.edu.au/units/CITS3001/Resistance.pdf

Wikepedia, 2021. *Bayesian Inference.* [Online]
Available at: https://en.wikipedia.org/wiki/Bayesian_inference

Wikepedia, 2021. *Gibbs Sampling.* [Online]
Available at:
https://en.wikipedia.org/wiki/Gibbs_sampling#:~:text=In%20statistics%2C%20Gibbs%20sampling%20or%20a%20Gibbs%20sampler,multivariate%20probability%20distribution%2C%20when%20direct%20sampling%20is%20difficult.

Wikipedia, 2021. *The Resistance (game).* [Online]
Available at: https://en.wikipedia.org/wiki/The_Resistance_(game)