G31 - Final Report - Multi-Agent Spy Tag

Carolina Pereira
IST192433
carolina.m.pereira@tecnico.ulisboa.pt

Daniela Castanho
IST192442
daniela.castanho@tecnico.ulisboa.pt

Manuel Domingues
IST182437
manuel.domingues@hotmail.com

ABSTRACT

SpAI Tag is a multi-agent game that consists of an environment with three types of agents designated Police, Criminal, and Citizen Agents. We conducted an experiment on this multi-agent system, using three different agent architectures for the Police Agents: reactive agents, deliberative agents, and cooperative agents with social conventions. In this report, we analyze the results of the experiment and discuss the causes of the results and the relations between observations, as well as the overall performance of each implemented architecture. We conclude that the most fitting architecture for this playground game out of the ones we implemented is the cooperative architecture.

1 INTRODUCTION

1.1 Motivation

Playground Games are a fun and safe environment to test and train multi-agent systems which can be mapped to real-world problems. One example of these playground games is the "Spy Tag" or "Cowboys and Indians", known in Portuguese as "Polícias e Ladrões" which consists of a tag game in which there are "good" agents who try to capture the "bad" agents. Multi-agent systems can be used for patrolling, meaning training them to identify suspicious behaviors and pursue the suspicious subject brings value to this research field.

1.2 Problem Definition and Objectives

The goal of the project is to study different agent architectures so that the Police Agents have the best possible performance. For this, we created a dynamic system with multiple agents that interact with each other independently. There are three types of agents spread on a simple grid map: Police, Criminal, and Citizen Agents. The criminals' objective is to rob all the citizens and not get caught by the police, resulting in an overall loss if this happens. To win the game, the Police Agents need to capture all the criminals before they rob all citizens. Additionally, if none of the objectives are met within a time limit of sixty seconds, the game ends without anyone winning. The architecture and approach of the Police Agents to catch criminals can change depending on initial user input. We implemented three different Police Agent architectures: reactive agent, deliberative agent and cooperative agents with social convention. However, the criminals' and citizens' behavior is always the same. With this in mind, we aim to compare and analyze the different results according to the different architectures taken by the Police Agents. We expect to see the best results with the cooperative approach.

AASMA Project, IST, 2022, Lisbon, Portugal

https://doi.org/

2 APPROACH

2.1 Agents

There are three different categories of agents in our environment:

- **Citizen** - Simple wandering agents, each one owning an object which can be stolen by a Criminal Agent. The Citizen Agent features an awareness attribute used to determine whether they realize who robbed them or not.
- **Criminal** - Wander the world looking for citizens to rob and evade the police. The Criminal Agent features a skill level that determines their success in stealing an item.
- **Police** - Identifies and arrests Criminal Agents in the environment and communicates with other Police Agents to warn about suspicious agents, when in Cooperative Mode. The Police Agent features a morality attribute used to aid the decision of accepting or rejecting a bribe from a Criminal Agent.

The Police Agents cannot visually tell apart Criminal Agents from Citizen Agents, unlike Criminal Agents who can identify Police Agents and Citizen Agents.

2.1.1 Sensors. All agents have sight which allows them to detect other agents around them in a given radius. Sight is used by: Citizen Agents to identify Criminal Agents; Criminal Agents to identify both Citizen and Police Agents; Police Agents to identify suspicious agents (which can be Citizen or Criminal Agents) and determine the innocence of the caught agent before arresting them. Besides hearing, all agents can speak. Speech is used by: Citizen Agents to denounce a robbery; Criminal Agents to bribe Police Agents; Police Agents to negotiate with Criminal Agents. Police Agents have hearing which allows them to detect when a Citizen Agent is denouncing a robbery. Moreover, Agents are capable of touching each other. In the case of Criminal Agents, it allows them to steal from the Agent they touched, and, for Police Agents, to arrest a given Agent.

2.1.2 Goals. In this section, the goals of each agent type are enumerated and described. Citizen Agents have no goals since they are purely reactive agents.

CriminalGoals = [Freedom, GetRich] where "Freedom" consists of not being arrested and "GetRich" consists of robbing the maximum number of items. These goals oppose each other since "GetRich" puts the Criminal Agent at risk of being arrested by the Police Agents, while "Freedom" prevents the Agent from attaining any riches for fear of being arrested.

PoliceGoals = [GetRich, CaptureThieves] where "GetRich" consists of being paid for arresting Criminal Agents or receiving money from bribes and "CaptureThieves" consists of arresting Criminal Agents. These goals complement each other since capturing thieves

will always, in the case where a Criminal Agent is arrested, lead to the Police Agent receiving money.

Each goal has a different weight which is used to determine which action a given agent chooses to do at a given time. Weights are influenced by the result of executing actions and the action chosen to be executed takes into account the weight of the goal when considering the "reward" the Agent will receive for completing said action. Goals are the core of the Agent's belief system, representing their "wants" and willingness to achieve them through the goal's weight.

2.1.3 Actions. In this section, the set of actions for each Agent type is enumerated and described. All actions feature a CanExecute function that, besides telling the Agent if the action is available for execution at a given time, also works as a means of providing the Agent with observations from the environment.

CitizenActions = [Rest, Scream, Wander]
CriminalActions = [Bribe, Flee, Rest, Steal, Wander]
PoliceActions = [Arrest, Investigate, Patrol, Pursuit, Rest]

- Rest - Action common to all Agents, since all of them contain an attribute for tiredness. The tiredness of an Agent reflects how much labor they have done, be it in terms of walking, stealing, or arresting a suspect. This action has a duration of two seconds, on which the Agent is unable to do anything else. Resets the tiredness back to 0.
- Wander - Action common to the Citizen and Criminal Agents. This action can be executed when the tiredness attribute is lower than its specified maximum value. The direction and finish point of this action are randomized and, upon reaching that point, the tiredness value is incremented.
- Scream - Action specific to the Citizen Agents, can only be executed if the Citizen has been robbed and successfully acknowledges this fact (using the "awareness" trait of the Agent) or if a Criminal fails to rob a Citizen, in which case the Citizen will shout out the name of the robber.
- Bribe - Action exclusive to the Criminal Agents. The Bribe action is the last resort of a captured Criminal, only being able to execute it if it is captured by a Police Agent. With this action, the Criminal tries to buy their freedom and is dependent on the Police Agent's willingness to accept bribes. If the action fails the Criminal Agent will be arrested, else it will be eliminated from that Police Agent's suspect list and set free.
- Flee - Action specific to the Criminal Agents, this action allows them to avoid the Police Agents. If the Agent is within a specified radius of a Police Agent it may decide to flee. At the end of the action, the tiredness of the Agent is increased.
- Steal - Action specific to the Criminal Agents, it allows them to walk up to a Citizen and attempt to rob them. The target agent must be within a limited radius of the Criminal so that it can be "seen" by the Agent. If the Agent loses sight of the target Citizen then the action is no longer executable. If the Criminal Agent "touches" the target, then it may be robbed. The outcome of the action is dependent on the Agents skill level.
- Patrol - The most basic action of the Police Agent class and comparable to the "Wander" action, this allows the Police Agents to move between two designated spots, patrolling that area. If a suspect is "seen" within that area the Agent may decide to pursue it. This action increased the Agent's tiredness.
- Pursuit - Action exclusive to the Police Agents, allows them to chase a suspect Agent so that it can be arrested afterward. This action increased the Agent's tiredness and can only be executed if the Police Agent can "see" the target suspect, i.e., the target is within a designated radius of the Police Agent.
- Arrest - Action exclusive to the Police Agents that can only be executed if they have "touched" a suspect. The outcome of this action is dependent on the willingness of the Police Agent to accept a bribe if offered by a Criminal. If a Citizen Agent is arrested it will not be eliminated from the board, and it's assumed it was cleared of all suspicion.
- Investigate - Although written like an action, Investigate serves as a means to provide the Police Agents with an observation of their environment. This is the "action" that allows Police Agents to add suspects to the suspect list by merely observing the environment. It's only executable upon "hearing" a Citizen scream.

2.2 Environment Definition

The agents are run on a grid map, 86 by 67 Unity units and each agent occupies an area of 1 by 1 Unity units. In the center, we have the map with all the agents and to the left and right of it, we have information display boxes, one for each agent type, which displays information in real time about each action an agent chooses to perform.

The environment we designed is dynamic and constantly changing due to the movement and actions of the Agents. Visually, we have 4 Police Agents, 4 criminals, and 8 citizens. To make it easier to understand what is happening, each agent has a different color: Police Agents are blue, Criminals are red and Citizens are green. All these agents are set on a fixed initial position of the map at the start of each run. This starting position is always the same regardless of the architecture chosen, but the patrol and wander movements are randomized. We also display a Scoreboard to make it easier to infer the winner. Police Agents get one point for each Criminal successfully arrested and Criminal Agents get one point for every two objects successfully stolen, for a total of four points. A timer is displayed right below the Scoreboard, to keep track of the round's duration.

Additionally, to aid in understanding the sometimes rapid changes in the environment we added sound effects. When a Criminal is arrested, the Police will say "objective complete" or "OK" if they are successfully bribed by the Criminal. When a Criminal successfully robs a Citizen, the robber will say "Woo", and the Citizen will scream upon realizing they have been robbed.

2.3 Implemented Architectures

- **Reactive Agents** - A very simple algorithm where the Agents make decisions based solely on their observations of the environment (they "react" to the changes in the environment).

- **Deliberative Agents -** A medium complexity algorithm that grants Agents the ability to choose the action that will better fit their wants and goals.
- **Cooperative Agents with Social Conventions -** A more complex algorithm that adds communication to the Deliberative algorithm. Police Agents are ordered by their "personality trait", i.e., their willingness to accept a bribe. The one with the greatest personality trait value is deemed the leader and is responsible for assigning targets to the other Police Agents. The remaining agents share their suspect list with the leader and report whenever a suspect is arrested and cleared of all charges.

2.4 Empirical Evaluation

To test our multi-agent system, we implemented a script that runs each agent architecture setup 100 times, calculates the defined metrics, and saves the output to a text file. A run can end in one of three states: "PoliceWin" (where the Police Agents Team wins), "CriminalWin" (where the Criminal Agents Team wins) or "TimeOut" (where the run ends after 60 seconds without any of the Agents fulfilling the goal of getting four points). The output file includes all the metrics we found necessary and applicable to analyze the results. Bellow we enumerate and describe the relevant metrics:

- **Number of victories -** absolute frequency of "PoliceWin" and "CriminalWin" final states, which allows us to reason about the performance of the Police in terms of victory ratio;
- **Average time to win -** arithmetic average of the duration, in seconds, of a run that ends in a "PoliceWin" or "CriminalWin" final state. This metric allows us to measure the efficiency of the agents in terms of time;
- **Average time between consecutive arrests -** arithmetic average of the duration, in seconds, for another arrest to occur. This metric, allows us to measure the efficiency of Police Agents and distribution in terms of time;
- **Average time between consecutive robberies -** arithmetic average of the duration, in seconds, for another robbery to occur. This metric allows us to see how efficiently Police Agents pursue and arrest suspects as the fewer criminals there are the higher this time will be;
- **Average number of arrests -** arithmetic average of the number of arrests per game. This allows us to measure how efficiently Police Agents are being distributed and stopping suspects, as well as to measure the occurrence of false positives, i.e., Police Agents arresting Citizen Agents;
- **Average number of robberies -** arithmetic average of the number of robberies per game. This allows us to measure how efficiently Police Agents are arresting Criminal Agents, as less active Criminals should lead to fewer robberies occurring;

Both the number of victories and the average time to win metrics are calculated for both Police Agents and Criminal Agents teams. By running the simulation 100 times for each architecture, we aimed at increasing the variety of generated personality traits, leaving our data less biased by the randomness of the agents' skills. Be careful when analyzing the data on the Criminal Agents team, since they always have the same architecture. This means the values for criminal agents are related to their performance against Police Agents of different agent architectures and not related to a change in their own architecture. Finally, when referring to the "average" of a value, we are referring to its arithmetic average.

3 EXPERIMENTAL ANALYZES

In this section, we will discuss the results obtained following the method described in section 2.4 Empirical Evaluation. The baseline for our comparisons is the Reactive Agents architecture.

3.1 Number of Victories Per Team

Counting the number of times each team (Police Agents Team and Criminal Agents Team) won, allows us to see the evolution of the game depending on the agent architecture of the Police Agents. Note that the sum of wins of both teams for a given architecture does not correspond to the total number of runs, since there are runs that end in a "TimeOut" state where there is no winner. In figure 1 the number of victories is compared in vertical grouped bars for each team. The colors represent the agent architecture of the Police Agents and the height of the bars is the average of the absolute frequency of the occurrence.

We observed that the number of victories of the Police Agent team increases as we change to more complex and intelligent architectures. Compared with the Reactive Police Agents, the Deliberative Police Agents won 11.5 times more than the Reactive Police Agents. This means the deliberative agent architecture allows for a better choice of actions because each agent "remembers" what happened in the past steps and keeps track of a list of suspects. Furthermore, deliberative agents have beliefs and goals, so instead of simply reacting to an external stimulus, they evaluate the several stimuli and choose the action with the greatest reward and that will lead them to accomplish their goals. Since arresting a criminal is an action with a big reward, Police Agents prioritize arrests or pursuits which lead to arrests. It is also noticeable how the gap in the number of victories between each side for the deliberative agents architecture is the smallest. The occurrence of bribing is an explanation for this behavior since it allows Criminal Agents to avoid being arrested even if the Police Agent succeeds to catch them, increasing their active time on the simulation. Another possible explanation is the fact both agents share the same architecture, making them equally efficient chasing their goals. Compared with the Reactive Police Agents, the Cooperative Police Agents won 16.25 times more than the Reactive Police Agents and 1.413 times more than the Deliberative Police Agents. By having Police Agents communicate among them the list of suspects, it makes it more difficult for Criminal Agents to escape a pursuit since all Police Agents will target those suspects. Plus, if a Criminal Agent manages to successfully bribe a Police Agent, they can still be arrested by other Police Agent who may not accept the bribe. Similar to the Reactive Police Agents scenario, the difference between the teams' number of victories is bigger because the Cooperative Police Agents architecture is more sophisticated than the Deliberative Criminal Agents architecture.

On the Criminal Agents Team side, the results are the inverse, which makes sense since when a team wins the majority of runs, the other loses. The discussion is similar to what was referred above.

Against Reactive Police Agents, it becomes easy to win since the Deliberative Criminal Agents have a better performance in action decision-making than the weaker reactive agents.

Finally, the number of runs that ended in a "TimeOut" state decreases from 41 with the Reactive Police Agents to 18 with the deliberative architecture and 11 with the cooperative architecture. This means that the Police Agents' architecture also has an impact on how long it takes for a team to win. Besides that, more robust agent architectures, such as the deliberative or the cooperative architectures, have a smaller probability of agents getting stuck, since they will have a bigger motivation to pursue their goals.

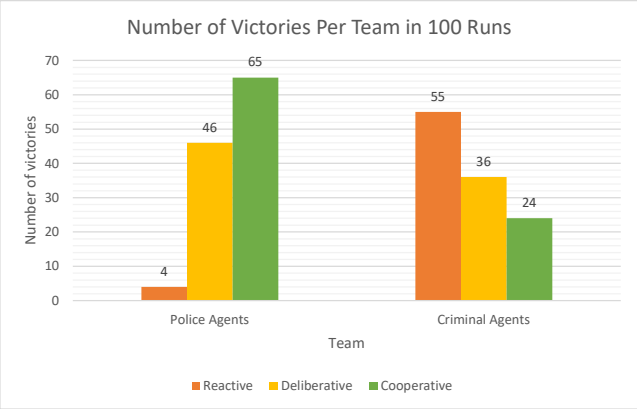

Figure 1: Number of times each agent team won per agent architecture in 100 runs.

3.2 Average Time to Win

Keeping a record of how long it takes for a run to end in a "PoliceWin" or "CriminalWin" state allows us to analyze the time it takes for each team to achieve its objective and observe how a more complex decision-making mechanism and actions impact the duration of a run. Figure 2 illustrates the results for this measure.

Starting with the Criminal Agents team side, it is interesting to notice that the agent architecture of their rivals has little to no impact on the time taken to win. This means the duration of a run depends more on the complexity of the actions. Since the actions of the Criminal Agents are always the same, the time is stable, with a slight trend to decrease, which can be related to the results presented in figure 5 where the time between robberies also decreases.

On the Police Agents team side, we observe that the few times the Police Agents won, it took them little time to accomplish the goal. Given that the values for the reactive agent architecture are based on only four wins, we can only conclude that the random generation of the personality traits on those four runs benefited the Police Agents by giving Citizen Agents a higher awareness level, which implies more pursuits and giving the Police Agents a greater morality level, making them less prone to accept bribes, which implies more arrests. Compared with the Reactive Police Agents, the Deliberative Police Agents need 187.8% more time to win, a little less than double the time. Unlike the criminal actions, the police actions increase their complexity now that they keep a record of past events and they have a suspect list. For example, if a citizen screams next to a Police Agent, but that Police Agent is currently pursuing the criminal with the highest suspicious level, the Police Agent will not withdraw the action to arrest the other criminal, even if they are closer to that criminal. This also explains the results for the deliberative architecture in figure 4. On the other hand, the duration of a run with Cooperative Police Agents decreases by around 1 second when compared to deliberative Police Agents, since the tasks are shared by all Police Agents.

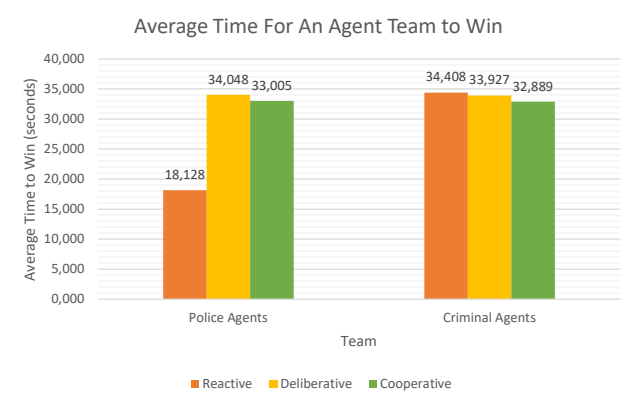

Figure 2: Average duration, in seconds, of a run where one of the teams wins per agent architecture.

3.3 Number of Arrests and Robberies

In this section, we compare the number of arrests per run with the number of robberies for each Police Agent architecture. In Figure 3, the blue bars represent the average number of arrests performed by the Police Agents and the orange bars represent the average number of robberies performed by the Criminal Agents. Keep in mind that for each run the maximum number of possible correct arrests is 4 and the maximum number of robberies is 8, respectively the same as the number of Criminal Agents and the number of Citizen Agents. This implies that when the number of arrests is greater than 4, citizens were arrested. This can also happen if the number of arrests is less than 4, but in that case, with the Reactive Police Agent implementation, the Police Agents are only able to pursue agents accused of robbery, i.e., robbers "caught in the act" by their victim, who then alerted the Police.

The results correspond to our expectations given the results we presented above. Once again, and for the same reasons we have already pointed out, the number of robberies is stable on the deliberative and cooperative architectures. It increases against reactive Police Agents since the number of victories is also bigger and there are few arrests, meaning more Criminal Agents will be active at the same time. This is an interesting result either way because even if there are more active criminals, figure 5 shows us that the reactive architecture has the biggest average time between robberies. With more robberies happening, it becomes increasingly harder to come across a Citizen that hasn't been robbed yet.

We can also see that the more complex the Police Agents' architecture, that is, the more it relies on the computation of rewards,

beliefs, and communication in the case of the cooperative agent architecture, the greater the number of arrests. The significantly greater number of arrests with the cooperative agent architecture is explained by the communication and update of the suspect list among Police Agents, making them focus more on agents who have been on crime scenes. However, these results are a bit controversial as they make us question if it is an efficient architecture when, on average, the system needs to arrest 9.46 agents when there are only 4 criminals. At least 60.9% of the arrests are false positives with the cooperative agents architecture. Still, the average time between arrests seen in Figure 4 and the number of victories in Figure 1, restores our confidence stating the Cooperative Agents Architecture is the one with better performance.

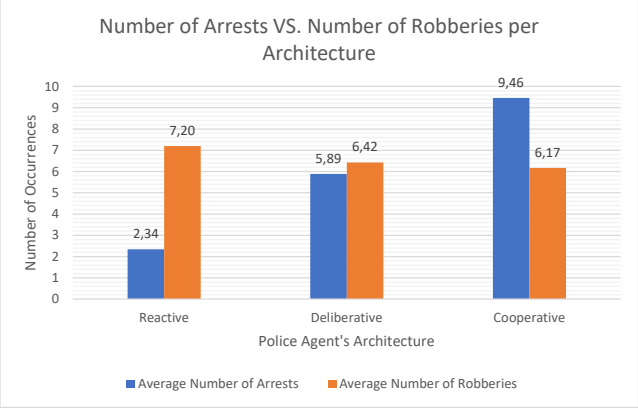

Figure 3: Number of arrests compared to the number of robberies per agent architecture.

3.4 Arrests Rate

In this section, we will analyze and compare the average time between arrests for each Police Agent architecture. From looking at Figure 4 we can see a big spike in the average time between arrests when going from Reactive to Deliberative architecture, although this might mislead us into concluding that Reactive had better results than Deliberative, we must recall that the rate of Police Agent wins is extremely low for Reactive architecture. Since this metric does not take into account which team wins and since from the previous graph in Figure 2 we can see the average time to win for a Police Agent team is a bit more than half than the other architectures, we conclude that these results for Reactive architecture are due to a random luck factor, meaning that in the very few times that the Police Agents manage to execute arrests and win the game, they do it in a short time, since if they do not do it in that short time they lose, as shown by the very low amount of wins they have. When they can't win, they might execute a few fast arrests but they end up losing as the Criminal Agents reach their goal, as seen on Figure 3, the number of arrests is less than half the other architectures.

Regarding the Deliberative architecture we see more straightforward results, the Deliberative Police Agent team wins more often than the Criminal Agent team, and since there is no coordination the pursue and arrest operations take longer as each Police Agent goes arbitrarily one by one to each suspect, some being far away from the robbery and not being notified of it, so sometimes not all agents are pursuing suspects and they might keep patrolling which delays the time between arrests further.

Finally, we have Cooperative architecture, this architecture had the best results. By sharing the suspect list with all other Police Agents we ensure that no agent is idle, even if they're far away from the robbery and the agents can also divide themselves between multiple suspects so no two agents go to the same suspect if there are more suspects to be pursued. Ensuring all agents are in pursuit and multitasking ensures a shorter time between arrests as they can be done simultaneously.

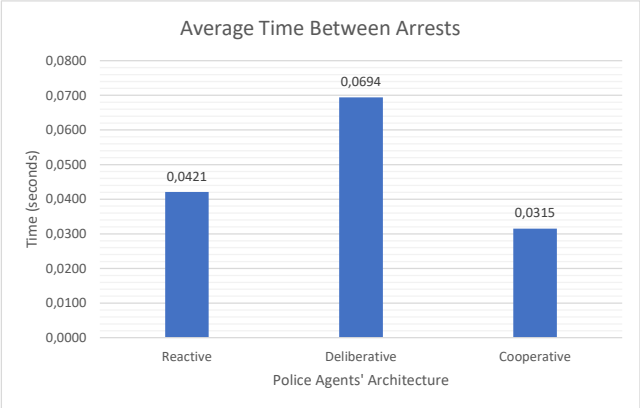

Figure 4: Average time between the occurrence of consecutive arrests.

3.5 Robberies Rate

Figure 5 shows us the average time between consecutive robberies when Criminal Agents play against Police Agents with different architectures. There is a correlation between the number of criminal victories (Figure 1, the average number of arrests (Figure 3 and the average time between robberies. The longer the duration of a run, the longer it is taking for either Police Agents to arrest all criminals or Criminal Agents to steal all citizens. The biggest the number of arrests, the fewer active criminals there are in the environment. Grouping these two affirmations, we can state that the longer the Criminal Agents are active, the hardest it becomes to steal a citizen because the options start to shorten as they have to find the few citizens who haven't been robbed yet. Therefore, the time interval between robberies will be longer as is the case of the scenario with the Reactive Police Agents architecture. With the deliberative and cooperative agent architectures, Criminal Agents are caught earlier, meaning the robberies will happen when there are still many citizens yet to be robbed, accelerating the rate at which the robberies happen.

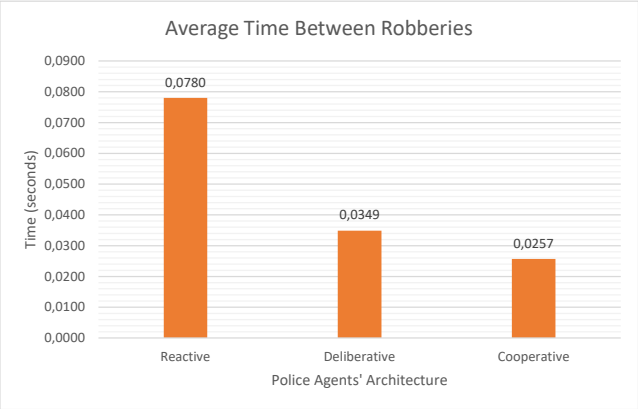

Figure 5: Average time between the occurrence of consecutive robberies.

4 CONCLUSION

Conducting this experiment and analyzing the data gave life to our idealization and allowed us to better understand the impact of each agent architecture. Overall, the results match our expectations. Reactive agents are the less successful for this model, having a low chance to win the game. the deliberative agents architecture offers a more balanced outcome between both teams, having a better performance than the reactive agents, but it can still be better as it takes longer both to win the game and to perform consecutive arrests. Cooperative Police Agents with Social Conventions have the best performance, even if it raises some false positives. However, when mapping this environment to a real-world patrolling and capturing problem, as long as the suspect is checked before being punished, these results show a "better safe than sorry" strategy, which is beneficial.

To conclude, in a multi-agent system, when the task does not radically increase its complexity by implementing cooperation, cooperative architectures, such as the one we studied in our solution, are the best response for team-based playground games such as Spy Tag.

5 SOURCE CODE DELIVERY

Due to maximum file size restrictions, we are delivering the source code via Google Drive:

https://drive.google.com/drive/folders/18uifeqpI8lke6K-f3iATDpSqNYvdp80D?usp=sharing