Thesis Dissertation

ANALYSIS AND IMPLEMENTATION OF CYPRIOT GAME ANTPIN

STYLIANI VAKI

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

May 2024

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

ANALYSIS AND IMPLEMENTATION OF CYPRIOT GAME ANTPIN

STYLIANI VAKI

Supervisor

Yiannakis Sazeidis

Thesis submitted in partial fulfilment of the requirements for the award of degree of

Bachelor's in Computer Science at University of Cyprus

May 2024

Acknowledgements

Primarily, I would like to express my gratitude to my thesis supervisor Dr. Yiannakis Sazeidis. During the process, he has been a source of support, giving me any assistance I needed at any time. Additionally, I am extremely appreciative of his guidance and his advice along with his constant feedback, which provided me with the motivation to continue. It is worth noting that he gave me a fresh perspective on things, helping me see various subjects from a different angle.

Furthermore, I would like to extend my appreciation to all my professors over the past four years. Their commitment to academic growth has been essential for my academic development. I have been provided with a solid foundation upon which I was able to complete this thesis.

Lastly, I would like to state the support and encouragement of my friends and family, that has been a constant source of motivation. Their never-ending belief in my abilities has been guiding me through all this process.

To all those mentioned and to those whose contributions may not be explicitly stated, I offer my sincere thanks.

Abstract

This thesis focuses on a traditional Cypriot two player game called “Αντρίν” . My supervisor brought this game to my attention, and after some discussion we were eager to look deeply to unravel its complexities. This game falls into the strategy game category and is similar to many widely played strategy games, for example Tic-Tac-Toe, Nine Men’s Morris, or Achi. This thesis aims to analyze some of Αντρίν computational properties but also to the preservation of this traditional game and potentially the inspiration for future work.

The game takes place in two phases. In phase one, players must take turns and place their tokens on the board in hopes of reaching a winning position. In phase two, players must take turns and move their tokens on the board in order to reach a winning position. The concept of this project was to conduct an in-depth analysis of the game. It is safe to say that an analysis of this kind requires a significant amount of time, so my contribution to the analysis reached the first game phase.

Throughout the journey of analysis, a board encoding system, and a set of operations have been put to place to help the process. The primary goal was to find the number of unique board placements in phase one of the game. That was achieved through an algorithmic implementation of the operation and encoding combination.

Finally, a basic console-based version of the game was developed, to close the thesis smoothly.

Contents

Chapter 1

Introduction

1.1 Idea and Objective

Taking it from the start, I had no previous knowledge of the Αντρίν game. I was not even aware it existed. After some discussion with my supervisor, I was introduced to an explanation of the game. The fact that the game is a traditional Cypriot one got my attention, thus I started asking people older than me if they ever played the game. People remembered the game and were always enjoying the fact that they retrieved an old, buried memory of their childhood. As a result, I chose this thesis to dig deeper into the game.

The Αντρίν game is a two-player strategy game resembling other classic strategy games. It is safe to say that the documentation of this game was minimal thus, an in-depth analysis of it was nonexistent. The primary objective of this thesis is to go through a deep and thorough analysis of the game. Primarily this thesis deals with phase one of Αντρίν. During the first phase, the players take turns to place their tokens on the board in hopes of a win. In the second phase, players take turns again by moving their tokens around the board trying to reach a winning condition. To accompany and make the analysis easier an efficient encoding system of the board as well as a set of operators is recommended and used. This encoding is taken as the base to represent all possible board game placements with the operators being used as the roots for the following goal.

The next goal was to find the exact number of unique board placements in phase one. The task was accomplished by firstly addressing the algorithm, followed by a Java program with strong consideration of the encoding and the set of operators. The result will give a significant understanding of the game but also serve as a foundation for future work. Additionally, a winning strategy possibility was examined for phase one, giving valuable insights on the game's strategy.

The last part of the thesis was the design of a basic console version of the game representing the practical application of the game as well as giving the chance of an experience of Αντρίν game. Moreover, a basic user-friendly interface was implemented, allowing people to easily engage with the game.

To conclude, this thesis has a goal of analyzing this strategy game, specifically its first phase using rich and methodical tools. Countless insides and mechanics of this game are unveiled with so many more left for possible future work.

1.2 Contribution and Findings

To start off, this thesis aims firstly to contribute to the understanding and preservation of the traditional game Αντρίν. Given the lack of documentation and analysis of this game, such work can be of importance for the maintenance and protection of the game, by safeguarding it for future generations. Through the analysis and results, tools were developed, giving useful basics for future study of Αντρίν. Examples of them are the encoding system, the set of operators, the algorithm to find the number of unique boards as well as the console game.

This research yielded significant findings as well. The exact number of unique board placements amongst the total 60480, was determined, which provides the baseline for the Αντρίν game complexity and understanding. Furthermore, the possibility of a winning strategy was thoroughly examined giving out valuable insides of the game's strategy, for players trying to understand the dynamics of Αντρίν.

Lastly, the console version of the game is a push for an interest to arise for the game. Using the computer as the middleman for the users to play gives a modernized and more accessible way to play and experience the game. However, it is safe to say that future work on further analysis of phase two of the game as well as a complete game development approach is suitable for future work.

Chapter 2

History and Future

2.1 Board Game History

Board game history [2, 3 12 17] goes back to 5000 BC when the very first board game came to surface. That was the Dice. The piece, despite its simplicity, dates back so many years but is one of the essentials when playing a modern board game. Archeologists found 50 small carved and painted stones, withing a 5000-year-old grave. As years went by dice was made from glass or marble.

Early games like Sanet (3100 BC) started to take a spiritual level. Some believed that the player who won was under the protection of the major gods of the time. That led people to place the Sanet board in the tombs to join them in the afterlife. Moving into centuries, The Royal Game of Ur (2650 BC) was the oldest game with preserved rules, therefore people still play it. Its' highly decorated wooden boards are significant, as well as the fact that it was played by people of all classes in ancient Mesopotamia.

Strategy games emerged around 1300 BC with Ludus Latrunculorum being one of them. This game mirrored the military tactics of that era, and it resembles chess as it is known today. Moving on to 1100 AD an ancestor of dominos, was brought up in Ancient China. Later, in 1974 role playing games begun to rise, Dungeons and Dragons being the base of them. A lot of variations, single player and multiplayer boards came in followed by computer games. This genre of games relied on imagination led entertainment as the adventure was different every time they played.

Furthermore, Catan that came out in 1995, was a widely played game with over 24 million copies sold in 30 different languages, that made a milestone in the board game history, Lastly, crowdfunding platforms like Kickstarter, launched in 2009, made a revolution in the board game industry. Anyone with an idea could have a chance to publish and share it with the community, bypassing the publishing barriers and allowing diverse innovative games to reach the market.

2.2 Board Game Future

Reflecting [14, 21] how many people and how many diverse board games are for sale today it is more than obvious that they are never going to stop being updated or become extinct. With the rise of computers and AI a huge and entirely new field of games is going to be developed in the future.

It is safe to say that nowadays digital entertainment is dominating a massive part of the population. New technologies featuring augmented reality or virtual reality are inviting a game experience far beyond the physical board. Applications accessible through mobile phones and tablets are resetting the rules of board games, minimizing the distance between players with remote play and generally playing a huge role in the accessibility sector. Creators and developers will definitely find a demanding, diverse and expanding audience as well as a huge gap open for creativity and boundless possibilities.

On the other side, a hybrid version of board games has been introduced, by merging physical and digital technologies. Both worlds create a new way of playing with the flexibility of technology and the hands-on experience of the physical. It is safe to say that these kinds of games are set up and explained easier thus, reducing the rejection of more complex games.

AI has become more accessible than ever, making the testing and mechanics of games done in no time, leaving room for improvement and quality brainstorming. Furthermore, it can assist in simulations or even evaluations of ideas and objectives. Artificial intelligence is upgrading the user experience, adding a virtual strong opponent to games bringing a fascinating new era.

To conclude, the future of board games is brighter than ever. Countless possibilities, flexibility, creativity, and personalization are key factors in today's fast-paced technology-driven world. This approach will result in fresh and unique ideas.

Chapter 3

Αντρίν Game

3.1 Αντρίν Game Explanation – Rules

This game falls into the strategy game genre. It is played by two players that are opponents. To play the physical game, a board and six tokens (three for each player) are required.

To set it up, the board is positioned between the players with each player holding three tokens of the same color. The layout of the board is shown in Figure 1.1. In the beginning there are nine spots in total that a player can put a token.

Figure 1.1: Andrin game setup

Winning boards are all the boards that either one of the players has three tokens on three consequent spots, either vertically or horizontally. It is important to note that diagonals are not winning positions in this game. That states that there are six distinct ways a player can win. All the ways a player can win are shown in Figure 1.2.

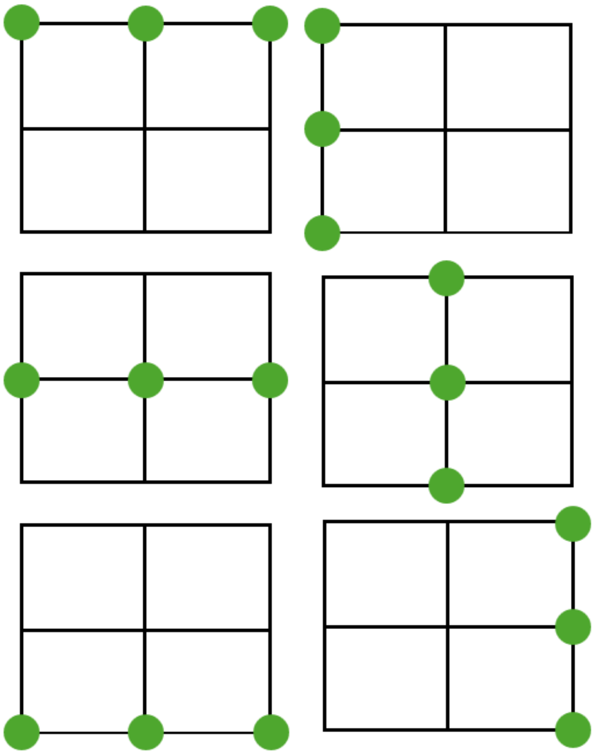


Figure 1.2: Set of winning boards

Αντρίν consists of two phases:

In the first phase, the game begins with an empty board. The two players decide who plays first and then take turns (three turns in total for each player). In each turn, the player must place one token on any empty spot on the board. If by the end of phase one, either player successfully achieves a win, that player immediately wins and the game ends.

During the second phase, all six tokens of the two players are on certain spots on the board. Players now alternate turns. In each turn, the player must move one of his tokens to an empty, adjacent spot. Players continue this routine and try to achieve a win. If a player successfully achieves a win, that player immediately wins and the game ends.

3.2 Similar Game Findings and Comparison

During my research, the first goal was to find any documentation or implementations of Αντρίν. The following represent the google searches I tried to find the game:

1. Αντρίν
2. Αντρές
3. Αντρές παιχνίδι
4. Αντρίν παιχνίδι

5. Παραδοσιακά παιχνίδια του Πάσχα
6. Παραδοσιακά παιχνίδια
7. Παραδοσιακά παιχνίδια αντρή

After multiple searches, the only reference I found [6] was a website that presented some traditional Cypriot games and one of them is a brief explanation of the game "Αντρές ή Αντρίν". The board mentioned in that website is slightly different to the one this thesis deals with.

Furthermore, my research continued with the goal of finding similar games. The results showed that similar games exist but with different board and slightly different rules that match the board. Some of the games include:

Achi

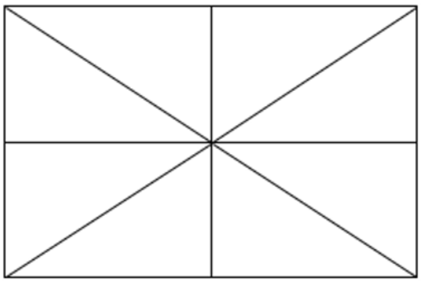


Figure 1.3: Achi game board

Nine Men's Morris

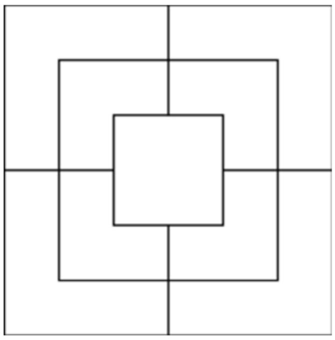


Figure 1.4: Nine Men's Morris game board

Other similar game findings

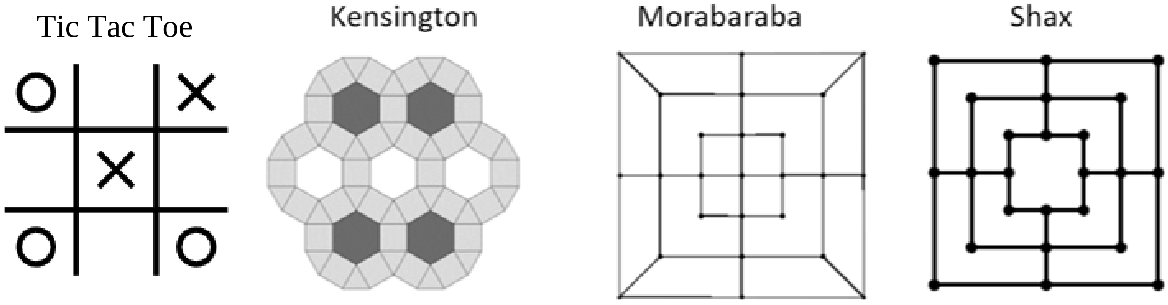


Figure 1.5: Other similar games

The research ended with a brief comparison of the 3 most similar games, Achi, Nine Men's Morris and Tic Tac Toe with Αντρίν. The table in Figure 1.6 visualizes the results.

	Andrin	Achi	Nine Men's Morris	Tic Tac Toe
Phase 1: put all tokens	✓	✓	✓	✓
Phase 1: three turns each player	✓	✓	✓	✗
Phase 2: move tokens	✓	✓	✓	✗
Win: three tokens at same line	✓	✓	✓	✓
Move only one token at turn	✓	✓	✓	✗
Move only one space at turn	✓	✓	✓	✗
Can jump over opponent	✗	✗	✓ (under condition)	✗
Possible starting points	9	9	24	9

Figure 1.6: Game comparison

What follows this comparison is the fact that the four games gave distinct features and probably strategic ideas. The analysis aimed to provide a deeper understanding of possible similarities and differences as well. Lastly, the games mentioned above will be of use in later analysis of Αντρίν.

Chapter 4

Game Analysis

4.1 Simple N = 3 Game Analysis

Αντρίν game as mentioned before, has a board that contains vertical and horizontal lines, forming a grid. The initial game of Αντρίν consists of three horizontal and three vertical lines. It is important that the board must be a square, in other words vertical lines must be equal to horizontal lines. Setting N as the size of the board, in essence the number of lines horizontally and vertically. Some examples of various N values are illustrated in Figure 4.1.

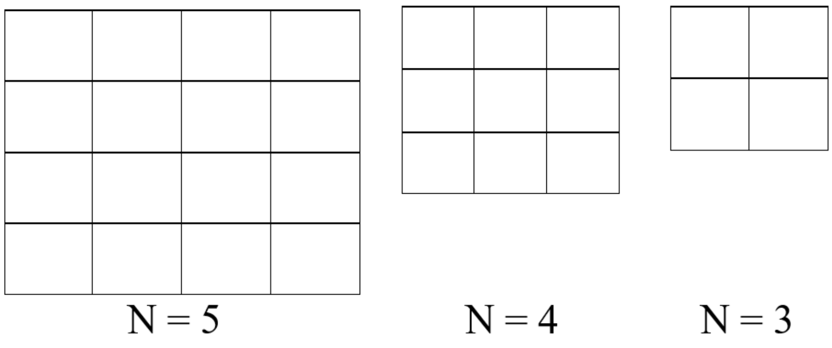


Figure 4.1: Board with different N values

In this thesis the board is always going to be for N = 3 and the following analysis takes in place the first phase.

As previously discussed, in the first phase players take turns to place their tokens on the board. After a quick study of the board, it becomes evident that there are nine available spots in total. However, each time a player takes a turn the available spots for the next turn start to decrease. In order to communicate the analyis, each player has three turns and there are six steps in total . It is important to note that the sixth step will occur in the case that the first player did not win dyring step five. This decrement in the number of availble spots follows the below order:

Step 1: first turn of player one – 9 available spots
Step 2: first turn of player two – 8 available spots
Step 3: second turn of player one – 7 available spots
Step 4: second turn of player two – 6 available spots
Step 5: third turn of player one – 5 available spots
Step 6: third turn of player two – 4 available spots

Regarding the above, for the abstract analysis of the number of availble spots in each player's turn it is important to remember that the players play to win. With that in mind, in theory the number of availble spots for each step are the ones shown above, in practise some of them lead the player to lose, thus not advantagous. This leads to the conclusion that part of some numbers representing the available spots are not in the player's advantage, but against them. Meaning that sometimes even though the player, for instance, has six available spots infront of them, the opponent might win in the next round if the player does not block them in this round, leading them to decrease there availble spots from six to one non losing choice. It is important to note that this is not the case every time, it can occur when the opponent has two tokens on the same line and it is the player's turn to add a token. This could be easily understood from Figure 4.2.

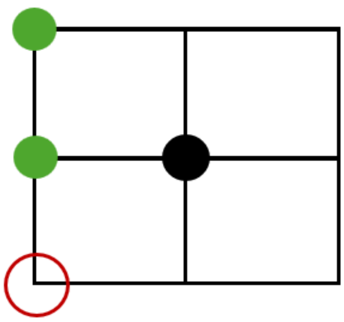


Figure 4.2: Available spots decrease

In Figure 4.2 player one (green) finished two turns and player two (black) has finished one turn. Player two must take his second turn now and there are six available spots to choose from. In practice since the players are playing to win or in other words to not lose, player two has now only one choise (red). If player two places his token in any other spot, player one is most likely to win.

The last small part of the analysis is the possibility of a player not being able to move in phase two. This is not a common occurrence in the game but it is a possibility. The only way a player will be blocked is the one shown in Figure 4.3, in which player two (black) cannot move anywhere on the board. This happens and it makes sense to mention it since none of the players is in a winning position and all of the tokens of the player whose turn is are blocked.

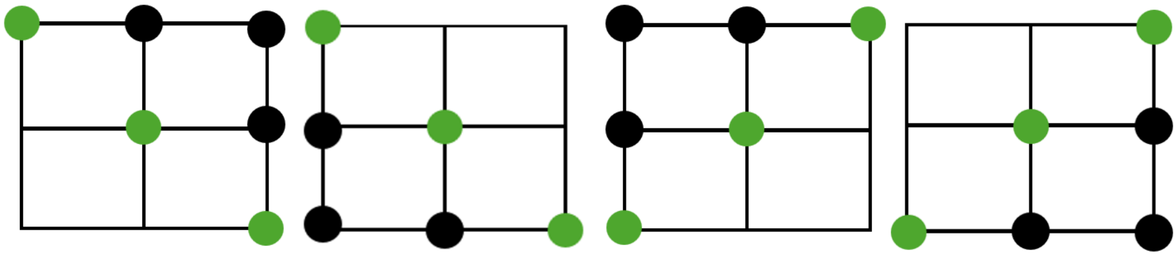


Figure 4.3: Blocked player boards

4.2 The goal

As a continuation of the analysis in the previous subchapter, the following is the primary goal of the biggest fragment of this thesis.

To start off, the previous N = 3 analysis stated that the decrease in available spots on the board while the game moves forward are as follows: 9, 8, 7, 6, 5, 4. Taking that into consideration, it is easy to calculate the total number of possible boards through the game's phase one which are 9*8*7*6*5*4 = 60480. The result 60480 represents the number of potential game states withing the first phase of the game. It is understandable that the number indicates the complexity of the game when trying to comprehend the different board placements that can occur in phase one.

However, given that there are 60480 total board placements, a question regarding whether they are all unique has arisen. This question came to the surface because given the nature of the board, there is symmetry. This is a question that this thesis answers. Analytically, the total number of possible board placements in each step follows:

Step 1: 9

Step 2: 9*8 = 72

Step 3: 9*8*7 = 504

Step 4: 9*8*7*6 = 3024

Step 5: 9*8*7*6*5 = 15120

Step 6: 9*8*7*6*5*4 = 60480

An important note, is that during steps one through four none of the players can win. However, in step five there is a possibility that player one might win and in step six the socond players has a chance of winning.

4.3 Board Encoding

For effective communication of each distinct spot on the board, an encoding of the board is required. The board encoding used in this thesis is the one illustrated in Figure 4.4.

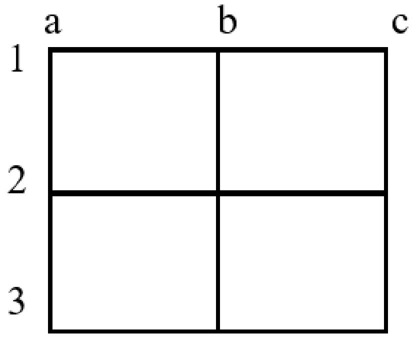


Figure 4.4: Board Encoding

Clear declaration of the nine spots on the board proceeds:

Top left – a1

Top middle – b1

Top right – c1

Middle left – a2

Middle – b2

Middle right – c2

Bottom left – a3

Bottom Middle – b3

Bottom right – c3

4.4 Initial Brute Force Pattern Analysis

In the beginning, the first thought that comes to mind is to try and illustrate from the beginning what are the possible board placements of phase one. By doing that it will be visually easier to spot boards that are the same or could be the same. It is obvious that when trying to think of the 60480 possible board placements it is not possible to do a correct analysis with valid conclusions without missing something.

As discussed before in the first steps of phase one, the board placements are not that many. That means that a small scale analysis could take place. In Figure 4.5 all the possible board placements for step one are illustrated.

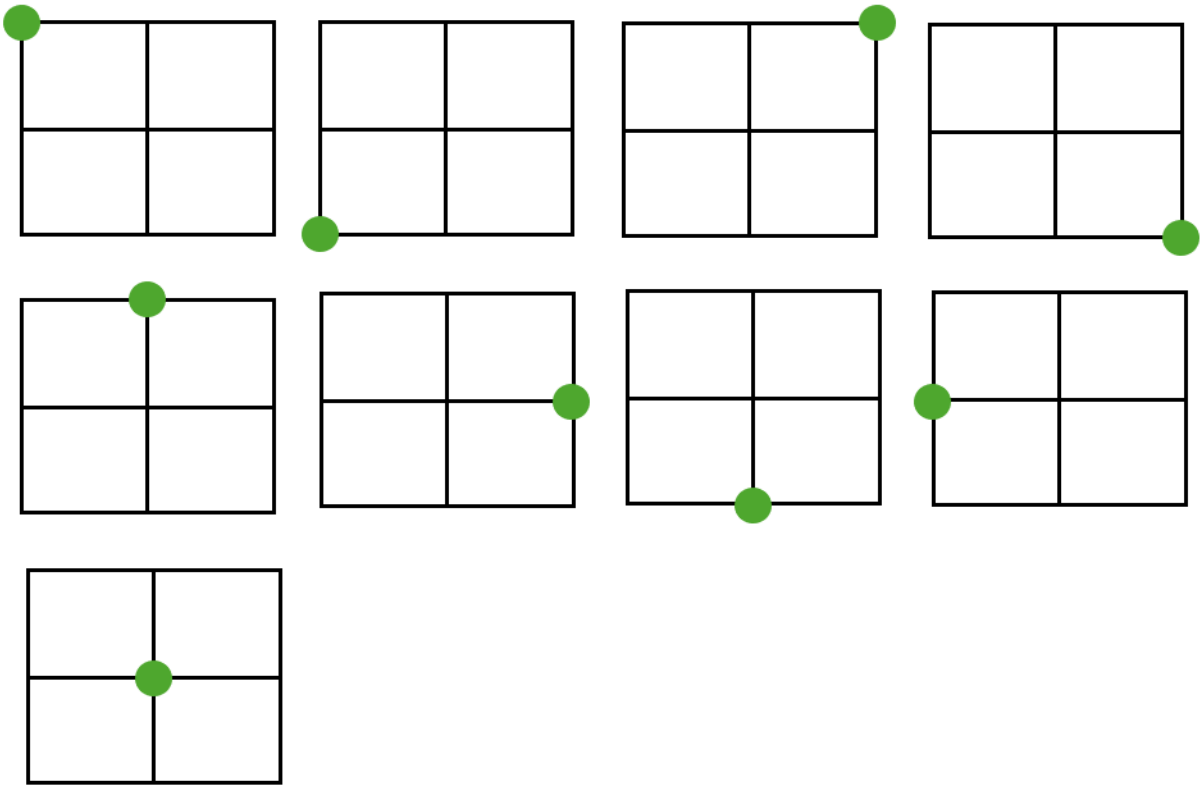


Figure 4.5: Step One Possible Board Placements

After a brief analysis of the boards some patterns became visible. Specifically there are three patterns. The first one is noticable in the first line of Figure 4.5 in which the four corners are chosen as the move in step one. It is safe to say that when one of the four specific choices is made the board is still the same, in other words the four boards can be thought of as one unique board with weight four. The same applies to the second line of Figure 4.5 in which any one of the four choices is still the same, thus one unique board with weight four. Lastly, the last line in Figure 4.5 the middle spot is chosen and that can be thought of as one unique board with weight one. Therefore, in step one there are three unique boards, cornern with weight four, middles with weight four and center with weight one. For verification the sum of the weights, 4+4+1 = 9, equals to all the empty spots there are in step one.

If the analysis continues in step two, the second player must take a turn. Given that in the previous step there are only three unique boards, step two is going to follow with only those.

Let's assume that player one chose upper left corner for step one. Taking advantage of the board's symmetry, the sub patterns of step two and corner as the first choice would be five.

That is illustrated in Figure 4.6. For example, if player two chooses either one of the spots marked as 1, it would give the same result, thus there is one unique board with weight two, and so on. That results in weight four for the corners, and five sub patterns (1 – weight two, 2 – weight two, 3 – weight two, 4 – weight one and 5 – weight one). For verification each sub pattern should be multiplied with the weight of the corners, four. That leaves us with result 32.

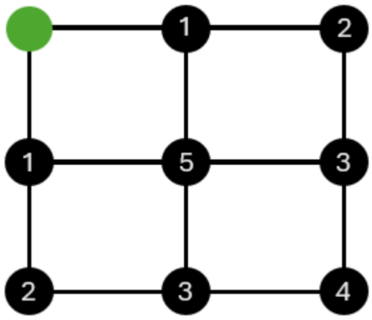


Figure 4.6: Corner – Step Two Choices

Moving on, it is assumed that player one chose upper middle for step one. Taking advantage of the board's symmetry, the sub patterns of step two and middle as the first choice would be five. That is illustrated in Figure 4.7. For example, if player two chooses either one of the spots marked as 1, it would give the same result, thus there is one unique board with weight two, and so on. That results in weight four for the corners, and five sub patterns (1 – weight two, 2 – weight two, 3 – weight two, 4 – weight one and 5 – weight one). For verification each sub pattern should be multiplied with the weight of the middles, four. That leaves us with result 32.

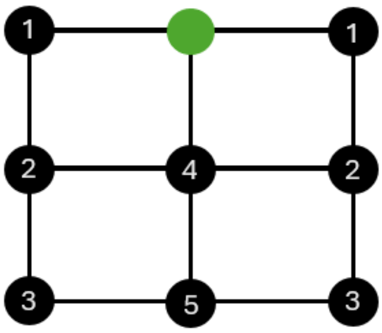


Figure 4.7: Middle - Step Two Choices

Lastly, it is assumed that that player one chose center for step one. Taking advantage of the board's symmetry, the sub patterns of step two and center as the first choice would be two. That is illustrated in Figure 4.8. The results for the weights would be one for the center and two sub patterns (1 – weight four, 2 – weight four). For weight verification each sub pattern should be multiplied with the weight of the center, one. That leaves us with result 8.

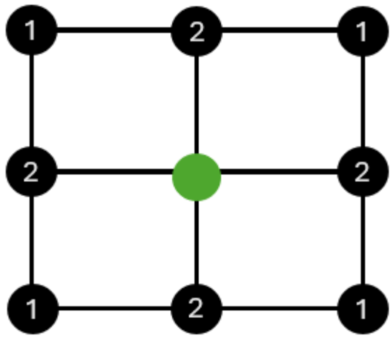


Figure 4.8: Center - Step Two Choices

After this brief analysis for step two, there are 32 + 32 + 8 = 72, verified correctly with the numbers in section 4.2.

To conclude, given the large number of possible board placements in phase one, which is 60480, it would be hugely time consuming and with a lot of room for error, to do this analysis mannually and by inspection. Thus, a strong need for another solution emerged, either another aproach or with the use of some operators and a program.

4.5 Retrograde Analysis Approach Rejection

During the process of research to find a possible effective approach to find the number of unique board placements in phase one, the retrograde analysis sprouted. This approach seemed to be an excellent fit for the purposes of this thesis [10, 15, 18, 19].

Retrograde analysis is primarily used in chess analysis. However, during searches on whether it was ever used for other games, there was a work that was used on the Nine Men's Morris game. This game is similar to Αντρίν. Generally, this analysis is used when there are multiple scenarios that could happen in a gameplay and are happening exponentially. Additionally, some of the resources mentioned it as "the procedure of "playing" the game".

Specifically, what happens is that an outcome of the game is picked and then by continuously going back a step every time, a critical path is formed leading up to a starting position. In other words, a guide on how to get from a starting position to a desired outcome, in a backwards manner. It is important to note that all the moves chosen must be legal in order to find the optimal play for all possible board positions in a specific endgame.

In conclusion, even though it could be a good approach for the problem of the unique board placements it was rejected in this thesis. This analysis could be more fitting for the second phase. A much simpler way was to move forward for the analysis of phase one of the game, is with a set of operators, explained in the next chapters accompanied with a program that automatically calculates the number of unique boards, limiting the room for error.

Chapter 5

Operators

5.1 Initial Set of Operators

As mentioned before, a need to see the board from a different perspective grew as the question about how many boards in the first phase are really unique arose. Given the large number of board possibilities, it became inevitable that something giving insights and patterns on the boards would be wanted. Given the nature of the board, the operators appeared to be straightforward. In this section rotation and four different mirrors will be introduced. This set was the initial and broader set of operators which later was assessed and reduced to only the necessary.

The first operator is called Rotation. The name is self-explanatory, and it signifies a complete clockwise turn of the board. A simple example is shown in Figure 5.1.

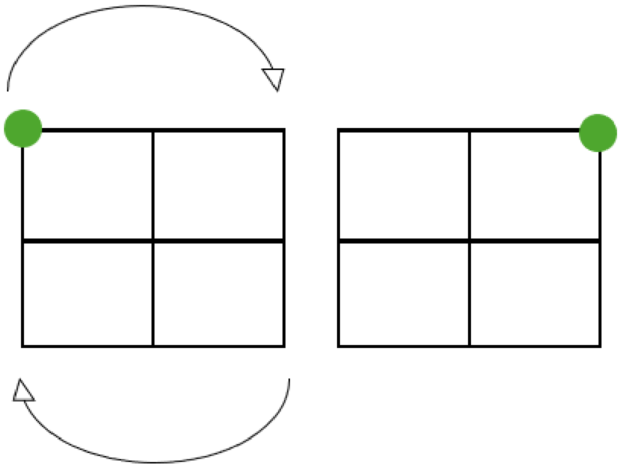


Figure 5.1: Simple rotation example

Furthermore, the Rotation operator has five stages which are illustrated with the help of an example in Figure 5.2 with the first board being the initial board. It is important to note, that this operator applies with any number of tokens on the board and the boards shown on Figure 5.2 are an example. In more detail, Rotation 0, the board remains unchanged, Rotation 1, one clockwise rotation, Rotation 2, two clockwise rotations, Rotation 3, three clockwise rotations and Rotation 4, returning the board to the starting position with four clockwise rotations. This approach to rotation enables a new way of exploring the board placements and identifying patterns.

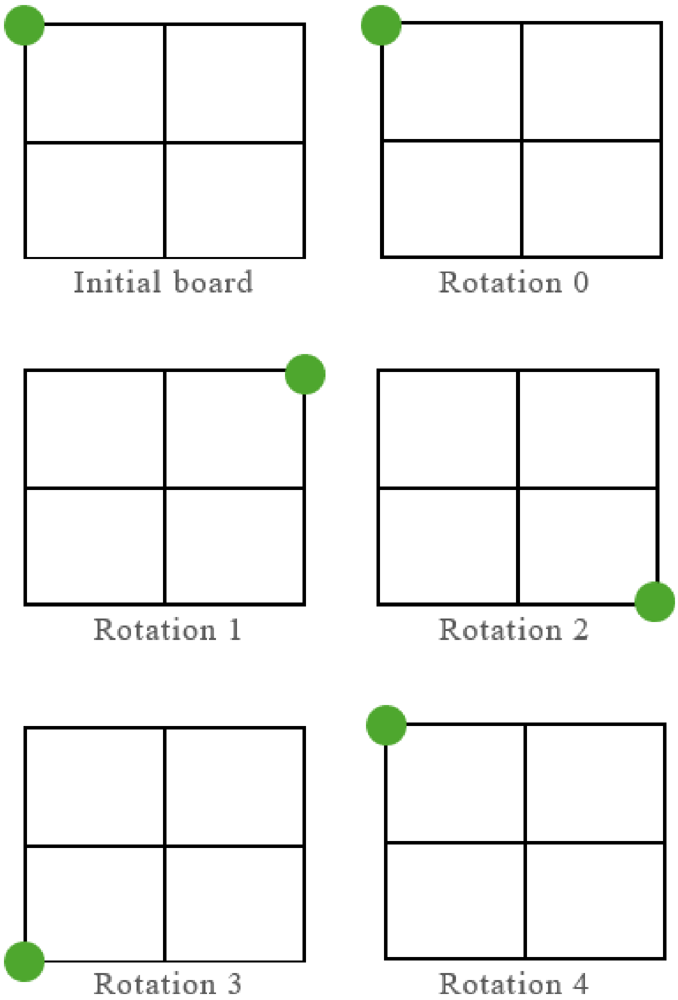


Figure 5.2: Rotation 0-5 operator

Moving on, the other important operator is called Mirror and at this stage it has four different variations: Vertical Mirror, Horizontal Mirror, Upward Diagonal Mirror, and Downward Diagonal Mirror. It is important to state that this operator uncovers possible symmetries, and it resembles imagining a mirror places on the board. This operator is easier to understand with the illustration in Figure 5.3. In this figure, at the left column there are the starting board placements, and at the right column there are the corresponding mirror variations mentioned above. The red line indicates the imaginary mirror and the empty circles the mirrored tokens. It is important to note that the board placements shown in Figure 5.3 are just examples and the operators can be applied on boards with any number of tokens.

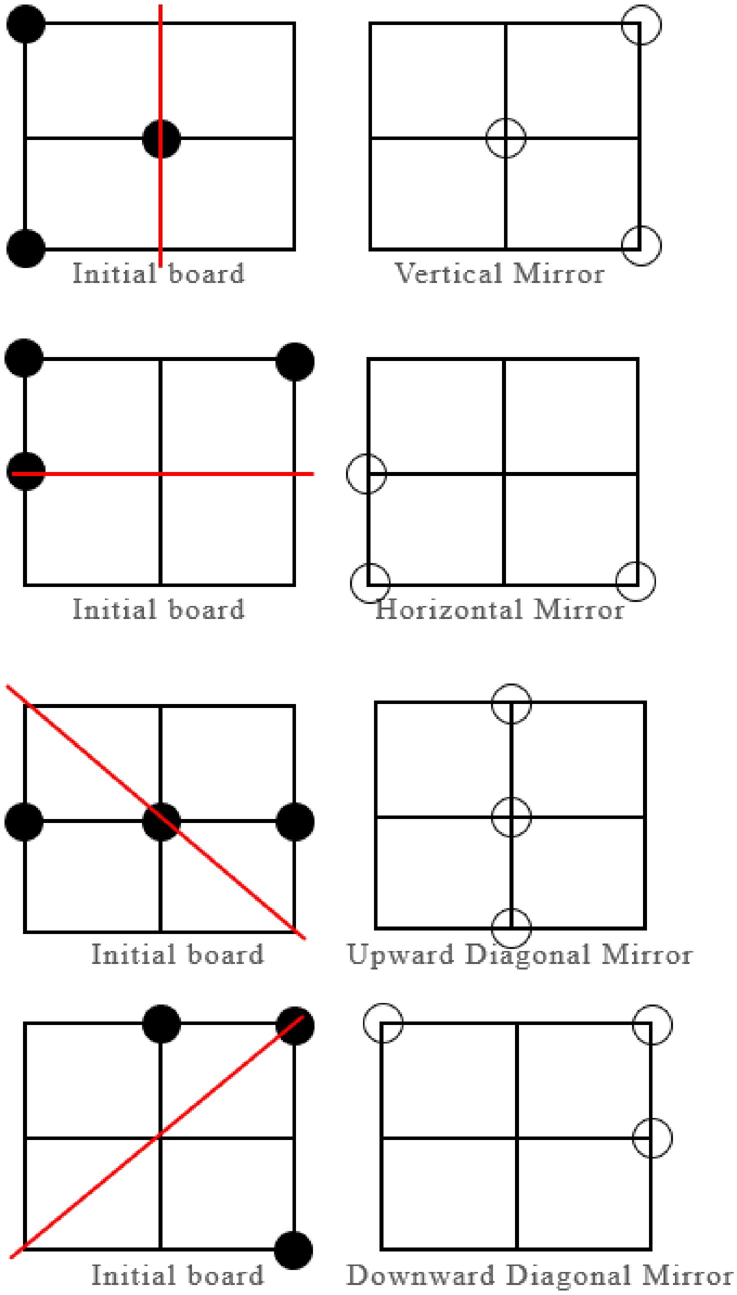


Figure 5.3: Mirror Operator Variations

After thorough application of these operators, the aim was to deepen the unserstanding of the game and help minimize the number of unique boards in phase one as mentioned before, using them as the most important tools.

Lastly, for simple understanding there are two definitions that need to be explained, in order to establish a common ground. The two words are same (τα ίδια) and equivalent (ισοδύναμα). The definitions are straightfoward and make it easy to identify the difference. When it is said that two boards are the same, it means that the placement of the tokens on the board is exactly the same. For example, board one has player one's token on a1 and board two has player one's token on a1. That concludes that board one and two are the same.

Furthermore, when it is said that two or more boards are equivalent, it means that two or more different token placements result in the same game play or the same board after applying any sequence of operators on them.

5.2 Evidence Against Necessity of Subset of Operators

Revisiting the previous subchapter, the conclusion was that there is a set of operators consisting of the five Rotations, Vertical, Horizontal, Downward and Upward Mirror operators. Since the problem is computationally hard and time consuming, by adding such a big set of operators the problem could become more complicated. With that in mind, the question of whether all the operators proposed are necessary arose. There was a possibility that some of them overlapped or produced the same result as another one.

The first step was to assess whether both vertical and horizontal mirrors were necessary. These mirrors appear to be very similar to each other and considering the nature of the board they might not be needed. It is assumed that rotations, upward mirror, and downward mirror are necessary operators. To eliminate horizontal and vertical mirror operators, every board placement these operators can produce should be generated by the other essential operators. In Figure 5.4 and Figure 5.5 there are some examples in which it is illustrated that the results a horizontal and vertical operator produce, can also be produced using a series of the other operators.

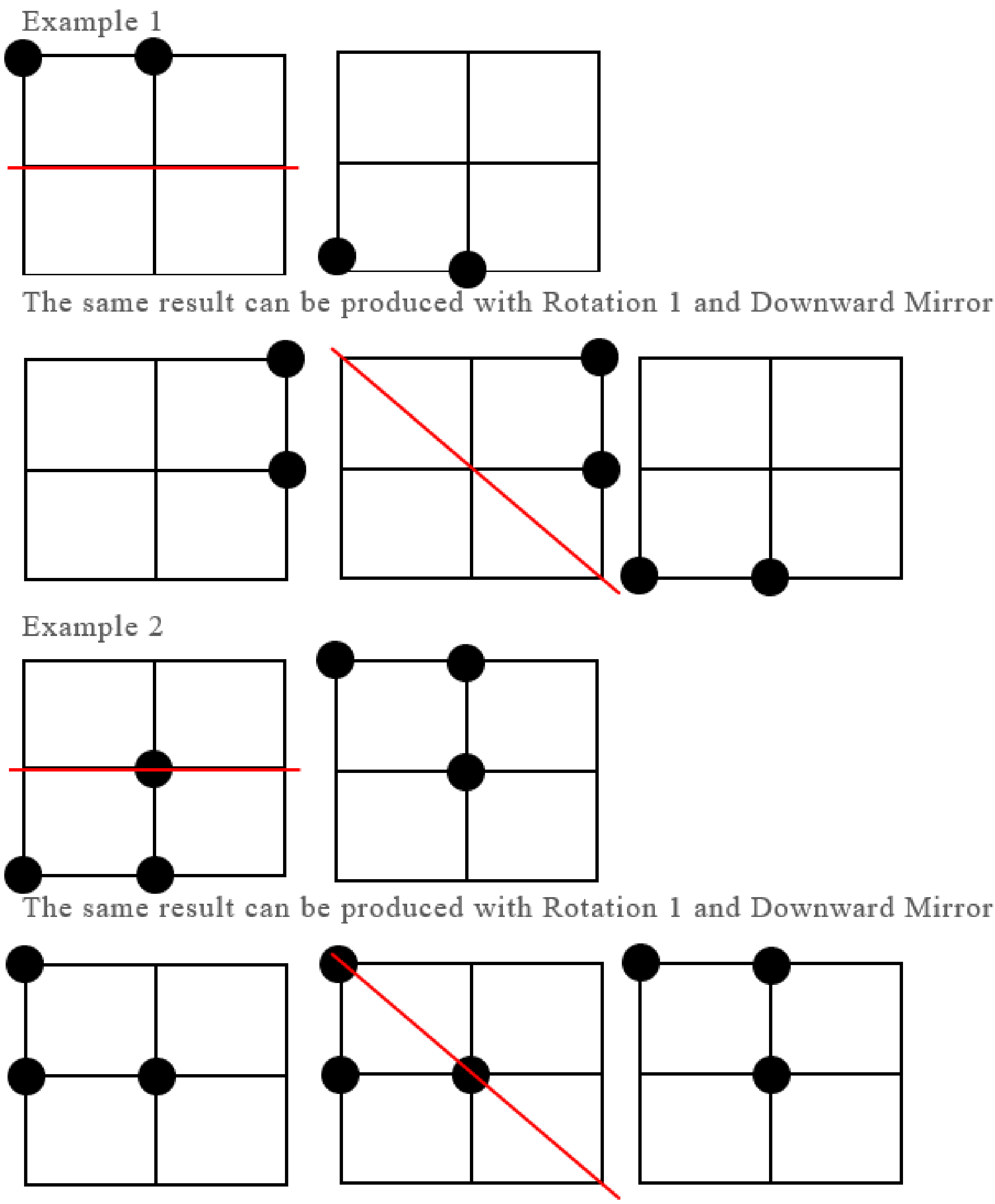


Figure 5.4: Horizontal Mirror Rejection

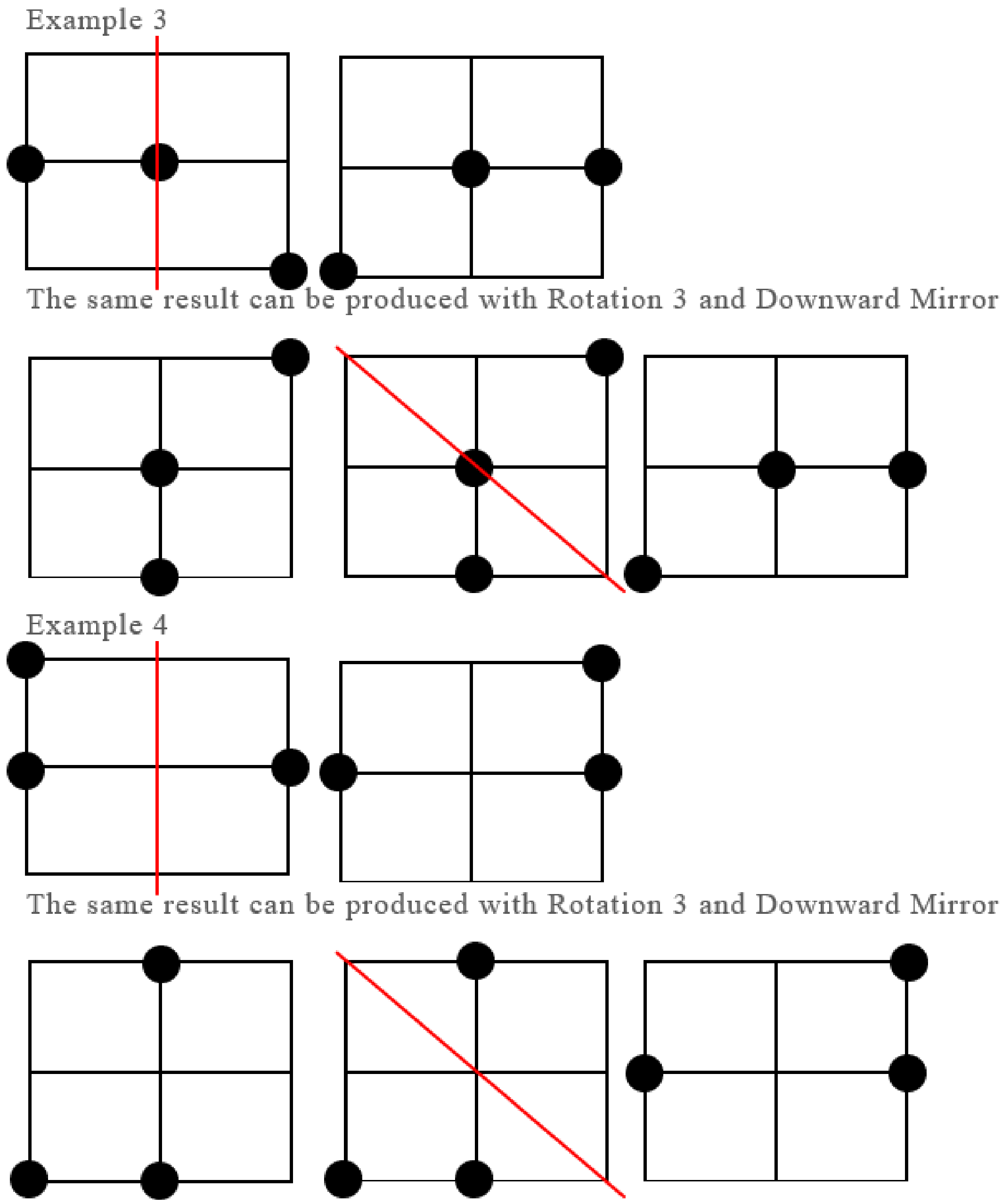


Figure 5.5: Vertical Mirror Rejection

It is concluded that horizontal and vertical mirror operators are no longer in need as the other essential operators can produce the same results as them. The set of operators now is reconstructed with only one rotation, downward mirror, and upward mirror. It is important to note that this is not proof but evidence that the operators are not needed.

Moving forward, the significance of both downward and upward mirror came under judgment. The way this was assessed was similar to the way vertical mirror and horizontal mirror were assessed. Similarly, taking into consideration the nature of the board there are leads indicating that both operators might not be of use. Considering that rotation, is a necessary operator, to eliminate one of the upward and downward mirrors or both of them, the other operators must be able to produce any board the mirrors can. In Figure 5.6 there are some examples which the necessity of only one of them seems to be necessary. In other words, any board the downward mirror can generate, the rotations and upwards mirror can do it as well. Likewise, any board the upwards mirror can generate, the rotations and downward mirror can do it as well. It is important to state that since rotation alone cannot produce all the board the mirrors can, then it cannot be standalone, which is explained in following paragraphs.

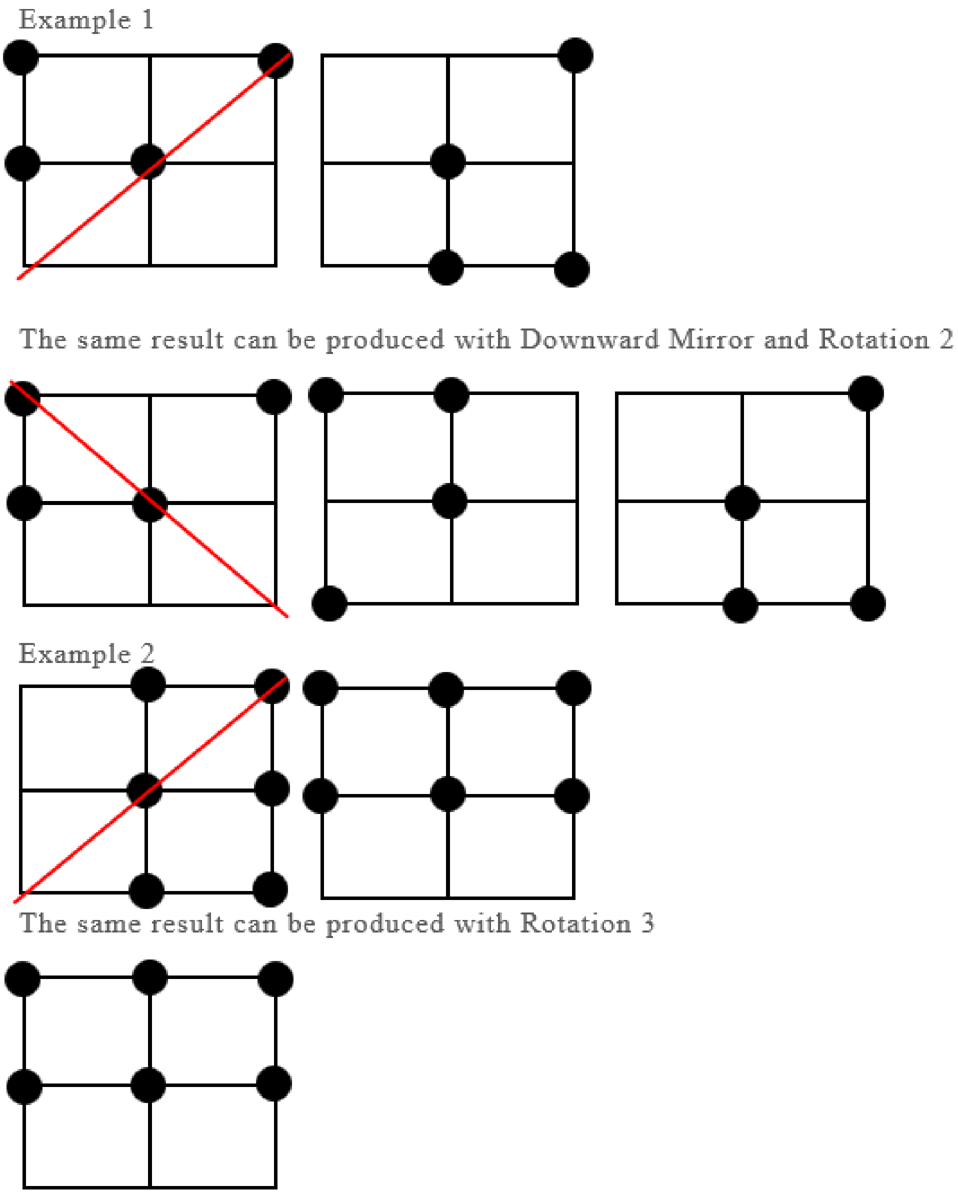


Figure 5.6: Upward mirror rejection

It is concluded that, downward and upward mirror are not both needed but only one of them. For this thesis, it was chosen to move forward with the downward mirror and eliminate the upward mirror. As a final point, it should be mentioned that now the set rests to the rotations and downward mirror operators. It is important to note that this is not proof but evidence that the operators are not needed.

Now the only one thing remains, to evaluate the possibility of Rotation operator not being able be standalone. In logic, since the nature of rotation and mirror are different, the rotation should not be able to exist on its own. To prove that rotation can exist without the downward mirror the following procedure takes place:

Firstly, it is assumed that the rotation can exist alone. It is indicated that taking a random example like the one illustrated in Figure 5.7, can be produced only with the downward mirror. All number of rotations, zero through four, are tested and illustrated. It is obvious that there are no number of rotations that can be done to generate the same result of downward mirror. This is, in fact, a counter example and thus the above assumption cannot be proved, hence the rotation cannot be a standalone operator.

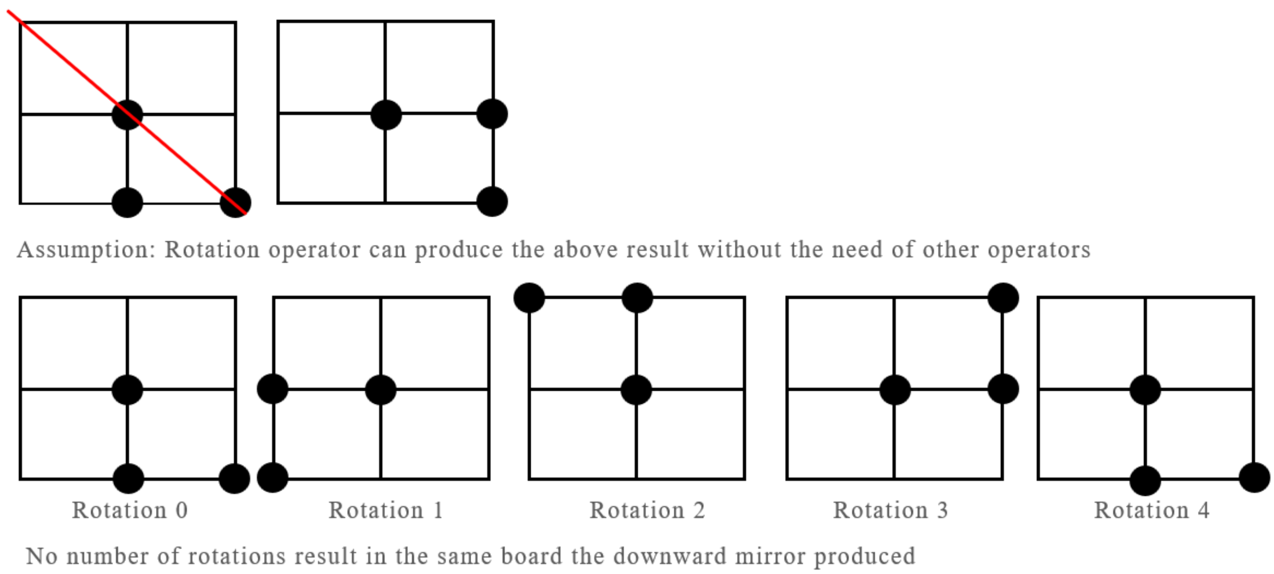


Figure 5.7: Rotation Sufficiency

To conclude, after the assumptions and all the revisits of the set the final set that is used from now on contains the five rotations and the downward mirror. It is important to note that this is not proof but evidence that the operators are not needed.

Chapter 6

Algorithm for Unique Board Placement

6.1 Board representation for the Algorithm

To start off, when trying to transfer the game board into a code it is necessary to have a way to represent it and be able to easily change and manipulate it. After assessing the possible choices and ways to imprint the board, the best pick was a one-dimensional array with nine cells. Each cell signifies one of the nine spots on the board.

The primary reason for the one-dimensional array choice is the following. It is visible that an array is easily manipulated and changed. Taking into consideration the nature of the two operators, rotation and downward mirror, the following observations, on consistency have been made:

1. Spot b2 (center) never changes even after operators are applied on the board.
2. Corners (a1, c1, c3, a3) always change up with corners after any operator is applied on the board.
3. Middles (b1, c2, b3, a2) always change up with middles after any operator is applied on the board.

Keeping the above three observations in mind, the encoding of the board is illustrated in Figure 6.1. That specific array was chosen in order to group the three types of spots on the board (corners, center, middles) for easier manipulation. In other words, the operators are applied on each board placement in a simpler way.

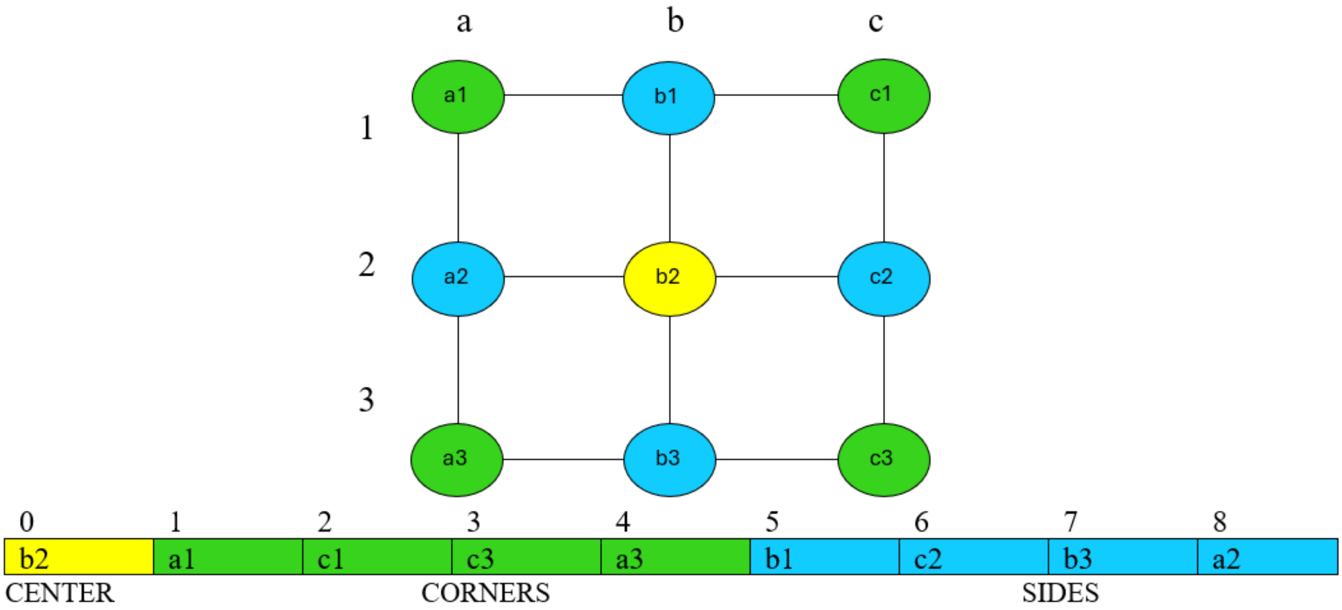


Figure 6.1: Board representation on algorithm

To simplify matters, a set of colors is used to indicate the corresponding board spots on the encoded array. The color green shows the four corners (a1, c1, c3, a3), color yellow shows the center (b2), and the blue shows the four middles (b1, c2, b3, a2).

Since there are two players, to indicate on the board which player is on what spot at any board placement, wherever player one has a token there will be a 1, wherever player two has a token there will be a 2 and lastly, wherever there are no tokens there will be a zero.

- 1 -> player 1 has a token on corresponding spot
- 2 -> player 2 has a token on corresponding spot
- 0 -> no token on spot

An example is illustrated in Figure 6.2.

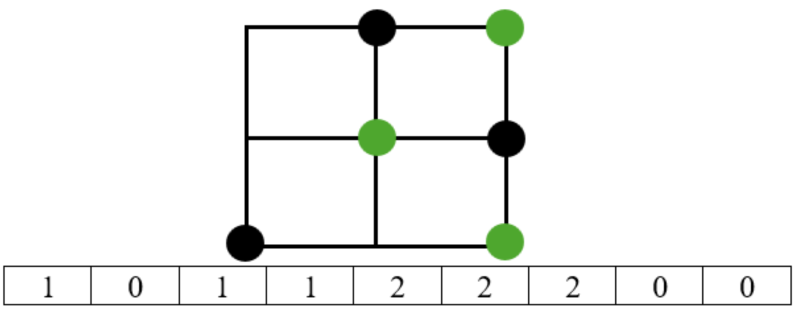


Figure 6.2: Board representation

To perfectly understand the board representation and how it works when applying operators Figure 6.3 helps. The figure illustrates how the board changes when Rotation 3 and downward mirror is applied on the starting board.

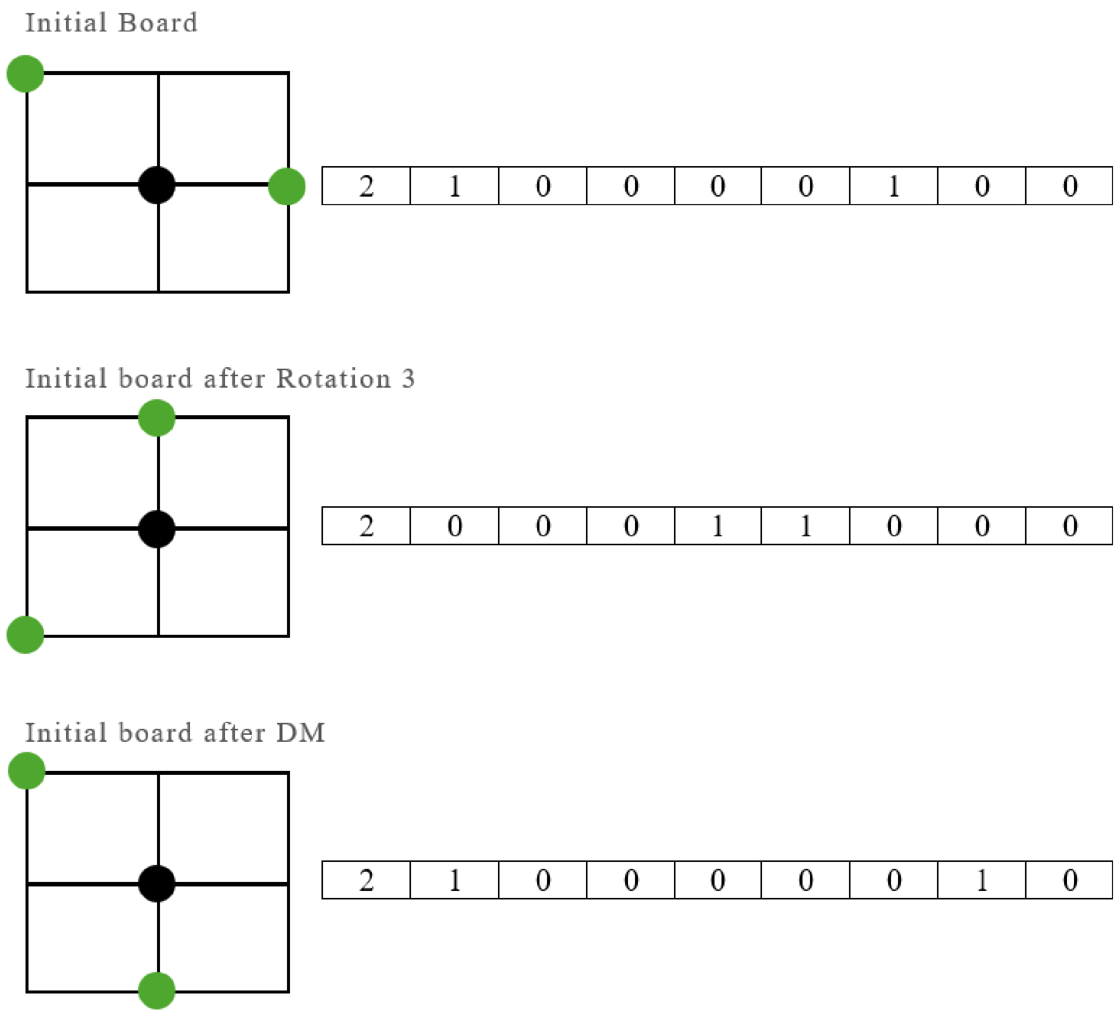


Figure 6.3: Board operator manipulation

To conclude, the use of a one-dimensional array with nine cells effectively simplifies the process of manipulation and representation of the game board at any instance.

6.2 Algorithm

To start off, it is important to note that this is one of the most significant and attention worthy subchapters of the whole thesis. Remembering the goal mentioned in a previous subchapter and summing up all the prework done with the board encoding, board representation and the making of an operation set this is the time to build an algorithm that finds an answer to the question. To refresh the memory, the big question was how many unique possible board placements are in phase one of the game.

When developing an algorithm, it is crucial to have cleanliness, understanding and correctness. The following algorithm, after a lot of thought and editing, is accurately counting the number of unique boards in each one of the six steps of phase one of the game.

First of all, some important declarations to be able to follow on with the algorithm:

- Array G – initial empty array
- Operators – R1, R2, R3, DM
- Sets of unique sequences (for each of the six steps) with corresponding weight – s1, s2, s3, s4, s5, s6
- Board encoding: b2, a1, c1, c3, a3, b1, c2, b3, a2
- Symbolism: 0 – no token on corresponding spot

  1 – token of player one on corresponding spot

  2 - token of player two on corresponding spot

Algorithm

---

1. Empty array G
2. For every cell with value 0 in G produce a new sequence by adding **value 1** to that cell

   2.1 Check if that NEW sequence exists in s1

   2.1.1 If it exists increase the weight of that sequence in s1 by 1 and move on to the next sequence.

   2.2 Apply to the sequence R1, R2, R3, DM, DM + R1, DM + R2, DM + R3

   For every sequence produced after EACH of the operators is applied, check if it exists in set s1

   2.2.1 If it exists increase the weight of that sequence in s1 by 1 and move on to the next sequence

   2.2.2 If it does not exist after all the operators are applied, add it in s1 with weight = 1 and move on to the next sequence.

Result of step2 -> s1 full of every unique sequence of 1 token with corresponding weight.

3. Get as input s1 and for every sequence and for every cell with value 0 add **value 2** to that cell

   3.1 Check if that NEW sequence is in s2

   …
4. Get as input s2 and for every sequence and for every cell with value 0 add **value 1** to that cell

   …
5. Get as input s3 and for every sequence and for every cell with value 0 add **value 2** to that cell

   …

6. Get as input s4 and for every sequence and for every cell with value 0 add **value 1** to that cell

   …

7. Get as input s5 and for every sequence and for every cell with value 0 add **value 2** to that cell

   …

8. After step 2 through step 7 the result is the unique sequences with the corresponding weight in the sets s1, s2, s3, s4, s5, s6 (each one containing the unique in each step, from 1 token through 6 tokens)

   Calculate total unique sequences = s1.size + s2.size + s3.size + s4.size + s5.size + s6.size

---

As previously mentioned, this algorithm produces and evaluates all possible configurations of the board and ends with the number of unique board placements for every step, and thus for whole phase one.

The algorithm starts off with an empty board, G. From steps two to seven one token is added at each time for the corresponding player, three for player one and three for player two alternatively. Starting from step two, there is only one board which is empty, and it is given as input, the G board. For every cell with value 0 in G, the algorithm produces a new sequence by adding value 1 to that cell. Now there are nine different boards to assess in total.

Every time a new board (sequence) is produced it is checked whether it exists in the corresponding set (s#). If it exists then just increase the weight of that sequence on the set and move on to the next board placement. Since it is mentioned, the weight represents the number of times that sequence was found in total. If the sequence does not exist in the set, move on to the application of the operators one by one. Each time check if the produced sequence exists in the set. If it exists then increase the weight by one and move on the next board placement. If it does not exist move on to the next operator. If all the operators are applied and there was no sequence that was the same in the set, then add it to the set with weight equal to one and move on to the next board placement. When this procedure comes to an end, the set consists of only the unique board placements with their corresponding weight.

A crucial mechanism is that, every step from three through seven, the previous step's set of unique board placements is passed on to produce the new ones. That is important and it happens because there is no reason to produce all the possible new board placements for each step, since the goal is to find the unique each step is based on the unique of the previous step.

The procedure is the same from step three through step seven. In order to produce every possible board placement each time a token is placed in every empty spot of the boards from the set (s#) given as input.

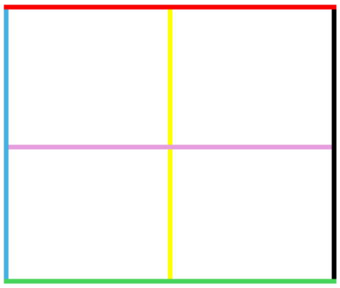
When the algorithm terminates, the s1, s2, s3, s4, s5 and s6 sets are full of unique boards with the corresponding weights. When adding the sizes of all the sets, the result represents the total number of unique board placements in phase one of the Αντρίν game.

6.3 Implementation

The algorithm described above was implemented using the Java programming language in the Eclipse IDE workspace. The implementation was carried out using OpenJDK version 11.0.11 and Eclipse IDE 2021-06 (4.20.0) version. Since the program does not rely on any specialized implementation features, it should be compatible with any Java versions and can be executed in alternative workspaces beyond Eclipse.

Before continuing with the output explanation and analysis, it is important to note two additional functions that were added to the program. Firstly, the winning positions were counted alongside the counting of the unique boards. Secondly, some kind of verification was necessary to guarantee the accuracy of the results.

To count the winning boards, the conditions in which there is a win were necessary to be defined. It was previously mentioned that in order for a player to win, three tokens of the same player must be on the same line, vertically or horizontally. Now that the board representation for the program is established it is important to show the winning positions of the board using that representation. In Figure 6.4 the six possible winning boards are represented using colors to easily match the line with its encoding. The example assumes that player one is the winning player, but the same applies for player two. Additionally, the zeros, where the board is empty could be anything else depending on the game instance. Apart from that, it should be mentioned that no winning boards can occur during step 1 through step 4, because players do not have three tokens on the board. In step 5 player one can have a chance of winning and in step 6 player two can has the chance to win.



011001000
100000101
000110010
010010001
100001010
001100100

Figure 6.4: Corresponding encodings for winning

Moving on to the second addition on the program, the verification. The number of all possible boards that was mentioned in previous subchapters, is the key for the verification. The numbers are presented below for easy access:

Step 1: 9

Step 2: 9*8 = 72

Step 3: 9*8*7 = 504

Step 4: 9*8*7*6 = 3024

Step 5: 9*8*7*6*5 = 15120

Step 6: 9*8*7*6*5*4 = 60480

The verification is rather easy to understand. The sum of the weights that correspond to the sequences in a set of a certain step must be equal to the total possibilities (listed above) for the equivalent step.

To make it straightforward it must be broken down. Step one is the easiest of all the steps, since the sequences are the “parent” to all others, it is the root. Thus, the sum of the weights of all the sequences in set s1 must be equal to nine. The other steps are a little bit more complicated. It is a fact that every sequence is derived from the sequences of the previous set, in other words they are the “children”. Therefore, to calculate the weight sum for this step and make the verification, each sequence must be multiplied with the weight of its parent. Finally, after all multiplications are made the weights are summed up. That number must be equal to the total number of possibilities for the equivalent step (above list).

As a final step the algorithm was connected with the two extra functionalities leading to a well-structured and validated program. The results and output of the implementation are discussed and analyzed in the next subchapter.

6.4 Output Analysis

After implementing the program, the result for the unique board placements for each step, in phase one of the game are as follows:

```
Total winning boards in unique PLAYER1: 33
Total winning boards in unique PLAYER2: 15
Total winning boards in unique : 48

Step1 weight verification:
Total: 9
Unique: 3

Step2 weight verification:
Total: 72
Unique: 12

Step3 weight verification:
Total: 504
Unique: 38

Step4 weight verification:
Total: 3024
Unique: 108

Step5 weight verification:
        Winning:
Total: 1080
Unique: 15

        Non Winning:
Total: 14040
Unique: 159

        Step5 total:
Total: 15120
Unique: 174

Step6 weight verification:
        Winning:
Total: 8208
Unique: 33

        Non Winning:
Total: 52272
Unique: 195

        Step6 total:
Total: 60480
Unique: 228
```

Figure 6.5: Program Output

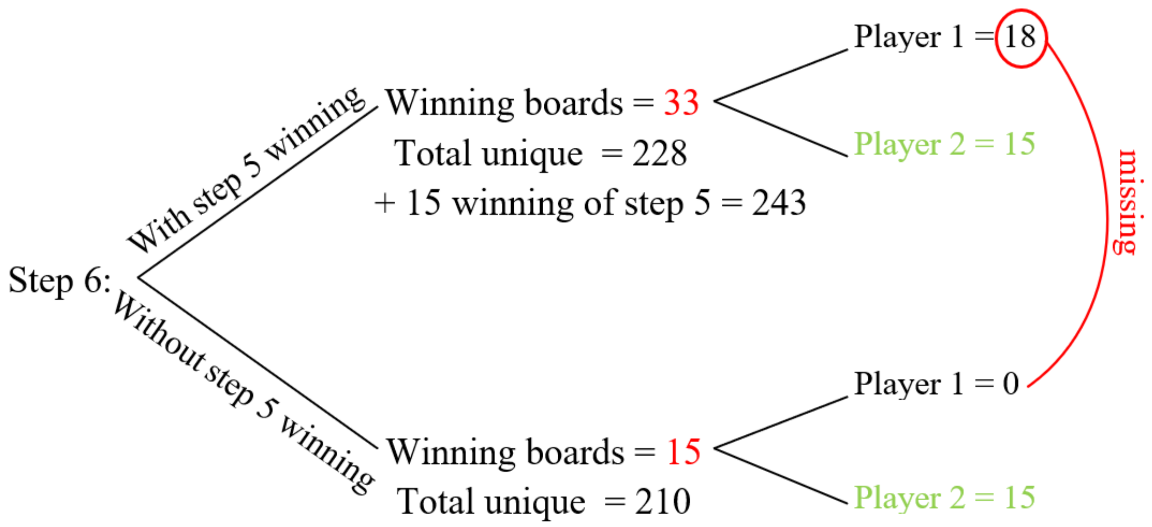
For better readibilty the following table presents the results:

Step 1: 9	3 unique boards	
Step 2: 9*8 = 72	12 unique boards	
Step 3: 9*8*7 = 504	38 unique boards	
Step 4: 9*8*7*6 = 3024	108 unique boards	
Step 5: 9*8*7*6*5 = 15120	174 unique boards • 15 winning • 159 non winning	
Step 6: 9*8*7*6*5*4 = 60480	228 unique boards • 33 winning • 195 non winning	210 unique boards + 15 winning of step 5 • 15 winning • 195 non winning
	When the input of step six is the whole set of step’s five unique boards	When the input of step six is only the non winning set of step’s five unique boards

There is one critical point to understand in order to comprehend the output of the program. There is a distinct difference in running the program with the complete set of unique boards of step five versus with the non winning set of unique boards of step five as input to step six. After some thought and conteplation whether the results make sense or not, the conclusion was that there is a completly logical explanation for it. It is visible from the table that step six has 9*8*7*6*5*4 = 60480 possible board placements. It is obvious that in that multiplication there is no condition that leaves out the winning boards, meaning that the third token of the second player is placed in all boards without exception (*4 in the equation). That serves as the main headline but further explanation is necessary.

Lets brake it down:

Step 5 : winning = 15



It is obvious that when running the program with step six's input step five's non winning unique boards the total weights for the validation are not 60480. That is not because the program does not calculate correctly the weights and unique boards. That happens because even though in the possibilites 9*8*7*6*5*4 = 60480 no attention is given to the winning boards in the case that only the non winning boards are given as input to step six, some boards which are the winning boards of step one are not counted. That is responsible for for the missing weights and it is also shown above. In the winning boards 18 boards that make player one as the winner are not calculated. In other words, when giving the winning boards to step six in the last steps *4 are counted the boards that player one won but player two put a token nevertheless. The program when given the non winning boards of step five, correctly makes output of 15 winnings for player two and zero for player one because practically player two never plays if player one has already won.

Chapter 7

Phase One Winning Strategy

7.1 Similar Games Winning Strategy

During the process of analyzing the game and especially the first phase, it is inevitable for someone to start wondering whether there is a winning strategy. Since the scope of this thesis is around phase one, the thought of the possibility of having a winning strategy was if the player had a way to win from phase one and terminate the game. The first thing that comes to mind is to determine whether there are similar games with comparable rules which can provide a winning strategy or an idea for one.

The most common game that is comparable to the first phase of Αντρίν is Tic-Tac-Toe. Tic-Tac-Toe is a well-known, fast paced game in which there are two players. There are nine available spots on the board for players to put tokens. The goal for a player in order to win is to put three of their tokens in the same line, vertically, horizontally, or diagonally.

From various searches on the internet the winning strategy became clear. There are two possibilities that look like the following:

- When playing first, the winning strategy is called “double threat”. A double threat is when a player has two of their tokens on the board forming two potential lines, increasing the possibility to win in two different directions on their next turn. Since the opponent will be able to block only one of the potential winning lines the first player has a guaranteed win.

- When playing second, there is no winning strategy, the best thing to do is try to prevent the opponent from forming a double threat. The best-case scenario is getting a draw.

Tic Tac Toe	Αντρίν
9 possible spots on board	9 possible spots on board
2 players	2 players
3 tokens in line to win (vertical, horizontal, diagonal)	3 tokens in line to win (vertical, horizontal)
Player 1 -> five turns Player 2 -> four turns	Player 1 -> three turns Player 2 -> three turns

Figure 7.1: Tic Tac Toe and Αντρίν

Figure 7.1 above, the first three rows indicate why this game was chosen for winning strategy ideas but the last line might be the drawback on why this could not work. This is discussed in the next subchapter.

7.2 Non – Applicable Winning Strategy for Αντρίν Game

After thorough examination of the winning strategy of the previously mentioned similar game, Tic-Tac-Toe, and getting in the mindset to find possible ways a player can win and terminate the game from phase one, a significant conclusion was reached. This conclusion is that there is no possible winning strategy to be applied in phase one.

The huge difference between the two games is mentioned in the last row of Figure 7.1. While Tic-Tac-Toe has five turns for player one and four turns for player two, Αντρίν has three turns for each player. The conclusion for the Tic-Tac-Toe winning strategy is the huge role the number of turns each player has when trying to win the game. The fact that the players have multiple turns gives them the chance to set up the double threat. Even though the tokens needed to win are three, multiple turns are needed to set up the double threat. That number must be at least one more than the required amount to win :

$$\text{tokens/turns} > \text{winning tokens}$$

In Αντρίν, both players only have three turns. The above condition does not apply and thus there is no way a player can set a double threat. The players do not have time to set one up, so the primary focus shifts to play defensively. In other words, aiming to prevent the opponent from winning. It is important to note that this could give huge purpose to phase two of the game.

The defensive play could be defined as a map for not losing and could only be applied from step four onwards:

- If the opponent has two tokens in the same line (row or column) and it is your turn, add a token to the empty spot in that line (row or column) to prevent the opponent from winning.

An important observation after countless play sessions of the game, it is concluded that the only way to win from phase one and terminate the game is if the player does not follow the above defensive play and misplaces their token. In simple terms, the player lets the empty space in the line in which the opponent has two tokens empty and chooses to put their token on another spot.

To sum up, there was an observation that could be considered a privilege when playing Αντρίν for phase two. Considering both the nature of the game as well as the winning positions, it is safe to say that any player who owns the center spot (b2) has a benefit over the other player. This is because the center gives a head start for phase two. To put it simply, any other token on any other spot can be moved in two different directions, whereas a token in the center spot could possibly move in four different directions. This is just an observation and could prove to be nothing at the end.

Chapter 8

Game Implementation

8.1 Console Game

After finishing the thorough analysis of the game, and evaluating the gameplay of phase one, the final and necessary step to close nicely the thesis involved the implementation of the console game as well as a more visually pleasing game. This subchapter outlines the console version of the game, which serves as a practical demonstration of the game. Furthermore, having a console game could be the righthand of future work. In simple words, it could help debug, or even analyze to a deeper level the second phase, revealing new concepts and techniques for Αντρίν. Lastly, it is possibly more useful for the development and analysis side of the game rather than being considered a user-friendly implementation.

The console game was developed using Java programming language in the Eclipse IDE workspace. The implementation was carried out using OpenJDK version 11.0.11 and Eclipse IDE 2021-06 (4.20.0) version. Since the program does not rely on any specialized implementation features, it should be compatible with any Java versions and can be executed in alternative workspaces beyond Eclipse.

The console game is an exact replica of the actual game. There are two phases, the placement phase, and the movement phase. It is a two-player implementation allowing users to play by inserting in the console their choices according to the corresponding step or phase. Every time a player makes a choice the board is updated, and the new board is printed on the console, so that players can keep up with their progress.

It is important to note the user interaction part. The game prompts players to input their moves via the console. Additionally, there are validations for every user input ensuring the correctness of the process.

For the purpose of this thesis in Figure 8.1 below the screenshots are shown of a possible game play where the player incorrectly gives input tokens for phase one. Furthermore, in Figure 8.2 there are screenshots where one of the players wins the game from the first phase. Additionally, In Figure 8.3 there is one more example of wrong inputs given by a player in phase two. Lastly, in Figure 8.4 there is a different game play where a player wins in phase two of the game.

```
 Current board
  0---0---1
  |   |   |
  0---2---0
  |   |   |
  0---0---0
Player 1 - Where to put your token?
c1
Position already taken - Insert choice again:
b2
Position already taken - Insert choice again:
aa
Wrong input - Insert choice again:
34
Wrong input - Insert choice again:
```

Figure 8.1: Console Game Error Messages – Phase one

```
 Current board
  0---0---0
  |   |   |
  0---0---0
  |   |   |
  0---0---0
PHASE 1
Player 1 - Where to put your token?
a1
 Current board
  1---0---0
  |   |   |
  0---0---0
  |   |   |
  0---0---0
Player 2 - Where to put your token?
b2
 Current board
  1---0---0
  |   |   |
  0---2---0
  |   |   |
  0---0---0
Player 1 - Where to put your token?
c3
 Current board
  1---0---0
  |   |   |
  0---2---0
  |   |   |
  0---0---1
Player 2 - Where to put your token?
a2
 Current board
  1---0---0
  |   |   |
  2---2---0
  |   |   |
  0---0---1
Player 1 - Where to put your token?
c1
 Current board
  1---0---1
  |   |   |
  2---2---0
  |   |   |
  0---0---1
Player 2 - Where to put your token?
c2
 Current board
  1---0---1
  |   |   |
  2---2---2
  |   |   |
  0---0---1
Game Over - Player 2 wins
```

Figure 8.2: Console Game Win – Phase one

```
  1---0---2
  |   |   |
  0---2---0
  |   |   |
  1---2---1
PHASE 2
Player 1 - Insert position of token to move?
a1
Player 1 - Insert where to move?
a1
Not valid move - not your token or not empty spot or not adjacent
Player 1 - Insert position of token to move?
a1
Player 1 - Insert where to move?
c2
Not valid move - not your token or not empty spot or not adjacent
Player 1 - Insert position of token to move?
c3
Player 1 - Insert where to move?
b3
Not valid move - not your token or not empty spot or not adjacent
Player 1 - Insert position of token to move?
b2
Player 1 - Insert where to move?
b1
Not valid move - not your token or not empty spot or not adjacent
Player 1 - Insert position of token to move?
aa
Wrong input - Insert choice again:
a5
Wrong input - Insert choice again:
```

Figure 8.3: Console Game Error Messages – Phase two

```
PHASE 2
Player 1 - Insert position of token to move?
a2
Player 1 - Insert where to move?
a3
Current board
  1---1---0
  |   |   |
  0---2---2
  |   |   |
  1---2---0
Player 2 - Insert position of token to move?
c2
Player 2 - Insert where to move?
c1
Current board
  1---1---2
  |   |   |
  0---2---0
  |   |   |
  1---2---0
Player 1 - Insert position of token to move?
a1
Player 1 - Insert where to move?
a2
Current board
  0---1---2
  |   |   |
  1---2---0
  |   |   |
  1---2---0
Player 2 - Insert position of token to move?
b2
Player 2 - Insert where to move?
c2
Current board
  0---1---2
  |   |   |
  1---0---2
  |   |   |
  1---2---0
Player 1 - Insert position of token to move?
b1
Player 1 - Insert where to move?
a1
Current board
  1---0---2
  |   |   |
  1---0---2
  |   |   |
  1---2---0
Game Over - Player 1 wins
```

Figure 8.4: Console Game Win – Phase two

To conclude, the console game provides a functional and interactive way to demonstrate the mechanics and rules of the game analyzed in this thesis. This digital adaptation not only serves as a foundation tool for future development and improvements but also validates the feasibility of transforming the physical game into a digital format.

8.2 Graphical Game

Since Αντρίν goes back years it is obvious that people used physical objects to play it on, paper or the ground. Whereas, with the digital adaptation of the game, it makes it more accessible and easier to play. The interface is extremely simple, as well as user-friendly with clear to the eye components and functionalities. It is important to note that after the console adaptation of the game, a more graphical game was necessary for a more pleasurable experience. This more graphical interface, rather than the console one, might attract more attention and preserve its existence.

This version of the game was used using the same Java and Eclipse versions with one addition, the Java AWT. Java AWT is provided by Java for creating graphical user interfaces and it includes components like windows, buttons, text fields and event handling classes. Furthermore, Java AWT is included in the OpenJDK.

To implement this interface, an expansion-like Java file was made from the console game. The logic was exactly the same, with the user input checks, the two phases and the prompts to help the user insert the correct input. The new addition was a button, on the top left of the window which shows the game rules when pressed. To be able to fully understand this implementation Figures 8.5 through 8.8 illustrate the various stages of the game: initial window, the help button information, a phase one snapshot and a phase two snapshot. The errors presented to the users are the same as the ones in the console game.

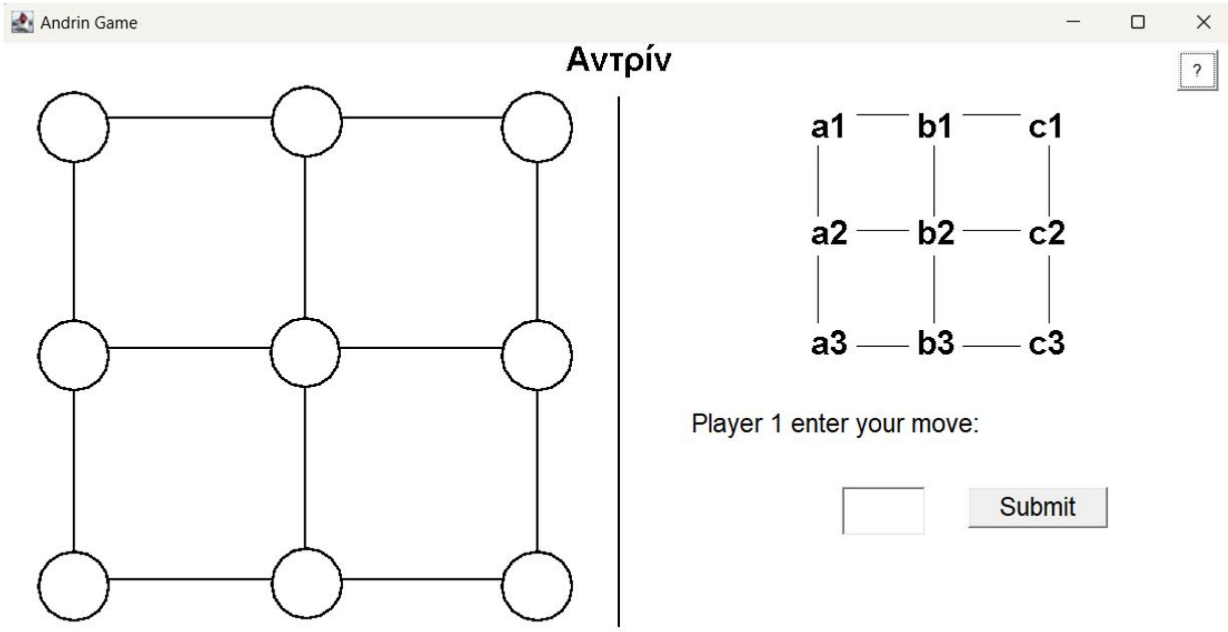


Figure 8.5: Graphical game – Initial Window

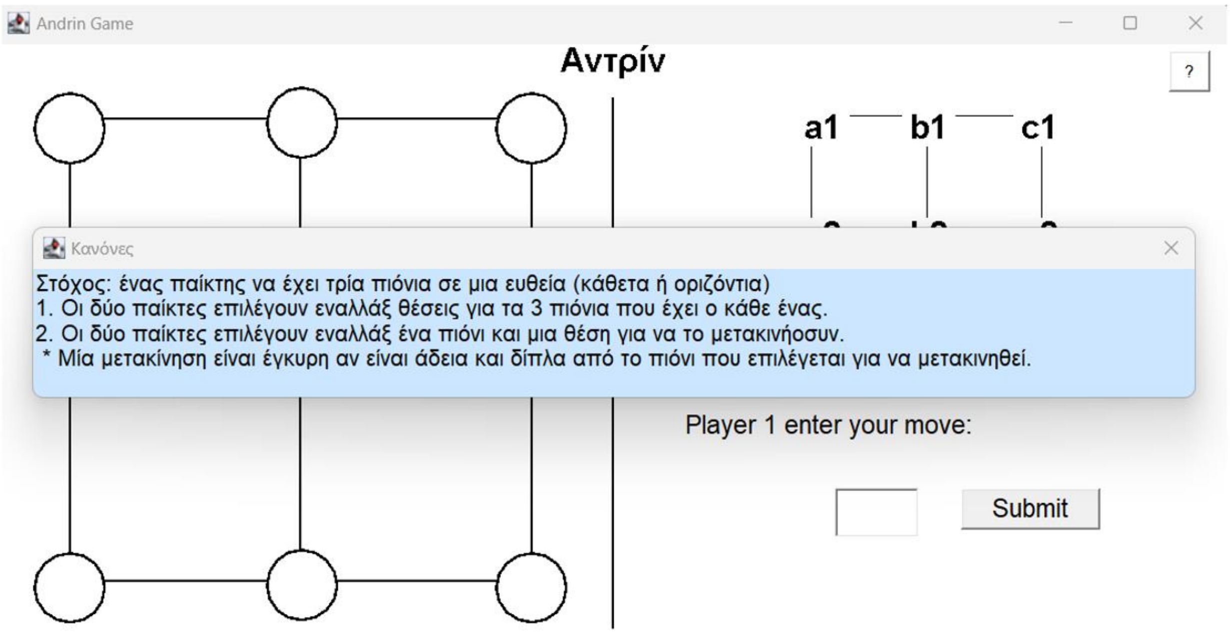


Figure 8.6: Graphical game – Help Button

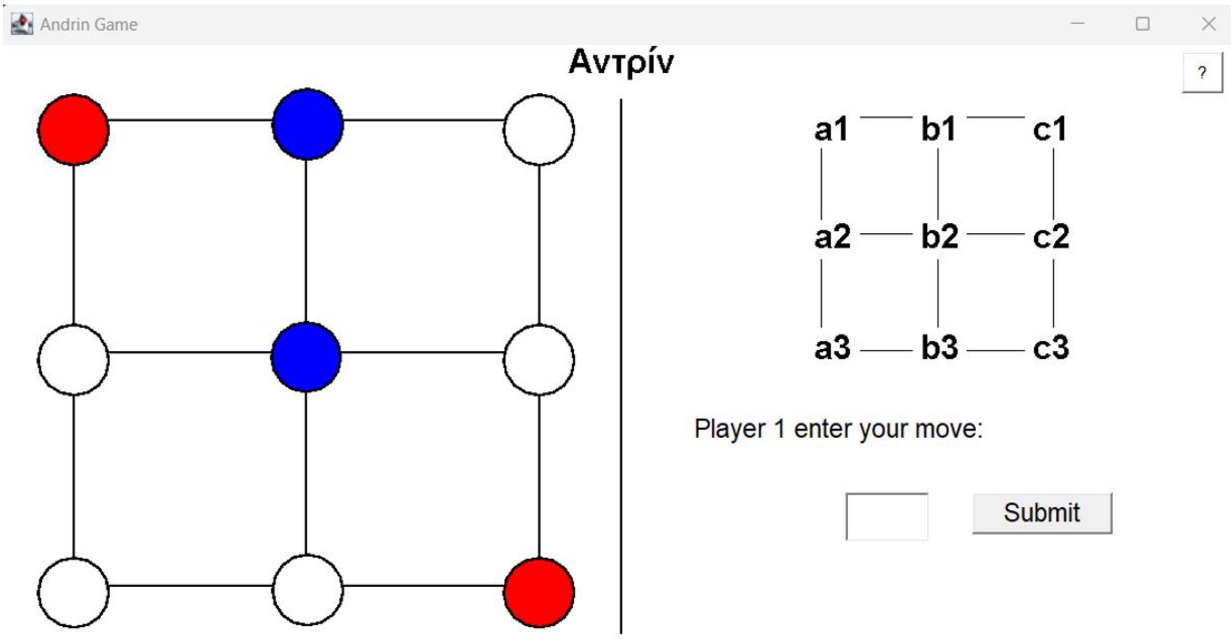


Figure 8.7: Graphical game – Phase One Snapshot

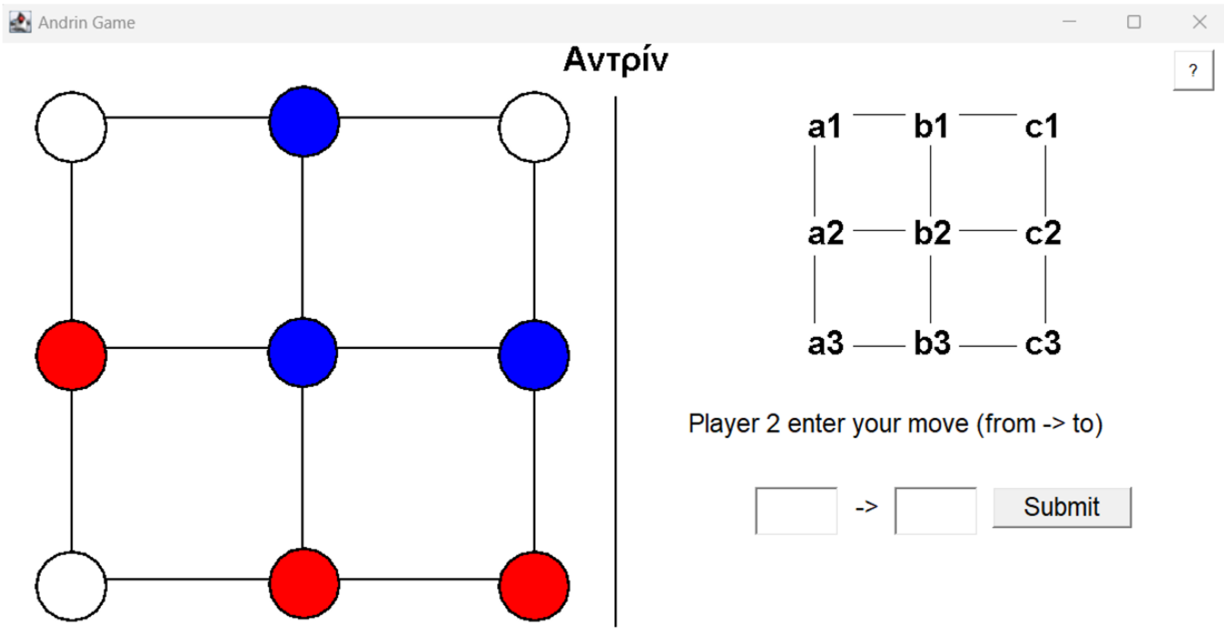


Figure 8.8: Graphical game – Phase Two Snapshot

To finalize, with this graphical interface the circle of the thesis closes smoothly. Apart from that, it maintains the essence of the original game while the console version ensures to be a useful tool for greater understanding of the game's insides.

Chapter 9

Conclusion

9.1 General Conclusion

Reaching the end of this thesis, it is evident that having Αντρίν as the core, several different kind of work has been done. There were various goals some of which were, giving a deeper understanding of phase one of the game, finding the number of unique boards that could be produced during the six steps of phase one and assessing whether there is a winning strategy that could take place in that phase. Apart from this, a console game and a more graphic interface was something necessary to close off the thesis smoothly and creatively.

It was inevitable that, to achieve the above goals some groundwork and foundational elements needed to be established. Those included, the board encoding as well as the board representation for the algorithm implementation, a set of operators as well as research amongst other similar games that could give potential insights into how a game analysis works. With all these in mind, an algorithm was formed giving insights to understanding the complexity of the game, winning strategies were assessed, and a playable game came to secure the thesis.

Given the whole process, which entailed extensive research, algorithm development and analysis, many existing skills were cultivated, while countless new ones were brought to the surface. This thesis serves as an important contribution to the limited documentation available for this traditional game. To roundup , this work not only enriches the understanding of Αντρίν but also gives a foundation for possible future work.

Lastly, it is worth mentioning that all code implementation files are uploaded to GitHub. There are three files, one file for calculating the number of unique placements in phase one (Total_Unique_Validations_Final.java), another for the console game (Console_Game.java) and lastly a third file for the graphic implementation of the game (Interface_Checks_Final.java). The files can be accessed via the following link: https://github.com/StylianaV/BachelorThesis_Andrin_2024.git . While the files are open for use by anyone interested, it would be greatly appreciated if the author's name, Styliani Vaki, as well as the purpose of the implementation, Bachelor Thesis of 2024, was mentioned in any alternations or use of the code.

9.2 Future Work

There are several prospects for future work that could be assembled on the foundation of this thesis built. The first key area is game development. There could be enhancements that could be made in order to improve the user experience as well as the functionality. Some examples features are counting of moves, board tracing in which the boards from the initial state to the winning point are shown along with, parallel view or switching view of console - graphic interface for analyzing. Furthermore, a feature allowing user to insert a board on a certain state and continue playing from there may end up being super useful for practice.

Moreover, the implementation could be enriched by adding a feature for opponent choice increasing the challenge of winning the game. In more detail, the following modes could be attached for more interesting game play, player vs player that already exists, player vs computer, player vs computer with machine learning and AI making the winning of the game a little bit more difficult. All these modes could help users improve their skills given the more puzzling game play. With machine learning a reinforcement learning approach could be used. At that point the analysis and algorithm output giving the number of unique boards in phase one of the game could be of significant use. In other words that set of unique boards could be used as a training set to serve as an extra training set for machine learning. As for the AI, the agent would be able to explore various strategies and receive feedback. Over time it would be able to recognize patterns and gameplays in order to develop tactics. An important note, is that the phase one analysis and the algorithm's output of the unique boards in phase one could be used as a starting point for machine learning or AI implementation.

As of the analysis part, there could definitely be a deeper analysis of phase two of Αντρίν. Possible winning strategy during phase two or even winning strategy combined with phase one could give deeper understanding of the game and unveil strategies. On that note, there could be research on whether there is a specific set of gameplays that could lead to a guaranteed tie. Finally, since the operators and algorithm used are based on evidence, proof could be incorporated instead of the heuristic approach used in this thesis.

Last but not least, after a full understanding of what the Αντρίν game has to offer in regard to strategy and analysis there could be more to explore. The possibility of expanding the board to a bigger one could lead to exciting findings which, of course, would increase the complexity too.

Resources

[1] H. Allen, “Never Lose at Tic-Tac-Toe: Winning Strategy and Tactics for Perfect Play”, https://hobbylark.com/board-games/Tic-Tac-Toe-Impossible-Never-Lose-Perfect-XO-Strategy-Tactics , 2022.

[2] D. Amlen, “The Royal Game of Ur: How to Play the Oldest Board Game on Record”, https://www.nytimes.com/2022/11/17/crosswords/ur-first-board-game.html , 2022.

[3] P. Attia, “The Full History of Board Games”, https://medium.com/@peterattia/the-full-history-of-board-games-5e622811ce89 , 2016.

[4] AQA, “Teaching Guide: Run-Length Encoding”, https://filestore.aqa.org.uk/resources/computing/AQA-8525-TG-RLE.PDF , 2020.

[6] Α. Αρχοντούς, “ Κυπριακά Παραδοσιακά Παιχνίδια 2”, https://blogs.sch.gr/kypros2gymchrys/kypriaka-paradosiaka-paichnidia-2/

[7] D. Berlinski, “Math Games Compendium”, https://ec.europa.eu/programmes/erasmus-plus/project-result-content/9fa79d5d-ecee-4b1b-a154-9647e7097dc9/Math-GAMES%20IO1%20EN.pdf , 2013 p 10-15.

[8] Board and Pieces, “Achi”, https://sites.google.com/site/boardandpieces/list-of-games/achi?authuser=0

[9] R. Gasser, “Nine Men’s Morris is a DRAW”, https://ics.uci.edu/~eppstein/cgt/morris.html , 1993.

[10] R. Gasser, “Applying Retrograde Analysis to Nine Men’s Morris”, https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=8b658eb4b3f22b1e9e2a70930eaf546ef5fb3f9e

[11] Gather Together Games, “How To Play Nine Mens’s Morris”, https://gathertogethergames.com/nine-mens-morris

[12] J. P. Hinebaugh, “Board Game Education”, https://books.google.com.cy/books?hl=en&lr=&id=gUK_AAAAQBAJ&oi=fnd&pg=PR3&dq=history+of+board+games&ots=5jtuKMddmb&sig=boKPkiQ4qtrJGOifJHhApZ1dTFo&redir_esc=y#v=onepage&q=history%20of%20board%20games&f=false , 2009.

[13] L. Ho, “Retrograde Analysis: Work Backwards to Solve Problems Effectively”, https://www.lifehack.org/938420/retrograde-analysis, 2023.

[14] Joyful Games, “The Future of Board Gaming: Predictions and Emerging Trends”, https://rb.gy/d0ka2n , 2024.

[15] C . Luvz, “Winning Tic-Tac-Toe Strategies”, https://www.instructables.com/Winning-tic-tac-toe-strategies/

[16] MIT OpenCourseWare, “ Introduction to Game Studies”, https://ocw.mit.edu/courses/cms-300-introduction-to-videogame-studies-fall-2011/071e671dbde4e7e4578448c75815abf8_MITCMS_300F11_GameAnaGuide.pdf , 2011.

[17] I. Ristov, “The Complete History of Board Games”, https://boardgamesland.com/the-complete-history-of-board-games/ , 2020.

[18] P. Roberts, “Retrograde analysis, a tool, and a mindset to deal with high uncertainty”, ttps://www.icopilots.com/retrograde-analysis-an-innovation-tool-to-deal-with-high-uncertainty/, 2020.

[19] P. Schnoebelen, J. Heer, O. Janko, “The Retrograde Analysis Corner”, https://www.janko.at/Retros/index.htm , 2001

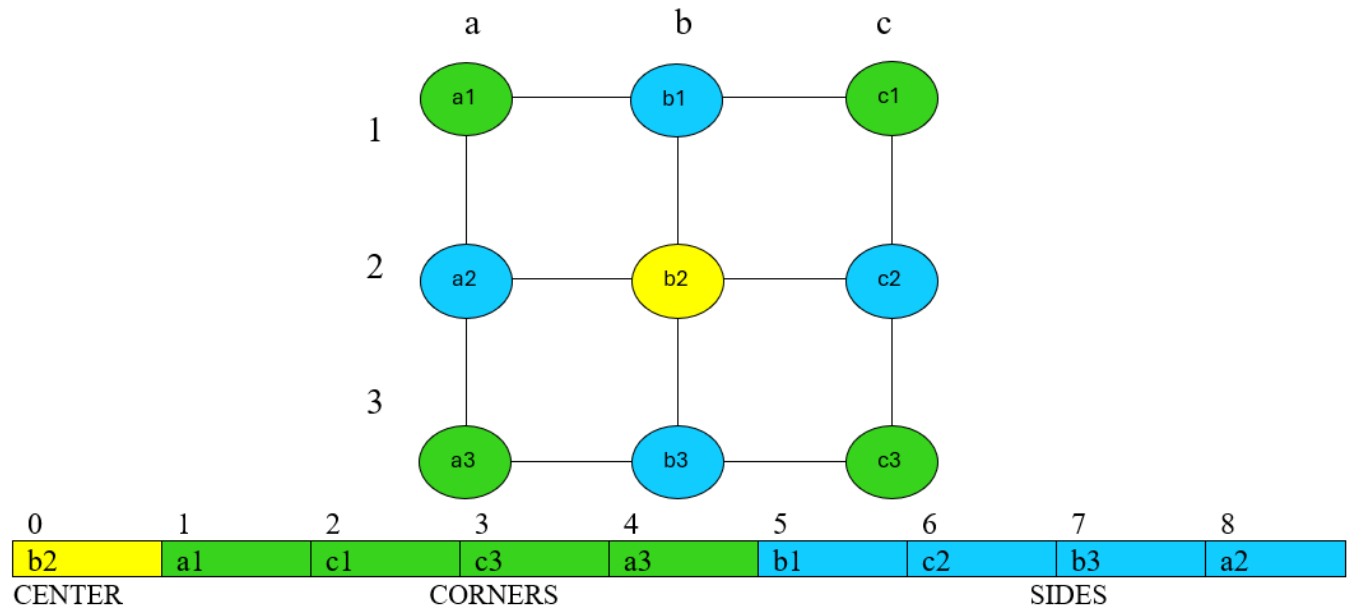
[20] D. Shaw, “Play Achi”, https://blog.doublehelix.csiro.au/play-achi/ , 2016.

[21] “The Future of Board Games and AI: An Emerging Player in Design and Gameplay”, https://boardgamenerd.medium.com/the-future-of-board-games-and-ai-an-emerging-player-in-design-and-gameplay-b17f6f0e4fe6 , 2023.

[22] C. Whipkey, T. Hwa, “Retractor”, https://xenon.stanford.edu/~hwatheod/Retractor/

Appendix A

In this appendix, the user guide regarding the board encoding, the board representation in the algorithm alongside the operators are illustrated for easy access.



Operators:

- Rotation:
  1. R1()
  2. R2()
  3. R3()

  There are also R0() meaning zero rotations and R4() that the board gets in its initial position, so for the purpose of this analysis they are not needed since we need to discover new positions.

- Downward Mirror

DM()