Understanding fully expanded games

Supervisor: Cameron Browne & Walter Crist & Matthew Stephenson
Working Group: Bas von den Borg (i6229916)
Katharina Erbach (i6243767)
Georgios Koutidis (i6233809)
Markus Niebisch (i6123084)
Tom Scholer (i6223586)
Zhangyi Wu (i6214955)

Group 10
Research Project AI and DSDM

Department of Data Science and Knowledge Engineering
Maastricht University
Netherlands

Abstract

Predicting the reception of a game by human players is useful for various ends, from historic game retrieval to automated game testing. In this work a series of strongly solved historic games are examined, that is to say that the effects of moves and the perfect strategies are known. An extension to the Ludii framework is developed that allows full game tree expansion for games defined in Ludii, which includes the games analysed. To find possible features that can approximate the reception of a game, features extracted and scored are compared with survey results that test the reception of the same games by human participants. The games selected for the analysis had to score a game complexity rating below a threshold, as higher complexity games would result in trees that are too big to compute with, using the hardware available. In addition to the search of predictive features, tools for representation of the game trees and a selection of features is developed as well.

Contents

1 Introduction

Ludology is a field that studies games and analyses them from various angles. The approaches to analysing games range from analysing the mechanics embedded in the games' design, to the culture surrounding the games themselves. One of such analytical angles occupies itself with measuring what makes games successful, what makes them fun and what engages players to the extent that they may want to return to play the game again. In other words, what combinations of a game's elements, its 'DNA', make for an enjoyable game. These questions have commonly been approached in abstract ways and are what game design theory originates from. However, such abstract philosophies on game design are difficult to compute, or measure automatically. Mathematical approaches have been few, but are much better suited at answering the questions about games consistently by computation and would better allow automated game evaluation. Automatically measuring a game's worth would be valuable for various ends, such as game testing and archaeological games. Game testing is mostly done by humans who have to play the games multiple times before making a verdict on enjoyability. This is known to be a slow process and would largely benefit from automated estimation of how the game will be received by players.
As of yet, there appear to be no successful metrics to approximate a game's reception by human players. Several papers have discussed the analysis of games by generating their respective game trees using methods from graph theory (Ascher, 1987) and intelligent search (Van Den Herik, Uiterwijk, & Van Rijswijck, 2002). The trees represent all the possible states the game can have and it may be possible that the value of the rule sets can be measured from these trees. In this paper, the aim is to search for features that can predict a game's reception by analysing the game's game tree. It is possible that certain aspects of a game can only be observed when the entire tree is known. Thus is important to have as much information about a game's workings available as possible. Therefore, this paper will focus on fully expanded game trees, to exploit the possibly unique information full expansion may hold. However, game trees can become so large that it becomes difficult to work with them. Small is relative and therefore we must clarify that in this paper, a game is considered small if its game state tree does not exceed 3 GB of ram, as limited by the available hardware. However, this is not a clear boundary for the trees itself, therefore we make use of metric prefixes that define a game's size based on the amount of edges it has, for example a kilo game would be a game that has one thousand edges. It is not possible to define a game tree as small based on board size or other attributes such as the amount of play pieces. This is due to rules being of high influence on the way a game's graph may be structured. The rules can increase both the breadth of a game as well as it's depth. There is no particular set of rule elements that turn a game's tree into one that is too big to work with.
Analysing games based on their underlying game trees is not a straightforward task. Game trees are abstract representations of the games and their rules; they likely leave out information, some of which can be very useful to predict game reception. Thus the question becomes: how should one approach the analysis of fully expanded game trees for the purpose of approximating the human reception of the respective game? This question, will be approached from two angles. First, there is the game design theory approach. The game entertainment industry has seen rapid growth over the course of the last few decades. With that growth has come a lot of game design theory that tries to describe and approximate what abstract elements of games can make for a successful game. What makes games fun? What engages players? Previous research 2.2 has already described and turned several game design aspects into measurable elements of games. However, these have not been implemented for trees. In this work, these methods of measurement are altered to fit their application to trees. The second approach this paper will take is a graph theory approach. It may be possible that some concepts of graph theory are applicable to the game trees, and are capable of identifying good or bad aspects of games. Several graph theory based graph structures will be searched for in the game trees, with the hopes that some may correlate strongly with human enjoyment. Determining whether the game design theory aspects and the graph theory structures can successfully predict the reception of a game by humans ultimately requires measurement of human players reception of various games first. To this end, a survey was conducted where players were asked to give ratings to games, to measure how the rating by people may correlate with the proposed metrics on the same games. Finding a metric that correlates very positively or negatively would be what is sought after and could greatly help analysis of games for other purposes. Along the search of telling metrics, an extension will be developed that can automatically generate game tree graphs. Such graphs can be helpful in the analysis of games

as they are able to highlight relations between moves in local parts of the graphs, something that is more difficult to do by global metrics. It will make it easier to see how a game progresses towards it's end states, as well as documenting how tension changes through the course of a game. Furthermore, it is possible that the local sections in visualised graphs highlight a group of states that involve specific elements of the game interacting or exercised more strongly or weakly. Such sections can provide insight on what possible changes may improve a game. This would slightly move away from the mathematical approach the majority of the paper is focused on, as it relies on visual interpretation. Yet, such visualisation can still be a helpful contribution to the analysis of games by use of their game trees.

2 Related concepts

2.1 The Digital Ludeme Project

The Ludii platform will be helpful to answer such questions. It is a computer program that offers a language for expressing game rules as well as providing tools to automatically play the respective game (Eric Piette et al., 2019). With this platform, it is possible to fully expand a game state tree of games given their rules, enabling the collection of state trees for games defined in the framework. Different rules will create different trees that have different properties. Trees can be compared with one another and which may be able to indicate certain differences between rule sets that could show a certain rule set to be more likely to be enjoyable than its alternatives and therefore a better approximation to the original rules.
Although the historic aspect itself is not what this research focuses on. The Ludii platform will be used extensively for this research, as it includes a vast collection of games to work with and compare, as well as providing search methods that can be modified for the purpose of full tree expansion.

2.1.1 Interesting Board Games

From here on, the positive reception by a human player will be referred to as interestingness. The focus lies mostly on games where the fun should lie in the challenge presented by the game, rather than a social activity that may surround the game while it is being played. We further define a board game to possess interestingness when it has several of the following aspects:

- The Game allows for various strategies, meaning that players can separate themselves in terms of skill from other players. A game that allows various strategies to try and attain a win is more interesting than a game that is dominated by a single strategy.
- The game approximates the right balance between being predictable and chaotic to the player. If the game is predictable, the players do not need to be engaged with the game as outcomes are already know. However, when a game is too unpredictable it leaves the players feeling as if they do not have any impact on the outcomes of the game, leading to the same result of not being engaged. (Lantz, Isaksen, Jaffe, Nealen, & Togelius, 2017)
- The depth of a game can be described as the number of leagues can be created, where every member of a league dominates every member in a lower league. It is assumed that more depth games are more interesting (Thompson, 2000).

During this project, various methods of gauging the levels of intestines will be experimented with. The focus will be on obscurity, tension and length of the principal variation and compared with depth. Further explanations of these concepts can be seen in 3.5.1.

2.2 Related Literature

Previous research that will be very helpful for creating measurements for game trees, reasoned from game design theory, includes:

- (Iida, Takahara, Nagashima, Kajihara, & Hashimoto, 2004): Provides useful descriptions of a general game mechanic that will be refereed to as uncertainty.

- (Margulies, 1977): Describes different principles, which make positions in chess beautiful. One of those, violate heuristic, will be described as obscurity of a move in section 3.5.1. That means, that a move is better then expected at first glance.
- (Works of Sam Loyd): A puzzle designer known for his chess puzzles, who has written several books on designing puzzles. This will be relevant as good turn based games can easily be turned into puzzles as every turn is somewhat of a puzzle itself. Which of his works will be most useful too this research is yet to be determined. One classic is his work "Chess Strategy", where he recommends the optimal number of moves for a chess riddle to keep the reader interested is three (Loyd, 1878, p.21).
- (Lantz et al., 2017): A paper that describes an aspect of games called 'depth', it essentially describes the proper balance of a game being too predictable or too disordered. One of the author's thesis tries to give better understanding, how the balance of a game can be quantified (Jaffe, 2013).

2.2.1 Computational Complexity of Games

A formal study on a game's difficulty can be given through Complexity Theory (CT) as described by Eppstein for example. To elaborate, this offers methods for identifying which complexity classes solving the games belong to and showing previously unrecognized relations between many different games. Tic Tac Toe is in the class P, because there is an algorithm that finds the optimal play in a polynomial time in the size of the game. Go, on the other hand, is in EXPTIME-Complete. This is a class in which algorithms cannot do any better than solving in $O(2^{n^k})$ time.

If a game is in P, it becomes uninteresting once you learn "the trick" to perfect play, but high levels of difficulty and obfuscation imply that there is no such trick to learn: the game is inexhaustible. And of course NP-completeness or PSPACE-completeness does not rule out the possibility of computing game values exactly; true, it seems to imply that worst-case-exponential algorithms are required, but there is still plenty of interesting work in designing, analyzing, and implementing such algorithms. For two-player games, one encounters a similar phenomenon at a higher level of complexity. The tree of potential interactions in a game typically gives rise to PSPACE-completeness results. However, some games are harder, EXPTIME-complete, which may indicate that it can sometimes be necessary for a well-played game to go on for a excessively long sequence of moves (Eppstein, 2020)(Arora & Barak, 2007, p.22,p.68,p.502)(Fraenkel & Lichtenstein, 1981).

Van Den Herik in comparison separates game complexity along two axis. Game state complexity and game tree complexity and classifies into 4 groups depending, if the properties are high or low (Van Den Herik et al., 2002).

1. Games in class 1 have low game state and low game tree complexity.
2. Games in class 2 have low game state and high game tree complexity.
3. Games in class 3 have high game state and low game tree complexity.
4. Games in class 4 have high game state and high game tree complexity.

As stated by 'Games solved: Now and in the future' (Van Den Herik et al., 2002), class 2 games can only be solved, if at all, by knowledge based methods, whereas class 3 games by brute force methods. Class 4 games are not solvable. During this project only games in game complexity class 1 will be looked at.

To more effectively select games for full tree expansion that fit the scope of this project, it would be useful to estimate the complexity of a game beforehand. Yong provides methods that can approximate such effectively(Yong & Yong, 2019).

2.2.2 Solved Games

There is extensive literature about solving games. Summaries of solved games and procedures how to solve them are given by Heule and by van der Herik (Heule & Rothkrantz, 2007) (Van Den Herik et al., 2002).

Allis uses three different terms, to distinguished between different types of solved games (Allis et al., 1994).

1. **Ultra-weakly solved** These are games, where one can proof that there must be a strategy for one player to win. The famous example would be Hex, where due board shape no draw is possible and it can be reasoned that no move is bad, except for the lost initiative, so if there is a winning strategy for the second player, the first player can just make a random move and become the virtual second player.
2. **Weakly Solved**. Weekly solved games are games, where one can proof a winning strategy for either player at the start or if that one does not exist, a forced draw for the starting player. One example would be checkers (Schaeffer et al., 2007).
3. **Strongly Solved**. Strongly solved games are games, where for every possible state the optimal move is known. The games, which are being focused upon during this research, will all be strongly solved.

Notable as well is the solving of 11 by 11 Domineering (Uiterwijk, 2016), the solving of Line of action 6 by 6 (Winands, 2008) and the solving of Fanorana (Schadd, Winands, Uiterwijk, Herik, & Bergsma, 2008). In the realm of games with imperfect information, the biggest breakthrough was the solving of limit heads up poker (Bowling, Burch, Johanson, & Tammelin, 2017).

3 Methods

3.1 Data

To retrieve the fully expanded game tree, every possible move of a game needs to be played out within the Ludii system. In doing so, all the states will be represented by a tree node be it a win, loss or draw, all according to perfect play. Additionally, every possible move will be evaluated with different player strengths using the internal Monte Carlo Tree Search, an heuristic method to evaluate the value of a move using sampling. (Browne et al., 2012)

3.2 Selecting Games

There is one important property of the games we select for this work. As mentioned before, depending on factors such as board sizes and rules, game state trees can get immensely big to the point that they do not fit into the memory of the used hardware or take too long to work with within the time budget. Therefore, the games are selected as candidates for analysis by using an estimate to game complexity. Through some preliminary tests, we found that the build in game complexity estimator in Ludii can roughly predict the amount of edges a game will have. Furthermore, we found that megagames, games with at least a million edges, start to reach the physical upper limit of the 3 GB of RAM. At times, it could still occur that the trees predicted to belong to games that had a complexity rating within bounds would overflow the memory. The complexity estimate is not a prefect measure to select the games, but it was an appropriate filter that could be computed quickly without the need of the tree itself.

3.3 State Symmetries

In many games it can happen that states of the game are reachable from different sequences of moves, meaning that a game state tree can have nodes that are symmetric to nodes elsewhere in the tree. Tackling these symmetries would not only make computation involving the trees more efficient, but it would also allow for a better evaluation of the game as unique states only get valued once. After all, the challenge presented by a unique state does not change based on the history of moves prior to the state. State symmetry does not simply come down to checking one state with another as there may be multiple elements to a game that may influence whether a state is truly symmetric to another state. To avoid searching the entire tree for a symmetric node every time a new node is found, hashing is used to quickly store and retrieve observed unique states. This method has shown big potential, but creates several special cases for different games and currently confidence could been only expressed for games played on the Mu Torere boards.

3.4 Graph Cycles

Without recognising cycles, tree expansion would in many cases end up as an infinite process. It is important to recognise a loop and note where the loop starts. This process is also dependant on the hashing method used to give state an identifier.

3.5 Methods

After generating the game state trees of the selected games, feature extraction is conducted. A variety of features are extracted as listed below. Most values are not only stored as an average, but median, min, max, variance, skewness and kurtosis are generated as well, if applicable.

3.5.1 Extracted Game Design Theory Features

1. **Game Theoretic Value**: A game can either be a win, loss or a draw. This is encoded with -1,0,1 respectively. As the maximum player size is one, it allows for easy conversion, when states between player 1 and player 2 are compared using symmetry. Also the game theoretic value of the root is stored, to check for games like Mu Torere with the first move win rule.

2. **Tension**: In the arena of an interesting game, the players are supposed to experience the ups and downs from decisions made by their opponents and themselves. Tension marks the importance of each move by computing the proportion of the poor moves to the total moves, so that when both players have the perfect understanding of the game mechanic, the tension can be predisposed to stay on a lower level because of the perfect play. According to 'What Makes a Game Good' (Kramer, 2000), low tension lasting for a long period should be avoided in any game. In contrast a preferable tension curve is one where high values are achieved more frequently, a game with such a tension distribution is likely to be considered interesting.
   In this context: Tension can be described by the term $1 - \frac{opt\_moves}{moves}$. Essentially, it is the ratio of optimal moves to total moves from a given state.

3. **Obscurity**: An agent will make a decision based on a heuristic value of the current state of a game. However, a agent might choose an option that will prove itself to be a poor decision subsequently, compared to an alternative. The frequency of events where the state is misleading to the AI can be described as obscurity. The optimal move was overlooked in exchange for a worse move that appeared to be a better move at first glance, but poor when evaluated more thoroughly. To quantify it, the average distance of the predicted game theoretic value (percentage of win in our case) to the factual game theoretic value of a move is calculated. To receive the final value of obscurity the average of average of different agents of skill is taken.
   qo formulate obscurity of a playout, first let $Moves$ be the set of all moves happened in a playout, $Agents$ the set of agents of different skills. The obscurity of a single move $move$ with respect to an agent $agent$ is:
   $Obsc(move, agent) = |valuePrediction(move, agent) - gameTheoreticValue(move)|$.
   The obscurity of a move is the average value:
   $Obsc(move) = \sum_{a \in Agents} Obsc(move, a)/|Agents|$.
   The final obscurity is the average obscurity across all moves of a game:
   $Obsc(game) = \sum_{move \in game} Obsc(move)/|Moves|$.

4. **Longest non looping principal variation**: The length of the path as a result of perfect play. The hypothesis motivating the use of this metric is that games that end their principle variant deeper within the tree make it harder to assess the value of moves as the results are not easily determined. In case the game contains forcible cycles it may be possible for the measured length of the game to be infinite. In such a case the enforcement of a cycle is ruled as a draw, as neither player will ever win when the cycle is perpetuated indefinitely.

5. **Average length of principal variations**: Both previous measures do not necessarily have to be taken from the root, they can be taken with regards to any particular node in the tree, as long it is not a leaf

node. The reasoning behind this measure is that players usually do not play optimally all the time. By measuring the average principle variant length it may be possible to gauge how much opportunity there is left for the game to make a sudden turn as a result from a mistake by one of the players.

6. **Depth**: The depth of a game means that a game can have several levels of understanding. The more levels it has, the deeper it is considered to be. For example the first level would be getting to know the rules of a game and being able to play correctly. The second level would be developing simple strategies.

   According to J. Mark Thompson (Thompson, 2000), the depth of a game can be measured by counting the number of leagues, in which players of different skill level can be categorized in. Therefore, in order to calculate the depth of a game, tournaments were simulated where several AIs of different thinking times competed against each other. After each match-up the Elo-Ratings of the AIs were adjusted according to the outcome of the match. After the play out of a tournament, the resulting Elo-Ratings were analysed and a suited number of leagues was determined.

7. **Ridgeness** As a ridge is a path on top of a mountain, where each side leads to a descend, ridgeness determines how save the steps are one is trodding. It is the smallest distance from any move to an imminent terminal state and may tells in how forgiving a game may be which may potentially relate to the tension of game. It may also be the case that ridgeness is related to obscurity. The lower the ridgeness, the closer end states lie to the average move, the easier the quality of possible moves can be determined.

3.5.2 Extracted Graph Features

In addition to the metrics originating from game design theory, several graph features are measured as well. These features are compared to and correlated with the game ratings from the player survey on enjoyably, novelty and challenge. This is done in the hopes of finding graph structures that indicate good games that are not easily reasoned from the game design perspective.

1. **Branching factor**: The average branching factor can quickly be calculated as the number of non-root nodes (the size of the tree, minus one) divided by the number of non-leaf nodes.

2. **Diameter**: The longest shortest path within two vertices. In this interpretation all edges are considered bidirectional

3. **Wiener Index**: The sum of all shortest path between vertices.

4. **Graph depth**: The shortest distance of a node to the start.

5. **Node Edge ratio**: This is related to the branching factor. It is left in, because it shows not a perfect correlation.

6. **Heuristic branching factor**: Measures the overhead of a iterative deepening search compared to a depth first search. (Zhang, 1999, p.52)

3.5.3 Game theoretic Value Sampler

To find simple indicators, which game could be interesting even in bigger games, the extracted features are correlated with sampled results of different games. For this 3 approaches have been tested, where for each game 30 matches were conducted:

1. 30 Ludii Random Playouts

2. 30 Ludii Matches using a slower MCTS Agent with mean 250 expansions per move.

3. 30 Ludii Matches using a faster MCTS Agent with mean 100 and standard deviation 50 expansions per move.

To encourage exploration of the agents in different part of the trees, the number of expansions before each match is determined by a Gaussian random variable which uses a standard deviation of 50 and respectively 12. The number of turns has been limited to 150, to allow reasonable draws. It can already been said, that the variance of results of a single game type using the slower agents, correlates with fun value determined by the survey.

3.6 Depth Estimation: Tournament Structure and Implementation

As mentioned in section 3.5.1 depth is the number of potential leagues a game possesses. Tournaments were conducted within the Ludii Java framework in a round-robin fashion to determine the leagues. Ten AIs with different thinking times were matched-up against each other. The thinking times differed from one another by a multitude of two, where an AIs lower counterpart had half the thinking time and an AIs upper counterpart had twice the thinking time. Every AI played against every other AI once, which leads to nine rounds with 5 matches per round. The rule best of four was applied for every match-up, which means to win one match the AI needed to beat its opponent three times. Two wins would end up in a draw. Within one match the AIs would alternate as starting player to eliminate any bias towards the starting player. A maximum amount of 2000 moves was determined for each match to prevent two AIs from going on endlessly to avoid a loss. After every match the Elo-Rating of the competing AIs would be adjusted according to the outcome of the match. The Elo-Rating would be lowered if the match was lost and raised if the match was won. Every AI starts with an Elo-Rating of 1000.

A tournament produces data in the form of a .csv file, containing all the play outs of the matches and the outcome of the tournament. The outcome of a tournament is further used to analyze the resulting Elo-Ratings to find out into how many leagues the AIs can be clustered. The number of clusters gives a notion of the depth of a game. The more leagues that can be formed the deeper the game is considered to be.

3.7 Jenks Natural Breaks Clustering

In order to find the right clusters, a python script was created which uses the Jenks natural breaks optimization method to arrange the Elo-Ratings into different classes by reducing the variance within classes and maximizing the variance between classes. The number of leagues is initially set to 2 and incremented by 1 until the goodness of variance fit (GVF) reaches a certain threshold of 0.8. The GVF can be calculated as follows:

$$\frac{SDAM - SDCM}{SDAM} \tag{1}$$

where SDAM is the sum of squared deviations from array mean and SDCM is the sum of squared deviations of class mean

3.8 Custom Generated Classifier

As Jenks natural breaks clustering did not reach the desired resolution as alternative a custom made classifier has been created using increasing step functions.

Hereby, the agents are sorted by their Elo-Rating and the worst and the best Elo-Rating determine the the minimum and the maximum of the step function.

The in-between step sizes are determined by:

$stepSize = x * (maxELO - minELO)/10$. Where x is a number between 1 and 10. This allows to generate all possible increasing step functions between minimum Elo-Rating and maximum Elo-Rating. This includes a step function, which has 10 steps, or only one big step in the middle. These functions are then mapped to the Elo-Rating of the ordered agents. The sum of the absolute error determines the quality of fit. The number of steps being 'occupied' by agents determines the final depth value using square error gives the same result.

3.9 Validation by Survey

General information To find the possible relations between the extracted features and the reception of the respective game. A survey of 34 people is taken, where the participants are tasked to play the selected games and rank them by enjoyably, challenge and novelty. The results are analysed for possible correlations with the aforementioned features, using the concrete features:

- Novelty: How novel do you find and how surprised were you by the game?
- Enjoyably: How much did you enjoy playing the game?
- Challenging: How difficult did you find the game in comparison to other games?

Survey structure The survey consists of three parts: Presentation of the LUDII tool to be used, presentation and execution of the 12 games and finally a game ranking. The former guides the participant through the installation of the required tools and explains LUDII and the (for the survey) necessary functions by means of pictures and brief explanations. The participant is asked to play the twelve games in the next phase, each game being structured as follows:
first the rules are explained in simple steps, after which the user switches to the Ludii application, opens the demanded game and plays against an AI at least twice. Then it is asked to give three assessments. Precisly the features mentioned above in the range of -5 (not applicable) to 5 (applicable). Always three games belong to one game category. The categories and their allocation can be found in the list below.
After three games (or exactly one category) the participant is asked to compare the games directly and to choose one game out of two that he/she prefers. The last part of the survey refers to the ranking of all games from all categories. First, the participant is asked to choose the best game, whereupon a rank of all games by preference is asked. Finally a preference order of the superior categories is to be determined.

Survey Games The survey sample four categories of all categories of Ludii games. This categories are indicators as they contain a variety of small games - three games of each category were selected. Those games are strong indicators as they represent in a wide range their category, while differ on rules and boards and give a deeper insight of small board games. As follows the four categories and the list of games are provided in detail.

- **Blocking category**: Blocking games are useful for our analysis as players can avoid defeat while repeating a specific move so the game have a lot of cycles repetition of the same state.

  Play out: Move a piece to a next adjacent point to force a state where the opponent is unable to move.

  Blocking games: Ho Bag Gonu, Madelinette, Mu Torere

- **Hunting category**: Hunting games are useful for analysis as players with more pieces always win so they are pretty trivial, it was worth to see how trivial games correlate with the interestingness.

  Play out: In the initial state players have unequal amount of pieces to play with. The player with one piece try to eliminate the opponents pieces. As the opponent tries to stop first player from moving.

  Hunting games: Haretavl, Kaooa, Hat Deviyan Keliya

- **Line Cateogry**: Line games is the most known games while having the simplest rule set, it is really important to see how the testers feel about them.

  Play out: Place and move a piece to arrange your pieces in a line thus leads to a win.

  Line games: Three Men's Morris, Tic-Tac-Chess, Tant Fant

- **Capture category**: Capture games are really interesting as you can plan your next moves and create a simple strategy.

  Play out: Eliminate the opponents pieces.

  Capture games: Boseog Gonu, Ja-Jeon-Geo-Gonu, Felli

The Rule sets of the games can be found in the Appendix B.

Survey Participants The Participants were all older than 18, and informed that non of their personal data was collected. Invitations to participate on the survey were distributed by two methods. First, participants could be invited through direct invitation by any member of the research group. Second, invitations have been send out by use of a mailing list, consisting of recipients at the University of Maastricht. As a result, a total of 34 responses were received. That amount of people represents on one hand the academically involved individuals like students or researchers. On the other hand individuals that are interested in games as such. Those two groups build a representative basis, due they partly overlap and all at least have some prior knowledge about games (in form of research and/or actual playing). This leads to a representative result in form of high quality answered questions. Participants are able to compare a game to other survey games, as well as to other known games, what leads to representative answers on the novelty of a game.

3.10 Representation

One of the most important visualisation of this project is using a game graph to describe a small game. The typical game graph contains only the current situation and possible decisions at different moments. In this study game graphs containing nodes and edges of different visual properties, which is capable of revealing more aspects, are expected. The program starts composing the position graph once the extraction of the structure of a game, and the back-propagation of game theoretic value are finished.

In order to encode the knowledge of each node and edge, GML (Graph Modelling Language) file format was used. A GML file consists of a hierarchical key-value lists and is convenient for describing graphs. Software called yEd(yWorksGmbH, 2019–2020) and Python's NetworkX(developers, 2014–2020) are used to read GML files and realise the representations. During the early stages of the project, yEd was very adequate at creating well structured graphs. However as soon as it was decided that the graph nodes should include visual icons describing the game state, yEd was not enough by itself. NetworkX was introduced as an additional graph network library. yEd would require every icon graphic to be imported separately and manually, whereas NetworkX allowed for more flexible and efficient solutions to include the icons. This change did come at the cost of various automatic graph structuring methods that organized the graphs in a more readable manner in yEd. This was a trade off we deemed worth taking, as we believe that the icons help the interpretation of the graphs more than the more organised representation that yEd offers. It is unfortunately not possible to export a NetworkX graph with its icons to yEd to handle the structure afterwards.

3.10.1 Principal Variation

To visualise the general obscurity of the solution, the principal variation (i.e. the edges which lead to the best possible outcome) will be marked red.

3.10.2 3D Representation

To help visualise more convoluted game trees, the 3rd dimension is involved for more complex representation, where the nodes are layered according to their depth. The planar arrangement of the nodes was optimised using a simple simulated annealing hill climber, which takes into account the graph distance of the nodes. To help perceive the perspective this graph is rotating as the parallax effect helps then better to distinguish different layers. It can be noted all ready that showing only nodes can help make things clearer, but introducing edges shows the limitation of that approach.

4 Results

4.1 Expanded Game Metrics

Games in the range up to three million edges could be extracted, then the limitation of memory was reached. For extracting values like diameter the limit was found at around 10000 to 30000 nodes. Tension, game theoretic values and similar simple measurements could all be extracted reasonably fast and the whole data set can be extracted and processed in around 40 minutes of times and the system allows data to be stored and loaded.

4.2 Positions Graphs

The game tree extractor used to retrieve the full game trees is implemented in a depth first search manner. It defaults to starting its search at the root node of the tree, similar to what humans playing the game do. There are two conditions that end the extraction algorithm. Either the entire tree is explored, or the amount of explored states exceeds the set threshold at which games are considered too big to work with. In addition to exploring and retrieving the full trees, the extraction algorithm assigns a game theoretic to the leaf nodes whenever it encounters them. This is later used to propagate the game theoretic values up the tree. These values indicate what the best subsequent move of a state will lead to, assuming perfect play from both players. For every legal game state, the specific layout of game pieces can be retrieved from Ludii. This feature enables us to create a visualisation of every boardstate, which is used in the graph visualisation.

Figure 1 shows the position graph of game Mu Torere.

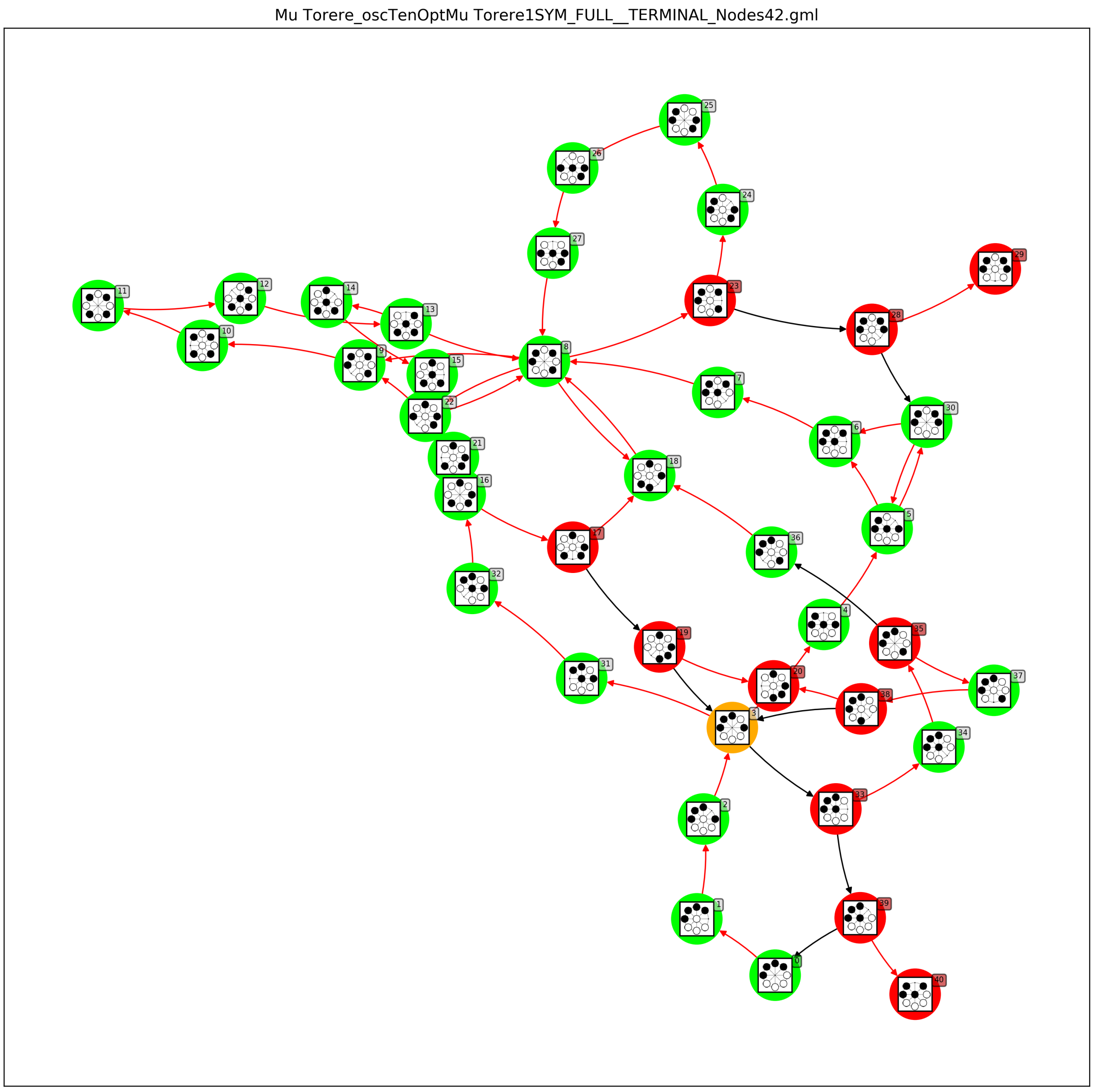
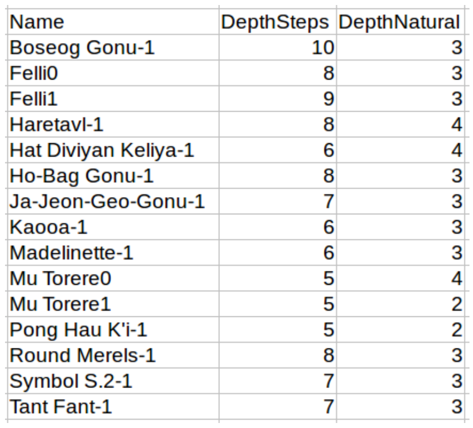


Figure 1: Position graph for Mu Torere

In this section a raw result of the position graph is provided, further insights about it will be discussed in the next section. We also extracted the position graphs for four other games than Mu Torere, all of they can be found in Appendix A. The unfortunate thing is, we do not have the graphic templates for displaying every game in the survey and most are to big to be feasible.

4.3 Tournament Outcomes And Analysis

The tournaments were performed in order to collect data. The data is composed of ten AIs and their resulting Elo-Rating after having completed the tournament. These Elo-Ratings were clustered by using the Jenks natural breaks optimization method and also with the custom step based clusterer. The outcome can be seen in fig 2 where the outcome of both methods are compared to each other. The outcomes of the individual tournaments are listed in Appendix C.
When using the Jenks natural breaks optimization method for clustering, we obtain three different results for our data. The AIs are devided into 2, 3 or 4 leagues. The games that had the lowest number of leagues (two leagues) were Mu Torere 1 and Pong Hau K'i. The games that had the highest number of leagues (four leagues) were Haretavl, Hat Diviyan Kelyia and Mu Torere 0. The rest of the games only had three leagues. This somewhat also mirrors the results that were received from the survey where the most challenging game was considered to be Haretavl, which was one of the games with higher depth.
In comparison, the stepcounter clusterer method assigned the AIs into leagues comprised between five to ten. Both clusterings show some correlation with the maximum tension with about 0.7. Especially the step clustering reacts highly on graph complexity measures like node edge ratio or branching factor.



Name	DepthSteps	DepthNatural
Boseog Gonu-1	10	3
Felli0	8	3
Felli1	9	3
Haretavl-1	8	4
Hat Diviyan Keliya-1	6	4
Ho-Bag Gonu-1	8	3
Ja-Jeon-Geo-Gonu-1	7	3
Kaooa-1	6	3
Madelinette-1	6	3
Mu Torere0	5	4
Mu Torere1	5	2
Pong Hau K'i-1	5	2
Round Merels-1	8	3
Symbol S.2-1	7	3
Tant Fant-1	7	3

Figure 2: Comparison between Depth using stepcounter and jenks natural breaks optimization

4.4 3D Representation

Overall this methods shows some potential, as there is more space for some relationships, which would already break a classic planar arrangement. The fact that the representation rotates helps to distinguish partly overlapping layers. In theory if one manages to increase the frame rate a simple 3d effect can be achieved using a red and green filter in front of one's eyes. As the brain's processing speed differs for different colors a rotating object can be used to create a virtual 3d effect (Morgan & Thompson, 1975). The biggest problem is the exploding number of edges as can be seen in fig 4, but fig 3 helps visualise the general ratio of more tense to low tense nodes and as higher nodes are closer to the root it also can help to show the distance to different terminal states. In the current version though the artistic value outweighs the utility.

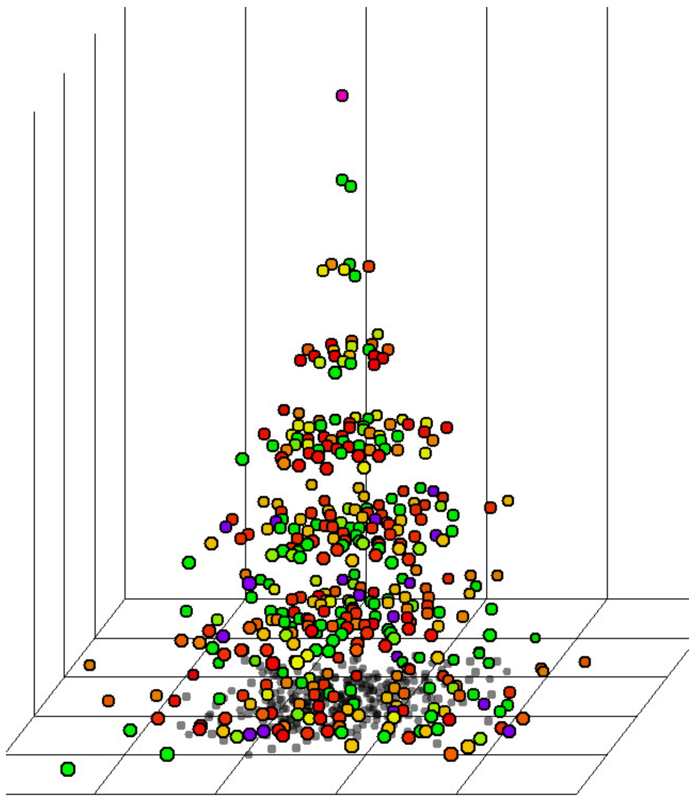


Figure 3: Showing the Round Merrels with full symmetry from the root's point of view. Terminal states are marked purple, other states show their tension

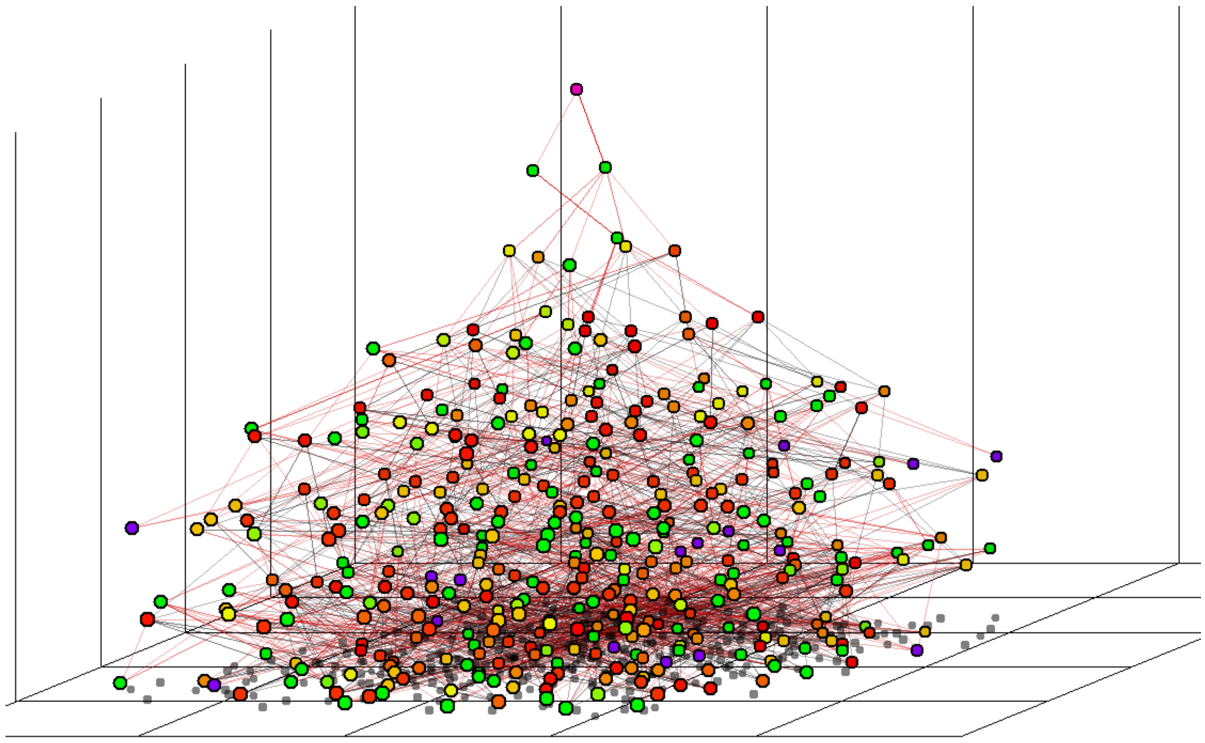
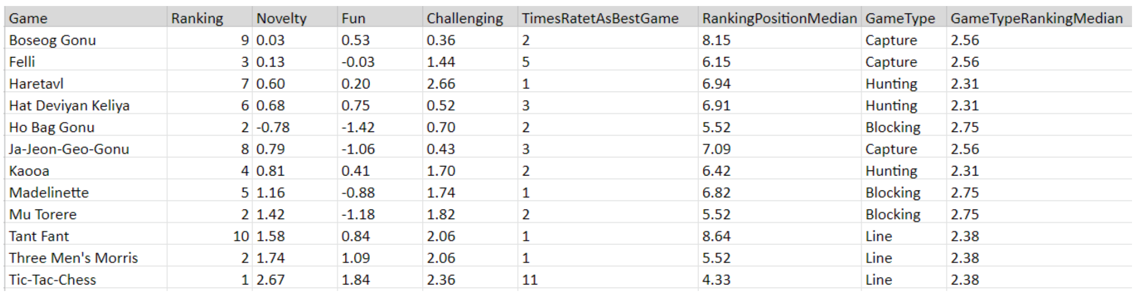


Figure 4: Round Merrels, this time with enabled edges

4.5 Survey Results

The survey reveals the preferences of the players and offers the possibility to relate these directly to the features. The survey results in three different evaluations of the games. Firstly, the voting within a category, secondly the voting per game directly related to the features and finally a superior ranking of all games. The absolute results are shown in the table of fig 5 below, as it represents survey games and their overall rated position in the first two columns. The values shown in the table represent the mean values of those



Game	Ranking	Novelty	Fun	Challenging	TimesRatetAsBestGame	RankingPositionMedian	GameType	GameTypeRankingMedian
Boseog Gonu	9	0.03	0.53	0.36	2	8.15	Capture	2.56
Felli	3	0.13	-0.03	1.44	5	6.15	Capture	2.56
Haretavl	7	0.60	0.20	2.66	1	6.94	Hunting	2.31
Hat Deviyan Keliya	6	0.68	0.75	0.52	3	6.91	Hunting	2.31
Ho Bag Gonu	2	-0.78	-1.42	0.70	2	5.52	Blocking	2.75
Ja-Jeon-Geo-Gonu	8	0.79	-1.06	0.43	3	7.09	Capture	2.56
Kaooa	4	0.81	0.41	1.70	2	6.42	Hunting	2.31
Madelinette	5	1.16	-0.88	1.74	1	6.82	Blocking	2.75
Mu Torere	2	1.42	-1.18	1.82	2	5.52	Blocking	2.75
Tant Fant	10	1.58	0.84	2.06	1	8.64	Line	2.38
Three Men's Morris	2	1.74	1.09	2.06	1	5.52	Line	2.38
Tic-Tac-Chess	1	2.67	1.84	2.36	11	4.33	Line	2.38

Figure 5: The results of the survey

measurements. The novelty, fun and challenging metrics can be measured in a range of -5 and 5 where the highest value represents the best game, and vise versa. The overall best rated game is Tic-Tac-Chess, with 11 rates for the best game and a mean rating of 4.33 of 12 possible positions. It combines a high novelty rate with high challenging and highest value of fun.

After conducting the survey and analysing the trees from the games involved, we arrive at the following results. These are the highlights worthy of attention, for the full results, please refer to the appendix. The survey data indicates that *tension* is one of the strongest metrics to predict interestingness of a game.

- Challenging and fun to play measurements are indicators for the testers while they rank the games. High challenging with high fun is related to an interesting game (e.g Tic-Tac-Chess) although high challenging with low fun represents not an interesting game (e.g Haretavl, Madelinette). A new metric from the survey was introduced which is composed of 0.65 times the "mean challenging" metric and 0.35 times the "mean fun" metric as the average of challenging and fun.
- Average Tension is strongly correlated with the average of challenging and fun. Average Tension represents the critical state of the game. Games with high tension contain a lot of random moves that are not influenced by the game progress so following the average tension is a way to investigate if a game is interesting or not for each game.
- Fun measurement is moderately correlated with the loop states of a game, since games with cycles, where a cycle represents the repetition of a game state, end up to be boring for the testers (e.g blocking games).
- Games that were rated to be very novel mostly ended up in the lower third of the rankings. It mostly depends on the rule set and the structure of the game, as a player may not immediately understand the rules and thus does not approach the game the same as if they were already familiar with a similar game. As an example, Three Men's Morris is a game of which many people are familiar with a similar game; Nine Men's Morris, which is essentially a larger version of the game. This can cause the player to view the game as less novel, as it is familiar to another game. Additionally, as participants play some of the unique games only twice, they end up ranking them with high novelty (e.g Ja-Jeon-Geo-Gonu about the formation of the game , Ho-Bag-Gonu about the rules) although do not give us any hints about the interestingness of the games (e.g Haretavl and Ho-Bag-Gonu).

4.6 Data relation

Correlation matrix is used to summarize the correlation between theoretic matrices computed by our algorithm. Correlation matrix is a table showing correlation coefficients using Pearson method a linear correlation between variables. Compute pairwise correlation of columns, each cell in the table shows the correlation between two variables. The Pearson Correlation Coefficient is used to generate the correlation matrix shown in Figure 6. It evaluates the linear relationship between two continuous variables, that in case of this research represents the extracted approach values of all used games. A relationship is linear when a change in one variable is associated with a proportional change in the other variable(Adler & Parmryd, 2010). The colours are in a range of dark red to dark blue, where dark red relates to a correlation value of 1 and dark blue of -1. White or almost white fields relate to correlation values of exactly or about zero, what means that no relation is feasible. The relation value can be interpreted as following (Bartelt & Evans, 1996): Values from 0 to 0.19 are very weak, 0.2 to 0.39 are weak, 0.4 to 0.59 are moderate, 0.6 to 0.79 are strong and 0.8 to 1 very strong related. The same takes into account for negative values. All values higher than 0.4 and lower than -0.4 are significant for the research due they represent values that are not equal to each other but related in certain intensity.

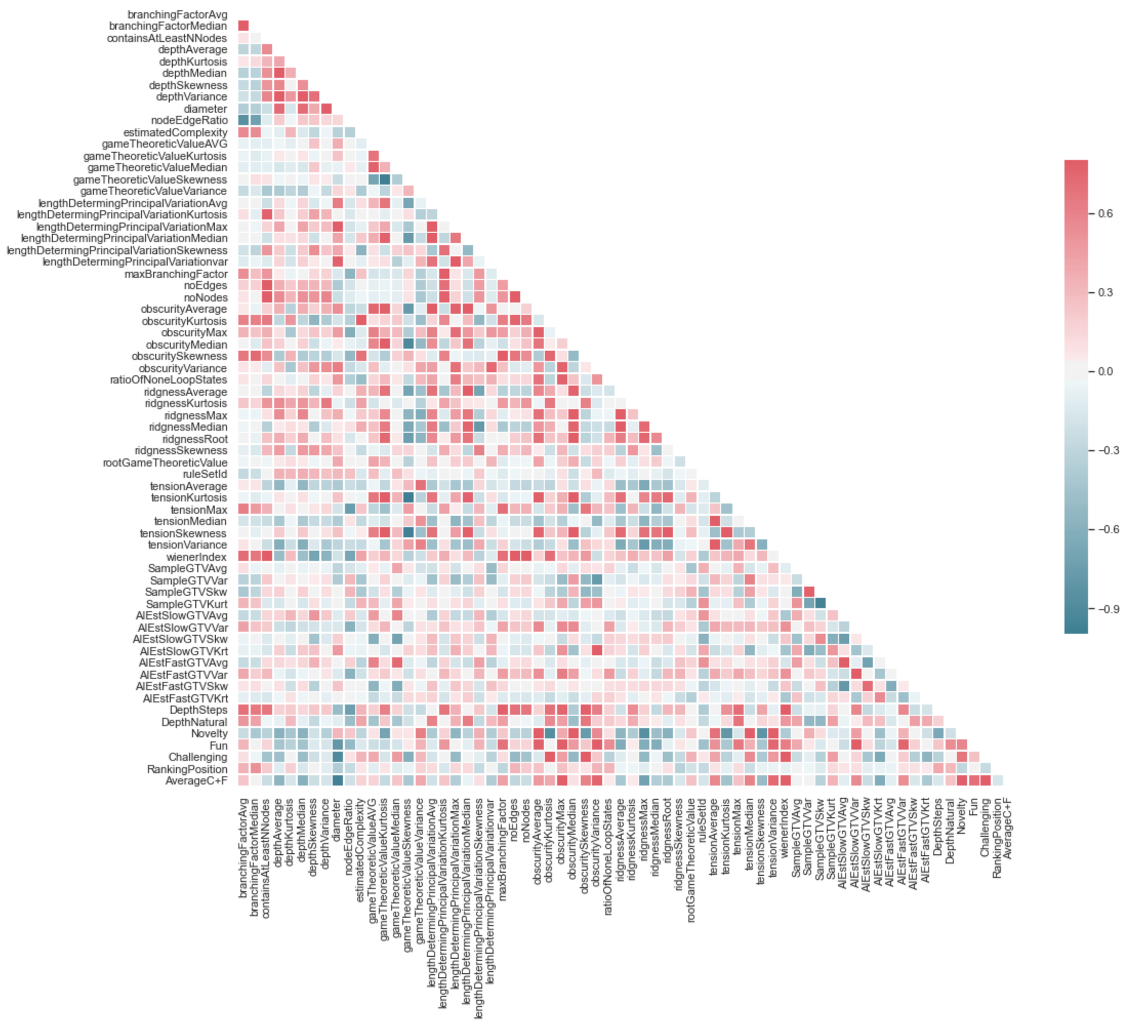


Figure 6: Correlation Matrix

5 Discussion

In this section we deliver some analysis on the results from Section 4.

5.1 Game Graph

The graphical representation of Mu Torere is depicted in figure 1. The nodes represent the unique reachable game states and they have been coloured according to their measured tension, from green to red representing from low to high tension respectively. The green states can be basically considered safe for the mover because player can hardly make mistake here. Oppositely red ones are theoretically the most dangerous cases in the playout, any wrong move here can be deadly for the mover. Take the move Orange nodes with mediocre level of tension standing somewhere in-between green and red. Edges coloured red indicate that for the starting state, the target this edge leads to is the considered optimal from the perspective of game theoretic value.

Keep on inspecting Mu Torere's position graph with the knowledge that this game only score a negative value in terms of fun in the survey. Considering the rule of Mu Torere, we can easily notice that in whatever states, there are at most two moves for players to decide. This property is well represented in the graph, where most of the nodes have only one or two descendants. The game can even progress in a normal manner without both players making much efforts to ponder. And if the current situation is in a middle of a cycle, it can be suggested that there are very few changes players can make to escape the infinite cycle. The situation is even worse especially when human player is against a strong AI, human might have to make an unfavorable move in exchange for escaping the cycle. State 23 can best exemplify such dilemma. The black player (say black is a human and white an AI) is the current mover in state 23, if it chooses 24 which is optimal because a red line connects 23 and 24, then both players will enter a cycle, but it if chooses 28, black player will be immediately terminated by its opponent who programatically goes for state 29.

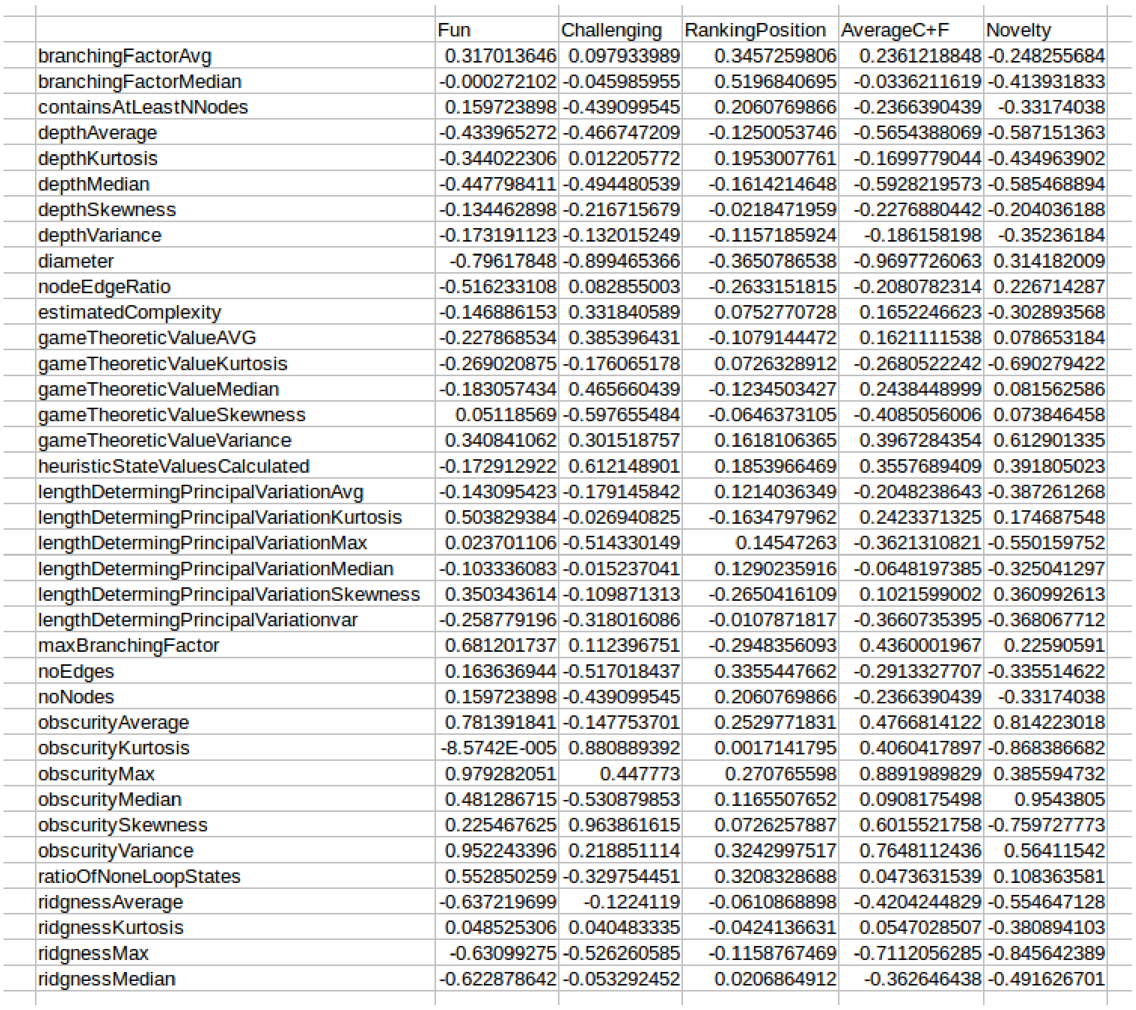
5.2 Sampling Game Theoretic Values

It can be seen that Ludii MCTS playouts can be used to quickly estimate the quality of a game as a high variance of outcomes is wished for. This value is correlated with 0.84 with fun. This makes sense and this means fairness and a low drawishness. A smaller kurtosis correlates with challenging as determined by the survey, which also expected, because in our games draws were achieved by reaching a maximum number of turns without a winner. As seen in fig 7 and 8.

5.3 Survey vs statistical measures

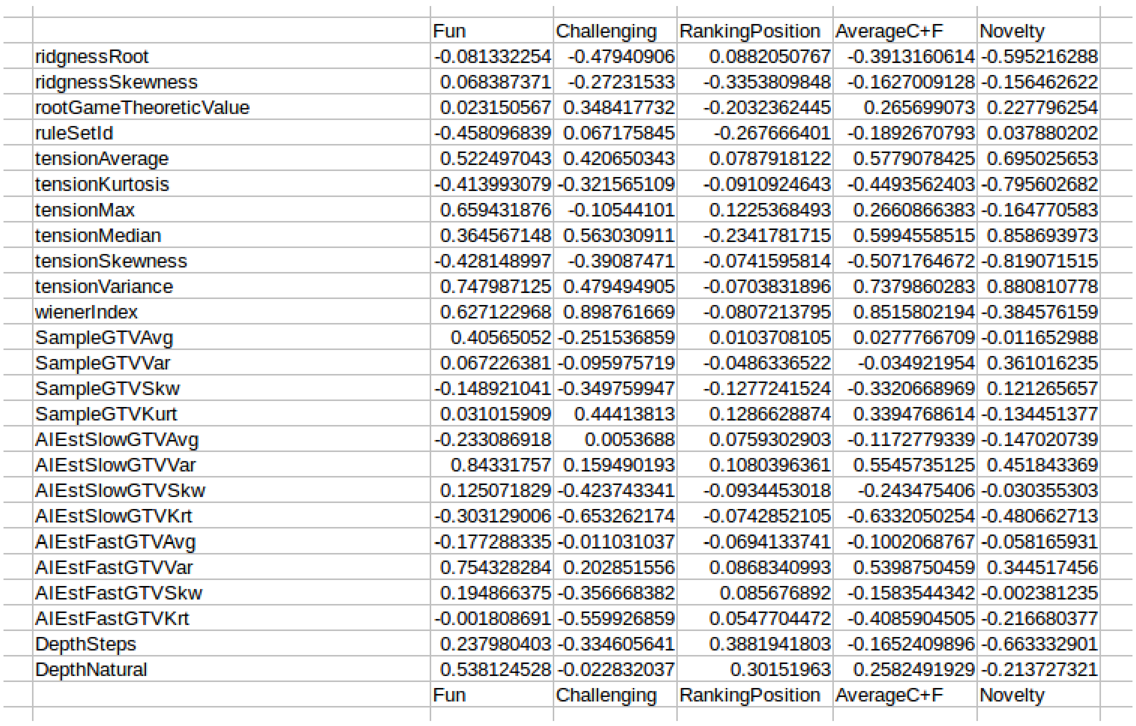
It is possible that some of the survey results are not directly caused by the games themselves. For example, we find that novelty negatively correlates to interstingness. While it can be the case that novel games are less interesting than more traditional games, we reason it might be the case that games that were difficult to understand were rated as the most novel. This could have been accounted for by including and extra option to rate how well participants understood the rules and/or aims of the game.

Diameter first surprised us with a -0.96 correlation to the combined fun and challenging value. These correlations could be statistical accidents, as not for all graphs diameter, wiener index, obscurity or tournament depth have been calculated, comparing those with the survey result, leads to extremer correlation then they might otherwise would have as those have overlaps with each other and the survey result only at five datapoint. Especially diameter could just mean how quick this game could be understood and the variance of obscurity needs to be investigated further due We are confident though about the tension variance as this would mean one reaches different situations in play. Also ridgeness showed promising correlation of -0.64 with fun. A lower node edge ratio also lead to more fun with a weak correlation of -0.5. The variance of tension is the most reliable measurement for the combined fun and challenging metric with a value of 0.73. These values can be seen in fig 7 and 8.



	Fun	Challenging	RankingPosition	AverageC+F	Novelty
branchingFactorAvg	0.317013646	0.097933989	0.3457259806	0.2361218848	-0.248255684
branchingFactorMedian	-0.000272102	-0.045985955	0.5196840695	-0.0336211619	-0.413931833
containsAtLeastNNodes	0.159723898	-0.439099545	0.2060769866	-0.2366390439	-0.33174038
depthAverage	-0.433965272	-0.466747209	-0.1250053746	-0.5654388069	-0.587151363
depthKurtosis	-0.344022306	0.012205772	0.1953007761	-0.1699779044	-0.434963902
depthMedian	-0.447798411	-0.494480539	-0.1614214648	-0.5928219573	-0.585468894
depthSkewness	-0.134462898	-0.216715679	-0.0218471959	-0.2276880442	-0.204036188
depthVariance	-0.173191123	-0.132015249	-0.1157185924	-0.186158198	-0.35236184
diameter	-0.79617848	-0.899465366	-0.3650786538	-0.9697726063	0.314182009
nodeEdgeRatio	-0.516233108	0.082855003	-0.2633151815	-0.2080782314	0.226714287
estimatedComplexity	-0.146886153	0.331840589	0.0752770728	0.1652246623	-0.302893568
gameTheoreticValueAVG	-0.227868534	0.385396431	-0.1079144472	0.1621111538	0.078653184
gameTheoreticValueKurtosis	-0.269020875	-0.176065178	0.0726328912	-0.2680522242	-0.690279422
gameTheoreticValueMedian	-0.183057434	0.465660439	-0.1234503427	0.2438448999	0.081562586
gameTheoreticValueSkewness	0.05118569	-0.597655484	-0.0646373105	-0.4085056006	0.073846458
gameTheoreticValueVariance	0.340841062	0.301518757	0.1618106365	0.3967284354	0.612901335
heuristicStateValuesCalculated	-0.172912922	0.612148901	0.1853966469	0.3557689409	0.391805023
lengthDetermingPrincipalVariationAvg	-0.143095423	-0.179145842	0.1214036349	-0.2048238643	-0.387261268
lengthDetermingPrincipalVariationKurtosis	0.503829384	-0.026940825	-0.1634797962	0.2423371325	0.174687548
lengthDetermingPrincipalVariationMax	0.023701106	-0.514330149	0.14547263	-0.3621310821	-0.550159752
lengthDetermingPrincipalVariationMedian	-0.103336083	-0.015237041	0.1290235916	-0.0648197385	-0.325041297
lengthDetermingPrincipalVariationSkewness	0.350343614	-0.109871313	-0.2650416109	0.1021599002	0.360992613
lengthDetermingPrincipalVariationvar	-0.258779196	-0.318016086	-0.0107871817	-0.3660735395	-0.368067712
maxBranchingFactor	0.681201737	0.112396751	-0.2948356093	0.4360001967	0.22590591
noEdges	0.163636944	-0.517018437	0.3355447662	-0.2913327707	-0.335514622
noNodes	0.159723898	-0.439099545	0.2060769866	-0.2366390439	-0.33174038
obscurityAverage	0.781391841	-0.147753701	0.2529771831	0.4766814122	0.814223018
obscurityKurtosis	-8.5742E-005	0.880889392	0.0017141795	0.4060417897	-0.868386682
obscurityMax	0.979282051	0.447773	0.270765598	0.8891989829	0.385594732
obscurityMedian	0.481286715	-0.530879853	0.1165507652	0.0908175498	0.9543805
obscuritySkewness	0.225467625	0.963861615	0.0726257887	0.6015521758	-0.759727773
obscurityVariance	0.952243396	0.218851114	0.3242997517	0.7648112436	0.56411542
ratioOfNoneLoopStates	0.552850259	-0.329754451	0.3208328688	0.0473631539	0.108363581
ridgnessAverage	-0.637219699	-0.1224119	-0.0610868898	-0.4204244829	-0.554647128
ridgnessKurtosis	0.048525306	0.040483335	-0.0424136631	0.0547028507	-0.380894103
ridgnessMax	-0.63099275	-0.526260585	-0.1158767469	-0.7112056285	-0.845642389
ridgnessMedian	-0.622878642	-0.053292452	0.0206864912	-0.362646438	-0.491626701

Figure 7: Correlations between survey and all graph measurments



	Fun	Challenging	RankingPosition	AverageC+F	Novelty
ridgnessRoot	-0.081332254	-0.47940906	0.0882050767	-0.3913160614	-0.595216288
ridgnessSkewness	0.068387371	-0.27231533	-0.3353809848	-0.1627009128	-0.156462622
rootGameTheoreticValue	0.023150567	0.348417732	-0.2032362445	0.265699073	0.227796254
ruleSetId	-0.458096839	0.067175845	-0.267666401	-0.1892670793	0.037880202
tensionAverage	0.522497043	0.420650343	0.0787918122	0.5779078425	0.695025653
tensionKurtosis	-0.413993079	-0.321565109	-0.0910924643	-0.4493562403	-0.795602682
tensionMax	0.659431876	-0.10544101	0.1225368493	0.2660866383	-0.164770583
tensionMedian	0.364567148	0.563030911	-0.2341781715	0.5994558515	0.858693973
tensionSkewness	-0.428148997	-0.39087471	-0.0741595814	-0.5071764672	-0.819071515
tensionVariance	0.747987125	0.479494905	-0.0703831896	0.7379860283	0.880810778
wienerIndex	0.627122968	0.898761669	-0.0807213795	0.8515802194	-0.384576159
SampleGTVAvg	0.40565052	-0.251536859	0.0103708105	0.0277766709	-0.011652988
SampleGTVVar	0.067226381	-0.095975719	-0.0486336522	-0.034921954	0.361016235
SampleGTVSkw	-0.148921041	-0.349759947	-0.1277241524	-0.3320668969	0.121265657
SampleGTVKurt	0.031015909	0.44413813	0.1286628874	0.3394768614	-0.134451377
AIEstSlowGTVAvg	-0.233086918	0.0053688	0.0759302903	-0.1172779339	-0.147020739
AIEstSlowGTVVar	0.84331757	0.159490193	0.1080396361	0.5545735125	0.451843369
AIEstSlowGTVSkw	0.125071829	-0.423743341	-0.0934453018	-0.243475406	-0.030355303
AIEstSlowGTVKrt	-0.303129006	-0.653262174	-0.0742852105	-0.6332050254	-0.480662713
AIEstFastGTVAvg	-0.177288335	-0.011031037	-0.0694133741	-0.1002068767	-0.058165931
AIEstFastGTVVar	0.754328284	0.202851556	0.0868340993	0.5398750459	0.344517456
AIEstFastGTVSkw	0.194866375	-0.356668382	0.085676892	-0.1583544342	-0.002381235
AIEstFastGTVKrt	-0.001808691	-0.559926859	0.0547704472	-0.4085904505	-0.216680377
DepthSteps	0.237980403	-0.334605641	0.3881941803	-0.1652409896	-0.663332901
DepthNatural	0.538124528	-0.022832037	0.30151963	0.2582491929	-0.213727321
	Fun	Challenging	RankingPosition	AverageC+F	Novelty

Figure 8: Correlations between survey and all graph measurments cont.

6 Conclusion

In this work we present a measure by which a game's game tree size can be expressed, although it may not fully encapsulate the entirety of the trees attributes, it will approximate the memory size of a tree well. Small is relative and while a megagame sounds big, in comparison to other games such as chess, the tree sizes of the games analysed are very small. A game can be estimated to be interesting to humans by sampling MCTS based AI playouts with variable skill levels. If a game produces more variance in its sampled playouts, it will be more likely to be received positively by human players. The challenge of a game can be estimated by looking at the kurtosis of the playouts. Visualised game tree printouts can be useful for analysing games and make it easy to recognise particular aspects of a game. However, there is an upper bound to the games state space as an abundance of nodes will start to clutter the visualisation making it hard to interpret properly. Unfortunately, this limit was observed to be reached quite quickly, at the size of nodes approaching a few hundred. Among the metrics tested for their predictive ability on the reception of games, the diameter, Wiener index and obscurity show to highly correlate. However, this outcome will have to be taken with caution as the amount of data points is a limited 5. Higher variance in the tension metric throughout a game scored a high correlation of 0.74 on predicting a fun challenge for a player. This coincides with the game design philosophy that even the harder games require moments to 'relax'. Another strong predictor is the variance of the game theoretic value. A possible explanation is that a highly varying game theoretic value may indicate that player moves have a lot of impact on the course of the game and that mistakes made during game play provide the opposing player the opportunity to improve their standing.

All of the metrics used on the game trees made full use of the fully expanded, perfect information nature. It is possible that some of these metrics are applicable to non-expanded trees. However, it is not certain whether the metrics are still as successful under incomplete sampling. For instance, measures such as obscurity rely on being able to see the truth of a move that is not visible at first glance.

6.1 Future Works

In this paper, the research has been contained to only fully expanded trees. However, the majority of games are quick to generate trees that are not feasible for full expansion with current hardware. It would therefore be of value to test whether the more successful metrics proposed in this work can be adapted and used in non-expanded trees.

The visualisation of the graphs is beneficial if the number of nodes allows it, for further experiments adding the third dimensions better arranging algorithms should be used and the number of selected edges should be limited or it should be determined, which nodes to omit. A game like French Military game, should mostly contain nodes which contain tension. This could be useful for example to show which nodes have been reached during a search, where the number of used edges is significantly lower. Graphs like these can as well serve as an assistant for human to find the right strategy and win the game.

During the development of the metrics, the boxicity metric had to be omitted. We reason this might have possibly been an insightful metric as it seems to approximate the convolution of a game's state transitions. Every vertex is assigned to a hyper-rectangle. If two vertices share an edge, their hyper-rectangles also overlap. The boxicity is the lowest dimension in which the hyper-rectangles can be represented, without false overlaps. Every planar graph for example has a boxicity of maximum 3.

References

Adler, J., & Parmryd, I. (2010). Quantifying colocalization by correlation: the pearson correlation coefficient is superior to the mander's overlap coefficient. *Cytometry Part A*, *77*(8), 733–742.

Allis, L. V., et al. (1994). *Searching for solutions in games and artificial intelligence*. Ponsen & Looijen Wageningen.

Arora, S., & Barak, B. (2007). Computational complexity: A modern approach.

Ascher, M. (1987). Mu torere: An analysis of a maori game. *Mathematics Magazine*, *60*(2), 90–100. Retrieved from http://www.jstor.org/stable/2690304

Bartelt, M., & Evans, J. W. (1996). Exact island-size distributions for submonolayer deposition: Influence of correlations between island size and separation. *Physical Review B*, *54*(24), R17359.

Bowling, M., Burch, N., Johanson, M., & Tammelin, O. (2017). Heads-up limit hold'em poker is solved. *Communications of the ACM*, *60*(11), 81–88.

Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., … Colton, S. (2012). A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, *4*(1), 1–43.

developers, N. (2014–2020). *Networkx home page.* Retrieved 2020-06-24, from https://networkx.github.io

Eppstein, D. (2020). *Computational complexity of games and puzzles.* Retrieved 2020-03-18, from https://www.ics.uci.edu/ eppstein/cgt/hard.html

Eric Piette, Soemers, D. J. N. J., Stephenson, M., Sironi, C. F., Winands, M. H. M., & Browne, C. (2019). Ludii - the ludemic general game system. *CoRR*, *abs/1905.05013*. Retrieved from http://arxiv.org/abs/1905.05013

Fraenkel, A. s., & Lichtenstein, D. (1981). Computing a perfect strategy for nxn chess requires time exponential in n.

Heule, M. J., & Rothkrantz, L. J. (2007). Solving games: Dependence of applicable solving procedures. *Science of Computer Programming*, *67*(1), 105–124.

Iida, H., Takahara, K., Nagashima, J., Kajihara, Y., & Hashimoto, T. (2004). An application of game-refinement theory to mah jong. In *International conference on entertainment computing* (pp. 333–338).

Jaffe, A. B. (2013). *Understanding game balance with quantitative methods* (Unpublished doctoral dissertation).

Kramer, W. (2000). What makes a game good. *Game & Puzzle Design, vol. 1, no. 2, 2015 (Colour)*, 84.

Lantz, F., Isaksen, A., Jaffe, A., Nealen, A., & Togelius, J. (2017, 1 1). Depth in strategic games. , *WS-17-01 - WS-17-15*, 967–974. (31st AAAI Conference on Artificial Intelligence, AAAI 2017 ; Conference date: 04-02-2017 Through 10-02-2017)

Loyd, S. (1878). *Chess strategy: A treatise upon the art of problem composition.* Retrieved from https://books.google.be/books?id=E5FZAAAAYAAJ

Margulies, S. (1977). Principles of beauty. *Psychological Reports*, *41*, 3–11.

Morgan, M. J., & Thompson, P. (1975). Apparent motion and the pulfrich effect. *Perception*, *4*(1), 3–18.

Schadd, M. P. D., Winands, M. H. M., Uiterwijk, J. W. H. M., Herik, H. J. V. D., & Bergsma, M. H. J. (2008). Best Play In Fanorona Leads To Draw. *New Mathematics and Natural Computation (NMNC)*, *4*(03), 369-387. doi: 10.1142/S1793005708001124

Schaeffer, J., Burch, N., Björnsson, Y., Kishimoto, A., Müller, M., Lake, R., … Sutphen, S. (2007). Checkers is solved. *science*, *317*(5844), 1518–1522.

Thompson, J. (2000). *Defining the abstract. the games journal.*

Uiterwijk, J. W. (2016). Domineering is solved: The first player wins. In *International conference on computers and games* (pp. 129–136).

Van Den Herik, H. J., Uiterwijk, J. W., & Van Rijswijck, J. (2002). Games solved: Now and in the future. *Artificial Intelligence*, *134*(1-2), 277–311.

Winands, M. H. (2008). 6x6 loa is solved. *ICGA Journal*, *31*(4), 234–238.

Yong, A., & Yong, D. (2019). An estimation method for game complexity.

yWorksGmbH. (2019–2020). *yed graph editor home page.* Retrieved 2020-06-24, from https://www.yworks.com/products/yed

Zhang, W. (1999). *State-space search: Algorithms, complexity, extensions, and applications*. Springer Science & Business Media.

A Appendix - Position graphs for games on Mu Torere boards, Pong Hau'ki and Madelinette

As mentioned in the main part every possible legal state in a small board game was extracted by an AI. Thereafter the game theoretical values are back-propagated. The position graphs can be found in this section. Nodes are coloured according to tension, and red edges are indicators of an optimal move.

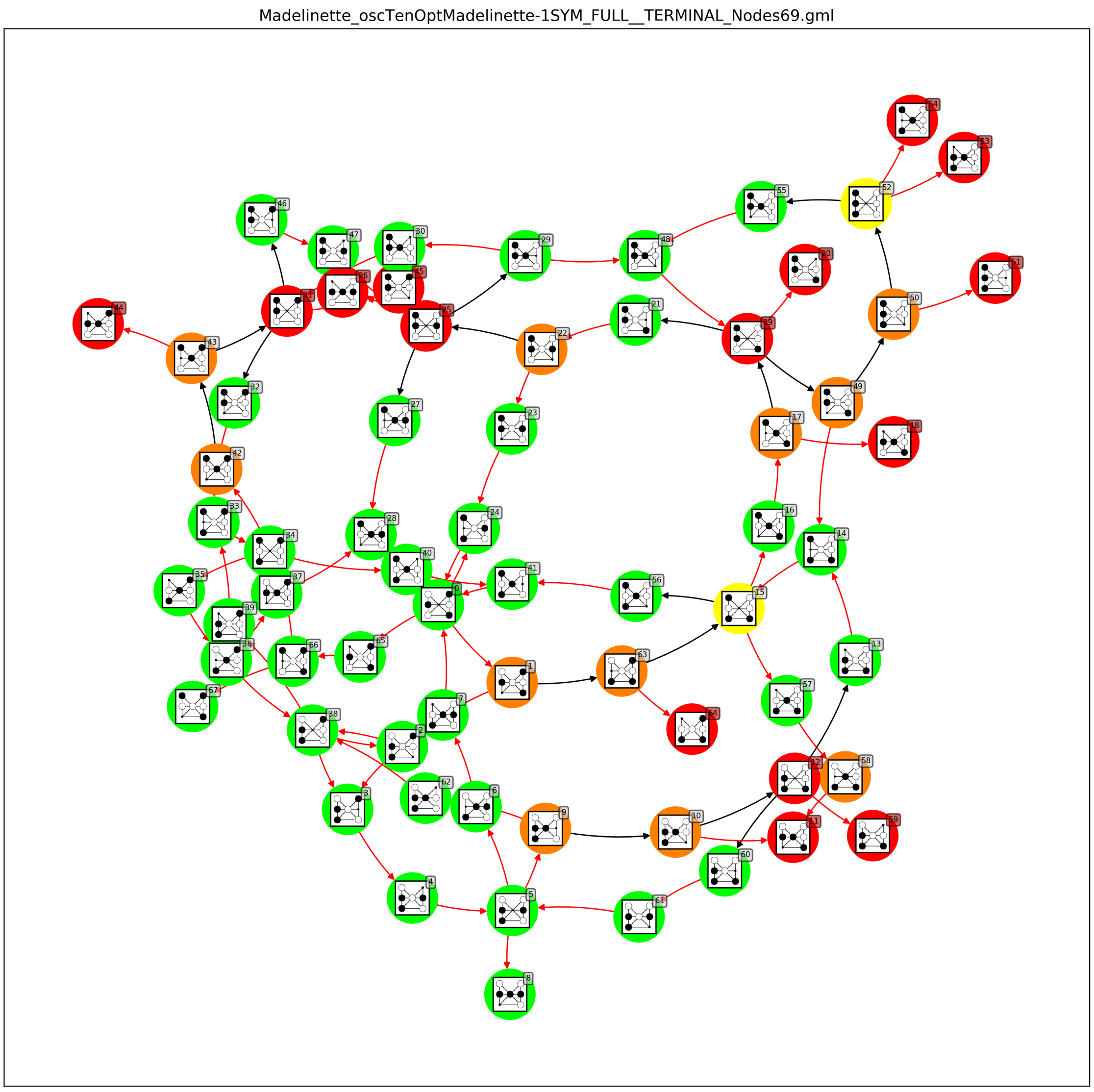

Madelinette_oscTenOptMadelinette-1SYM_FULL__TERMINAL_Nodes69.gml

Figure 9: Position graph for Madelinette

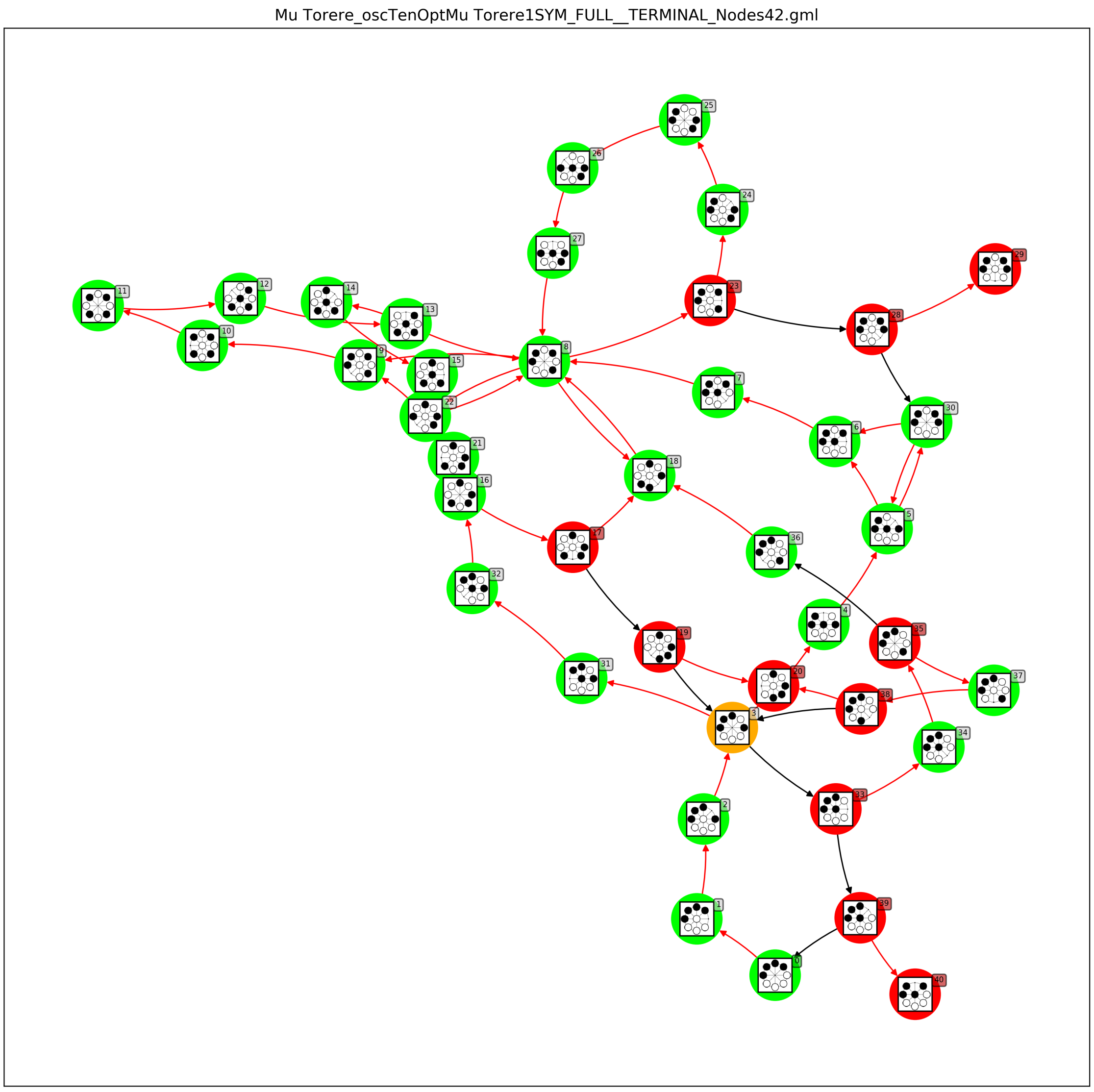


Figure 10: Position graph for Mu Torere

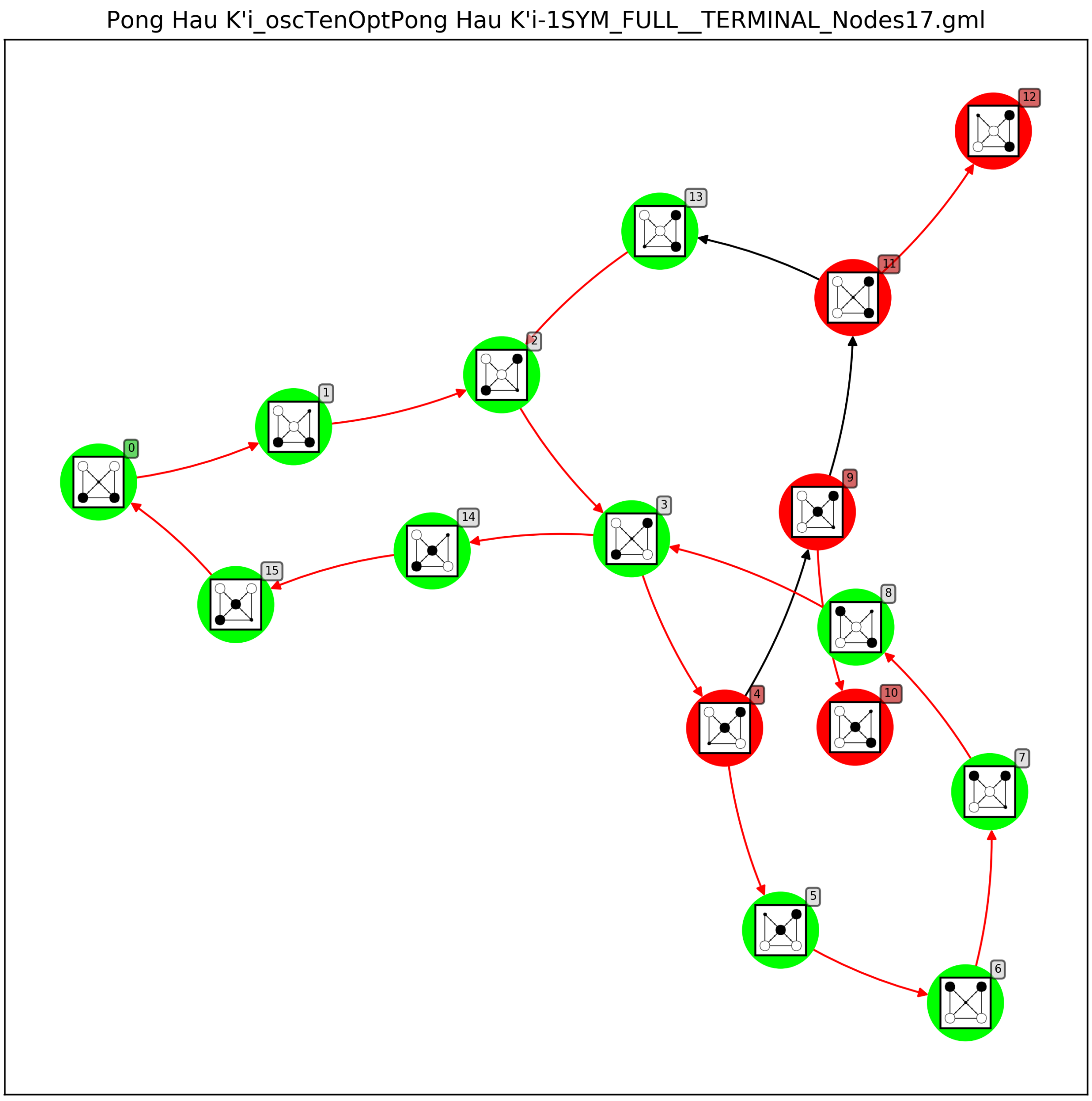

Pong Hau K'i_oscTenOptPong Hau K'i-1SYM_FULL__TERMINAL_Nodes17.gml
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Figure 11: Position graph for Pong Hau K’i

Shisima_oscTenOptShisima-1SYM_FULL__TERMINAL_Nodes115.gml

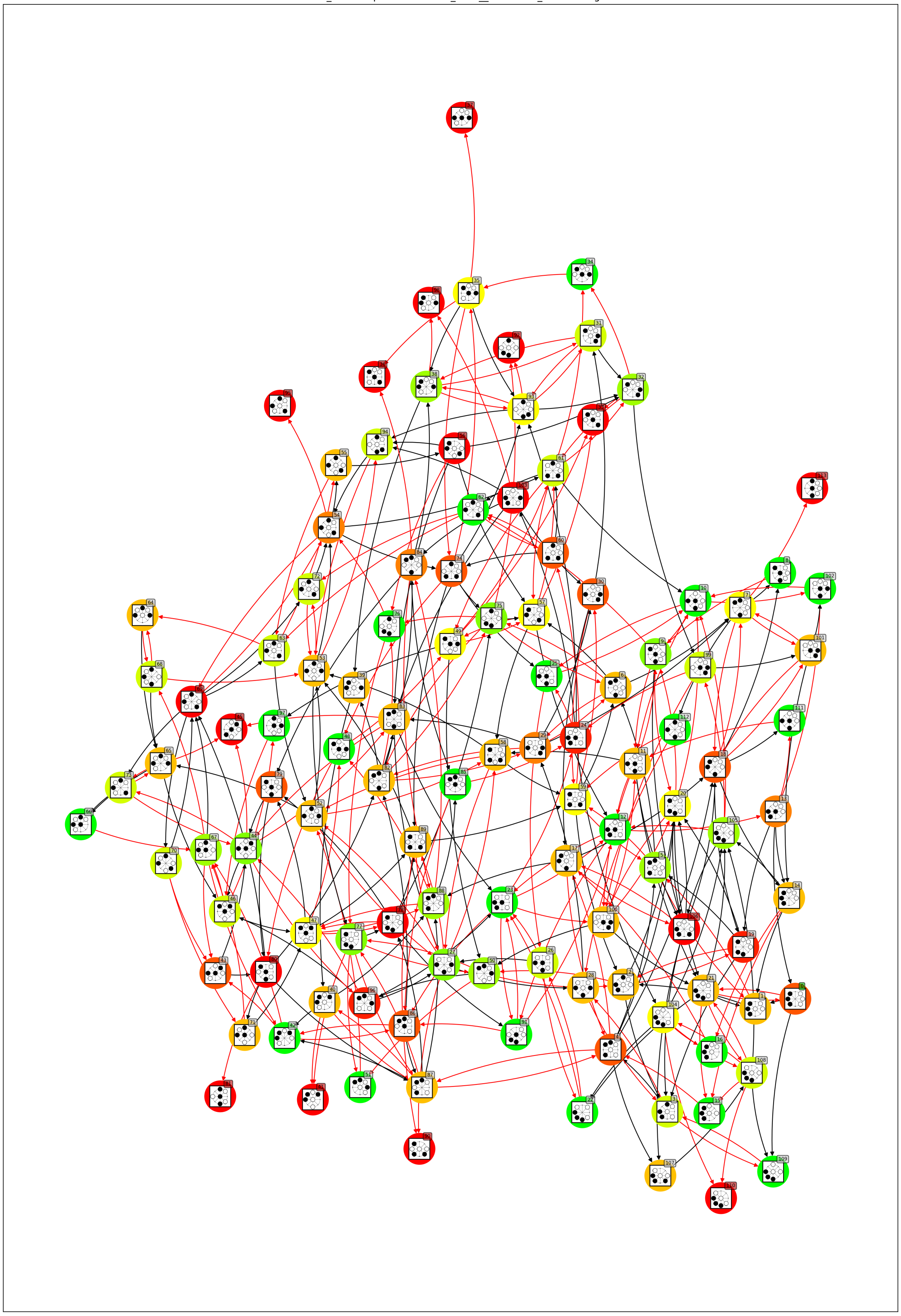


Figure 12: Position graph for Shisima

Wagner_oscTenOptWagner-1SYM_FULL__TERMINAL_Nodes43.gml

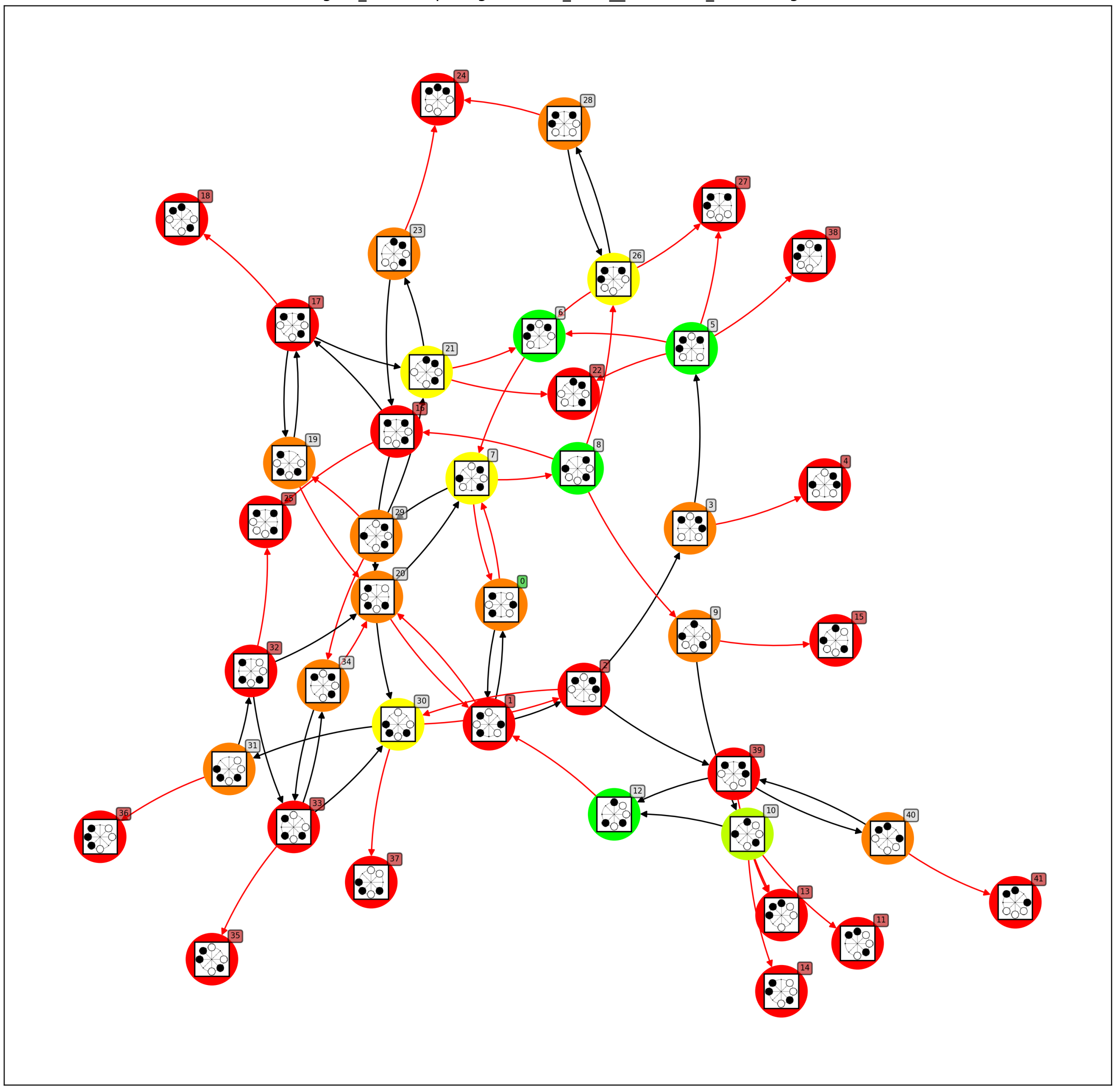


Figure 13: Position graph for Wagner

B Appendix - Rulesets of considered games

B.1 Blocking games

Ho Bag Gonu
Rules:

- Each player has three pieces.
- Players take turns moving a piece to an empty dot.
- Players cannot return to their starting three dots.
- Pieces that move into the opponent's starting three dots cannot exit them.
- A player wins by blocking the other player from being able to move

Madelinette
Rules:

- The game is played by two players, each having three pieces, which start the game on the edges.
- Each player in his turn moves a piece from its point, along a marked line, to an adjacent empty point.
- There is no jumping as found in other games, nor is there capture.
- The game is over when one of the players is trapped and cannot move. His opponent is declared the winner.

Mu Torere
Rules:

- The game is played by two players, each start with four pieces.
- The pieces are arranged on the board so that each player occupies one half of the board. The middle is empty.
- Players take turns moving a piece to the empty hole.
- Pieces can only be moved if one of the opponent's pieces is next to one of the player's pieces.
- A player wins if the opponent can not move anymore.
- Variant: There is no restriction on moving a piece to the center hole, except that the first move must allow the second player to move.

French Military Game
Rules:

- The French Military Game is played by two players.
- One player takes the part of three hunters, the other the part of a single prey piece.
- First the hunter player moves one of his pieces, then the prey moves, play alternating thereafter until the game is ended.
- A hunter piece may move one step along a marked line in any forward or sideways direction. Hunter pieces cannot move backwards, diagonally or otherwise, towards the end of the board from which they started.
- The hare may move one step in any direction along a marked line.
- The prey wins by passing the hunters and reaching the end of the board from which they started. The hunters win by trapping the prey so that it cannot move in its turn.
- Variant: The single piece may place their piece on any empty spot at the beginning of the game. The rest stays the same.

L Game
Rules:

- Played on a board of 4x4 squares.
- Each player controls a 3x2 L-shaped piece, and there are two 1x1 pieces which either player can manipulate.
- On a turn, a player moves the L piece to a new available position, and then has the option to move one of the 1x1 pieces.

- When a player cannot move their L piece, they lose.

Pong Hau K'I
Rules:

- Pieces begin on opposite sides of the square.
- Players take turns moving the piece to an empty spot either orthogonally or diagonally, but one orthogonal direction is forbidden.
- The player who blocks the other player from being able to move wins.
- Variant: The piece are already placed on the board.

B.2 Hunting games

Haretavl
Rules:

- One player has three hound pieces and the other player has a single hare piece.
- Players take turns adding one of their pieces to an empty spot (the hare gets a single placement).
- Players take turns moving a piece of theirs to an adjacent empty spot.
- The hounds win by surrounding the hare so that it cannot move
- The game is played by two players, each having three pieces, which start the game on the edges.

Kaooa
Rules:

- Played on a five-pointed star shaped board.
- One player plays with one piece, the "tiger," and the other plays with seven pieces, the "kaooas."
- The player with the kaooas attempt to checkmate the tiger by moving to one of the points where the lines of the board intersect.
- The tiger captures kaooas by hopping over them. The tiger wins by capturing all the kaooas.

Hat Deviyan Deliya
Rules:

- One player plays as the tiger, and the other plays as seven leopards.
- The tiger plays their piece on a point where lines intersect first, and then on subsequent turns the leopards are placed one-by-one.
- Moves occur along the lines to an adjacent intersection.
- The tiger may capture a leopard by hopping over it.
- The tiger's goal is to capture four of the leopards; the leopards' goal is to block the tiger so it cannot move.

B.3 Line games

Tic-Tac-Chess
Rules:

- Players take turns placing a piece of theirs at an empty cell. The pieces are the chess figures King, Queen and Rook.
- When all pieces have been placed, players take turns moving one of their pieces.
- The pieces move like the equivalent Chess pieces but do not capture.
- Any piece can hop over an adjacent enemy piece to an empty cell beyond (without capturing it).
- First to make a line of 3 of their pieces, at any time, wins the game.

Three Men's Morris
Rules:

- Two players each start with three pieces.
- Players take turns adding a piece to an empty hole.
- Players then take turns moving one of their pieces along a line to an adjacent empty point.
- A player wins, if the he make a consecutive line of three of their pieces at any time.

Tant Fant
Rules:

- Two players each start with three pieces placed in a row on their home side.
- Players take turns adding a piece to an empty hole.
- Players then take turns moving one of their pieces along a line to an adjacent empty point.
- A player wins, if the he make a consecutive line of three of their pieces at any time.

Picaria
Rules:

- Two players each have three pieces.
- Players take turns placing one of their pieces at an empty hole.
- Players then take turns moving one of their pieces to an adjacent empty hole.
- A player wins by making a straight line of three of their pieces through the centre, at any time.

B.4 Capture games

Boseog Gonu
Rules:

- Two players each start with four pieces each.
- Players take turns adding a piece to an empty dot.
- A player may capture an opponent's piece by playing a piece on either side of an opponent's piece.
- If a player moves their own piece between two of the opponent's pieces, it is not captured.
- A player wins by reducing the opponent to one piece.

Felli
Rules:

- Each player's six pieces are set up on their respective triangle.
- A piece is moved one space in any direction per turn onto an empty point along the lines of the board.
- A piece captures an enemy piece by hopping over it as in draughts.
- Only one piece may be used to move or capture per turn. Players alternate their turns throughout the game.

Ja-Jeon-Geo-Gonu (Bicycle Game)
Rules:

- Players begin with four pieces, arranged in the square of spaces in their bottom right corner.
- Players take turns moving a piece to one adjacent spot.
- To capture an opponent's piece, you must move along the curved loops.
- The spot at the immediate end of the loop must be empty, but the piece may continue as far as the player wishes, including moving along successive loops, or until they make a capture or are stopped by their own piece.

B.5 Mathematical games

Chopsticks
Rules:

- Players begin with one finger extended on each hand.
- They take turns tapping a hand, either the opponent's or their own.
- The number of fingers on the tapping hand is added to the tapped hand, the tapping hand is unchanged. The tapped hand must extend the appropriate number of fingers.
- If the total exceeds five on a hand, that hand then shows the number over five (for example, if it has three and is tapped by three, it shows one.).
- Points can be transferred between a player's hands by tapping them together. If the score is exactly five on one hand, it is out and goes to zero.
- The goal is to force a player to either have all five fingers on both hands showing, or to force them to have one hand out and one with all five fingers showing.

Nim
Rules:

- Essentially, the game consists of a number of objects in 'heaps,' and players alternate turns taking any number of objects from one of the heaps.
- This continues until no objects are left, and the last player to remove an object loses.
- Variant: The last mover loses.

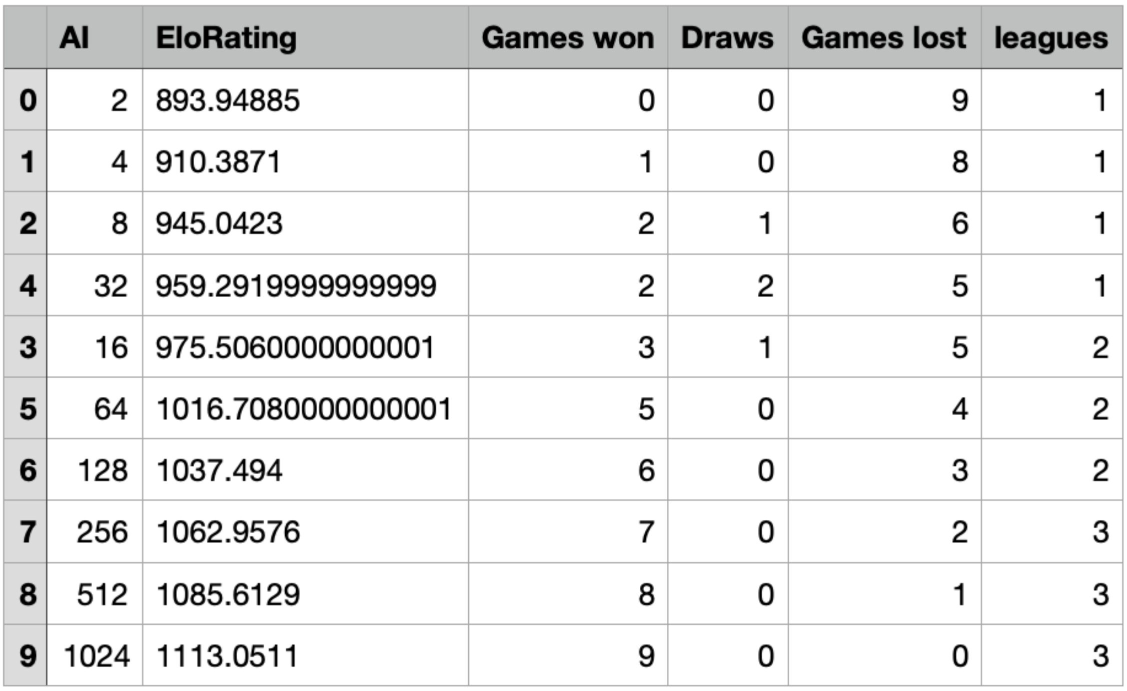
B.6 Planning games

Insanity Rules:

- One Player
- The pieces start with each color on opposite ends of the board, with two empty holes in between.
- Pieces can only be moved forward and can jump over one piece of the other color at a time.
- One color can be moved as long as it can be moved upon the rules.
- The goal is to move the pieces so that they end up on opposite sides of the board from their starting position.

C Appendix - Outcomes of tournaments

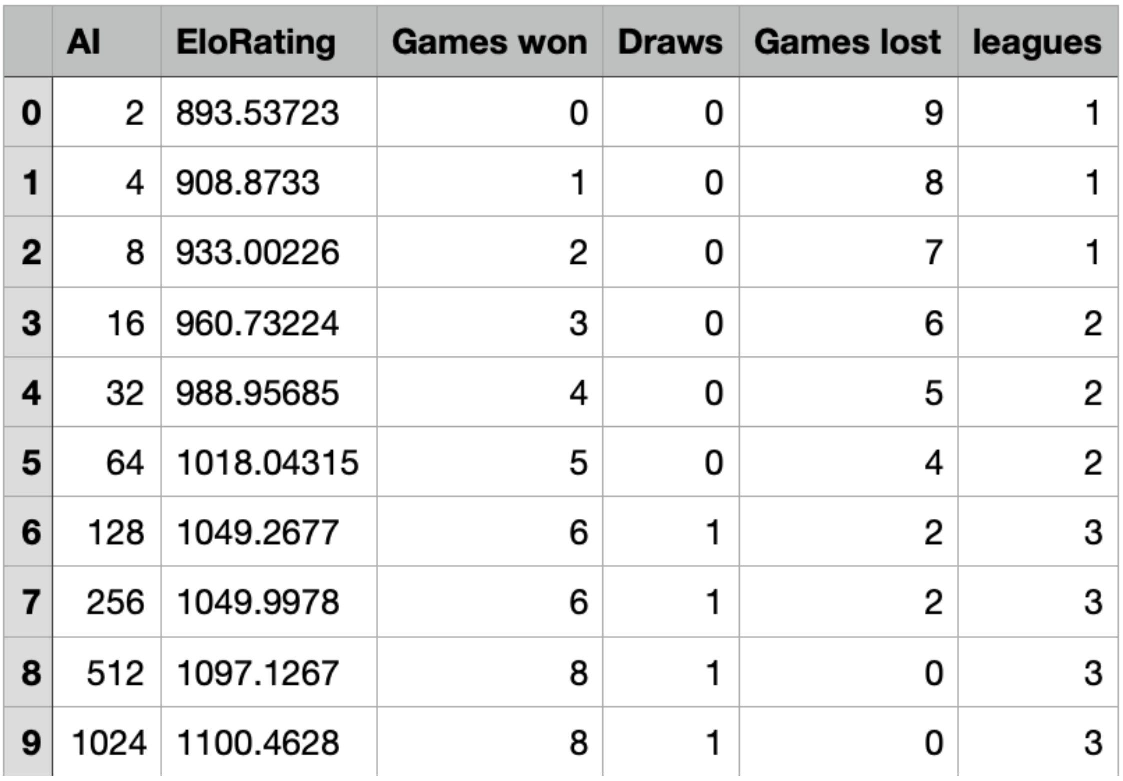
C.1 Boseog Gonu



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	893.94885	0	0	9	1
1	4	910.3871	1	0	8	1
2	8	945.0423	2	1	6	1
4	32	959.2919999999999	2	2	5	1
3	16	975.5060000000001	3	1	5	2
5	64	1016.7080000000001	5	0	4	2
6	128	1037.494	6	0	3	2
7	256	1062.9576	7	0	2	3
8	512	1085.6129	8	0	1	3
9	1024	1113.0511	9	0	0	3

Figure 14: Tournament outcome with leagues from Jenks natural breaks optimization for Boseog Gonu

C.2 Felli 0



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	893.53723	0	0	9	1
1	4	908.8733	1	0	8	1
2	8	933.00226	2	0	7	1
3	16	960.73224	3	0	6	2
4	32	988.95685	4	0	5	2
5	64	1018.04315	5	0	4	2
6	128	1049.2677	6	1	2	3
7	256	1049.9978	6	1	2	3
8	512	1097.1267	8	1	0	3
9	1024	1100.4628	8	1	0	3

Figure 15: Tournament outcome with leagues from Jenks natural breaks optimization for Felli 0

C.3 Felli 1

	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	893.9798599999999	0	0	9	1
1	4	909.51447	1	0	8	1
2	8	932.51447	2	0	7	1
3	16	961.1439	3	0	6	1
4	32	986.08636	4	0	5	2
5	64	1017.91364	5	0	4	2
6	128	1036.8561	6	0	3	2
8	512	1062.4855	7	0	2	3
7	256	1085.4855	8	0	1	3
9	1024	1114.0201	9	0	0	3

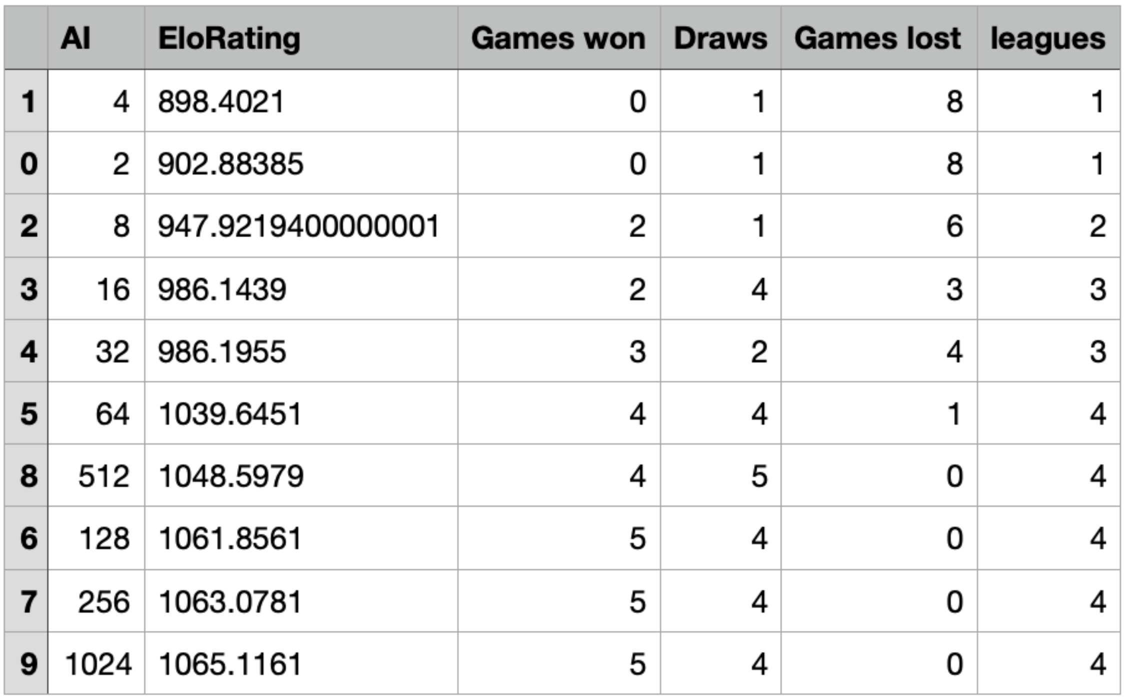
Figure 16: Tournament outcome with leagues from Jenks natural breaks optimization for Felli 1

C.4 Haretavl

	AI	EloRating	Games won	Draws	Games lost	leagues
1	4	897.43964	0	1	8	1
0	2	922.6261599999999	1	1	7	1
2	8	958.3969999999999	1	4	4	2
3	16	986.07745	2	4	3	2
4	32	1000.561	2	5	2	2
5	64	1028.5178	3	5	1	3
6	128	1038.9225	3	6	0	3
7	256	1039.603	3	6	0	3
8	512	1048.5603	4	5	0	3
9	1024	1078.3739	6	3	0	4

Figure 17: Tournament outcome with leagues from Jenks natural breaks optimization for Haretavl

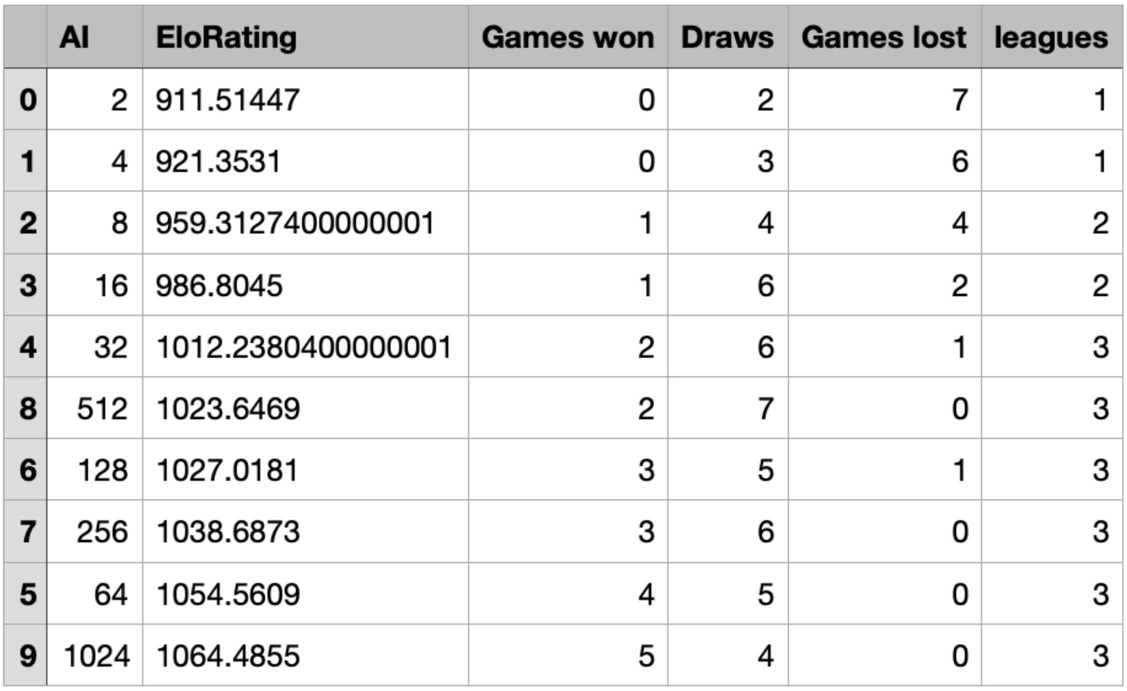
C.5 Hat Diviyan Keliya



	AI	EloRating	Games won	Draws	Games lost	leagues
1	4	898.4021	0	1	8	1
0	2	902.88385	0	1	8	1
2	8	947.9219400000001	2	1	6	2
3	16	986.1439	2	4	3	3
4	32	986.1955	3	2	4	3
5	64	1039.6451	4	4	1	4
8	512	1048.5979	4	5	0	4
6	128	1061.8561	5	4	0	4
7	256	1063.0781	5	4	0	4
9	1024	1065.1161	5	4	0	4

Figure 18: Tournament outcome with leagues from Jenks natural breaks optimization for Hat Diviyab Keliya

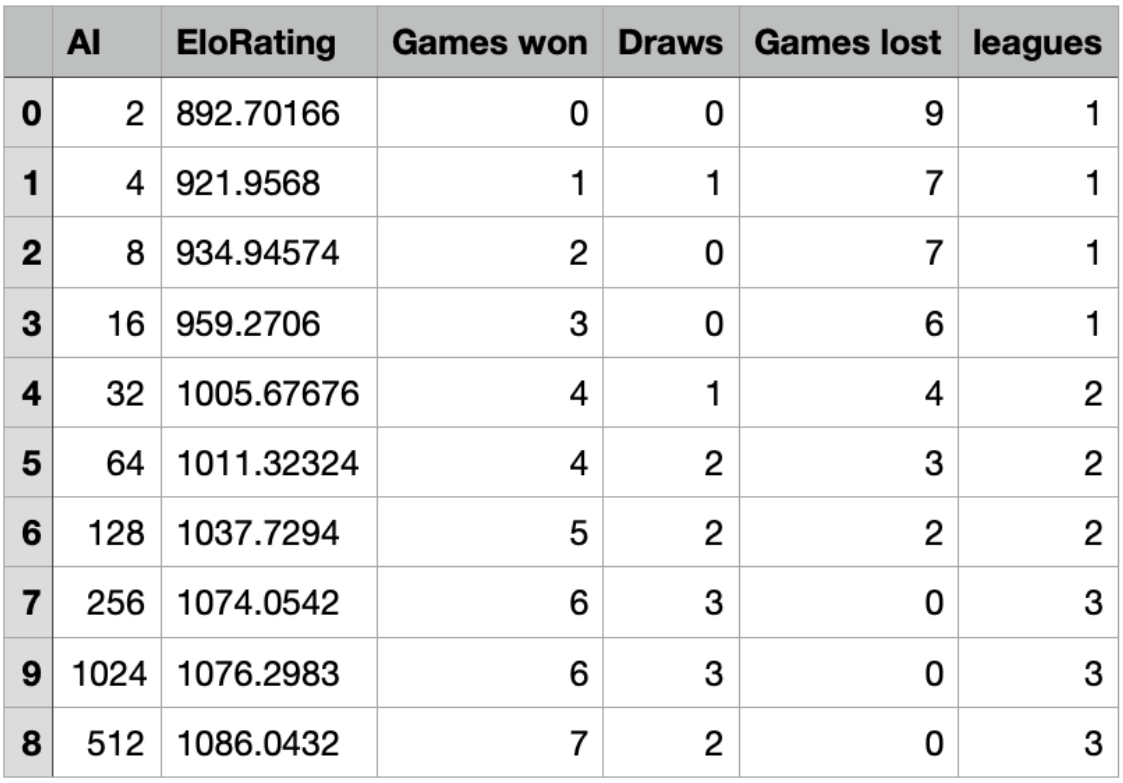
C.6 Ho-Bag Gonu



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	911.51447	0	2	7	1
1	4	921.3531	0	3	6	1
2	8	959.3127400000001	1	4	4	2
3	16	986.8045	1	6	2	2
4	32	1012.2380400000001	2	6	1	3
8	512	1023.6469	2	7	0	3
6	128	1027.0181	3	5	1	3
7	256	1038.6873	3	6	0	3
5	64	1054.5609	4	5	0	3
9	1024	1064.4855	5	4	0	3

Figure 19: Tournament outcome with leagues from Jenks natural breaks optimization for Ho-Bag Gonu

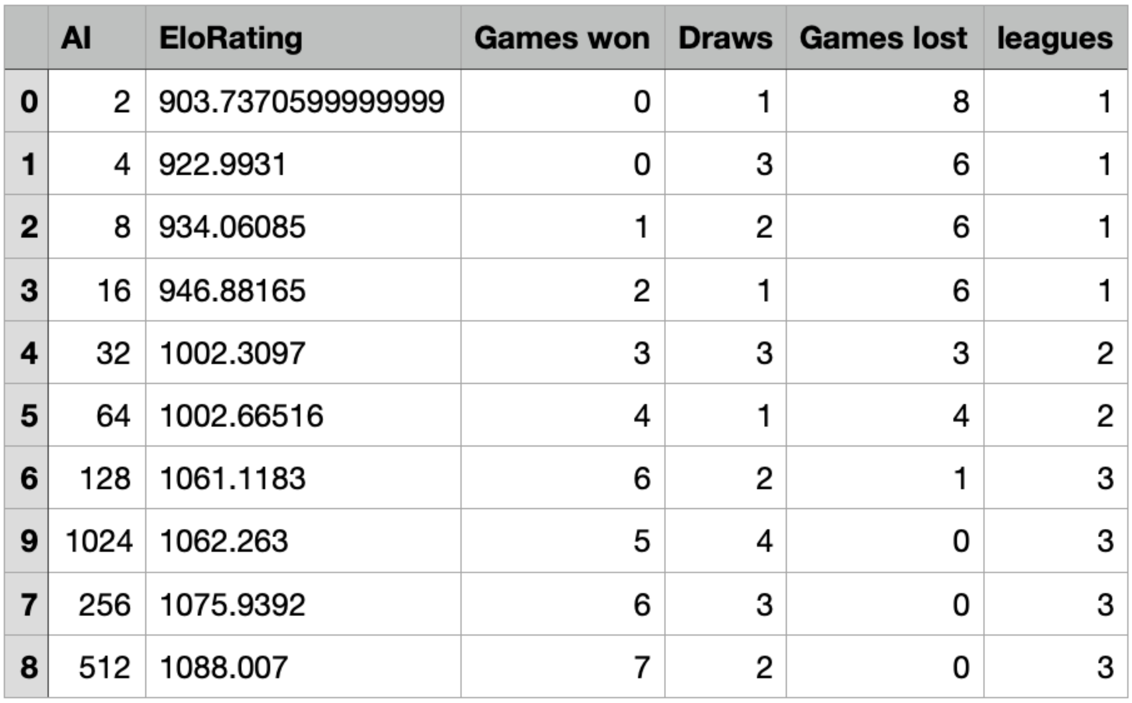
C.7 Ja-Jeon-Geo Gonu



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	892.70166	0	0	9	1
1	4	921.9568	1	1	7	1
2	8	934.94574	2	0	7	1
3	16	959.2706	3	0	6	1
4	32	1005.67676	4	1	4	2
5	64	1011.32324	4	2	3	2
6	128	1037.7294	5	2	2	2
7	256	1074.0542	6	3	0	3
9	1024	1076.2983	6	3	0	3
8	512	1086.0432	7	2	0	3

Figure 20: Tournament outcome with leagues from Jenks natural breaks optimization for Ja-Jeon-Geo Gonu

C.8 Kaooa



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	903.7370599999999	0	1	8	1
1	4	922.9931	0	3	6	1
2	8	934.06085	1	2	6	1
3	16	946.88165	2	1	6	1
4	32	1002.3097	3	3	3	2
5	64	1002.66516	4	1	4	2
6	128	1061.1183	6	2	1	3
9	1024	1062.263	5	4	0	3
7	256	1075.9392	6	3	0	3
8	512	1088.007	7	2	0	3

Figure 21: Tournament outcome with leagues from Jenks natural breaks optimization for Kaooa

C.9 Madelinette

	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	903.7739	0	1	8	1
1	4	908.4771	1	0	8	1
2	8	923.94574	1	1	7	1
3	16	960.73224	3	0	6	2
4	32	1000.67676	4	1	4	2
6	128	1049.2677	4	5	0	3
8	512	1060.5230000000001	5	4	0	3
7	256	1063.0542	5	4	0	3
9	1024	1063.2261	5	4	0	3
5	64	1066.3232	5	4	0	3

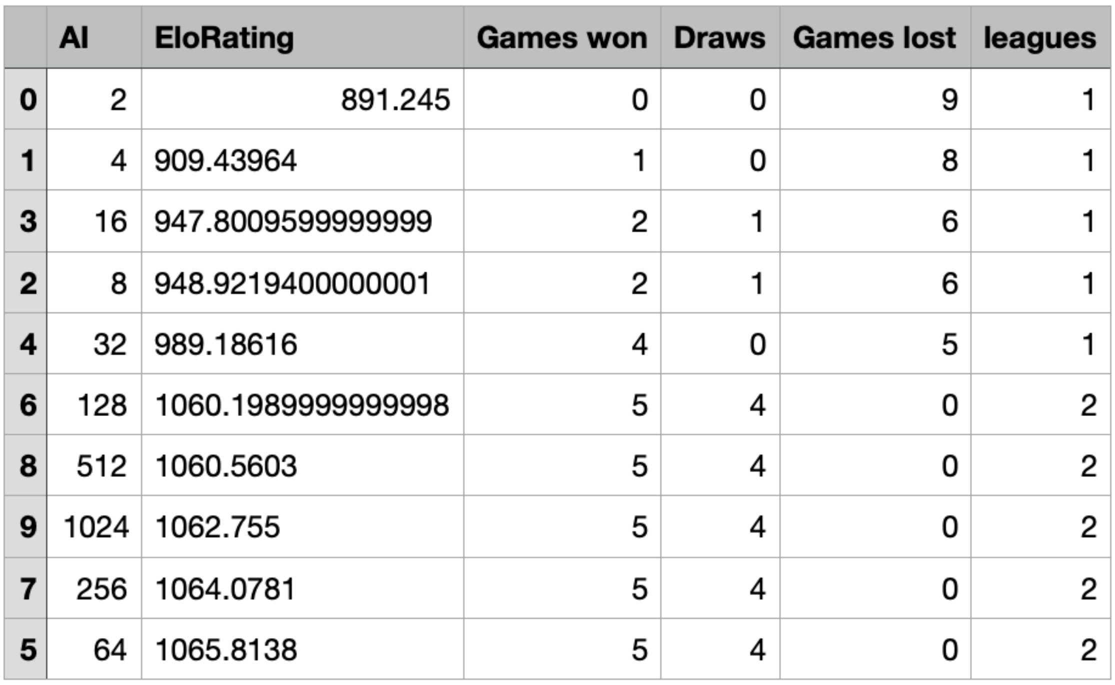
Figure 22: Tournament outcome with leagues from Jenks natural breaks optimization for Madelinette

C.10 Mu Torere 0

	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	886.4771	0	0	9	1
2	8	972.8045	1	5	3	1
1	4	984.6903	1	6	2	2
8	512	1011.0780599999999	1	8	0	3
7	256	1011.5649400000001	1	8	0	3
6	128	1012.0596	1	8	0	3
4	32	1013.1105	1	8	0	3
3	16	1028.3959	2	7	0	4
9	1024	1038.5230000000001	3	6	0	4
5	64	1041.1956	3	6	0	4

Figure 23: Tournament outcome with leagues from Jenks natural breaks optimization for Mu Torere 0

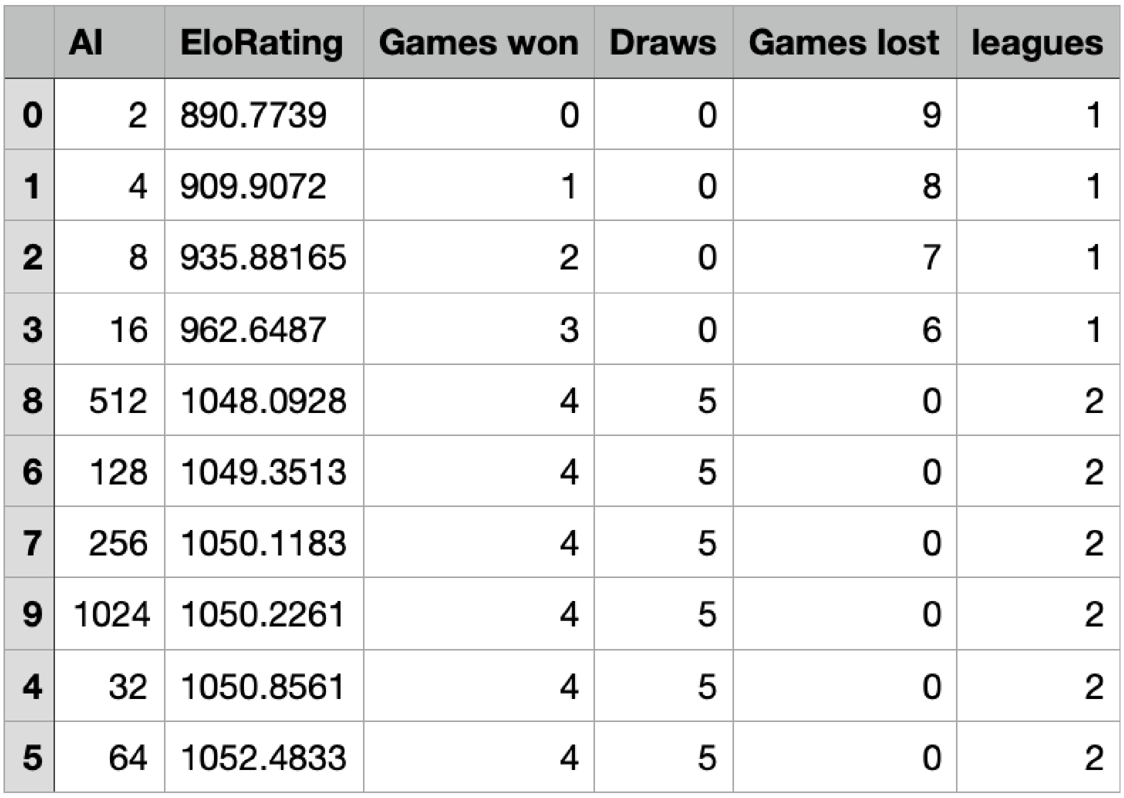
C.11 Mu Torere 1



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	891.245	0	0	9	1
1	4	909.43964	1	0	8	1
3	16	947.8009599999999	2	1	6	1
2	8	948.9219400000001	2	1	6	1
4	32	989.18616	4	0	5	1
6	128	1060.1989999999998	5	4	0	2
8	512	1060.5603	5	4	0	2
9	1024	1062.755	5	4	0	2
7	256	1064.0781	5	4	0	2
5	64	1065.8138	5	4	0	2

Figure 24: Tournament outcome with leagues from Jenks natural breaks optimization for Mu Torere 1

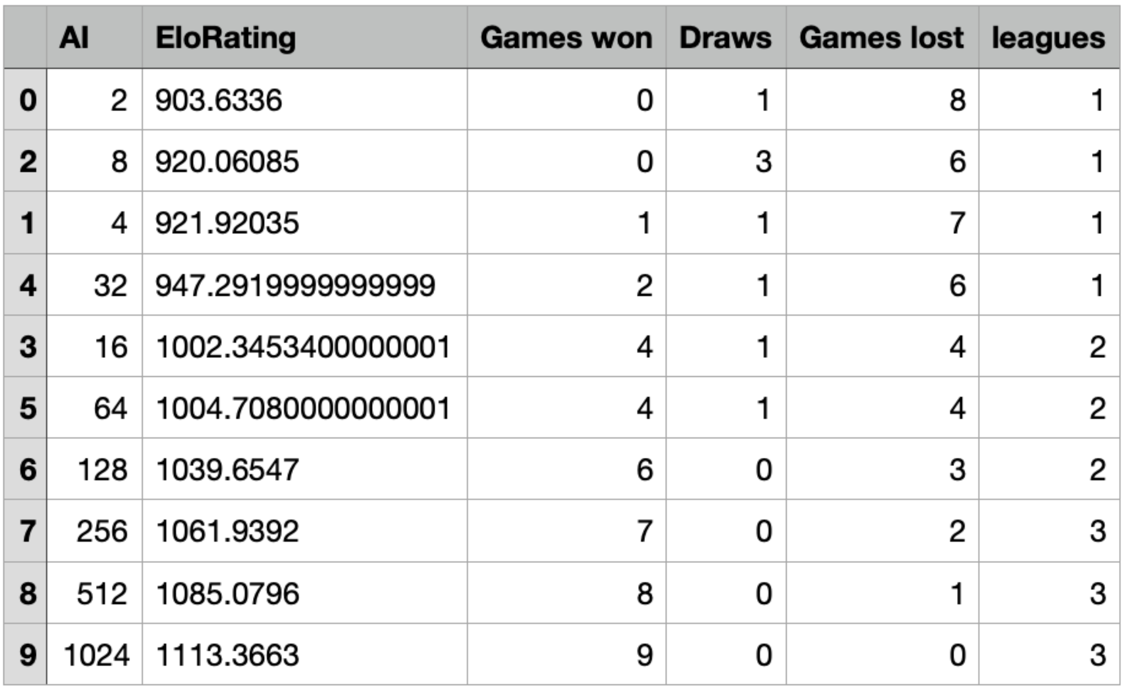
C.12 Pong Hau K'I



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	890.7739	0	0	9	1
1	4	909.9072	1	0	8	1
2	8	935.88165	2	0	7	1
3	16	962.6487	3	0	6	1
8	512	1048.0928	4	5	0	2
6	128	1049.3513	4	5	0	2
7	256	1050.1183	4	5	0	2
9	1024	1050.2261	4	5	0	2
4	32	1050.8561	4	5	0	2
5	64	1052.4833	4	5	0	2

Figure 25: Tournament outcome with leagues from Jenks natural breaks optimization for Pong Hau K'I

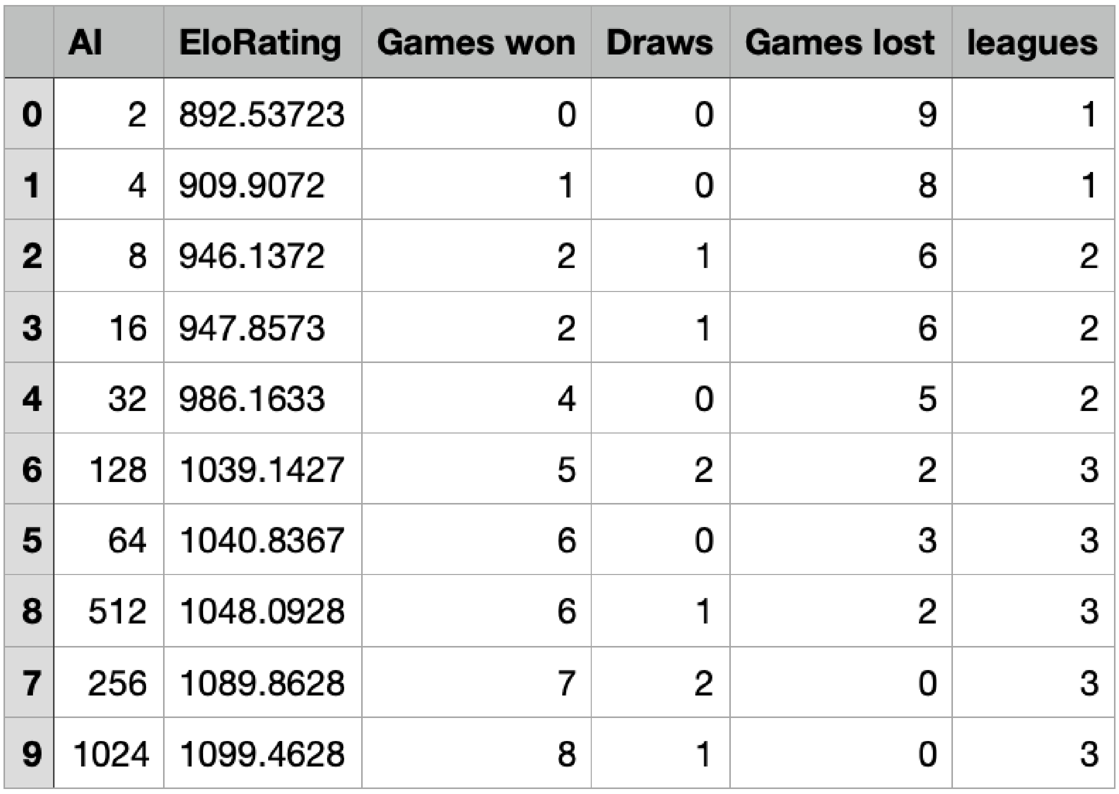
C.13 Round Merels



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	903.6336	0	1	8	1
2	8	920.06085	0	3	6	1
1	4	921.92035	1	1	7	1
4	32	947.2919999999999	2	1	6	1
3	16	1002.3453400000001	4	1	4	2
5	64	1004.7080000000001	4	1	4	2
6	128	1039.6547	6	0	3	2
7	256	1061.9392	7	0	2	3
8	512	1085.0796	8	0	1	3
9	1024	1113.3663	9	0	0	3

Figure 26: Tournament outcome with leagues from Jenks natural breaks optimization for Round Merels

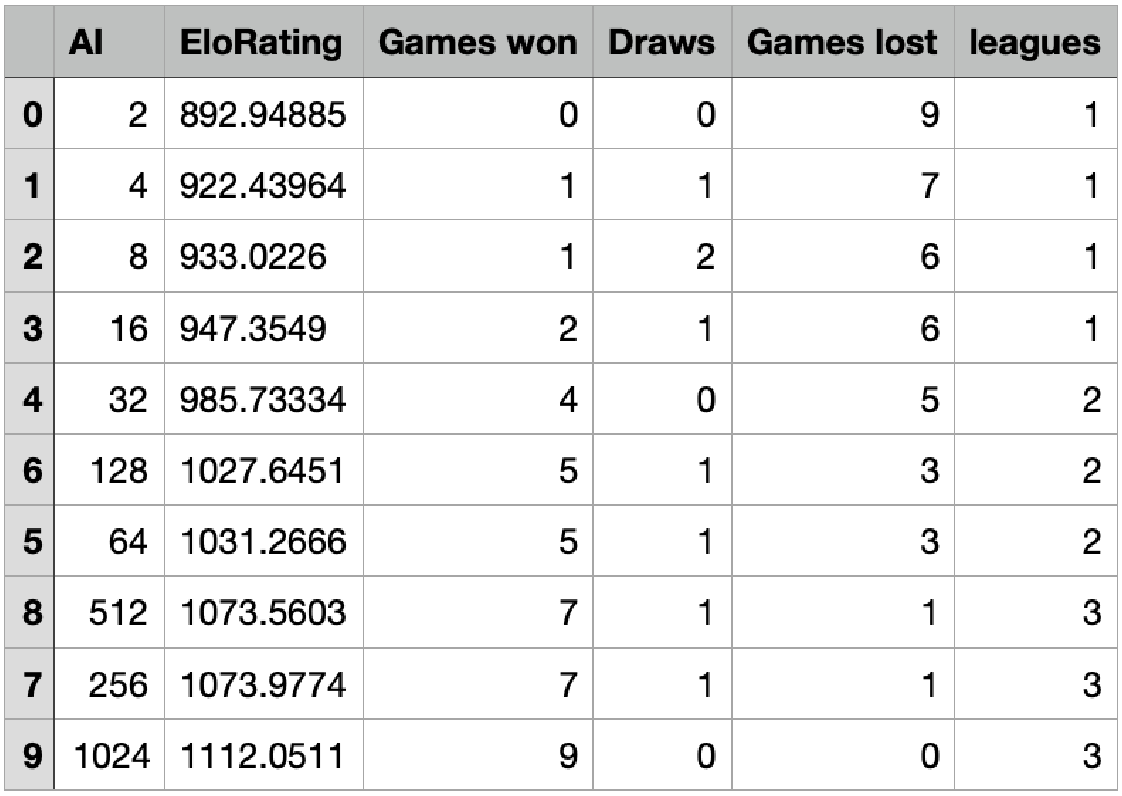
C.14 Symbol S.2-1



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	892.53723	0	0	9	1
1	4	909.9072	1	0	8	1
2	8	946.1372	2	1	6	2
3	16	947.8573	2	1	6	2
4	32	986.1633	4	0	5	2
6	128	1039.1427	5	2	2	3
5	64	1040.8367	6	0	3	3
8	512	1048.0928	6	1	2	3
7	256	1089.8628	7	2	0	3
9	1024	1099.4628	8	1	0	3

Figure 27: Tournament outcome with leagues from Jenks natural breaks optimization for Symbol S.2-1

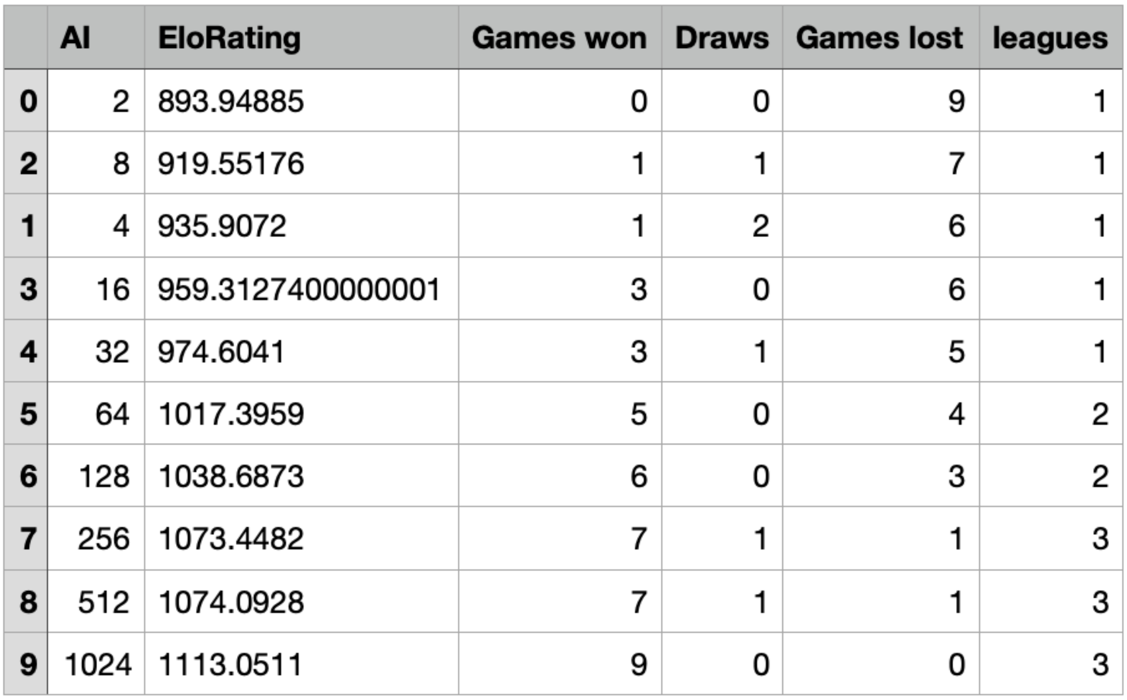
C.15 Tant Fant



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	892.94885	0	0	9	1
1	4	922.43964	1	1	7	1
2	8	933.0226	1	2	6	1
3	16	947.3549	2	1	6	1
4	32	985.73334	4	0	5	2
6	128	1027.6451	5	1	3	2
5	64	1031.2666	5	1	3	2
8	512	1073.5603	7	1	1	3
7	256	1073.9774	7	1	1	3
9	1024	1112.0511	9	0	0	3

Figure 28: Tournament outcome with leagues from Jenks natural breaks optimization for Tant Fant

C.16 Tic-Tac-Chess



	AI	EloRating	Games won	Draws	Games lost	leagues
0	2	893.94885	0	0	9	1
2	8	919.55176	1	1	7	1
1	4	935.9072	1	2	6	1
3	16	959.3127400000001	3	0	6	1
4	32	974.6041	3	1	5	1
5	64	1017.3959	5	0	4	2
6	128	1038.6873	6	0	3	2
7	256	1073.4482	7	1	1	3
8	512	1074.0928	7	1	1	3
9	1024	1113.0511	9	0	0	3

Figure 29: Tournament outcome with leagues from Jenks natural breaks optimization for Tic-Tac-Chess