PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

MOHAMED BOUDIAF UNIVERSITY - M'SILA

FACULTE OF MATHEMATICS AND COMPUTER SCIENCE

DEPARTEMENT OF COMPUTER SCIENCE

N° ……...

DOMAINE: Mathematics & Computer science

FILIERE: Computer science

OPTION: IDO

Dissertation presented for obtaining

The Academic Master diploma

By: Laaraf Ahmed Amine

&

Latelli Thamir

Entitled

Developing an online strategy game powered by an intelligent algorithm

Framed by: Dr. Barkat Abdelbasset

Supported in front the jury composed of:

……	University of M'sila	President
……	University of M'sila	Examiner

Academic year: 2020 /2021

DEDICATION

This paper is dedicated to our beloved parents, who have been our source of inspiration and gave us strength when we thought of giving up, who continuously provide their morale, spiritual, emotional, and financial support.

To our brothers, sisters, friends, classmates, and mentor who shared their words of advice and encouragement to finish this work.

All thanks to Professor Abdelbasset Barkat who addressed this interesting subject, and for the information and guidance he provided us

And lastly, we dedicated this work to the Almighty God, thank you for the guidance, strength, power of mind, protection and skills and for giving us a healthy life.

TABLE OF CONTENTS

TABLE OF FIGURES

INTRODUCTION

A video game or computer game is an electronic game that involves interaction with a user interface or input device – such as a joystick, controller, keyboard, or motion sensing device – to generate visual feedback. This feedback is shown on a video display device, such as a TV set, monitor, touchscreen, or virtual reality headset. Video games are often augmented with audio feedback delivered through speakers or headphones, and sometimes with other types of feedback, including haptic technology.

Video games are defined based on their platform, which include arcade games, console games, and personal computer (PC) games. More recently, the industry has expanded onto mobile gaming through smartphones and tablet computers, virtual and augmented reality systems, and remote cloud gaming. Video games are classified into a wide range of genres based on their type of gameplay and purpose.

In video games, artificial intelligence (AI) is used to generate responsive, adaptive or intelligent behaviors primarily in non-player characters (NPCs) similar to human-like intelligence. Artificial intelligence has been an integral part of video games since their inception in the 1950s. AI in video games is a distinct subfield and differs from academic AI. It serves to improve the game-player experience rather than machine learning or decision making.

Videogames have grown to resemble competition-based, interactive movies, and the COVID-19 pandemic has propelled the industry to make more money than movies and North American sports combined.

Global videogame revenue is expected to surge 20% to $179.7 billion in 2020, according to IDC data, making the videogame industry a bigger moneymaker than the global movie and North American sports industries combined. The global film industry reached $100 billion in revenue for the first time in 2019, according to the Motion Picture Association, while PwC estimated North American sports would bring in more than $75 billion in 2020.

Both of those industries suffered from the effects of the COVID-19 pandemic in 2020, however, while the videogame industry is expected to show double-digit growth this year following high-single-digit growth in the previous two years. Experts forecast that strong growth will continue in 2021, following the recent introduction of next-generation gaming consoles from Sony Corp. and Microsoft Corp. and new games to get the most out of those upgrades, even as COVID-19 vaccines are rolled out.

Our Objective:

Playing video games has had a positive effect on players' perceived well-being during the COVID-19 pandemic. Games have provided an enjoyable means of maintaining social contact, and a stress relieving and mentally stimulating escape from the effects of lockdown.

We decide to be a part of this awesome industry and we put some of goals to achieve it

- Our main target is to develop a fun online board game powered with artificial intelligence.
- Gaining new experience & skills by opening our mind to new fields rather than software's only.
- Giving community something fun to play with friend during quarantine

Encourage Algeria to invest in gaming industry.

CHAPTER 1: VIDEO GAMES & ARTIFICIAL INTELLIGENCE

1 A brief on video games

A video game or computer game is an electronic game that involves interaction with a user interface or input device – such as a joystick, controller, keyboard, or motion sensing device – to generate visual feedback. This feedback is shown on a video display device, such as a TV set, monitor, touchscreen, or virtual reality headset. Video games are often augmented with audio feedback delivered through speakers or headphones, and sometimes with other types of feedback, including haptic technology.

Video games are defined based on their platform, which include arcade games, console games, and personal computer (PC) games. More recently, the industry has expanded onto mobile gaming through smartphones and tablet computers, virtual and augmented reality systems, and remote cloud gaming. Video games are classified into a wide range of genres based on their type of gameplay and purpose.

The first video games were simple extensions of electronic games using video-like output from large room-size computers in the 1950s and 1960s, while the first video games available to consumers appeared in 1971 through the release of the arcade video game Computer Space, followed the next year by Pong, and with the first home console the Magnavox Odyssey in 1972. The quickly-growing industry suffered from the crash of the North American video game market in 1983 due to loss of publishing control and oversaturation of the market. Following the crash, the industry matured, dominated by Japanese companies such as Nintendo, Sega, and Sony, and established practices and methods around the development and distribution of video games to prevent a similar crash in the future, many which continue to be followed. Today, video game development requires numerous skills to bring a game to market, including developers, publishers, distributors, retailers, console and other third-party manufacturers, and other roles.

Since the 2010s, the commercial importance of the video game industry has been increasing. The emerging Asian markets and mobile games on smartphones in particular are driving the growth of the industry and shift to games as a service. As of 2020, the global video

game market has estimated annual revenues of US$159 billion across hardware, software, and services, three times the size of the 2019 global music industry and four times that of the 2019 film industry.

2 Types of video games

There are many different types of video games, and typically, they're categorized by their characteristics or underlying objectives—not by the type of gameplay they contain. Game categories or genres, then, can also have subgenres, and many games fit into multiple genres, sure it can be confusing, but by breaking down the game mechanics, we can start to understand how developers and publishers categorize their titles.

For an easy understanding of video games ‘s types we can categorize them as follows:

3.4 Action games

Action games are just that—games where the player is in control of and at the center of the action, which is mainly comprised of physical challenges players must overcome. Most early video games like *Donkey Kong* and *Galaga* fall into the action category. Because action games are usually easy to get into and start playing, they still, by most accounts, make up the most popular video games. We can name as subgenres for this type of games:

- Platformer games
- Shouter games
- Survival games
- Stealth games
- Fighting games

2.2 Adventure games

Adventure games are categorized by the style of gameplay, not the story or content. And while technology has given developers new options to explore storytelling in the genre, at a basic level, adventure games haven’t evolved much from their text-based origins. In adventure games, players usually interact with their environment and other characters to solve puzzles with

clues to progress the story or gameplay. Aside from an occasional mini-game, adventure games rarely involve any traditional video game action elements. Thus, the genre isn't very popular with mainstream gamers. The adventure games can be subcategorized into:

- Text adventures
- Graphic adventures
- Visual novels
- Real-time 3D

2.3 Role playing games

Probably the second-most popular game genre, role-playing games, or RPGs, mostly feature medieval or fantasy settings. This is due mainly to the origin of the genre, which can be traced back to *Dungeons & Dragons* and other pen and paper role-playing games. Still, hardcore RPGers don't discount sci-fi fantasy-themed RPGs like *Mass Effect*, *Fallout*, and *Final Fantasy*, which have helped put unique spins on the genre. Cultural differences have also had a bearing on this genre, as many gamers categorize RPGs as either WRPGs (Western-influenced) or JPRGs (Japanese-influenced). Finally, gamers are often given choices in this genre that influence the final outcome of the game, which means many RPGs have alternate endings.

2.4 Simulation games

Games in the simulation genre have one thing in common—they're all designed to emulate real or fictional reality, to simulate a real situation or event. The simulation games can be divided into three types:

- Construction and management simulation
- Life simulation
- Vehicle simulation

2.5 Strategy games

With gameplay is based on traditional strategy board games, strategy games give players a godlike access to the world and its resources. These games require players to use carefully developed strategy and tactics to overcome challenges. More recently, these types of games have moved from turn-based systems to real-time gameplay in response to player feedback. Some of the subgenres of strategy games:

- Artillery
- Real-time strategy (RTS)
- Real-time tactics (RTT)
- Multiplayer online battle arena (MOBA)
- Tower defense
- Wargame

2.6 Sport games

Sports games simulate sports like golf, football, basketball, baseball, and soccer. They can also include Olympic sports like skiing, and even pub sports like darts and pool. Opposing

players in these games are often computer-controlled but can also take the form of live opponents. The most known kinds of sport games are:

- Racing games
- Team-sports games
- Sport-based fighting

2.7 Puzzle games

Puzzle or logic games usually take place on a single screen or playfield and require the player to solve a problem to advance the action, and also the puzzle games can be:

- Logic games
- Trivia games (answer games)

3 AI in games

3.1 Artificial intelligence in video Games

Artificial Intelligence research has been used in many fields. In video game industry, game AI began to emerge not long after the birth of video games, it gained huge progress since then. These days, and has become an unavoidable component and is playing a significant role in raising the quality of games.

AI in video games is a distinct subfield and differs from academic AI. It serves to improve the game-player experience rather than machine learning or decision making. During the golden age of arcade video games the idea of AI opponents was largely popularized in the form of graduated difficulty levels, distinct movement patterns, and in-game events dependent on the player's input. Modern games often implement existing techniques such as pathfinding and decision trees to guide the actions of NPCs. AI is often used in mechanisms which are not immediately visible to the user, such as data mining and procedural-content generation.

3.2 History of AI in video games

Game playing was an area of research in AI from its inception. But video games were born without AI. The first game ever was created in 1960s by William Higinbotham who worked at the Brookhaven National Laboratory. The game is called “Tennis for Two”, and was made by wiring an oscilloscope up to an analogue computer. [1] The first game to run on a computer was Spacewar by Steve Russell from MIT. Spacewar is considered as the first computer game as it was made on PDP-1 mainframe computer. Neither of these two games included AI. During the early days of video games, AI is not a feature. This is because those games are relatively simple, and for most of the time, they were about completion among real people. No computer opponent

was added into the game.

In 1970, Atari released the first video arcade game, “Computer Space”. And it was not until then that game designers began the take their first attempt to incorporate AI into their games. AIs were designed primarily for arcade games with the purpose of ensuring people kept feeding quarters into the game machine. [2] Pong, Space Invaders and Donkey Kong were among the seminal games.

These games were running under very simple rules and scripted actions. The agents didn’t have the ability to make decisions. Sometimes decisions were designed to be made randomly so that the behaviors looked more unpredictable. Therefore, the so-called Intelligence was actually coded into the game and was not able to act at runtime. Early AI appeared in the form of stored patterns. An example of such hard-coded AI is the aliens design in “Space Invaders”. In this game, Player must shoot the aliens before they reach the bottom of the screen. The way these aliens move is pre-coded into the game.

More games were created based on this type of AI, but evolution of AI had just begun. The appearance of computer opponent in “Pong” made people believe that computer was thinking. It was also regarded as the earliest real artificial intelligence in games. The way that the game Pong goes made it impossible to script the behaviors of non-human objects. The paddles need to make decisions based on the actions of human players. The decisions may be not that hard to make, it’s a simple calculation of where the paddles need to go, but this made people experience the same feeling of playing against a real human player.

The influence that the AI design in *PacMan* had is just as significant as the influence of the game itself. This classic arcade game makes the player believe that the enemies in the game are chasing him but not in a crude manner. The ghosts are chasing the player (or evading the player) in a different way as if they have an individual personality. This gives people the illusion that they are actually playing against 4 or 5 individual ghosts rather than copies of a same computer enemy.

By the late-1980s, the arcade video game craze was beginning to fade. With the development of Computer industry, home computer became available. Home computers and consoles led the new direction of video game development. New games designed for these devices became more complex due to the higher ability of the modern processors.

In 1983, the Nintendo Entertainment System was released, marking the beginning of the modern gaming era. Home computers were starting to become practical and popular. Video games designed for home computers started being developed. The fact that home computers had more storage and computing power than cartridge-based console systems allowed for more complicated games. The console industry caught back up in 1995 when Sony brought the PlayStation to the United States. It was the first practical CD based system and a huge step forward. Currently, both console systems and computer games are pushing the limits of modern processors. Real-time 3D rendering, movie-quality video and sound, and intelligent computer agents all continue to amazing gamers worldwide [3].

Even though more resources are to be applied to contents and higher-quality graphic performance, AI still found its own need to develop. During the 1980's, more genres of games emerged rapidly. The old style of AI design was out of date. Designers had to treat game AI seriously since AI has become a standard feature of video games.

The last ten years has seen great strides in the field of AI in gaming. Most of these strides have been made in computer (as opposed to console) games because of their raw computing power and memory capacities. Currently, driving games like "Nascar 2002" have computer-controlled drivers with their own personalities and driving styles. Pass one of these guys while they are having a computer-controlled "bad day" and they will be riding your bumper for miles, trying to take you out. Strategy games like "Warcraft 3" call for users to create an army to defeat one or more computer-controlled villages with armies of their own. These villages will form alliances, scout surrounding areas, and devise appropriate battle plans to do their best to be the last village standing. "The Sims" has reinvented artificial life in gaming.

The game starts with the user controlling one person. Eventually the character may marry and then the user can control the spouse also, and then the children. These "people" are wonders of AI. They all have their individual characteristics, wants, and needs. They have the ability to fall in love, make friends, enemies, become hermits, strive for more in life, etc. [4]

We are still in the evolution of game AI. Video games have come a long way ever since 1950's, and so have the AI techniques that go along with them. The past few years have witnessed more and more novel ideas, and methods for games AI have joined into the game development process.

3.3 Academic AI vs Game AI

Normally, an AI programmer doesn't need significant amount of knowledge of Artificial Intelligence from academic field to create high quality game AI. The reason lies in two facts.

First, Artificial Intelligence is a grand branch of computer science. It is not easy to grab the complex models into the games.

Secondly, the main goal of game AI nowadays is to entertain people. There are tricks or other easy ways to solve the problems perfectly. People won't be interested in how AI works in a game. They won't buy games only for the fact that the designer cited complicated algorithms to solve a mathematics problem perfectly.

Actually, in most cases, what designers used for game AI is not what academic AI is about. But it's wise to understand the concepts of AI in academic field before investigating AI techniques in games.

The concepts of Artificial intelligence existed long before this term was first used. The intellectual roots of AI can date back to Greek mythology. Intelligent artifacts appeared in literature with some real mechanical devices performing some degree of intelligence. In 5th century B.C, Aristotle first invented syllogistic logic.

He also gave the first definition of intelligence: the ability to put things into categories. But it was not until modern history that research on Artificial Intelligence became more mature.

In 1950, A.M. Turing published "Computing Machinery and Intelligence" which introduced the Turing Test as a way of operationalizing a test of intelligent behavior. In its original illustrative example, a human judge engages in a natural language conversation with a human and a machine designed to generate performance indistinguishable from that of a human being. The concepts that Turing brought seemed obvious, but it formed an essential concept in the philosophy of artificial intelligence. For video games, this concept is also treated as the objective of game AI design, naturally.

In 1956, the term "Artificial Intelligence" first appeared. It was coined by computer scientist

named John McCarthy. At that time, there were no video games available. However, games engaged in research of AI in academic fields had already existed. In 1950, Claude Shannon published his work on how a computer plays chess. It is the first time that AI is used to create a virtual opponent. Since then, Chess has been a main component in AI research. After around half a century in 1997, Computer DEEP BLUE won in a 6-game match against Grandmaster Gary Kasparov. DEEP BLUE was able to evaluate 200 million positions per second compared to 2 per second by a human player. This achievement has demonstrated the success of AI research.

However, this didn't contribute the same to game industry. Even though digital chess games are available to video game players, the meaning of chess AI in the research field is far more important than how it is applied to entertaining game players. It would be a disaster to game industry if players began to consider if he could reach to certain level so that he gets a chance to win the computer opponent before he buys a game. Most of time, people play games for fun. AI is therefore supposed to let player win but in an entertaining way.

On the other hand, investing too much on game AI to hoist it to academic levels is unrealistic. Dating back to the time when arcade games emerged, developers have had a hard time to put even simple AI into games due to RAM constraints. [5] Even nowadays, percentage of CPU cycles left to process game AI is still limited, and this offends many academic AI researchers who might be interested in developing AI for the game industry. But the very reason that game AI doesn't need to be as good as academic AI is that: AI in video games doesn't need to be deep. The design of *Pac Man* is evidence that design may not to be as hard as possible. They way *Pac Man* works is to add randomness into the decision when ghosts come to a junction, which makes sure that ghosts do not follow the same route every time. (Anderson) It's about the variation of the same path-finding algorithm, and programmers don't need to make the algorithm more complicated to achieve such effect. AIs in many games are successful without being deep. The Puzzle game is another example.

Therefore, the heights game AI has enjoyed is far less than academic AI has achieved. But with the development of game AI, more and more techniques are applied to solving problems from academic AI field. The fact is that game AI has made full use of some basic theories and methods from real AI. Methods such as Finite State Machines (FSM), Decision Trees and Fuzzy Logic are simple but powerful tools, and are widely used for modeling AI agents in

games.

3.4 The future of game AI

The past 60 years has witnessed the evolution of game AI; however, we are still in its infancy. The way that game AI performs is still often not satisfying. Take RTS games for example: the reason why it has become successful is more due to the exciting competition between real human players. None of the most popular games these days is attractive because of the excellent job that has been done on AI performance. Game AI still has a long way to go. There still lies a huge gap between game AI and academic AI. With the development of computer industry, game AI would enjoy more space to show its possibility. With more and more scientific research on game AI, we could expect a promising game AI in near future.

CHAPTER 2: GAME DESIGN AND FEATURES

1 The classic LUDO

LUDO is a strategy board game for two to four players, in which the players race their four pawns from start to finish according to the rolls of a single dice. Like other cross and circle games, but simpler.

The game contains two, three, or four players, without partnerships(every one for himself) at the beginning of the game, each player's four tokens(Pawns) are out of play and staged in the player's yard (one of the large corner areas of the board in the player's color), when able to, the players will enter their tokens one per turn on their respective starting squares, and proceed to race them clockwise around the board along the game track (the path of squares not part of any player's home column).

When reaching the square below his home column, a player continues by moving tokens up the column to the finishing square. The rolls of a single dice [6][7] control the swiftness of the tokens, and entry to the finishing square requires a precise roll from the player.

The first to bring all their tokens to the finish wins the game. The others often continue to play to determine second-, third-, and fourth-place finishers.

Played by four players, Ludo is a board divided into four main areas:

- Red
- Bleu
- Green
- Yellow

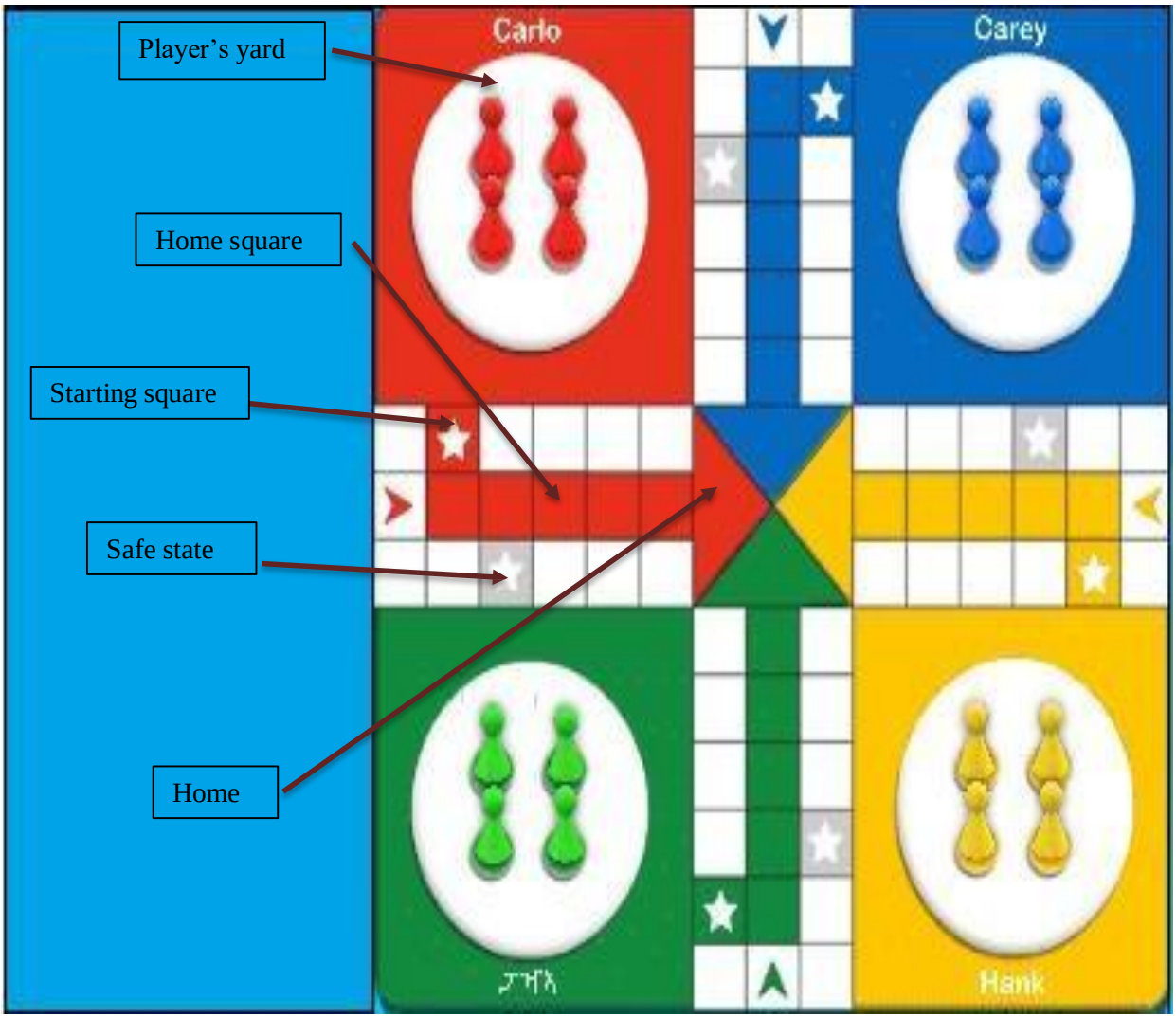


Figure 1 Ludo board components

Each player occupies one area of a color, in it, he is given four tokens of his chosen color, the Ludo board is generally a square with the game track, that is cross-shaped. Here each arm in the cross consists of three columns that are subdivided into again squares (generally six squares within a single column). Representing the home column of a player, there are five colored squares in the middle column of every cross part. There is a sixth colored square that is not a home column square but the initial point of every player to start the race.

At the board center there is a large square that is home for every player, and of course, if a player reaches in this area with all four tokens, he wins.

2 LUDO PARCHIS

First of all, the game was built from scratch, and the name was derived from the original Indian game PACHISI. LUDO PARCHIS is already lunched and properly working on Facebook (test period), the basic rules are quite similar to the classic game, but in order to make a more exciting (bit different) version, we've made some extra rules and features.

3 Game rules

We can resume the rules that we've made as following:

- The pawns will be ready to start moving on the board (come to life) only after the player has rolled a 6 on the dice.
- The pawns will move the number of units based on the number rolled by the player on the dice.
- Both kinds of pieces are safe if they land on a star.
- The player can choose to move any of his/her pieces on the board as long as the piece is ready to start moving and it has distance available to cover by the move.
- The detailed movement characteristics of the pieces are discussed above.
- If a player cannot move his piece upon rolling 6, he will not get a second chance to roll.
- If a square happens to have two or more pieces of same opponent and the current player's piece happens to land on the same square during his move, the opponent's pieces are considered safe and will not be killed, this rule only applies to the opponents whose 2 or more pieces are on the same square.

- A player can roll a dice again if he gets a 6. If he gets 6 on three consecutive rolls, then his turn is dismissed along with the last 6.

- If a piece completes the path around the board, it is called to have been home where it doesn't need to move anymore.

- A game is won by the player who is able to get all of his 4 pieces home before any other players. The game will continue until the game has 2nd and 3rd winners.

- After we get a winner, the winner's turn is skipped (but still remains in the game session as spectator) and the game continues with 3 players, after getting 2nd winner, the 2nd winner's turn is skipped (but remains in the game session as spectator) and the game continues with 2 players, and after getting 3rd winner, the game ends.

4 Used tools and software

In order to complete developing the game we used several tools and software, and we can recall them and why we used them as following:

- First of all, as a programming language we used the JavaScript because it covers perfectly all of our needs, it's easy to learn and implement, very fast since it can be run immediately within the client-side browse, thanks to that it reduces the demand on the website server, and finally because it's a very powerful high-level programming language made for web, same as our game target requirements.

- We also used a new game engine (2D/3D Game Creation Tool) called COCOS Creator, the reasons why we used it are: for being a Cross Platform engine, it means games can be quickly published to IOS, Android, Windows, Mac, Web, and every mini-game platform, allowing us to maximize the visibility and the success probability of our game, and also for being a very easy-to-use lightweight (small package size) tool, also the reason that made us use COCOS is being exactly what this engine creator offers is web-based game using JavaScript language knowing

that our main target platform was web and Facebook instant game platform.

- From Facebook, we used the instant games platform because it supports the exact type of games that we've made, easy to start playing (thanks to the magic of HTML5) from the moment a user decides to play an Instant Game on Facebook Messenger, all it takes is one tap and ten seconds and they are playing the game, it let people play games on any device, mobile or desktop, right in News Feed. The games are highly social, and it's easy to invite friends.

- The thing about our game coding is that the most of it was written in the server side. With the help of an open-source, cross-platform called Node.js we were able to run our game server and communicate with players by using JavaScript and a collection of "modules" that handle various core functionalities

- We used Artificial Intelligence to generate a list of fake faces to give player the feel of playing with another person while competing against the bot using an AI photo generator.

Figure 2 Some generated bot's face pictures by an AI photo generator.

- Since our game is distinguished by being an online social game, it requires a Matchmaking between player to start a game, and we could not find a better solution than Colyseus, it's simply the easiest solution that let you focus on the game itself rather than dealing with low level API.

- In order to make an honest game we provide a direct communication line throw Gmail to Feedback or send a complain, all of that thanks to the Nodemailer of Nods.js applications which easily allows sending messages, write HTML content and provides good security and a high range of emails support.

Figure 3 Screenshot taken from the official game email (sent by players as a Feedback).

- And also, to store players pictures, names and scores we used a small JSON database for Node, powered by Lodash called lowdb, just for being small, easy, contain no extra steps and easy to host in our backend. It helped us to classify the player to put them in the Leaderboard.

- We also get some help from a specific JavaScript library for generating fake data. Fake data is useful when building and testing our application. The faker.js can generate fake data for various areas, including address, commerce, company, date, finance, image, random, or name. We chose Faker.js just because it contains a massive number of names.

- In our game we have the ability to identify some information about player like: country, region, city, ZIP code, time zone, connection speed, ISP, domain name, IDD country code, area code, weather station data, mobile network codes (MNC), mobile country codes (MCC), mobile carrier, elevation, usage type and proxy, and that's by using a Geo Ip solution called Ip2location which is in our opinion the best solution that allows us to host the server in our backend and the easiest the use. We can see some of that information at the Feedback.

5 The in-game features

In order to make clients more excited about our game we've add some feature those we can resume as:

- **Multi language:** the game can be played by three languages English, French and Arabic, upon logging into the game (either by directly going to the app or by clicking a link shared by a friend's game request), the player will choose by checking the language and then confirming his choosing.

Figure 4 In-game multi language choice

- **Daily login coins:** after entering the game, the user is presented with a home screen, a gift icon will appear with a clock that define the remaining time to the next daily bonus, after pressing it, it will show a window with two options for collecting daily login bonus coins: Collect and Get Twice. Collect button will add 1000 coins in his account and get twice will lead to an Ad, after watching which they will get 2000 coins added in their account.

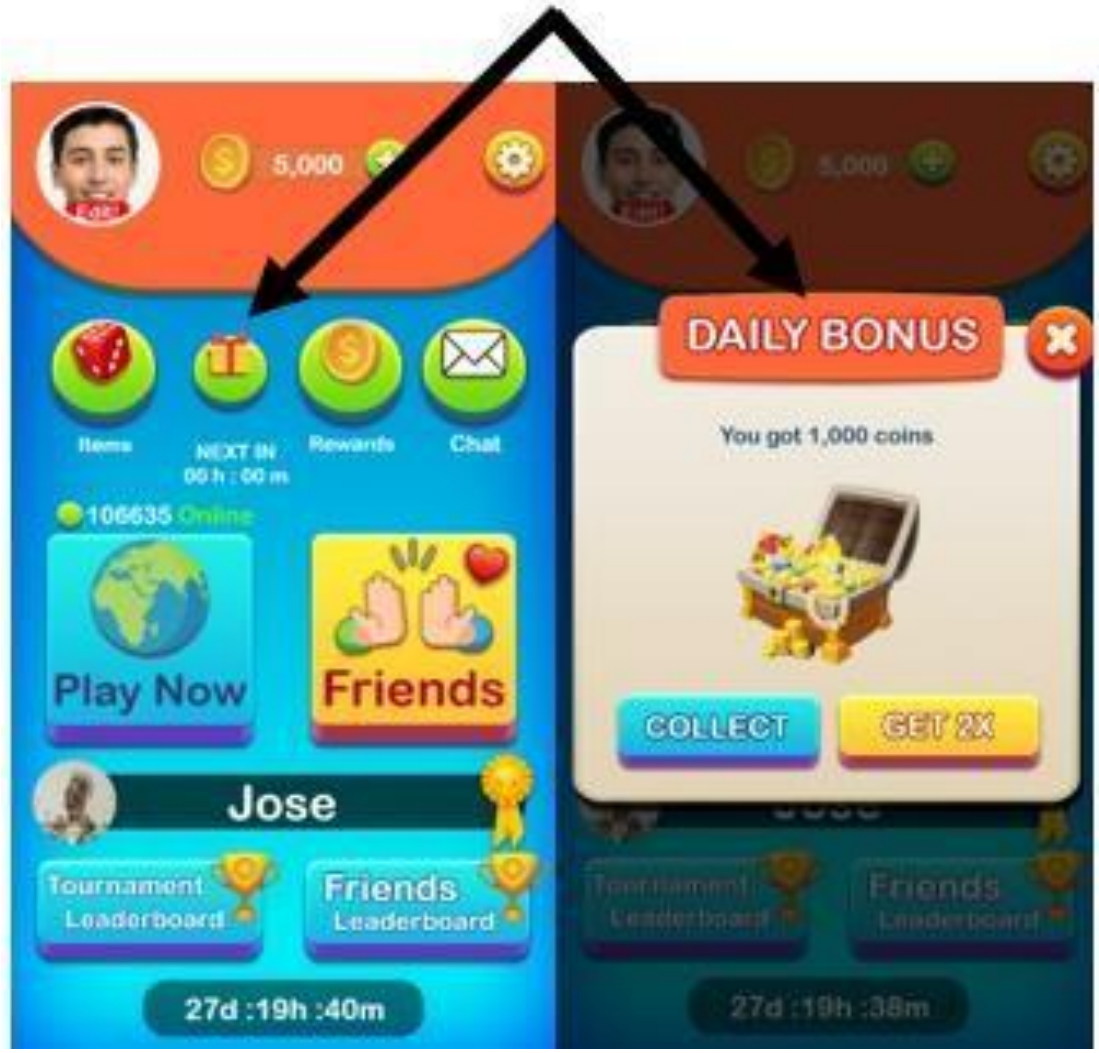


Figure 5 In-game coins reward

- **The leaderboard:** We have two kinds of leaderboards, this option will appear as two buttons at the home screen, one is for friends ranking and the other is for all players ranking. The players are ranked by their wins in the game. The ranking will update every 2 days and will be a moving indicator of the player's wins in the last 7 days.

Figure 6 In-game leaderboard

- **Play publicly or with friends:** The two options will be listed on the home screen and the player can choose the form of game he wants to play. This will take the player to the player count selection screen.

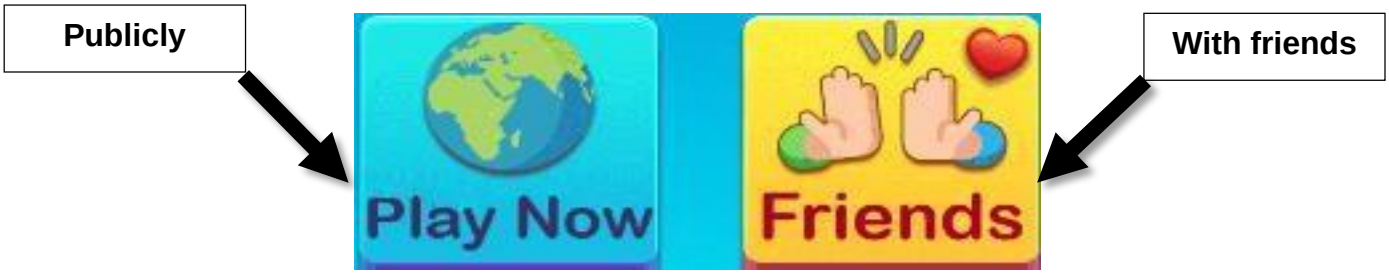


Figure 7 In-game playing mode

- **Multi playing:** After selecting any of the game options above, the player will select the number of players to play with, the app should start the timer and wait for the first 4 players to join. If at least 2 players and at most 4 players join the game within 120 seconds, the game will begin. Others joining the game will be able to see the game but will not be able to participate in the game. The player who sends the invite can start the game once at least a player joins in.

Figure 8 In-game number of opponents selection

- **Player profile:** By clicking at the profile picture displayed at the top left of the game screen, the player can choose to change his name and also see his results such as the winnings, loses…

Figure 9 In-game player profile

- **Music/Sounds:** The game have a background music which can be turned on/off from the game settings. Each action in the game will have sound effect, which also can be turned on/off in the settings.

- **The auto play (bot on):** If a player does not roll the dice within 10 second, the bot will be turned on which will play the game on the players' behalf until the player chooses to turn it off by pressing the button on the game screen.

Figure 10 In-game auto play bot countdown

- **Feedback:** Another option that can be accessed from settings in which the player can send a feedback directly to the game developers/owners.

Figure 11 In-game Feedback forum

- **Quick text:** Also, under the game settings, the player can add custom quick chat texts to use later in the game just by pressing on the expression. The only way to the player to do this is by watching an advertisement.

Figure 12 In-game quick message option

- **The inventory:** it's another option in home screen from where the player can choose the board type and also the dice type (from three kinds of dices, each dice has its power). The choosing is done by trading coins or watching ads.

Figure 13 In-game inventory menu

- **Animation:** the dice should be able to roll in his place next to the player's icon, and the 3D looking pieces would vibrate to show the available plays, and to move it would jump from square to the next. If a kill is made, the killed token should trace the path back to its den.

6 The use of ads in the game

We wanted to make our game beneficial in somehow, so knowing that the game target specifically the Algerian community, which means they won't buy coins with real money, for obvious reasons, but they will watch ads if they have proper motivation, like an ability to skip other turns with the expense of 5k coins when the other guy is about to win, requesting a rematch with the expense of some coins, the catch here is that we've restricted the use of such methods only while playing with friends, not with random people to make an extra motivation to play with friends and watch ads then we might have a good source of income, the game play is not affected by these types of ads, just to make sure that the player won't get bothered while enjoying the game.

CHAPTER 3 :
AI IN LUDO PARCHIS

1 Multiplayer games & AI algorithms

Multi-player games are games that can be played by more than two players. They have several properties that makes them an interesting challenge for computers. First, contrary to two-player games, pruning in search trees is considerably more difficult. Second, the opponents' moves are more unpredictable, as coalitions may occur, so we cannot simply use Min-Max algorithm.

Over the past years, Researchers have often observed deficiencies in the minimax algorithm and its approach to game playing. Russell and Norvig (1995), for instance, gave a prominent example of where minimax play can be flawed through slight errors in the value of leaf positions [8]. Others have shown that minimax search can be pathological, returning less accurate results as search depth increases (Beal 1982; Nau 1982) [9][10]. While new algorithms have been designed for better analysis of games (Russell & Wefald 1991; Baum & Smith 1997) [11], or for opponent modeling (Carmel & Markovich 1996) [12], these approaches have not been widely used in practice. There are a variety of reasons for this, but the primary one seems to be that minimax with alpha-beta pruning is simple to implement and adequate for most analysis.

Since our game is a multiplayer game, the approach we took is using the MAX-N algorithm just because it’s the most similar one to the MIN-MAX and returns the same result.

2 The MAX-N algorithm

Max-N (Luckhardt & Irani 1986) is the generalization of minimax to any number of players, while in a two-player, zero-sum game it will return the same result as minimax. The values at the leaves of a max-n tree (max-n values) are n-tuples, where i-th value in the tuple corresponds to the score or utility of a particular outcome for player i. The max-n value of a node where player i is to move is the value of the child node for which the ith component is maximal. In the case of a tie, any outcome may be selected [13].

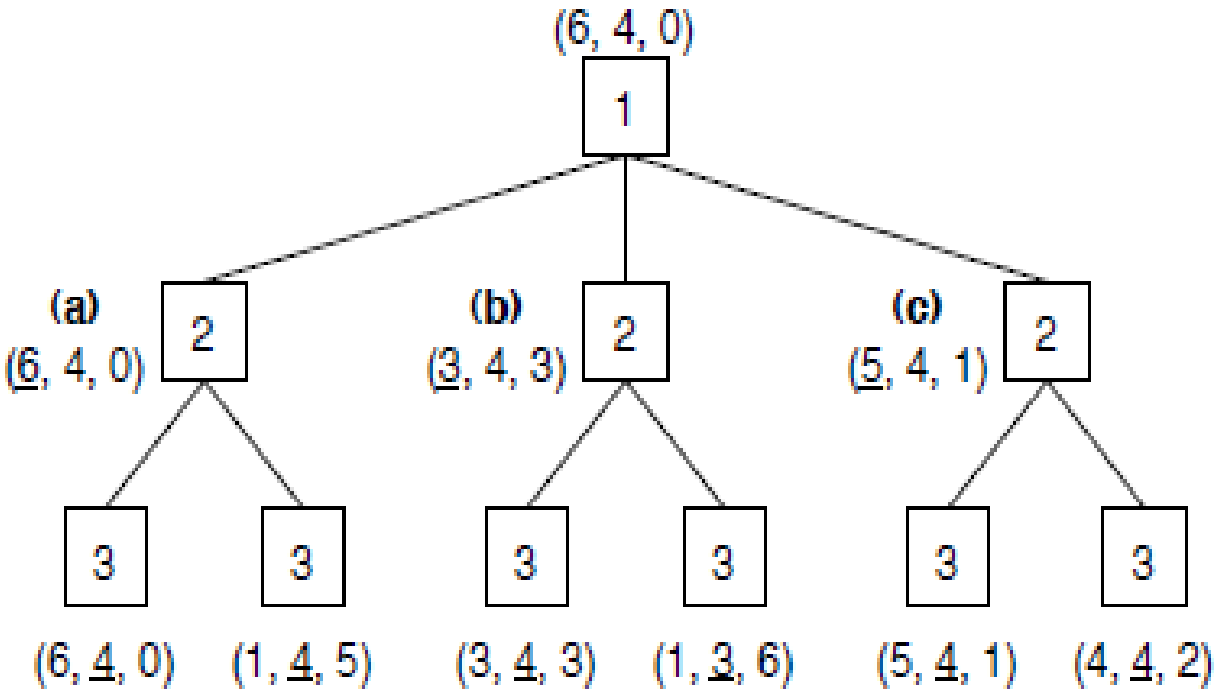


Figure 14 An example maxn tree.

Figure 14 demonstrates the maxn algorithm. Each node in the tree is a square, inside of which is the player to move at that node. At node (a) Player 2 can choose between two outcomes, (6, 4, 0) and (1, 4, 5). Because Player 2 gets 4 from either choice we arbitrarily break the tie to the left and return the value (6, 4, 0). At node (b) Player 2 will choose (3, 4, 3) to get 4, instead of (1, 3, 6) to get 3. Player 2 also has a tie at node (c), and chooses the value (5, 4, 1). At the root of the tree Player 1 chooses the left branch to get (6, 4,0), the final maxn value of the tree. If all players use maxn to search a game tree, and all leaf values are known, the resulting strategies will be in equilibrium, meaning that no player can do better by changing their strategy. But this analysis doesn't provide a worst-case guarantee. A player, for instance, may be able to change their strategy in a way that decreases another player's score without causing their own score to decrease. In fact, mistaken analysis at even a single node of a maxn tree can arbitrarily affect the payoff of the resulting strategy [14].

```
Max-N(board,depth,turn)
  If ((terminalNode()) or (depth == 0))
    Return evaluation(board)
  Score <- -infinite
  for I<- 1,2,3 … numberOfMove do
    board.doMove(I)
    val <- max-n(board,depth-1,next turn)
    board.undoMove(I)
    if (val[turn] > score)
      score <- val
  return score
```

Figure 15 Pseudo-code for max-n algorithm

Figure 15 shows the pseudo-code for the max-n algorithm. The process on that algorithm just like the minimax, but it uses max value for each array.

3 How MAX-N work in LUDO PARCHIS

First of all, in our case the board is represented by a combination of 3 objects and 3 giant arrays of those objects filled with information of player's ids and its pawns also it have specific information of whether it is in home place or safe place and so on…, the first array represents the player's yard, the second is the normal white squares, and the third contains the home squares, we also have in object Safe Square option, if the pawn is in it, we don't have to see if it can die or not.

The player's yard is represented in Player Object and to be specific in the Pawns key at start all 4 pawns are stored there also the Player object have extra information about the player like the player nickname, his own picture , his unique id, whether the bot is activate or not and what dice skin he selected

Note: In case of ai opponent we give fake name and picture to give the feeling of playing with real players.

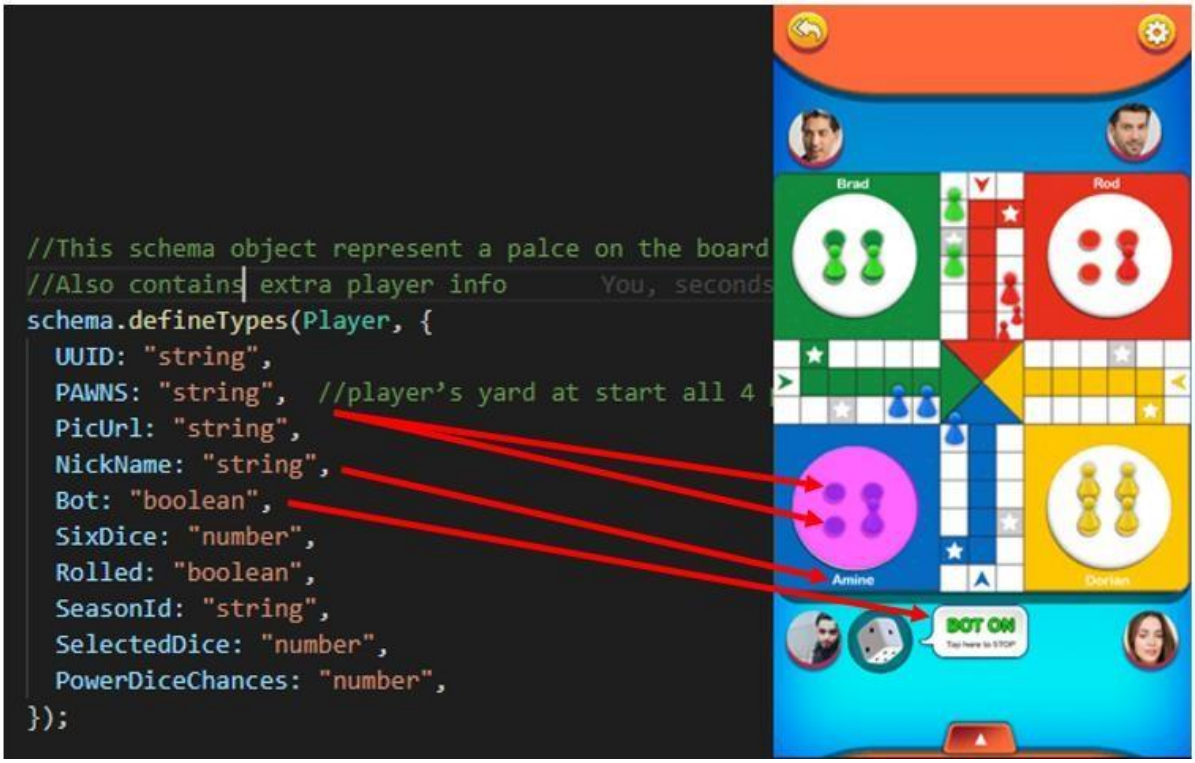


Figure 16: Object represent player's yard at start all 4 pawns are here, also contains extra info about player

Each white square is represented by Object called Board the safe square represented by true Boolean which is located in the key that called safe while the normal white square set to false. The pawns are stored in the key that called PawnID

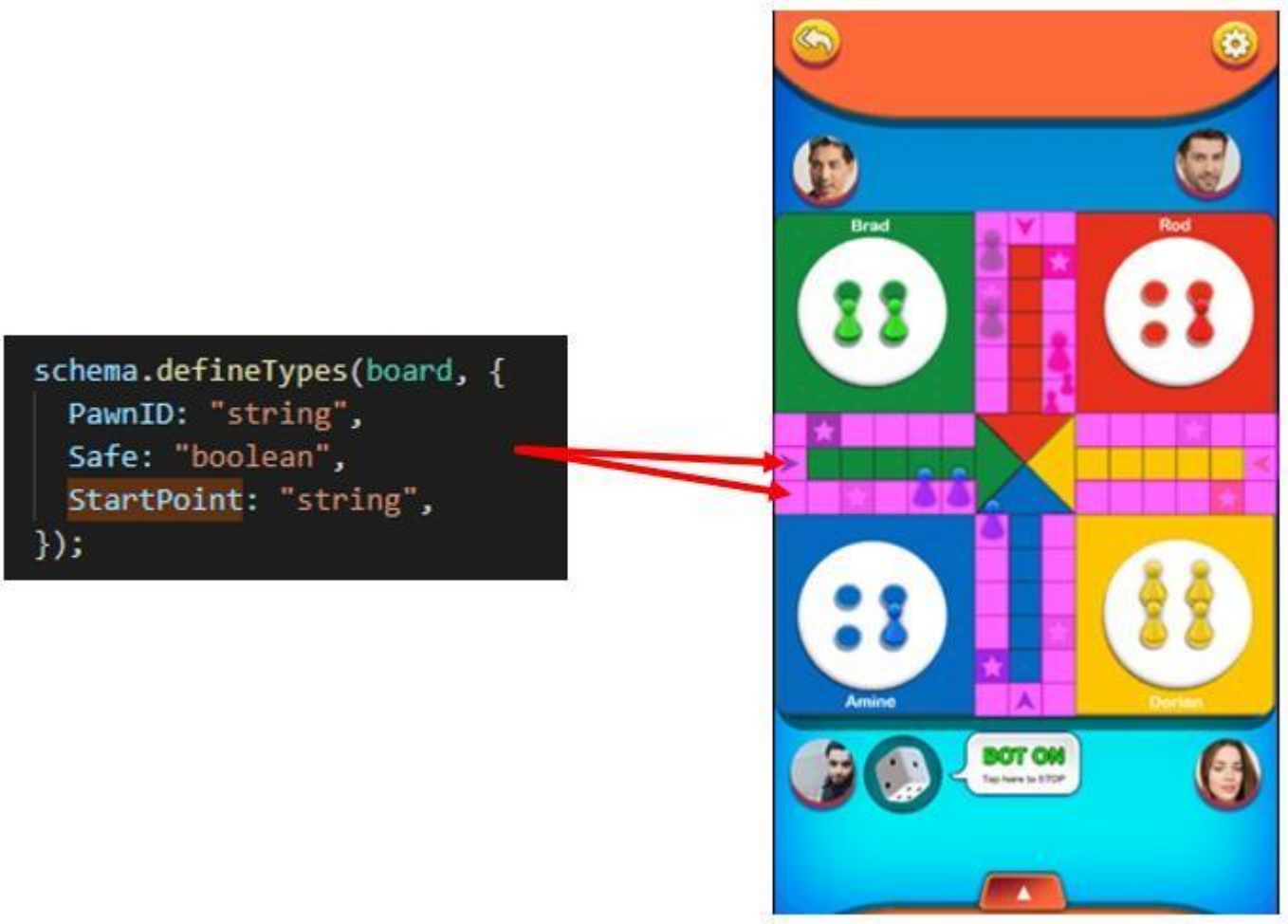


Figure 17: Object represent a single white square and whether if the square safe or not

The home square is represented by object called plusboard the pawns are stored in key called PawnId

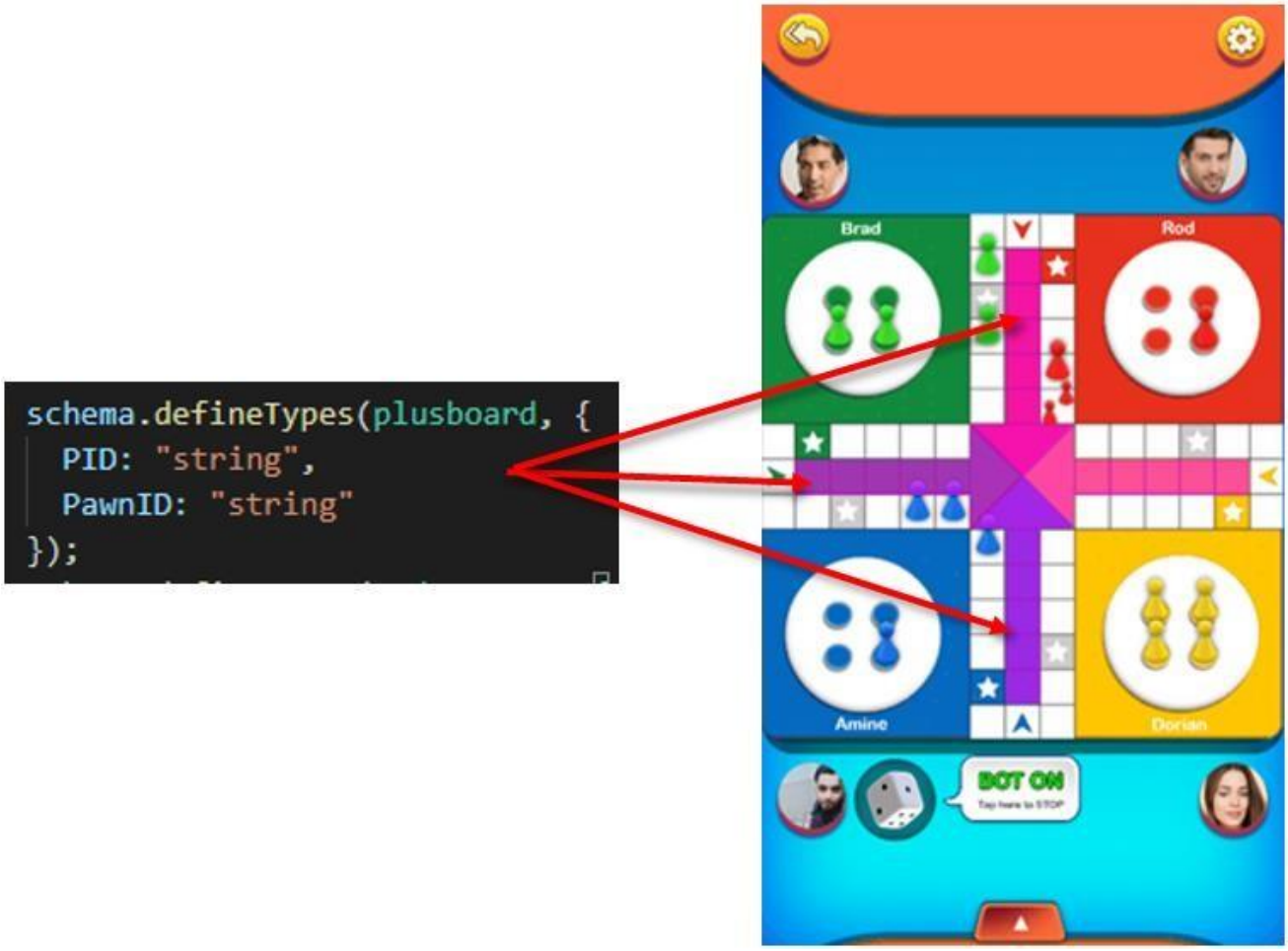


Figure 18: Object represent a single home square

To simulate the full board we make 3 arrays of all objects white, home and the player yard we use those giant array later to get and set information about the pawns place on the board though array iteration in JavaScript

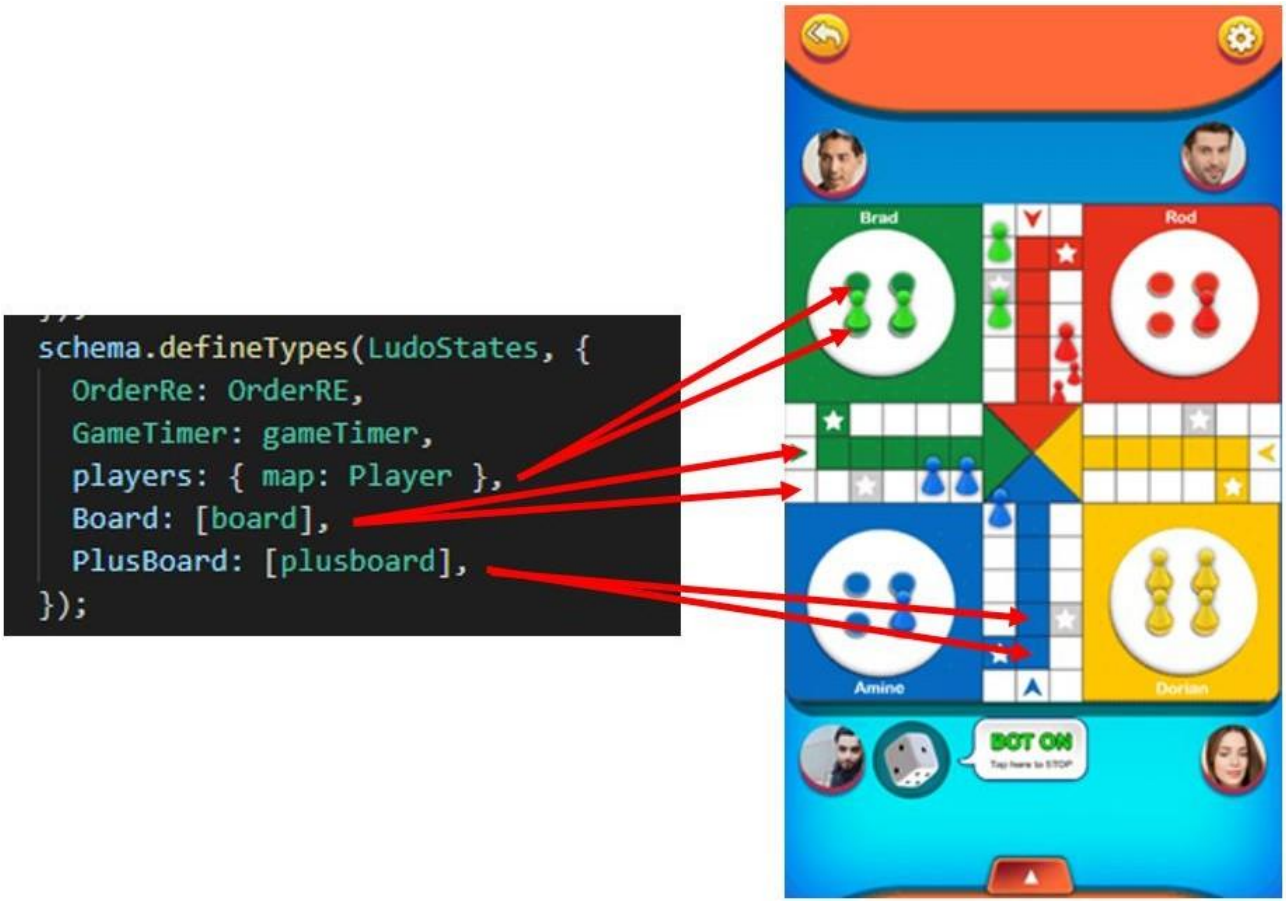


Figure 19: Object represent a giant array of squares white, home, and player's yard to simulate the complete ludo board

We pass to an iteration function the player ID and his rolled dice for the current player turn, it loop over each array to find in which place the player pawn located also to check if possibly can move it give us available moves and also it check if pawn could get killed, each place can also let us know if player moved to home or going out from the first yard …

Each node from the MAX-N tree is basically the full state of the board, implementing max-n was challenging since our game logic is synchronize over network and handled by server side itself, also due to hardware limits for the servers, Max-n can be heavy and take ages to finish a task, and in such games time matter, however we edited the algorithm to fit our exactly needs to run as fast as possible and smoother so other players will not feel any delay and without breaking the fun of the feeling of Ai resistance.

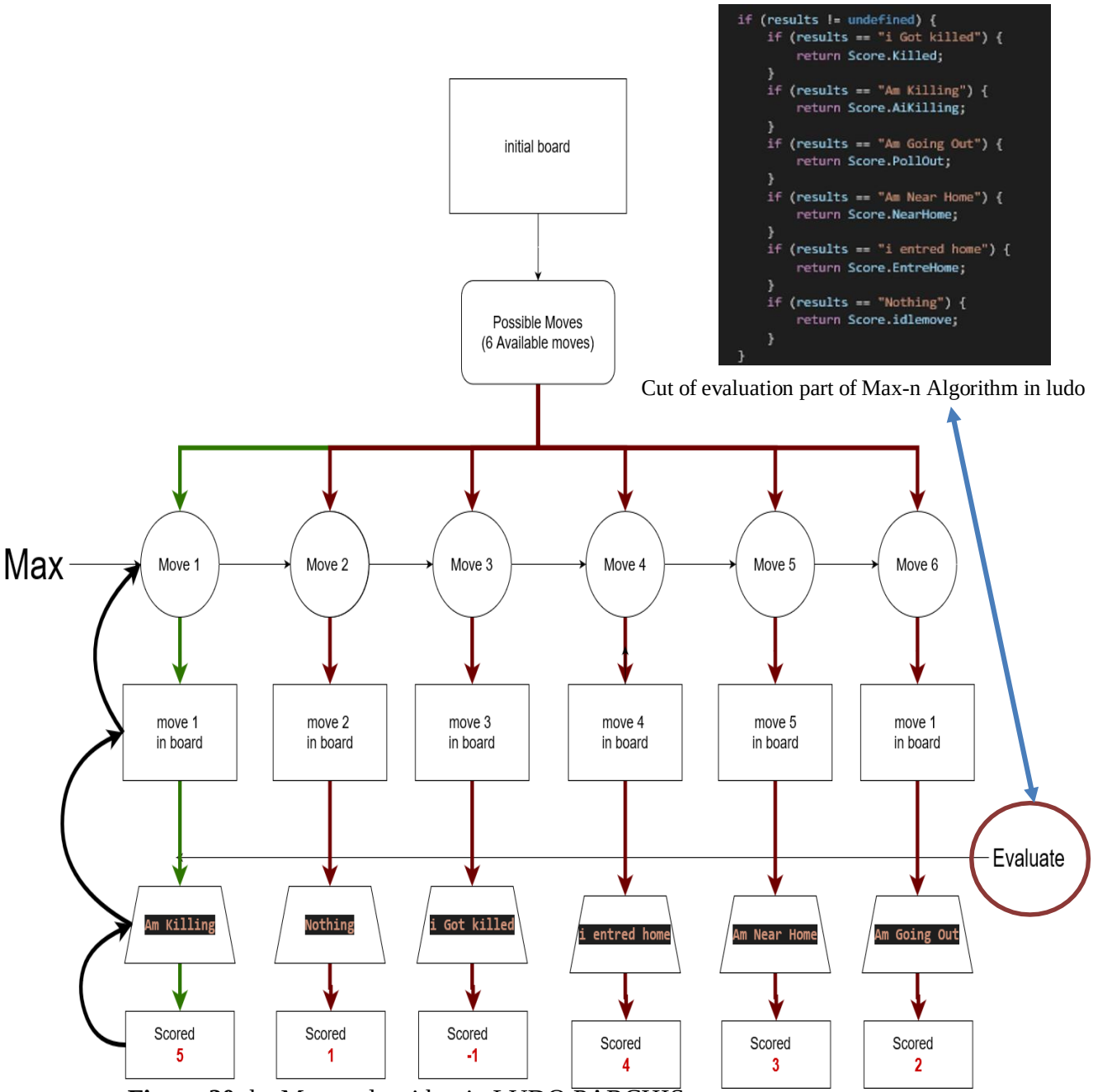
What we have done:

- The depth was not necessary since Ludo game was totally a stochastic game due to the randomness of dice for each player, looping over and over by predicting all dices and move for each player was just waste of resources in Ludo we decide to remove the depth because of that reason.

- By making a function that help us to get the state of each pawn after moving virtually, the function pass on all the objects to find a specific player id the it will extract the exact position of all its pawns, this made evaluating each move with max-n easy and super-fast.

- In our Ludo we have done six evolution and it was enough to make Ai pretty intelligent however to give the Ai more abilities we can increase the evolution inside the algorithm will make Ai more intelligent in many hard cases.

By following those steps, we were able to make efficient Ai for a real-time multiplayer game.




```
if (results != undefined) {
    if (results == "i Got killed") {
        return Score.Killed;
    }
    if (results == "Am Killing") {
        return Score.AiKilling;
    }
    if (results == "Am Going Out") {
        return Score.PollOut;
    }
    if (results == "Am Near Home") {
        return Score.NearHome;
    }
    if (results == "i entred home") {
        return Score.EntreHome;
    }
    if (results == "Nothing") {
        return Score.idlemove;
    }
}
```

Cut of evaluation part of Max-n Algorithm in ludo

Figure 20 the Max-n algorithm in LUDO PARCHIS

The Figure20 shows how the algorithm work in LUDO PARCHIS, the algorithm compares between the scores of the available plays then take the max score.

As an example, for that we can take two moves and let's say that:

The first move returns the message **"I Got killed"** while the other move returns the

message **"I entered home"** in this case the algorithm will evaluate both scores and returns which better, so getting you pawn killed is a high risk and will lead to lose, while entering home will give more chance to win the game, and as we already know entering 4 pawns to home will lead to total win, so in this case the algorithm will choose entering home, and same idea with the other cases.

The evaluation part done through terminal node or depth but in our case we have terminal node only as we can see in max-n algorithm we apply the same behavior in our ludo algorithm

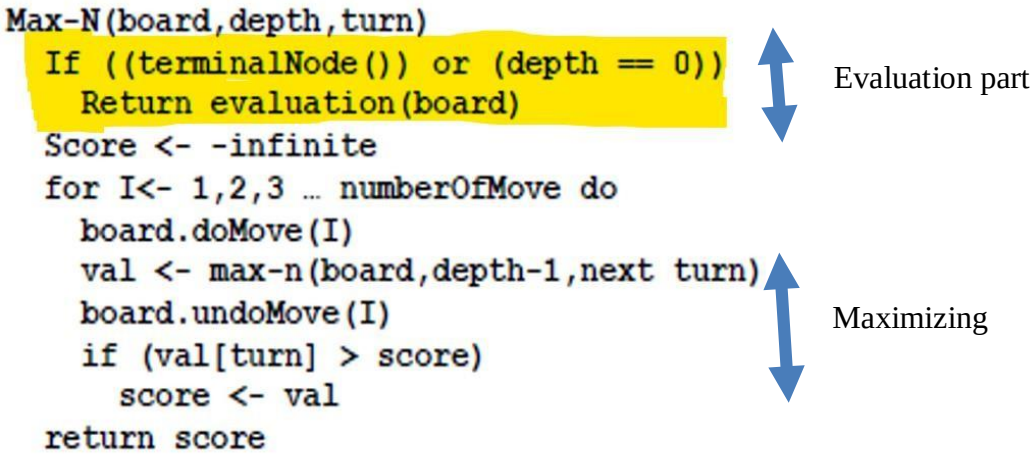


Figure 21 Evaluation part in Max-n Algorithm

Figure 22 cut of the Max-n algorithm in LUDO PARCHIS

The Figure 22 shows a cut piece from code of the edited algorithm of max-n inside LUDO PARCHIS, the virtual move function will move the available pawns in virtual board and return evolution of the movements, every move have a pre-defined score, the max-n algorithm will also evaluate each score and compare it with the previous score of other movements then it will decide which is the best play.

CONCLUSION

Today Games and intelligence go side by side. Ludo is an intelligently played mind strategy board game. It requires some basic knowledge for different moves that need to be taken on one’s move.

This project implements a Ludo game which is powered by an intelligent algorithm, LUDO PARCHIS is also an online game and even though that the Facebook administration changed developer's rules while doing this work and stopped our game we were able to keep it running in a test version, the game were designed colorful yet simple and easy to navigate at, and what's make it different of the other version is its plentiful features, they were designed just to give a better gaming experience for the user.

Games were, are, and will always be an ideal domain for exploring the capabilities of artificial intelligence within a constrained environment and a fixed set of rules, where problem solving techniques can be developed and evaluated before being applied to more complex real-world problems, however, in order to make a game intelligent means to apply an intelligent algorithm on it, so to make the computer play a multi-player game of chance we’ve modified the MAX-N algorithm just to fulfil our needs.

ABSTRACT

Games are an ideal domain for exploring the capabilities of artificial intelligence within a constrained environment and a fixed set of rules, where problemsolving techniques can be developed and evaluated before being applied to more complex real-world problems, however, this thesis seeks for implementing an online strategy game which is powered by an intelligent algorithm, so in order to make a game intelligent means to apply an intelligent algorithm on it, so the method we used to make the computer play a multi-player game of chance is a modified MAX-N algorithm. The result of this project is properly working strategy board game on Facebook(test period) that we've named LUDO PARCHIS.

Nowadays games and intelligence go side by side. Ludo is an intelligently played mind strategy board game. It requires some basic knowledge for different moves that need to be taken on one's move.

Keywords: LUDO, Board-game, MAX-N algorithm, Multiplayer, AI.

ABSTRAIT

Les jeux sont un domaine idéal pour explorer les capacités de l'intelligence artificielle dans un environnement contraint et un ensemble de règles fixes, où les techniques de résolution de problèmes peuvent être développées et évaluées avant d'être appliquées à des problèmes plus complexes du monde réel. un jeu de stratégie en ligne qui est alimenté par un algorithme intelligent, donc pour rendre un jeu intelligent, il faut lui appliquer un algorithme intelligent, donc la méthode que nous avons utilisée pour faire jouer l'ordinateur à un jeu de hasard multi-joueurs est un MAX modifié -N algorithme. Le résultat de ce projet est un jeu de société de stratégie fonctionnant correctement sur Facebook (période de test) que nous avons nommé LUDO PARCHIS.

De nos jours, jeux et intelligence vont de pair. Ludo est un jeu de société de stratégie mentale intelligemment joué. Cela nécessite des connaissances de base pour les différents mouvements qui doivent être pris lors de son mouvement.

Mots clés: LUDO, Jeu de plateau, l’ algorithme de MAX-N, multijoueur , IA.

ملخص

تعتبر الألعاب مجالًا مثاليًا لاستكشاف قدرات الذكاء الاصطناعي في بيئة مقيدة ومجموعة ثابتة من القواعد ، حيث يمكن تطوير تقنيات حل المشكلات وتقييمها قبل تطبيقها على مشكلات أكثر تعقيدًا في العالم حيث يمكن تطوير تقنيات حل المشكلات وتقييمها قبل تطبيقها على مشاكل العالم الواقعي الأكثر تعقيدًا ،ومع ذلك ، تسعى هذه الأطروحة إلى تنفيذ لعبة إستراتيجية عبر الإنترنت مدعومة بخوارزمية ذكية ، لذا فإن الطريقة التي استخدمناها لجعل الكمبيوتر يلعب لعبة حظ متعددة اللاعبين هي خوارزمية MAX-N المعدلة. نتيجة هذا المشروع هي لعبة إستراتيجية على تعمل بشكل صحيح على منصة FACEBOOK(فترة اختبار) والتي أطلقنا عليها اسمLUDO PARCHIS

في الوقت الحاضر ، تسير الألعاب والذكاء جنبًا إلى جنب ,LUDOي لعبة لوحة إستراتيجية ذهنية يتم لعبها بذكاء. يتطلب بعض المعرفة الأساسية للحركات المختلفة التي يجب اتخاذها أثناء التنقل.

<u>الكلمات المفتاحية</u>: ليدو, لعبة اللوحة,خوارزمية ماكس أن, متعدد اللاعبين, الذكاء الصناعي.

BIBLIOGRAPHY

[1] Parlett (1999), p. 49.

[2] Tozour, P. (2002). Introduction to Bayesian Networks and Reasoning Under Uncertainty. AI Game Programming Wisdom (ed. S. Rabin), Charles River Media, 2002,

[3] "AI in Games Reaches New Level". Ziff Davis Net News. 24 Jan 2001.

[4] Woodcock, S. "Game AI: The State of the Industry", Game Developer Magazine. August 1999. Volume 3, Issue 33

[5] Walther, A. (2006). AI for real-time strategy games

[6] Bell (1983), p. 113.

[7] McCorduck, Pamela (2004), Machines Who Think (2nd ed.), Natick, MA: A. K. Peters, Ltd., ISBN

[8] Russell, S., and Norvig, P. 1995. Artificial Intelligence: A Modern Approach. Englewood Cliffs, NJ: Prentice Hall.

[9] Beal, D. F. 1982. Benefits of minimax search. In Clarke, M. R. B., ed., Advances in Computer Chess, volume 3, 17– 24. Oxford, UK: Pergamon Press.

[10] Baum, E. B., and Smith, W. D. 1997. A Bayesian approach to relevance in game playing. Artificial Intelligence 97(1- 2):195–242.

[11] Nau, D. S. 1982. An investigation of the causes of pathology in games. AIJ 19(3):257–278.

[12] Carmel, D., and Markovitch, S. 1996. Incorporating opponent models into adversary search. In AAAI-96, 120–125.

[13] C. Luckhardt and K.B. Irani. An algorithmic solution of n-person games. In Proceedings of the 5th National Conference on Artificial Intelligence (AAAI), volume 1, 1986.

[14] Sturtevant, N. 2004. Current challenges in multi-player game search. In Proceedings, Computers and Games.