CITS3001 Project,

The Resistance:

Yunhui Rao: 22551999,

Chunguang Xie: 22555027.

Literature review

The Resistance is a modern board game focusing on hidden information, deductive reasoning, and the players themselves rather than the game mechanics. The game is played to a maximum of 5 turns, or rounds, or missions. Players vote on approving each single mission, and two groups have to turn 3 missions to their favor. Two roles are in the game, Spies, who have complete information, know hidden identity of all players, and can sabotage mission to win when they are selected in one, and Resistances, who need to figure out who they can trust, work together to find out spies in the game without fully trusting anyone, and make informed decisions on voting and proposing teams on missions. More information we have, we can use epistemic logic better to exclude impossible reality. The greater availability of information gathered could also help us to explore a broader range of approaches. (van Ditmarsch & Kooi).

Tree-search is a common technique used for AI agents in playing games. Game trees are a method of representing games and similar decision processes as directed graphs. Each node of the complete tree is a reachable state from the root and all actions available at each of these nodes. Usually, number of nodes describes the complexity of the game. Monte-Carlo method uses repeated random sampling of a distribution. Monte-Carlo Tree-Search does not require a complete game tree to make informed decision in normal tree search algorithm. The game tree, however, is constructed from the results of plenty of play and actions at decision nodes for player are randomly selected. And exploration of the tree is conducted by running multiple simulations to terminal states. Agents could try different sensible player configurations and predict others' action to get the optimized strategy and take actions.(Daniel P. Taylor)

Observation-based strategy on incomplete information relies on the past sequence of observations in game. Logic deduction, which uses game results observed before, for example, teams failed a mission, could works in game with imperfect information after the agent has accumulated certain amount of information of the game. The agent is able to eliminate uncertainty to some degree by analyzing and cross comparing data collected. And it is possible to produce useful information, for example, players' identity and heuristics supporting them take some certain actions. (Doyen and Raskin)

Opponent modelling, which is very common technique for AI in playing game like Poker, Chess etc., is a technique applying statistical and probabilistic model to opponents of the agent. Agents learn their opponent's behavior during long term game and build models. The model is aimed to help agents to select strategies countering their adversaries' or exploit their weakness. Agents could observe adversaries' behavior and fit their behavior into models created by the agent, so that it is able to predict actions of its opponent and it can exploit that to take actions have more advantages.

Games are consisted of certain rules. Expert rules specify what actions should to take at given certain conditions and are usually based on some intuitive, human reasoning about what appears suspicious, what would likely be disadvantage, and what can never be, or always will be profitable.

Employing computer learning technique to optimize the decision making is possible and interesting. With unsupervised reinforcement learning, co-evolution, and other learning techniques, agents are able to adjust its actions and change the strategy it used to gain more rewards.

Design description and rationale of selected technique

The Resistance, however, is not like most of other games. It is an imperfect information game and contains a lot of uncertainty.

In tree-search approach, for resistance, agent do not know other identity of other agents, and do not know what their gain is and lose, so that they can not make informed decision which maximum its gain. Also, the search space could be extremely large, and, in the tournament, there exists a time limit for our agent to take action. In the situation of without know identity of other agents, most of simulations produced by Monte-Carlo Tree-Search method could useless and meaningless except wasting time and computational power.

Opponent modelling could help agents learning their opponents' behavior in long term, so that agents can predict opponents' action, exploit their weakness, and take different, strong, and targeted actions helping them to get advantages. While in the tournament, our agent will compete with different and sizeable opponent, it is hard to track and build model for every different opponent. More than that, one single agent could be treated as different player, which decreases the practicability of opponent modelling in this project. Besides, even if our agent encountered the same agent met before, that agent could possibly had changed its strategy, which will dysfunction the model out agent built.

We wish our agent is able to figure out roles of other agent in one game, propose a team highly likely turn the mission in its favor, and vote for correct teams. In the first agent we designed, we apply the expert rules which we think is good enough to have a good performance. These rules help our agent, as a spy which has more information, to disguise like a resistance as much as possible, to attract less suspicion after sabotage missions , propose a team could fail the mission and get others approved. Combining with Role configuration over single game it observed, our agent could remember teams which failed a mission before, and it is able to decide how likely a team would fail and vote yes for team looks much safer by cross comparing the data and deduct information they need to know.

Expert rules, however, would normally limit the capability of an artificial intelligence as performance of decision made by hard code always limited by the intelligence of its developer, and we may meet some agents can change its strategy and are very good at observing as well. We apply a suspicion score modification and behaviors tracking technique to our second agent. Our agent is able to learn other player’s behavior and make adjustment by reinforcement learning in long term. Our agent assigns a suspicion score to every other agent and adjust this score during one game helping it identify other agents’ roles. In long term of playing, the agent could know the game better and expected to have a better performance.

We expect our first agent has good performance while playing with other simple agents, like random agent and agent with less knowledge. We also expect our second agent would improve its performance after certain time of learning. We wonder if learning really works for agents and whether an expert without keeping improving himself will be prevailed by others.

Validation

To test the performance of our agent code, we ran 10000 games with 5 agents between Random Agent and itself, and between Random Agent and one of our agents. Our agents include suspicion score agent with reinforcement learning, suspicion score agent without reinforcement learning, and expert rules agent. In the games, there are 4 Random agents and the other is one of the agents stated above to control variables. For our suspicion score agent with reinforcement leaning, we divided the games into 10-game tournament, 50-game tournament, 100-game tournament and 500-game tournament, to show whether the performance is improved with reinforcement learning.

Name	No. of game in one tournament	Spy Plays	Spy Wins	Res Plays	Res Wins	Win Rate	Spy Win Rate	Res Win Rate

Random Agent	10000	4065	801	5935	4803	0.56	0.2	0.81
Suspicion score without reinforcement learning	10000	3982	2533	6018	5177	0.77	0.64	0.86
Expert Rules	10000	3974	2345	6026	5195	0.75	0.59	0.86
Suspicion score with reinforcement learning	10	4018	2555	5982	5138	0.77	0.64	0.86
Suspicion score with reinforcement learning	50	4040	2628	5960	5109	0.77	0.65	0.86
Suspicion score with reinforcement learning	100	4017	2662	5983	5152	0.78	0.66	0.86
Suspicion score with reinforcement learning	500	4021	2754	5979	5157	0.79	0.68	0.86

From the table, it is clear that all of our agents play much better than Random Agents, especially when our agents play the spy role. This shows that using expert rules or suspicion score methods will both improve the chance of winning significantly.

Also, for our suspicion score with reinforcement learning, it is noticed that the winning rate does increase a bit as the number of games in one single tournament increases, and the increase in winning rate comes from the increase in winning rate of spy role, as we have only implemented reinforcement learning for the probability of betraying when the number of spies in a mission is greater than the required of fail vote to fail a mission. This shows the reinforcement learning can improve the performance of agents. The reinforcement leaning can also be implemented in other part of the agent, such as the suspicion scores the agent gives to others, and we believe that this will improve the performance of the agent as resistance. However, it is a pity that we could not implement such features due to time limit.

We also found that the winning rate of an agent not only depends on its own design, it also depends what strategy its opponents use. We ran two tournaments each consisting of 10000 games. The first tournament is played with four suspicion score agents without reinforcement learning and one expert rules agent, and the second is played with one suspicion score agent without reinforcement leaning and four expert rules agent.

Name	No. of game in one tournament	Spy Plays	Spy Wins	Res Plays	Res Wins	Win Rate	Spy Win Rate	Res Win Rate
Suspicion score without reinforcement learning	10000	4075	3160	5925	1528	0.47	0.78	0.26
Expert Rules	10000	3890	2478	6110	1031	0.35	0.64	0.17

Tournament with four suspicion score agents and one expert rules agent

Name	No. of game in one tournament	Spy Plays	Spy Wins	Res Plays	Res Wins	Win Rate	Spy Win Rate	Res Win Rate
Suspicion score without reinforcement learning	10000	3940	3063	6060	521	0.36	0.78	0.09
Expert Rules	10000	3997	3519	6003	920	0.44	0.88	0.15

Tournament with one suspicion score agent and four expert rules agents

It is obvious the winning rates of two agents changed significantly with respect to the change of opponents. Their winning rate almost exchange when the opponents changed. In this situation, we can not say that an agent with fixed strategy is strong. The agent only plays well when facing specific opponents. This is where the reinforcement learning plays a role. By implementing reinforcement learning, the agent can change its probabilities of taking certain actions depending what kind of opponents it's facing. In our implementation, only making use of reinforcement learning in a single place can improve the winning rate. We believe that using reinforcement learning nicely in other parts will increase the winning rate significantly and will adapt the agent to opponents quickly.

References

van Ditmarsch, H., & Kooi, B. *One Hundred Prisoners and a Light Bulb. Switzerland: Springer International Publishing.*

Doyen, Laurent, and Jean-François Raskin. *Games with Imperfect Information: Theory and Algorithms* .http://www.lsv.fr/~doyen/papers/Games_with_Imperfect_Information_Theory_Algorithms.pdf

Daniel P. Taylor. *INVESTIGATING APPROACHES TO AI FOR TRUST-BASED, MULTI-AGENT BOARD GAMES WITH IMPERFECT INFORMATION WITH DON ESKRIDGE'S "THE RESISTANCE"* https://teaching.csse.uwa.edu.au/units/CITS3001/Resistance.pdf

Pfeiffer, M. [2004], 'Reinforcement learning of strategies for settlers of catan', http://www.igi. tugraz.at/pfeiffer/documents/LAG-37_pfeiffer_2004.pdf.