AASMA Project Proposal

“Full House”

Ana Evans
86379
ana.s.evans@tecnico.ulisboa.pt

Catarina Gonçalves
86397
catarina.m.goncalves@tecnico.ulisboa.pt

Leandro Salgado
86460
leandrosalgado@tecnico.ulisboa.pt

1. PROBLEM

Learning a complex game is always difficult to new players and in the case of Poker, there is the added barrier of involving betting and monetary sums. There aren’t easy and accessible ways for a person to learn this game that don’t involve some degree of spending. The few ways to learn how to play is by knowing the rules, playing many games and learning from trial and error, which, as mentioned, is risky as well as time consuming.

2. PROPOSAL

For this assignment, we planned the development of *Full House*, which is an AI multi-agent system in which the objective of each agent is to play and win the game against each other. The type of Poker implemented in this system is Limit Texas Hold’em, a modality of Texas Hold’em in which the amount of money a player can bet in each round is limited by a minimum and maximum value, both of which are fixed. The currency in this game is money, which is represented by chips throughout the game. There are different types of **chips** depending on the value they represent. In each round of the game, players have either to make a **bet** or forfeit the round. The chips bet in each **round** are added to the middle of the table, being this amount the **pot.** The objective of each round is to win the pot, which is done by having the best hand at the end of the round or by having every opponent forfeit the round. A **hand** is what we call the combination of cards each player holds at the end of a round; the highest hand wins and, in case of a tie, the winners of each round share the pot. The game ends when all the players have run out of chips to bet with, and the last remaining player is crowned the winner.

In this game, the number of players may vary but in our system we will focus on optimizing a strategy between 3 and 6 agents. At each step, the agents will evaluate the state of the game and make a play, according to the cards in their possession, the cards on the table and how much they still have left to bet. Additionally, the agents should be able to decide whether to make a play also depending on the cards and possible plays of their opponents.

We will take a level-based approach, in which the first and second levels will be our baseline; any level beyond that will be conditional to the time remaining for the development. On the first level, we will focus on developing the system with the objective of recognizing the value of each card and each possible card combination, that is, implement the concept of “hand”. On the second level, we will further develop the decision process by also taking into consideration the chips available at a certain timestep to the agents and to their opponents (this includes the ability to **bluff**). On the third level, we would like to establish different behaviors and personality traits for each agent and condition the decision-making process according to those behaviors. Lastly, on a fourth level, we would implement a relation between the agent and the environment, the latter of which will have an impact on the actions taken by the agents.

3. PROPERTIES

3.1 Agent

To define the agents, we determined that each will be:

- **Autonomous:** the agent independently determines ways to achieve victory
- **Adaptive**: each agent in this system has an internal state and it can adapt to the surrounding environment
- **Rational:** all the agents try to maximize the utility function provided
- **Not curious**: the agent doesn’t try to come up with innovative ways to achieve victory
- **Proactive/Reactive**: the agents will be able to follow a well-designed plan to achieve the goal. If needed, they can interact more abruptly depending of the situation of the world
- **Competitive**: the agents will compete with each other to win the game
- **Believable:** the agents’ decisions are trustworthy towards the user
- **Immobile**: the agents will not move in the physical or virtual world
- **Personality traits**: different situations of the game can make the agent act differently of the other agents
- **Non-veracious**: the agent can deliberately lie to the other agents, in order to have better chances to win the game

Each agent in this system has the following sensors:

- **CheckMyCards:** See the cards in his hand
- **CheckTableCards:** See the cards upon the table
- **CheckTurn:** See what agent is currently making a play
- **CheckMyChips:** See the amount of chips he has
- **CheckTheirChips:** See chips in each of his opponents’ possession
- **CheckPot:** See the chips in the current pot
- **CheckBlind:** Know when to pay the initial blind bet
- **CheckPlayRecords:** Take into account record of plays
- **CheckProfiles:** See opponents’ facial expressions and game “tells”
- **CheckEnviroment:** Know the surrounding environment

Regarding to the actuators, each of the agents will have the following ones:

- **Call:** Put the value corresponding to the established bet value in the pot and keep playing in the round
- **Raise:** Raise the current amount of the bets, putting the increased amount in the pot and keep playing in the round
- **Fold:** Quit the current round, while losing the chips that were bet in this round
- **ShowHand:** Show his cards in the end of the game
- **PayBlind:** Pay the round's blind bet (minimum or maximum blind)

In certain situations, the agent can make a Call or Raise, even if it doesn't have a good hand in his possession. This may be done in order to intimidate an opponent by making them believe the agent holds a better hand, therefore forcing the opponent to abandon the round increasing the agent's odds of victory. This is called a **Buff**.

We will implement a **Hybrid Architecture** for each agent. We have chosen this option to better implement deliberative, emotional and learning behavior. To calculate everything regarding possible card combinations, we will be borrowing from decision-making in Game Theory and try to implement a **Monte-Carlo Tree Search Algorithm (MCTS)**. We consider this to be our best bet in order to deal with the large decision-making trees that will be necessary. As a backup plan, we have considered implementing **Reinforcement Learning,** although it will not be our first approach.

Having presented this, we can state that the agents are equipped to understand each other's actions and make decisions accordingly, as well as to react to the environment surrounding them. Combined with the internal state of each agent (which holds the probability of achieving combinations of cards, classifying them as better or worst depending on the situation), this multiagent system simulates a realistic game between players. We hope to achieve the most accurate and realistic simulation of a real-world situation from which a person, ideally our user, can learn from.

3.2 Environment

The game environment is:

- **Inaccessible:** the agents have only access to information about the cards in their hand and on the table
- **Non-deterministic**: the results of the agent's actions are on himself are predictable, however it is hard to predict accurately the results of the agent's actions on the other agents
- **Dynamic**: while the agents are deliberating, another agent can make an action that changes the state of the world
- **Continuous**: agents can be in any situation of the world and new information might be given in any point of time
- **Non-Episodic**: each round is dependent of the final state of the previous one, whether it's concerning the cards in the table or the amount and quantity of chips in each agent's possession).

4. REFERENCES

[1] Moody, David. "*Monte Carlo Tree Search in Texas Hold 'em Poker. Computer Science Tripos, Part II", *2011 May 19

[2] Woolridge, Michael. "*An Introduction to MultiAgent Systems:",* 2009, John Wiley & Sons Ltd.