CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Modeling Players Personality in General Game Playing

Master's thesis in Computer Science - Algorithms, Languages and Logic

Stefania Crotti

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2018

Master's thesis 2018

Modeling Players Personality in General Game Playing

Stefania Crotti

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden

Modeling Players Personality in General Game Playing
Stefania Crotti

© Stefania Crotti, 2018.

Supervisors: Christos Dimitrakakis,
Department of Computer Science and Engineering,
Chalmers University of Technology

Stephan Schiffel,
Department of Computer Science, Reykjavik University

Examiner: Staffan Björk,
Department of Computer Science and Engineering,
Chalmers University of Technology

Master's Thesis 2018
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2018

Modeling Players Personality in General Game Playing
Stefania Crotti
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Artificial agents' skills need to become more relatable to humans', and one approach to solve this problem would be to associate a personality to the agents. When games are used as a framework, General Game Playing (GGP) provides an unbiased environment where new games are played without any prior knowledge of the rules, and without applying any game-dependent heuristic. This thesis is expecting to infer preferences from human played games, depending on the personality the players recognised themselves in. The artificial player is aided with a Monte Carlo Tree Search algorithm with tunable parameters, which associate evaluation values to each move, consequently selecting the next state. The optimal set of parameters to fit the human gameplay is found with the subsidy of a Genetic Algorithm where individuals are represented as sets of parameters themselves. This approach is backed up with a Bayesian probability model, and, finally, the outputted sets of parameters are evaluated to determine if the artificial gamer has indeed learnt to behave accordingly to a certain personality. After an extensive research on personality models has been carried out to find a suitable one for the amount of data expected to be collected, the choice has fallen over the Hippocrates'-Galen Four Temperaments. The results however hint to the conclusion that a different model might have been easier to be fit. Although the results are not astonishing, this thesis can be considered as a first stepping stone into personality model fitting through Monte Carlo Tree Search parameters tuning.

Keywords: General Game Playing, Monte Carlo Tree Search, Genetic Algorithm, Personality Mapping, Bayesian Modeling.

Acknowledgements

All my gratitude goes to my supervisors, Stephan Schiffel and Christos Dimitrakakis, for proposing this project, getting me started on this icelandic experience, and making it possible. Thanks to both for giving me the opportunity of working on this thesis, and for all the support that has been given me throughout the last months. I would also like to thank Staffan Björk, my examiner, for putting up with this work.

Thank you to all the people that participated (more or less willingly) in the data collection, your help has been highly appreciated.

Grazie infinite to my family, who encouraged me to keep going.
Another big thank you needs to go to the people that have stuck around, being them in Italy, Sweden, Iceland, or anywhere else in the world. I hold you all dearly. Honorable mention to few people without whom I wouldn't have made it this far, and specifically had to put up with me in my swedish years: Cyrill, Divia, Fathi, Rachele, Ragnar, Tanya and Yuesong. I love you all.

Lastly, none of this would have been possible without such an amazing person by my side. Hjörtur, this thesis is for you.

Grazie mille. Tack så mycket. Takk kærlega.

Stefania Crotti
Reykjavík and Göteborg
October 2017, October 2018

Contents

List of Figures **xi**

List of Tables **xv**

List of Abbreviations **xvii**

1 Introduction **1**
1.1 Goal of the project . . . 1
1.2 Structure of the thesis . . . 2

2 Background **3**
2.1 Discrepancies between Players . . . 3
2.2 Personality Models . . . 4
2.2.1 Bartle's types . . . 4
2.2.2 Stewart's Unified Model . . . 5
2.2.3 Quantic Foundry . . . 6
2.2.4 The Cardboard Republic Archetypes . . . 6
2.2.5 Timmy, Johnny and Spike . . . 6
2.2.6 BrainHex . . . 7
2.2.7 Big Five Personality Traits . . . 8
2.2.8 Galen-Hippocrates' Four Temperaments . . . 8
2.2.9 Myer-Briggs . . . 9
2.2.10 Keirsey's Temperaments . . . 10
2.3 General Game Playing . . . 10
2.3.1 Games Structure . . . 10
2.4 Monte Carlo Tree Search Algorithm . . . 12
2.4.1 Monte Carlo Tree Search Enhancements . . . 13
2.5 Machine Learning . . . 16
2.5.1 Decision Theory and Preference Elicitation . . . 17
2.5.2 Markov Decision Process . . . 17
2.6 Genetic Algorithm . . . 18
2.6.1 General Structure . . . 19
2.6.2 Selection . . . 19
2.6.3 Crossover . . . 20
2.6.4 Mutation . . . 21
2.6.5 Termination . . . 21

3 Methods **23**
3.1 Problem Definition . . . 23
3.2 Personality Model . . . 24
3.3 Data Collection . . . 24
3.3.1 Selected Games . . . 25
3.4 Base Project . . . 27
3.5 Monte Carlo Tree Search . . . 28
3.5.1 Parameters . . . 29
3.6 Probability Model . . . 30
3.7 Genetic Algorithm . . . 32
3.7.1 Parameters Intervals . . . 34
3.8 Evaluation . . . 35

4 Results **37**
4.1 Comparison of Personality Models . . . 37
4.2 Data Collection . . . 39
4.3 Genetic Algorithm . . . 42
4.3.1 Results Overview . . . 42
4.3.2 Fittest Individuals . . . 45
4.4 Results Game Play . . . 46
4.4.1 Skirmish Games with Fittest Individuals . . . 47
4.4.2 Random Games with Fittest Individuals . . . 47
4.4.3 Skirmish Games with Random Individuals . . . 50
4.4.4 Random Games with Random Individuals . . . 51
4.5 Limitations and Improvements . . . 52
4.6 Evaluation Results . . . 52

5 Conclusion **55**

A Preliminaries **I**
A.1 Data Structures . . . I
A.1.1 Graphs . . . I
A.1.2 Directed Graphs . . . I
A.1.3 Paths and cycles . . . I
A.1.4 Connected Graphs . . . I
A.1.5 Trees . . . II
A.1.6 State Space . . . II
A.2 Finite State-Machines . . . II
A.2.1 Deterministic Finite Automata (DFA) . . . II
A.2.2 Non-Deterministic Finite Automata (NFA) . . . III
A.3 Probability . . . III
A.3.1 Elementary rules . . . III

B Collection of Graphs **V**
B.1 Single Individuals . . . VI
B.2 Evaluation . . . XXI

List of Figures

2.1 Bartle's taxonomy graph (adapted from [16]). 5
2.2 Example of Board Game Motivation Profile from Quantic Foundry[41]. 5
2.3 Example of one of the Cardboard Republic Archetypes (from [7]). . . 7
2.4 On the left, BrainHex classes, on the right the corresponding parts of the brain (red text) and chemicals (green text) involved. Figure from [79]. 7
2.5 Diagram of a classical GGP environment (adapted from [48]). 12
2.6 Iteration of MCTS (graph adapted from [22, 27, 116]). 13
2.7 Example of Single Point crossover between two individuals. 20

3.1 A game of Checkers. 25
3.2 A game of Nine Boards Tic-Tac-Toe. 26
3.3 A game of Connect Four. 26
3.4 A game of Skirmish. 27
3.5 A screen-shot of the Virtual Agent while playing a game of Checkers. 28
3.6 Bayesian Network representation. 30

4.1 Correlation between the Four Temperaments and Big 5 factors. . . . 39
4.2 The four graphs representing the correlation between each of the Four Temperaments and the factors that define the Big 5 model. 40
(a) Choleric graph 40
(b) Melancholic graph 40
(c) Sanguine graph 40
(d) Phlegmatic graph 40
4.3 The four graphs represent the running times (in ms) for all the individuals in each generation, on y and x axis respectively. 43
(a) Choleric graph 43
(b) Melancholic graph 43
(c) Sanguine graph 43
(d) Phlegmatic graph 43
4.4 Correlation between specific Charge Default (x-axis) and fitness (y-axis). 43
4.5 Charge Default gene for Phlegmatic personality, where the generations are on the x-axis, and Charge Default values are on the y-axis. . 44
4.6 Nice Threshold gene's results of the III run, the generations are on the x-axis, while the values for Nice Threshold are on the y-axis. . . . 44

4.7 Random Error gene's results of the III run, where the x-axis represents the number of generations, and the y-axis the value of Random Error. 45
4.8 Fitness evolution (y-axis) throughout the generations (x-axis), for each personality. 47
4.9 Evaluation of Skirmish matches, where the moves are represented on the x-axis, and their QValue on the y-axis. 48
4.10 Evaluation of Nine Boards Tic-Tac-Toe for Sanguine personality, where the moves are represented on the x-axis, and their QValue on the y-axis. 49
4.11 Evaluation of AAR with Fittest Individuals (percent). 49
4.13 Actions Agreement Ratio for Skirmish games with random individuals, divided by personality (percent). 51
4.14 Evaluation of AAR with Random Individuals (percentage). 51
4.12 Evaluation of the Skirmish games with random individuals, where the moves are represented on the x-axis, and their QValue on the y-axis. . 53

A.1 Example of simple DFA. II
A.2 Example of simple NFA.[1] III

B.1 Aggressiveness gene, with generations on x-axis, and value of each individual on y-axis. VI
B.2 Charge Default gene, with generations on x-axis, and value of each individual on y-axis. VII
B.3 Charge Depth gene, with generations on x-axis, and value of each individual on y-axis. VIII
B.4 Charge Discount gene, with generations on x-axis, and value of each individual on y-axis. IX
B.5 Defensiveness gene, with generations on x-axis, and value of each individual on y-axis. X
B.6 Epsilon gene, with generations on x-axis, and value of each individual on y-axis. XI
B.7 Exploration Factor gene, with generations on x-axis, and value of each individual on y-axis. XII
B.8 Grave gene, with generations on x-axis, and value of each individual on y-axis. XIII
B.9 Horizon gene, with generations on x-axis, and value of each individual on y-axis. XIV
B.10 Nice Threshold gene, with generations on x-axis, and value of each individual on y-axis. XV
B.11 Random Error gene, with generations on x-axis, and value of each individual on y-axis. XVI
B.12 Rave gene, with generations on x-axis, and value of each individual on y-axis. XVII
B.13 Tree Discount gene, with generations on x-axis, and value of each individual on y-axis. XVIII
B.14 Correlation between specific genes (values of which are on x-axis) and fitness (y-axis) (continued on the next page). XIX

B.15 Correlation between specific genes (values of which are on x-axis) and fitness (y-axis). XX
B.16 Evaluation of the Skirmish games with fittest individuals, with moves on x-axis and QValues on y-axis. XXI
B.17 Evaluation of Nine Boards Tic-Tac-Toe for Sanguine Personality, with moves on x-axis and QValues on y-axis. XXII
B.18 Evaluation of Nine Boards Tic-Tac-Toe for Phlegmatic Personality, with moves on x-axis and QValues on y-axis. XXIII
B.19 Evaluation of Nine Boards Tic-Tac-Toe for Melancholic Personality, with moves on x-axis and QValues on y-axis. XXIV
B.20 Evaluation of Nine Boards Tic-Tac-Toe for Choleric Personality, with moves on x-axis and QValues on y-axis. XXV
B.21 Evaluation of Connect Four for Sanguine Personality, with moves on x-axis and QValues on y-axis. XXVI
B.22 Evaluation of Connect Four for Phlegmatic Personality, with moves on x-axis and QValues on y-axis. XXVII
B.23 Evaluation of Connect Four for Melancholic Personality, with moves on x-axis and QValues on y-axis. XXVIII
B.24 Evaluation of Connect Four for Choleric Personality, with moves on x-axis and QValues on y-axis. XXIX
B.25 Evaluation of Checkers Games, with moves on x-axis and QValues on y-axis. XXX
B.26 Evaluation of Nine Boards Tic-Tac-Toe for Random Individual A, with moves on x-axis and QValues on y-axis. XXXI
B.27 Evaluation of Nine Boards Tic-Tac-Toe for Random Individual B, with moves on x-axis and QValues on y-axis. XXXII
B.28 Evaluation of Nine Boards Tic-Tac-Toe for Random Individual C, with moves on x-axis and QValues on y-axis. XXXIII
B.29 Evaluation of Nine Boards Tic-Tac-Toe for Random Individual D, with moves on x-axis and QValues on y-axis. XXXIV
B.30 Evaluation of Connect Four for Random Individual A, with moves on x-axis and QValues on y-axis. XXXV
B.31 Evaluation of Connect Four for Random Individual B, with moves on x-axis and QValues on y-axis. XXXVI
B.32 Evaluation of Connect Four for Random Individual C, with moves on x-axis and QValues on y-axis. XXXVII
B.33 Evaluation of Connect Four for Random Individual D, with moves on x-axis and QValues on y-axis. XXXVIII
B.34 Evaluation of Checkers games for Random Individuals, with moves on x-axis and QValues on y-axis. XXXIX

List of Tables

2.1 Associated gaming skills to each of the four temperaments, in a scale from 1 to 4, where 1 means it is a strong suit for the type. 9
2.2 Keirsey's types and associated Myer-Brigg's types. 10

4.1 Comparison of Galen-Hippocrates' Four Temperaments with Keirsey's Four Temperaments and Myer-Briggs personality types, as from [61]. 37
4.2 Correlation of Big 5 factors with Myer-Briggs (data from [71]). 38
4.3 Probability of each Myer-Briggs preferences given the Galen-Hippocrates' type. 38
4.4 Extension of table 4.2, with clarification of the preferences scale. . . . 38
4.5 Correlation between the Four Temperaments and Big 5. 38
4.6 Correlation of Big 5 factors with Myer-Briggs (table 4.5) normalised to 2. 39
4.7 Summary of the games collected and the personality the users self-associated to themselves. 41
4.8 Summary of the games collected and the perceived personality the users associated to their opponents. 41
4.9 Summary of the games where the users recognized successfully the opponent's personality. 41
4.10 Fittest individuals picked for evaluation. 46
4.11 Last moves in the Choleric-Skirmish game. 49
4.12 Random individuals picked for evaluation. 50

List of Abbreviations

AAR	Action Agreement Ratio.
ABC	Approximate Bayesian Computation algorithm.
AI	Artificial Intelligence.
AMAF	All Moves As First.
ANN	Artificial Neural Network.
DFA	Deterministic Finite Automata.
EA	Evolutionary Algorithm.
GA	Genetic Algorithm.
GDL	Game Description Language.
GGP	General Game Playing.
GRAVE	General Rapid Action Value Estimation.
MAST	Move-Average Sampling Technique.
MCTS	Monte Carlo Tree Search.
MDP	Markov Decision Process.
MEU	Maximum Expected Utility.
ML	Machine Learning.
MUD	Multi-User Dungeon.
NBTTT	Nine Boards Tic-Tac-Toe.
NFA	Non-Deterministic Finite Automata.
NPC	Non-Player Character.
RAVE	Rapid Action Value Estimation.
UCB	Upper Confidence Bound.
UCT	Upper Confidence Bounds for Trees.

1
Introduction

The early developers of Artificial Intelligence (AI) were fascinated by having smart agents that, when facing a problem, could find better solutions than humans, in a possibly shorter time. However, when this approach started to be applied to gaming agents, it eventually led to frustrating opponents[102], due to it being too-challenging of a game, or a too-easy and boring one. At the same time, games developers found themselves in need of offering more realistic content, with interesting and authentic-looking Non-Player Character (NPC)[109], and automated and reliable game testers[56]. To solve this problem, and make gaming a more enjoyable experience, research has started to develop interest in Artificial Intelligence behaviour, trying to mimic humans' game playing, mostly through Imitation Learning - the meaning of which is straightforward. To quote Gorman[51], Imitation Learning is "the acquisition of skills or behaviors through examination of a demonstrator's execution of a given task". Few different strategies have been applied so far, often associated to Artificial Neural Network (ANN), as in [102, 30, 118, 103, 74], but also Reinforcement Learning algorithms[104], or Genetic Algorithm (GA)[70], among others.
Monte Carlo Tree Search (MCTS) has been the object of focus in the process to "humanize" artificial agents, applied, for example, on Spades[33], play testing[56], or video games[62]. However, we find that the research could be exploited further with this thesis project, which merges several different tools to model *procedural personæ*, defined by Holmgård in [55] as: "Game playing agents that codify player decision making styles, either from the designer's holistic representation of these or from observations of players collected directly from the game". In this thesis, we will use the terms personality and procedural persona as interchangeable.
This thesis work has mostly been carried out at Reykjavík University, on the basis of an existing project that was conjoining virtual reality and general game playing[52]. The agent developed was a graphical representation of a version of CadiaPlayer[6], a general game player implemented by the Reykjavík University team that has three times won the international General Game Playing (GGP) competition[46].

1.1 Goal of the project

This project's goal is to deduce if personality is a feature that can be inferred from game playing. Aiming to do so, information about human played games have been collected and used to train a Genetic Algorithm over a Monte Carlo Tree Search

algorithm applied to the data gathered. The search algorithm would be returning a move selection for the current state of the game, and the Genetic Algorithm would return the set of parameters that would make the search match the same moves as the human players. The outputted parameters will then be evaluated over a new set of human-played matches, confirming or disproving the hypothesis behind this thesis.
In the case that the results will be satisfactory, it can lead to the development of more targeted agents, not necessarily only in the game playing field.

1.2 Structure of the thesis

The structure of this thesis is as follows: firstly, an introductory chapter with an overview on the relation between games and artificial intelligence, in addition of a general description of the problem. The following chapter is a survey on the theoretical notions necessary to understand the project and its implementation, starting with a review of the personality models we came across; the definition of General Game Playing; Monte Carlo Tree Search, as the algorithm we adopt in our General Game Player; an overview on Machine Learning; and, lastly, the definition of Genetic Algorithms. The third chapter then describes the methodologies applied at implementation time, justifying the choices made during the work process. Sequentially, we redefine the problem; describe the data collection process and the base project used as a starting point; define the Monte Carlo Tree Search variations adopted; formalize the probability model; specify the Genetic Algorithm's options selected; and outline the evaluation process adopted. In the fourth chapter we include the results of each step, followed by an explicative discussion over the outcomes at various stages of the project: the data collection, the output of the genetic algorithm, and, finally, the results of the game playing evaluation. Lastly, chapter five includes the conclusions drawn during this thesis work, together with a brief overview on how this project could be expanded in the future.

2
Background

Human behaviour is commonly associated to a personality that is generally unique for each specific person. However, over the years, Psychology has been trying to find common patterns in behaviours, mapping common traits into personality models. It is believed that those common personality traits would lead the players to have similar gaming styles, that would be inferred to our artificial gamer through tuning parameters plugged into a Monte Carlo Tree Search algorithm. This chapter will lay out the background requirements needed to understand and implement this thesis project. Firstly, a brief introduction on the differences that can be expected between agents' and human game playing, followed by a survey on personality models in section 2.2. In section 2.3 is an overview of General Game Playing, which is the general framework in this thesis. The following section, 2.4, is about Monte Carlo Tree Search (MCTS), as it is the algorithm used by the General Game Player agent, to decide which moves will be chosen at each state. Each set of tuning parameters specific to the MCTS algorithm can change the outcome of the search. In section 2.5 will be given some machine learning background definitions. A Genetic Algorithm is used upon the collected dataset, meant to find which sets of parameters are associable to different personalities, and will be described in section 2.6.

2.1 Discrepancies between Players

Humans' game playing considers some features that might sound obvious to the reader, but are not to the artificial player. The human player, for example, has common sense, which can influence the learning process of new tasks[58]. As a smart algorithm might figure out that some moves are useless for the agent, or even damaging for the outcome of the game, those choices still need to be evaluated a number of times before being discarded. However, this might be more evident with video games - as better exploited by Freed et al. in [42] -, than with classic board games, which are the ones considered in this project. In [62], Khalifa et al include along with those features the "recklessness" that seems to label human game playing, while algorithms seem to adopt less risky strategies, and the "thoughtfulness" given by the pauses that humans would take while playing, contrarily to the different artificial agents responsiveness.
Modeling players to behave in a human-like style has been an object of interest on multiple occasions, as literature shows, for example, in [70, 59, 108], however, the focus has been mostly on the video game environment. The reason behind this

could be that the physical reactions are more defined, and can be better associated to a certain pattern behaviour-situation, while strategies tend to change frequently in a short period of time, depending on the state of the game, mood of the player, and environment, amongst others[59]. In [101], Simon evaluates how grandmasters in Chess recognize patterns in the game, having memorised small parts of game trees. On a smaller scale, every human player recognizes patterns while playing, or is able to associate knowledge coming from other games. This is confirmed as one of the staple-points of Decision Theory: humans make decisions depending on subconscious reasons that might seem like they have no rational base[60]. We would like to point out that the General Game Player used in this project does not support this feature, and forgets everything it has learnt right after the game has ended. This limitation might affect our player's behaviour, reducing the similarities with that of the humans'. However, while playing, the search runs a 1000 simulation for each move, which is most likely more than the average human player would do, supplying our agent with some knowledge of the game. At this point, we find the difference in knowledge between a human with some gaming experience, and an agent running 1000 simulations, irrelevant of the scope.

2.2 Personality Models

In order to have a more human-like behaviour, it has been decided to provide the artificial agent with a personality. There are numerous models in Psychology to characterize human behaviours, but most of them resulted in being too complicated for our scope, being either too distinctive, or requiring a deeper analysis of our human players. While researching for the best fit for this project, we have focused on some game-related models, and on some amongst the most simplistic ones from classic Psychology. Here is an overview of the models came across during the development of this project, together with the reasons why they have or have not been adopted. We find useful to have this overview, in order to give the reader a better understanding of what is the state of the art, and the reasons behind the choices made in this project, regarding which model to adopt. The chosen personality model, which selection is discussed in section 3.2, will then be used to evaluate our data, trying to extrapolate the parameters correspondent to the common move choices for people belonging to the same personality group.

2.2.1 Bartle's types

Bartle's taxonomy has become the most famous personality model in the gaming environment, it divides 4 different types of people: Killers, Achievers, Socialisers and Explorers[16]. The taxonomy was created to suit Multi-User Dungeon (MUD) games, specifically to help game designers to understand which kind of modifications would have to be applied in order to get the attention of some specific players. As it is clear in figure 2.1, players that are of the Killer type will prefer games where actions involve other players, or NPCs (Non-Player Characters), while Achievers would prefer acting on the world and the environment. Socialisers and Explorer

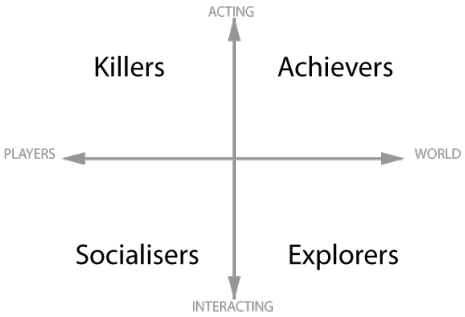


Figure 2.1: Bartle's taxonomy graph (adapted from [16]).

will favour interactions with other players or with the world, for example solving quests. Even though it seems to be the most used model in game designing and game analysis, it does not seem to suit our purposes. This taxonomy fits the role playing scenario, but not the classical board games one. Agreeing with Bartle himself: it is not always the right tool for the job[17].

2.2.2 Stewart's Unified Model

In his article, Stewart analyzes a few different models -together with Bartle's- and matches all of them into a unified model[105]. This shows how similar those models are, and that they could be almost interchangeable. Since Bartle's model has been excluded from the list of the ones that are suitable for this project, the ones mentioned in Stewart's article have been excluded too. However, a brief description of Keirsey's model will be given further on in the chapter, as it will be useful for this thesis as a middle step for models adaptation.

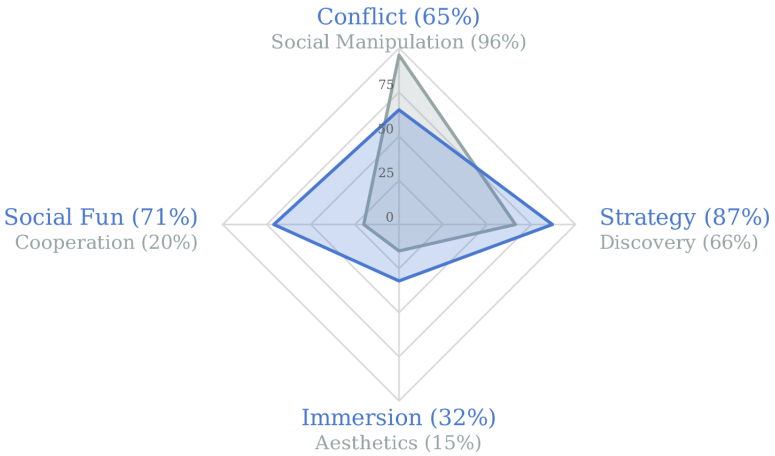


Figure 2.2: Example of Board Game Motivation Profile from Quantic Foundry[41].

2.2.3 Quantic Foundry

Quantic Foundry is a consulting company that works around psychology of gaming. They developed the Quantic Lab[81], which is a survey project aimed at understanding gamers preferences. They developed tools to test which gaming motivations are behind every player, and on their website there are two tests available, one for "Gamer Motivation Profile"[117] and one for "Board Gamer Motivation Profile". As the first one is focused on videogames, it is outside of our scope.
In figure 2.2 can be seen an example of what are the results of the test. The profile gives a percentile rank across various motivations, as Conflict, Social Fun, Immersion and Strategy -and only secondly across Social Manipulation, Discovery, Aesthetics and Cooperation. Even though the model seems to be pretty well-defined, it seems too complicated for this thesis' goal. It probably would be more appropriate if a bigger amount of data to work on would be available, so to force preferences over different types of games over the agent (i.e. the player that prefers immersive settings would not focus as well on classic board games, playing more sloppily that he would do if playing other games). Until the data is big enough, though, we prefer a simpler model.

2.2.4 The Cardboard Republic Archetypes

The Cardboard Republic is a website that collects board games news. They also have developed their own personality model, which consists of 6 different archetypes: Tactician, Socializer, Immersionist, Daredevil, Architect and Striker[7]. Their idea is to give an overview about those types, and suggest games that could be liked by the different personalities. There is a quiz on their website[9], and the model suggested here - based on gaming styles - seems to be highly appreciated among board gamers[10]. In figure 2.3 is part of the description of the Tactician archetype, to give an idea about how the the different personalities are defined. However, no academic resource about those archetypes has been found, limiting the credibility of being a valuable tool to use.

2.2.5 Timmy, Johnny and Spike

Timmy, Johnny and Spike are three different gamers personalities defined by Mark Rosewater and the "Magic: The Gathering" designers. It describes the three main different people that can be found while playing Magic. Timmy defines the player who plays to enjoy "big" wins, Johnny is the player that prefers to win using hidden or peculiar strategies, and Spike is the competitive type that plays to win at all costs[89].
Even though this model would be simplistic enough to be used for this thesis, it feels like its is too tied to the "Magic: the Gathering" game, as it can be perceived even more in the "revisited" version of the same model[90], where the finer distinctions are based on the different behaviours of the players depending on the deck they are playing with. It had been used as a personality model for other games, like Dominion[49], reaching the same conclusion about this model being game-dependent.

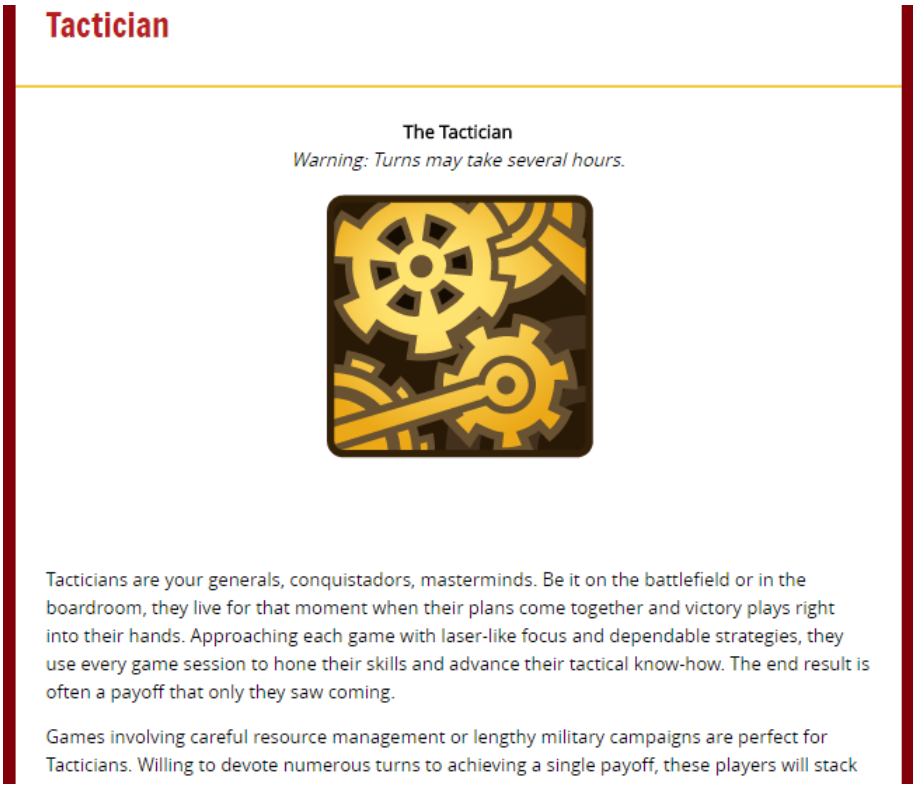


Figure 2.3: Example of one of the Cardboard Republic Archetypes (from [7]).

2.2.6 BrainHex

BrainHex is the tool developed by International Hobo Ltd to help categorize gamers while running a survey on their preferences. They associate personalities with the parts of the brain responding to games in the specific classes, as can be seen in figure 2.4. The personalities remind of the archetypes in the Cardboard Republic, and even

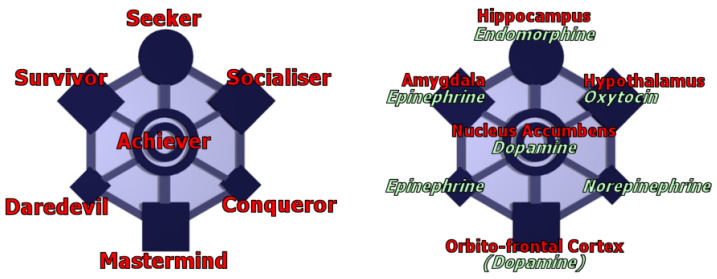


Figure 2.4: On the left, BrainHex classes, on the right the corresponding parts of the brain (red text) and chemicals (green text) involved. Figure from [79].

though it has a more scientific background -results of a survey can be found in the article [79]-, has the same complexity problem as some models described above.

2.2.7 Big Five Personality Traits

The Big Five Traits is a model widely used in psychology[50], and it analyzes how personalities fall into five different scales: Openness, Conscientiousness, Extraversion, Agreeableness and Neuroticism. Each personality corresponds to a combination of the factors, helping the comprehension of behaviours and social interactions.
The five factors are described as:

- Openness (to experience): general openness towards emotions, new ideas, and non-ordinary situations. Low scores would imply favouring routines, and more traditional approaches.
- Conscientiousness: high self-discipline, sense of duty, aim for achievement. Lower scores describe careless behaviour, spontaneity, and impulsive reactions.
- Extraversion: higher scores imply talkativeness, sociability, and tendency to seek company of others. Lower scores describe reservedness and introversion.
- Agreeableness: people with high scores are usually kind, trusting, trustworthy and considerate. Lower values imply skepticism, uncooperativeness and suspicion.
- Neuroticism: also called emotional instability, describe the tendency to experience negative emotions and be vulnerable to stress.

Different scores in the factors match up in a range of various traits, giving a overall better fitting description of different personality.[34]
Unfortunately, it is not really suitable for this project, as all the factors matter equally and it would become too complicated to match users and gameplay behaviours to specific combinations. However, this was the model adopted by the project this thesis is based upon, therefore an adaption to this model was needed in order to have some continuity. A better understanding of this adaption will be given in subsection 4.1.

2.2.8 Galen-Hippocrates' Four Temperaments

Dating back to Hippocrates and Plato, the four temperaments might be one of the oldest personality models in psychology. It divides in four different categories: Sanguine, Melancholic, Phlegmatic, and Choleric[72]. Shortly, the sanguine type is described as easily bored, somehow careless and as a risk-taker. Phlegmatic people are told to be diplomatic and to avoid conflicts. Cholerics are analytical, logical and goal-oriented. Melancholics are traditional and extremely ordered and accurate. Even though there is a difference between temperaments and personality (the first one refers to innate behaviour, while the latter is built on top of the temperament, i.e. through experiences), we use the two terms interchangeably here, as what we are looking for is a simplistic model to categorize our players' behaviours.
Using the descriptions of each type as a guideline, and associating the model to the time responses for each type, as analyzed in [28], those archetypes are manipulated a little, to get the results shown in table 2.1 describing the playing style of each type. In table 2.1 are shown the "specifics" for each personality type. A set of suits has been scaled from 1 to 4 depending on how fitting it is to each personality, where the suits are competitiveness, strategy, analysis paralysis and contrast, and they will be better described later. The scale system is applied instead of just pointing out if

Table 2.1: Associated gaming skills to each of the four temperaments, in a scale from 1 to 4, where 1 means it is a strong suit for the type.

	Sanguine	Choleric	Melancholic	Phlegmatic
Competitiveness	3	1	2	4
Strategy	4	2	1	3
Analysis Paralysis	4	3	2	1
Contrasts	1	2	3	4

they are descriptive of the archetype or not in a binary fashion, in consideration of this not being exact, because of people showing different traits even if labeled with the same personality type. The table has the purpose not to strictly define a player type, but more to give an idea to our human players on what is to be expected from each temperament.

- Competitiveness: shows how much a player cares about winning. It can be assumed that player types with a high value of competitiveness would do anything in order to win, without caring about how other players could perceive them.
- Strategy: indicates if a player is careless in his moves or if there is a long-term plan behind. Players scoring high in strategy might play unintuitive or non-obvious moves, in order to satisfy the long-term plan. The lack of it might end up with more of short-term game (obvious moves, which give gain in the following few steps but not ultimately the best moves to win the game).
- Analysis Paralysis: high scores result in the player taking a long time between moves, possibly getting overwhelmed by the game and stalling.
- Contrast: if the player likes contrasts, it can be assumed it would play in a more aggressive and less defensive way, favouring risky moves over more conservative ones.

There are more distinctions that could be added to the table, but this already defines four very different playing styles and it feels it could be easily matched to our human players. Unfortunately, the Phlegmatic type cannot really stand up for its skills in the games chosen in this project. A better perception of this type would be given with games that involve deception and contrast, like Werewolf[12] or the Resistance[5].

2.2.9 Myer-Briggs

The Myer-Briggs model distinguishes among 4 different bipolar traits: extroversion (E) and introversion (I), sensing (S) and intuition (N), thinking (T) and feeling (F), and judgment (J) and perception (P). It is assumed that people have a preference towards either ends of the scale, for each of those pairs, distinguishing then 16 different types of people[77, 78].

This model has being widely adopted in the business sector, although it is not considered as fully effective and reliable[87, 21]. We have never considered this model as suitable for our purposes -having a too broad variety of classifications-, but since it is widely used in other contexts, we could find an adaptation between it and Keirsey's types[61, p. 23].

2.2.10 Keirsey's Temperaments

Keirsey distinguishes between two basic human behaviours: communication and action. The first one could be further divided in either abstract or concrete, while actions could be defined as utilitarian or cooperative. The combination of those 4 traits forms the temperaments, which would be Artisans (concrete utilitarians), Guardians (concrete cooperators), Idealists (abstract cooperators), and Rationals (abstract utilitarians)[61].
Those four types can be further distinguished in 4 sub-types each, for a total of 16 different personalities that can be matched one-to-one to the Myer-Briggs' ones, as can be seen in table 2.2. This model was described in Steward's unified model[105],

Table 2.2: Keirsey's types and associated Myer-Brigg's types.

Guardian	Artisan	Idealist	Rational
Supervisor (ESTJ)	Promoter (ESTP)	Teacher (ENFJ)	Fieldmarshal (ENTJ)
Inspector (ISTJ)	Crafter (ISTP)	Counselor (INFJ)	Mastermind (INTJ)
Provider (ESFJ)	Performer (ESFP)	Champion (ENFP)	Inventor (ENTP)
Protector (ISFJ)	Composer (ISFP)	Healer (INFP)	Architect (INTP)

and being proposed as interchangeable with Bartle's taxonomy, not taken into consideration as base model for the project.

2.3 General Game Playing

Games have been considered suitable for the development of Artificial Intelligence, and have been adopted as a framework for many years now; in [95], Schaeffer and van Der Herik present an interesting overview on the topic. In this project, we will focus on the branch called General Game Playing (GGP), firstly described by Pell's thesis[85]. General Game Playing deals with agents learning new games without prior knowledge or ad-hoc strategy. The rules are given to the player only once the game has started, and the agent will have to work out a strategy while the match is ongoing. Contrarily to gaming agents like Deep Blue[24] or AlphaGo[100], a General Game Player is not supposed to know a-priori any game-specific heuristic, and the heuristics it is supposed to be supplied with need to be applicable to any game, hence the word General. The subject includes a broad range of topics in AI, as knowledge representation, search algorithms, machine learning and, of course, game playing[48].

2.3.1 Games Structure

Innumerable games could be proposed to General Game Players, as long as they satisfy the common requirements: the games are supposed to have a fixed number

of players, and can be defined as in definition 2.1.

Definition 2.1 *A game in GGP can be described as an automaton defined by a tuple* $< S, A, \delta, s_0, \Gamma, G_p >$*, where:*

- S *is a finite number of states;*
- A *is a finite set of actions;*
- δ *is the transition function, where* $\delta : S \times A \rightarrow S$*;*
- s_0 *is the initial state, where* $s_0 \in S$*;*
- Γ *is the set of final states, where* $\Gamma \subseteq S$*;*
- G_p*, the goal function associated to each player* p*, for each state* $s \in S$*.*

Only a subset of the games available for General Game Playing will be considered here: all of them are perfect information games, turn-dependent, and bounded to 2 players, as will be further described in section 3.3.1. Games are represented with state machines (see appendix A for further information), specifically, they can be seen as trees, where each node-successor represents the legal moves from the parental node. Additionally, games are also Markov Decision Processes[18] -better described in 2.5-, as the history of transitions between states is irrelevant in the decision process for the future, at any given point[48, p. 189].
When the game starts, the players are provided with the rules of the games, described in Game Description Language (GDL)[69] -a logic language, variant of Prolog. The language has recently been extended (GDL-II) to allow description of imperfect information games[107, 96].
In GGP, every game must be playable and weakly winnable - if multiplayer -, or strongly winnable - if single-player -, according to definitions 2.2 and 2.3, as found in [47].

Definition 2.2 (Playability) *A game is playable if and only if every player has at least one legal move in every non-terminal state.*

Definition 2.3 (Winnability) *A game is weakly winnable if and only if, for each player, there is a sequence of joint moves of all players that lead to a terminal state where that player's goal is maximal. A game is strongly winnable if and only if, for some players, there is a sequence of individual moves that leads to a terminal state of the game where that player's goal value is maximal.*

The administration of the matches is left to the Game Manager, which communicates to and with the players regarding the ongoing game, i.e. informing when the respective turns start -passing over the state of the board and the list of previous actions-, or when the game is over. In figure 2.5 we can see the idea behind a GGP system[48].

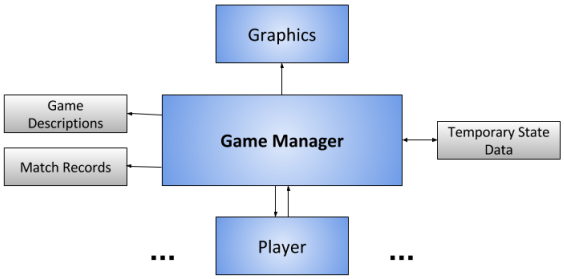


Figure 2.5: Diagram of a classical GGP environment (adapted from [48]).

The player used for this project uses a Monte Carlo Tree Search algorithm to find a strategy, and has been modified to go through already played games, and it will be further described in section 3.2.

2.4 Monte Carlo Tree Search Algorithm

Monte Carlo Tree Search has become more and more popular recently, due to the simplicity of implementation and the generality of application. It has been applied to a multitude of scenarios, giving satisfying results. Sticking to the games' development in the field of artificial intelligence, the Monte Carlo Tree Search algorithm has been applied to a variety of games: Go[44, 45, 35], Hearts[93], Settlers of Catan[106], Amazons[68], and Lines of Action[116, 115] among the others. Moreover, it can also be associated with the best players participating in the General Video Game Playing Competition[86].
The algorithm has been developed starting from the idea of Monte Carlo methods, which obtain results relying on random sampling. In the search case, the Monte Carlo sampling is associated to a tree structure, where - when applied to games - each node represents a state in the game, and contains the current QValue and a value n, where n is the number of visits for that specific state[27]. The algorithm could be summarised in four main steps[26], as can also be seen in 2.6:

1. *Selection*: starting from root, and until a terminal node is hit, a leaf child node is recursively selected for expansion;
2. *Expansion*: children nodes are added to the tree, depending on the available moves in the current state;
3. *Simulation*: one play-out is run through until a value *chargeDepth* limit is reached. A parameter *epsilon* decides whether a random move is picked, or if Move-Average Sampling Technique (MAST) which will be better described in the next paragraph is applied.
4. *Backpropagation*: the nodes in the tree get updated with the results from the simulation, associating a value $Q(s, a)$ to each node, defined as

$$Q(s,a) = \frac{1}{N(s,a)} \sum_{i=1}^{N(s)} \chi_i(s,a) z_i \tag{2.1}$$

where, using the same notation as [44] and [22], $N(s, a)$ is the number of times action a has been simulated from state s, $N(s)$ describes how many times a

game has been played out through state s, z_i is the result of the ith simulation played out from s, and

$$\chi_i(s,a) = \begin{cases} 1 & \text{if } a \text{ was selected from state } s \text{ during } ith \text{ playout;} \\ 0 & \text{otherwise.} \end{cases}$$

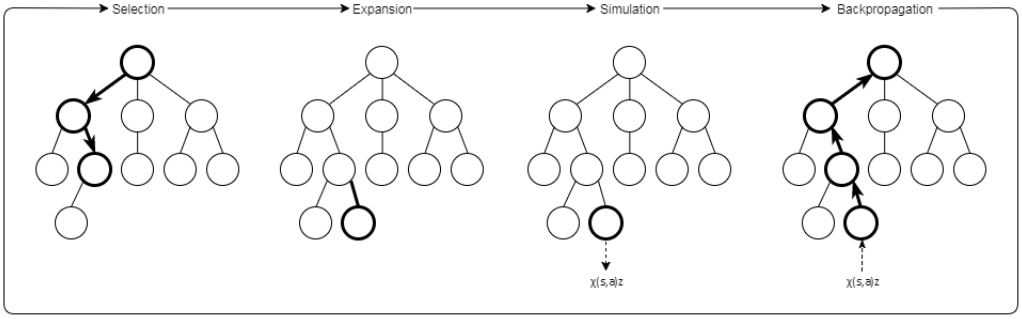


Figure 2.6: Iteration of MCTS (graph adapted from [22, 27, 116]).

In the case of this project, the search ends when *Limit* simulations - set as a constant $limit = 1000$ - is reached, but could be replaced by any time or memory constraints.

2.4.1 Monte Carlo Tree Search Enhancements

The search can be associated to a set of other algorithms and tree policies, that can lead to quicker results compared to the basic Monte Carlo Tree Search, and might be preferred in some particular circumstances. In [22] we can find a comprehensive survey of the Monte Carlo Tree Search methods that have been implemented so far, and we would like to refer to it if the reader is interested in further information. Different enhancements can be applied at different stages of the search, it is then possible - and suggested - to apply multiple algorithms at once[40]; in our case, our algorithm uses UCT with RAVE and GRAVE as extensions for the learning process, and MAST as a search-control policy, and they will be better described later in this chapter. First thing, though, it is necessary to define what bandit problems are.

Bandit problems and Upper Confidence Bound (UCB)

Bandit problems are a class of sequential decision problems that could be modeled over slot machines (known as "one-armed bandit" in Las Vegas), and describes the *exploitation-exploration dilemma*, which is the problem of finding equilibrium between exploiting an action that appears to be optimal in the short run, and exploring the search space for a temporarily less-optimal but overall better one. Assuming to be in a room with K slot machines, while having no knowledge of the past observations, our goal is to collect information about each slot machine by pulling their levers, while maximizing our cumulative reward at the same time. A K-armed bandit can then be described as a set of random variables $X_{i,n}$, with $1 \leq i \leq K$ and $n \geq 1$, and where $X_{i,n}$ represents the reward of pulling arm i for the nth time.

All $X_{i,n}$ are independent and identically distributed for all n, accordingly to an unknown distribution, with unknown mean μ_i. Which bandit to play is decided with the aid of a *policy*, whose goal is to minimize the regret of the player, defined as $\nu_n = \mu^* n - \mu_i \sum_{i=1}^{K} \mathbb{E}[T_i(n)]$, where $\mathbb{E}T_i(n)$ is the expected number of plays of arm i in the first n tries, and μ^* is the best possible reward[22, 13, 64, 65, 99]. It is known that the slowest growing regret policy is $O(\ln n)$, as described by Lai and Robbins in [67], and one the simplest applicable policies, with expected logarithmic growth, is *Upper Confidence Bound (UCB)*, first published by Auer et al. [13], and defined as

$$UCB1 = \overline{X}_i + \sqrt{\frac{2 \ln n}{n_i}}.$$

Upper Confidence Bounds for Trees (UCT)

Firstly proposed by Kocsis and Szepesvári [64, 65], Upper Confidence Bounds for Trees (UCT) considers every state in the game as a multi-armed bandit problem, taking advantage of *UCB1* and its simplicity and efficiency for the children selection. Then, the selection of child node i depends on maximizing

$$UCT = \frac{w_i}{n_i} + C_p * \sqrt{\frac{\ln N}{n_i}}$$

where w_i is the number of wins after move i, n_i is the number of simulations for move i, N is the total number of simulations for the current node, and C_p is a constant that we will be passing as a parameter called $ExplorationFactor$.

Rapid Action Value Estimation (RAVE)

Rapid Action Value Estimation (RAVE) is part of policies that are categorised under the name of *All Moves As First (AMAF)*[23, 53]. In this case, it is combined with UCT, as first done by Gelly and Silver[43, 44], and it is meant to speed up the nodes' values evaluation, updating the value of each action every time it is taken, even in subsequent courses of action, and not only when it is specifically selected at one state. RAVE is based on a class of AMAF algorithms called α-AMAF, which calculates the score for each action as

$$\alpha Q_a + (1 - \alpha) U$$

where U is the UCT score, and Q_a is the Q value for action a. In the RAVE case, we have a parameter β instead of α, which is calculated as

$$\beta = \sqrt{\frac{Rave}{3 * n + Rave}}$$

where n is how many times the node has been selected, and *Rave* is the value passed as parameter to our search, representing the number of visits a node can have before RAVE is not used anymore[22, 43, 44].

General Rapid Action Value Estimation (GRAVE)

General Rapid Action Value Estimation (GRAVE) can be considered a generalization of RAVE, where a node is considered to have reliable statistics only when a certain number of playouts has been carried out. This value is passed as parameter *Grave*, and when the number of playouts is greater than this parameter, then the values of the actions are calculated using general RAVE, otherwise the closest parent with a satisfying number of playouts is used as a reference to calculate the value for Q_a[25].

Move Action Sampling Technique (MAST)

The MAST method has firstly been described in [38] and applied in CADIAPLAYER [36]. This technique keeps track of the average value of each move, updated at backpropagation, and independent from the state it has been played in. From literature it is known that, on the long run, actions a with an higher average value $Q_{mast}(a)$ are most likely to be better than the ones with lower average[39]. During playout the move selection can then be biased, with probability distribution corresponding to Gibbs distribution, as

$$P(a) = \frac{e^{Q_{mast}(a)/\tau}}{\sum_{b=1}^{n} e^{Q_{mast}(b)/\tau}}$$

where τ is a parameter that controls the distribution, and $Q(a) = maxGGPscore = 100$, if a hasn't been explored yet[38, 36, 39].

Discounting

Two different discounts are applied to the moves' values, controlled by two separate variables. One discount, depending on a variable called $ChargeDiscount$, applied at backpropagation, and the second, depending on $TreeDiscount$, is applied as a discount in the tree. It is thought that the search tree results are more reliable than the ones found at charge, therefore only a smaller discounting needs to be applied[52].

Early Cut-Off

Running Monte Carlo Tree Search algorithms might be time consuming, so a limit can be introduced to try to skim the moves that look more unappealing, so to focus better on the moves that are more promising[37]. A limit $ChargeDepth$ is applied to the tree size, in order to make sure that a response to the opponent is being sent back in a decent amount of time. When this limit is hit by our search, a set of default values is returned, controlled by an array of variables called $ChargeDefaults$, one for each player[52].

Heuristics

Innumerable heuristics could be applied, either game-dependent (i.e. through piece counting[52]), based on the move selection history[94], or used as personality bias. The latter is the case applied in this project's search, with two arrays of variables

- one per each player - called $Aggressiveness$ and $Defensiveness$, and one value called $NiceThreshold$. As the names are mostly self-explicable, $Aggressiveness$ and $Defensiveness$ are controlling the goals of the game, being an increasing or decreasing factor, depending on the current state of the game. Considering that the maximum goal is 100, and the minimum is 0, we have that:

- If $goal_{current} > 85$, the game can be considered as already won, and no biases are applied;
- if $60 < goal_{current} < 85$, then the goal is biased using the $Aggressiveness$ value;
- if $goal_{current} < 35$, then the goal is biased using the $Defensiveness$ value;
- if $35 \leq goal_{current} \leq 60$, then a value r is picked uniformly at random in $[0, 1]$ used to select with probability $P_{a,d} = 0.5$ which bias to apply, $Aggressiveness$ or $Defensiveness$.

It is to be noted that $goal_{current}$ takes its value from the games rule, and most games only expect a win, a loss, or a draw, corresponding to 100, 0, or 50 points each. The values get then discounted, allowing to have some more flexibility with the $goal_{current}$ values described above.
Lastly, the value $NiceThreshold$ controls which difference in goals sets the AI to lower its standards and pick a sub-optimal move instead of the best one, to make the gaming experience more enjoyable for the opponent[52].
Although those parameters have relatively less influence in the search outcome itself, it is easier to intuitively relate them to specific personality types. For example, looking back at table 2.1, a higher $Aggressiveness$ value would be expected for higher "Contrasts", hence for Sanguine users, as well as lower $NiceThreshold$ for more competitive players, as Choleric and Melancholic people are supposed to be.

2.5 Machine Learning

Machine Learning (ML) refers to the concept of extrapolating patterns and information from data. There are three main approaches that are applied in machine learning: supervised learning, where the agent learns to map certain outputs depending on specific inputs; unsupervised learning, where the agent is given an input and is meant to discover patterns in the outputs, so to be able to draw any conclusion about the data; and lastly, reinforcement learning, where behaviour is meant to be learnt depending on rewards or punishments. In the case of unknown reward functions that need to be withdrawn from observed behaviour, the problem becomes an inverse reinforcement learning one. In the case of this thesis, the problem falls in the supervised learning field. Classical methods for supervised learning could not be applied in this project, due to the complexity of the model used. In this section will be defined the information needed to understand the Bayesian probability model in section 3.6. A Bayesian approach simply means calculating each hypothesis' probability and using them to infer predictions from the data[92].
The combination of a Bayesian framework and excerpting of preferences has gotten promising results in literature[91], as well as the combination of Bayesian statistics and evolutionary algorithms, as for example in [110], where an Approximate

Bayesian Computation algorithm (ABC) has been combined with a Differential Evolution one; in [84], where Bayesian Networks were used instead; or in [113], where Bayesian Inference has been matched with Genetic Algorithms.

2.5.1 Decision Theory and Preference Elicitation

Looking at the games' structure, it is easy to associate the problem of selecting the next move in a game to the topic of Decision Theory, which is an interdisciplinary field intuitively defined as the analysis of choices between options, each one having an expected utility value. The underlying idea in decision theory is called Maximum Expected Utility (MEU), where *"an agent is rational if and only if it chooses the action that yields the highest expected utility, averaged over all the possible outcomes of the action"*[92]. In the specific, not having a full picture of the actions choices for all states of each game, the problem tried to be solved in this thesis falls in the class of "Decision Making Under Uncertainty", which, as the name hints, refers to the problems where either information or knowledge are incomplete[97, p. 3-4]. Even more specifically, the goal to extract preferences depending on the personality of the player would fit in the Preference Elicitation problem, as it tries to identify the favourite events of a group of people, over the set of available actions that can be selected from being in a certain state, depending on the group's personality. The choices throughout games are here extracted from a Monte Carlo Tree Search algorithm, described previously, which has a set of controlling parameters the actions picked depend on. Different sets of parameters lead to different actions, hence the decisions should be able to be associated to a set of MCTS parameters. This set is meant to be found using a Genetic Algorithm -described later in this chapter-, with a fitness function that represents how close the AI's moves choices are to the humans'. Our hypothesis is assuming multiple major differentiation of people's behaviour in game playing, based on specific personalities. We are expecting that games played by people with different personality would develop with different types of underlying strategies, than if compared to people with the same personality. The implementation of our algorithm is foreseeing the application of a diverse set of parameters, depending on the personality of the player.

2.5.2 Markov Decision Process

It has been mentioned in 2.3 that each game in General Game Playing is seen as a Markov Decision Process. In order to clarify what a Markov Decision Process is, it is necessary to start defining the Markov property. Assuming to work in sample space Ω, let I be a state-space, X is a random variable with values in I where $X : \Omega \to I$, then

$$\lambda_i = \mathbb{P}(X = i) = \mathbb{P}(\omega \in \Omega : X(\omega) = i).$$

It is said that $(X_n)_{n\geq 0}$ is a Markov chain with initial distribution λ and transition matrix $H = (h_{ij} : i, j \in I)$ if

(i) X_0 has distribution λ;

(ii) for $n \geq 0$, conditional on $X_n = i$, X_{n+1} has distribution $(h_{ij} : j \in I)$ and is independent of $X_0, \cdots X_{n-1}$.

Theorem 2.1 *A discrete-time random process $X_{n_0 \leq n \leq M}$ is a Markov chain (λ,H) (or just called Markov(λ,H)) if and only if* $\forall i$ $i_1, \cdots, i_N \in I$

$$\mathbb{P}(X_0 = i_1, X_1 = i_2, \cdots, X_N = i_N) = \lambda_{i_1} h_{i_1 i_2} h_{i_2 i_3} \cdots h_{i_{N-1} i_N}.$$

Additionally:
Theorem 2.2 *Let $(X_n)_{n \geq 0}$ be Markov(λ, H). Then, conditional on $X_m = i$, $(X_{m+n})_{n \geq 0}$ is Markov(δ_i, H) and is independent of the random variables $X_0, \cdots, X_m$.*
The proofs are let to be read in [82], where we have taken the cue from for the above theorems and definitions. However, more informally speaking, the Markov property is assuming that the Markov Chain X is in a current state s_t, then no additional information is given by how s_t has been reached. Therefore, the current state s_t provides enough information for the reward to be computed without looking at previous states. Subsequently, Markov Decision Process can then be defined as sequential decision processes with a Markovian transition function and a reward function[92]. More formally, adapting from [88] and [80]:
Definition 2.4 *A (finite) Markov Decision Process (MDP) is a tuple* $< S, A, P_{sa}, G_{\cdot}(\cdot, \cdot), \xi >$, *where:*

- *S is a finite set of states;*
- *A is a finite set of actions;*
- *$P_{sa}(\cdot)$ are the state transition probabilities upon taking action a in state s;*
- *$G_p(s, s')$ is the reward function;*
- *$\xi \in [0, 1]$ is the discount factor.*

2.6 Genetic Algorithm

The information to be inferred from the dataset is a set of parameters which can be used as input to an agent, expecting that the output actions will match humans', when the human has the same personality as the agent being trained. In order to determine this specific set of parameters, an optimization algorithm is applied over the fitness function 2.1. The approach adopted is a Genetic Algorithm.
Genetic Algorithms (GA) are part of the family of stochastic optimization methods, and are based on Evolutionary Algorithm (EA), having the purpose to study the adaptation of individuals throughout time. First implemented by John Holland and his colleagues and students between 1960s and 1970s[54], there has been a growing interest in the topic in the past years. Evolutionary Algorithms take their terminology from biology, being inspired by it in the first place. The concept of *population* is then introduced as a set of *individuals*. Each *individual* has a *chromosome*, formed by a set of *genes*. The *individuals* are then compared using a *fitness function*, which is used to decide how the *population* should evolve throughout the *generations*. A more detailed description of the biological terminology can be found in [75, p. 5] and in [112, p. 35], while here the vocabulary will be limited to what is strictly related to the purposes of this project.
Termination methods may be applied to Genetic Algorithms, however, there is no complete certainty of finding the actual optimal solution, as the outputted solution might be due to premature convergence and only be a local optima.

2.6.1 General Structure

There is no final definition of GAs, but a general structure can be drawn from literature:

- *Individual*: a set of *genes*. The *genes* have a defined *encoding scheme*, that can be a binary encoding scheme, strings, real numbers, integers, and so on.
- *Population*: a family of Ψ individuals is initialized at random, forming the first *generation* of the population.
- *Fitness function*: is a function F whose value is calculated at evaluation time. The overall goal of optimization problems is to maximise (or minimise) the value of this function.
- *Evaluation*: each *individual* i is evaluated, after being decoded in the relative variable, and the value of $F(i)$ is calculated, where F is the *fitness function*, and $i = 1, ..., \Psi$.
- *Selection*: in order to form the next *generation* it is needed to select which *individuals* will survive evolution, and which will not. There are multiple strategies for selection, where the more popular are tournament selection, roulette-wheel selection and ranking selection. Those methods will be further described in subsection 2.6.2.
- *Elitism*: keeping track of our best individual generation after generation reduces the risk of having a decreasing fitness due to mutation or crossover. It has been shown that combining elitism to a selection algorithm improves the final results[15].
- *Crossover*: the reproduction of the individuals happens through a crossover algorithm, where, once two parents are picked, two new individuals containing genes from either parents are added to the next generation. Further information about the topic can be found in subsection 2.6.3.
- *Mutation*: Once the new generation is born, a mutation can happen to none, one or multiple genes in each individual. A description of how mutation happens over this project's encoding can be found in 2.6.4.

2.6.2 Selection

In this section we will describe three different algorithms for the selection process will be described. Only the roulette-wheel selection has been used in this thesis, as specified in 3.7, but the other two algorithms have been considered and might have been applied for comparison if there were no time constraints.

Tournament selection Tournament selection, as the name hints, it is the selection process carried out as a tournament. Once defined a tournament size t_s, t_s individuals are taken and compete for survival. The lower t_s is, the higher the competition. This flexibility makes this approach one of the most widely used in GAs[73].

Roulette-wheel selection The roulette-wheel strategy selects individuals proportionally to their fitness value. To obtain the resemblance with an actual roulette-

wheel, a cumulative fitness function is calculated as

$$\phi_j = \frac{\sum_{j=1}^{i} F(j)}{\sum_{i=1}^{\Psi} F(i)}, j = 1, ..., \Psi$$

where $F(i)$ is the fitness value of individual i. A value r is drawn uniformly at random in $[0, 1]$, and the individual with the smallest j that satisfies $r < \phi_j$ is selected[112].

Rank selection Firstly defined in [14], it ranks all the individuals depending on the value of their fitness function, from rank Ψ for the best individual, to 1 for the worst. Each individual i has then an associated probability of being selected, calculated as

$$P(i) = \frac{1}{\Psi}\Big(\eta^- + (\eta^+ - \eta^-)\frac{i-1}{\Psi - 1}\Big); i \in 1, ..., \Psi$$

where the conditions $\eta^+ = 2 - \eta^-$ and $\eta^- \geq 0$ are satisfied, since the population size is constant, and $\frac{\eta^-}{\Psi}$ is the probability of selecting the worst individual in this generation's population, while $\frac{\eta^+}{\Psi}$ is the probability of selecting the fittest[20].

2.6.3 Crossover

We will describe two of the most used crossover algorithms: one because we think it helps to have a better understanding of the crossover process; and the second one, which is the algorithm that has applied in this project. Further techniques can be found in [111].

Single Point Crossover A popular way of applying crossover is to decide on a "barrier" c, and create the offspring o_1 and o_2 with c genes from one of the parents, and $(G - c)$ genes from the other one, where G is the total number of genes in an individual. This approach is called Single Point, and it is represented in figure 2.7.

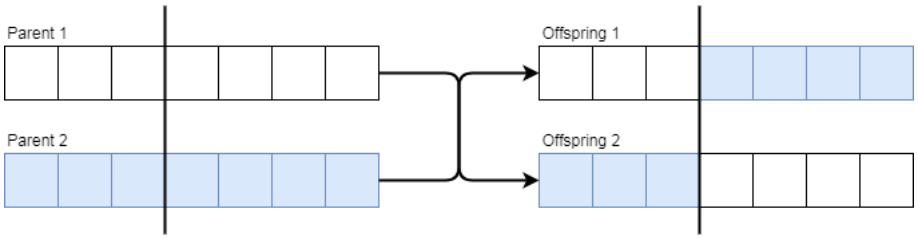


Figure 2.7: Example of Single Point crossover between two individuals.

Deriving from this idea, there is Two-Points crossover, as well as Multi-Points Crossover, which explanation at this point is trivial.

Averaging Crossover Given two individuals as parents, this approach returns two children o_1 and o_2 with genes calculated as an average from the parents' p_1 and p_2, as

$$g_{o_1} = \alpha g_{p_1} + (1 - \alpha) g_{p_2}$$

and

$$g_{o_2} = (1 - \alpha) g_{p_1} + \alpha g_{p_2}$$

where α is chosen uniformly at random in $[0, 1]$[66], for each gene g.

2.6.4 Mutation

Any gene in any individual could be subjected to mutation, depending on probability P_{mut}. When dealing with genes encoded with real numbers, the most common approach for mutation is the real-number creep, where a creep value limits the range of values the new gene can take. Let C_r be the creep rate, then the new gene is

$$g \Leftarrow g - \frac{C_r}{2} + C_r\gamma = g + C_r(\gamma - \frac{1}{2})$$

where γ is picked uniform at random in $[0, 1]$[112, pp. 53-55].

2.6.5 Termination

Genetic Algorithms usually stop when there is stagnancy: either the fitness function has not been increasing for generations, or the individuals converged to very similar ones (leading to a very similar fitness function too). In some cases it is preferred to give a fixed number of generations to run the GA for; in other, a decency fitness test is added, stopping the algorithm when a minimum fitness value is reached, finding an acceptable result instead of the optimal one. In this project, a stagnancy test has been adopted, and will be further described in section 3.7.

3

Methods

This chapter will contain the methodologies applied during the development of this project. Firstly, in section 3.1 there will be the definition of the problem this thesis tries to solve. The selected personality model to be applied in the project is described in section 3.2, followed by the explanation of how the data has been collected, in section 3.3. Section 3.4 introduces the base project, that is the baseline for the artificial agent the algorithms implemented in this thesis are training upon. The description of the gaming algorithm involved will be found in section 3.5, which is a Monte Carlo Tree Search algorithm. Section 3.6 includes the probability model in association with our problem. The Genetic Algorithm used to infer information from the dataset can then be found in section 3.7. Lastly, the evaluation of the set of parameters outputted by the above mentioned algorithms can be found in section 3.8.

3.1 Problem Definition

The aim of this thesis project is to understand if procedural personæ can be inferred from human game playing, and if it is possible to recreate those personalities by parameters tuning of a Monte Carlo Tree Search algorithm, so to return the optimal move for each state, where optimal in this case means similar to the human player's moves. A set of parameters $\vec{\theta_j}$ would return action a_j^s for a state s, while -for the same state s- we can expect a different a_k^s for parameters $\vec{\theta_k}$. What we aim to show by the end of this project, is that for a personality ρ, we can have a set of parameters $\vec{\theta_\rho}$ that leads the artificial agent to pick the same actions (or equivalent), as a human with personality ρ. In order to find the values associated to those set of parameters, we will apply a Genetic Algorithm over the Monte Carlo Tree Search, expecting the optimal set of parameters to match a set of training data. Finally, each set of parameters $\vec{\theta_\rho}$ will be evaluated on new clusters of data, part of which is of unknown users (with relatively unknown personalities), and part of known testers. The influence that the choice of parameters $\vec{\theta}$ will have on the search outcome, matching or not the correct personality, will confirm or disprove our thesis.

3.2 Personality Model

After an extensive research over personality models, the Galen-Hippocrates' model became the best suited for this project, as it seems like different players could be easily mapped to a specific type, giving us some variety and not too subtle distinctions. Furthermore, there is academic background, that gives it some reliability as a model. Some other models, like BrainHex, or the Big Five, would give a more accurate representation of personalities, however the expected amount of data retrieved will not give them justice. Every model listed in section 2.2 could succeed in different settings, but they are either too complicated for our scopes, or they are not backed up by enough academic research to be used as reliable models.

3.3 Data Collection

When dealing with Optimization Algorithms and Machine Learning, it is usually suggested to have a discrete amount of data available. Knowing how slow data collection could be, the best option for us would have been to find existing gameplay databases. Unfortunately, as the existing platforms were unresponsive, or the data was just not suitable for our purposes, this research was unsuccessful. Consequently, we set up our own data collection system, taking advantage of the Tiltyard server[8] as framework.
The Tiltyard server has been developed as a testbed for general game players, which also implies it not meant to be strictly used by human players. Because of the not extraordinarily user-friendly interface, we had to provide our testers with an instruction file[32], as well as a couple of social platforms (a Facebook group and an IRC Channel, to be specific) to find opponents to play against. Additionally, in order to keep track of the personalities of the players, we asked our testers to fill a questionnaire with the information not collected by the Tiltyard server, such as usernames, perceived personality of the adversary, and player's own personality.
In the instruction file[32], the testers can find a brief description of the personality models, with a table representation (see table 2.1) to help them define their own archetype, the rules of the games, as explained in subsection 3.3.1, together with the actual instructions on how to use Tiltyard.
To simplify the game set-up process, we have implemented a second version of the Tiltyard homepage, including the choice of playing against the latest version of CadiaPlayer[6, 36, 19], a player previously developed at Reykjavik University. However, time constraints have not played in our favour, and the page has not been available to the public early enough to make any actual difference in the data collection process.
The participation in the data collection has not been very high, even after organizing periodic events trying to simplify finding an opponent. Only a small amount of games were played in total, but we managed to gather an acceptably uniform distribution of games and personalities, as will be better described in section 4.2.

3.3.1 Selected Games

The games selected are two players, perfect information games: Checkers, Nine Board Tic-Tac-Toe, Connect Four, and Skirmish. In figures 3.1, 3.2, 3.3, and 3.4 we can see examples of the games while ongoing. Although a few of those games cannot be considered new for our testers, the version implemented in Tiltyard might have included a few variants of the rules as they are traditionally known.
The choice of games has been dictated by the list of available games in Tiltyard; once we selected the ones with a usable human interface, we checked the average number of moves, excluding the ones that had too little or too many, to avoid games to be too short or too long.

Checkers

The game is played on a 8x8 squares chessboard, where each player has 12 pieces. The goal of the game is to capture all the adversary pieces by jumping over them. Pieces can move only diagonally, and on unoccupied squares. In this version of the game, capturing is not mandatory, and it is possible to capture multiple pieces during the same turn (if there is an empty square between every two). When a piece reaches the adversary's side of the board, it becomes a "king", and gains the ability to move backwards. The player who has no more pieces on the board, or cannot move anymore, loses the game (see figure 3.1).

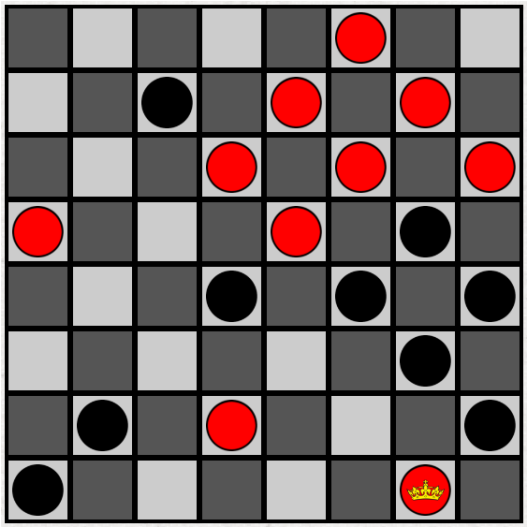


Figure 3.1: A game of Checkers.

Checkers has been selected because it is a classic game, where non-simple strategies can be developed. It is also the most time consuming game among our choice, having 99.91 moves on average for each game on Tiltyard[1].

Nine Boards Tic-Tac-Toe

The game is played on a grid formed by 3x3 simple Tic-Tac-Toe grids. The first player can play on any board, but his/her choice would force the second player to

play in the grid corresponding to the square of the previous move (if player 1 picks the central square, player 2 will have to play in the central grid). If the grid the current player is supposed to play on is full, then any other move is allowed. To win, the player has to win on any one of the boards (see figure 3.2).

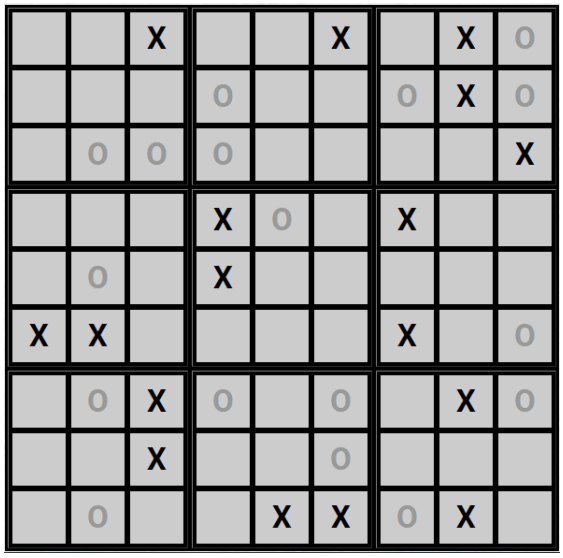


Figure 3.2: A game of Nine Boards Tic-Tac-Toe.

Nine-Board Tic-Tac-Toe has 35.07 moves on average on Tiltyard[3], and has been chosen because it is a variant of the more famous Tic-Tac-Toe, with the advantage of not having too obvious moves.

Connect Four

The game is played on a 6x8 suspended grid. Taking turns, the players drop a token of their colour in the grid. The goal is to have 4 tokens connected either horizontally, vertically, or diagonally (see figure 3.3).

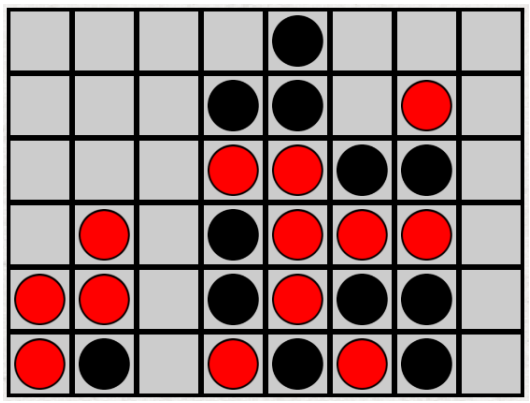


Figure 3.3: A game of Connect Four.

Connect Four has been selected for being the "quick game", with only 28.52 moves on average on Tiltyard[2], and being mostly well known worldwide.

Skirmish

The game uses a chessboard and the same pieces as chess, which are also moving in the same way. The pieces are all worth the same, fixed, amount of points -from the pawn to the king-. To win, the player has to eat the most amount of adversary's pieces. The game terminates when all the pieces of one player have been eaten, or one of the players cannot move anymore. The rules of chess are not rules in Skirmish (i.e. it is not possible to castle, there is no check mate, and so on...), however pawns get promoted to queens when they reach the other side of the chessboard (see figure 3.4).

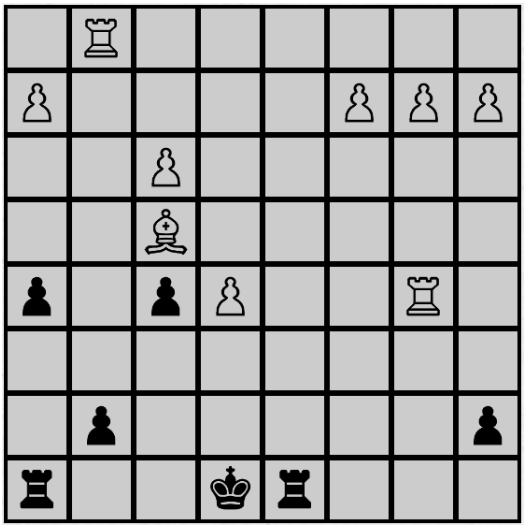


Figure 3.4: A game of Skirmish.

Skirmish has been chosen later in the thesis project, taking the place of another game that had been selected first (called Battle), but was proving to be problematic for our users to play. We then decided that a chess-like game, without falling into the obvious choice of chess, was the perfect game for our evaluation process, having a wide tree search as a consequence of its complexity. Only a handful of matches have been played on Tiltyard, so we do not think that the average number of moves of 59.67[4] is a reliable statistic.

3.4 Base Project

The implementation of this thesis project has been based on a work with the title of "Virtual General Game Playing Agent"[52], previously carried out at Reykjavik University. The authors have joined the General Game Playing features with a Unity Player who would mimic human behaviours when playing a game of Checkers or Nine Men's Morris. The player has a virtual reality interface, as we can see in figure 3.5, and it is meant to work with a joystick device and an Oculus Rift. Although the graphic interface is fascinating, we have not taken advantage of it, focusing only on the back-end GGP Engine.

The back-end implementation is based on the ggp-base framework[98], which pro-

Figure 3.5: A screen-shot of the Virtual Agent while playing a game of Checkers.

vides API for the Game Manager, supplying libraries to handle the GDL reasoner and the communication protocol, to create the game state machines, and to check which moves are legal at each state. This framework has been improved with the addition of a Monte Carlo Tree Search algorithm, tweaked depending on a wide set of parameters that are supposed to give the virtual player a human-like gaming style. Those parameters are the ones that are exploited in the current version of the search, and are described in detail in section 3.5.1.
The Monte Carlo Tree Search is paired with an heuristic dependent on the number of pieces on the board, which would affect the aggressiveness or defensiveness of the virtual player's next moves. However, having a more diverse selection of games in our implementation, we are not adopting the same heuristic, relying on the evolutionary algorithm to find a suitable balance in the 17 control parameters.
Lastly, in order to make the virtual agent more human, it has been provided with the choice of a personality. When starting the game, the user can select values to decide which personality the opponent will have. The personality model used here is the Big Five, described in subsection 2.2.7, and in subsection 4.1 we can associate the "older" model with the one chosen for the purposes of this thesis' work.

3.5 Monte Carlo Tree Search

The search to evaluate which move needs to be played is implemented as a Monte Carlo Tree Search algorithm with UCT. The algorithm satisfies the requirement of not being game-dependent, and has been giving optimal results in the gaming AI development. In section 2.4 we have seen the theory behind it and described the different strategies that can be applied to a basic Monte Carlo Tree Search Algorithm, while in this section we will understand the settings chosen for our purposes.
We run our Monte Carlo Tree Search over each game sample, with a set of 17 different

parameters that will affect the values in the tree and, possibly, the output. Those parameters could be hand picked and hard coded (as they were in the old project we based this thesis on), or chosen by the evolutionary algorithm described in section 3.7. As mentioned earlier, our algorithm is implemented allowing the addition of RAVE and GRAVE over UCT, MAST, and other enhancements, as early cut-offs, some personality-biased heuristics, and discounting.
Our algorithm runs over the matches collected, calculating the QValues for all the moves in each state, but eventually forcefully selecting the same move as the user. This is done to give us an easier way of mapping human behaviour over the states available, focusing only on the knowledge we have available.

3.5.1 Parameters

The Monte Carlo Tree Search parameters are what truly affects the values of each move, as we said earlier. Since each parameter represents and controls a different piece of structure in the search, it is easy to assume that each of them will have its own variable type and a range of meaningful values. More details about the intervals are given in section 3.7.
The parameters can be described as:

- *Rave* is the threshold at which only the QValues are considered for selecting moves, and Rave stops being used.
- *Grave* is the threshold setting when Grave stops being used, and only general Rave is used for moves evaluation.
- $ChargeDepth$ represents the depth limit for a play out.
- $Horizon$ represents the maximum depth of the tree.
- $ExplorationFactor$ is a UCT value that controls the exploration weight over exploitation in the MCTS algorithm.
- *Limit* is the number of simulations the MCTS executes. It has been fixed to be $Limit = 1000$, and will not be modified at any stage during the process.
- $TreeDiscount$ decides how much discount there is on every move after expanding a node in the tree.
- $ChargeDiscount$ represents the discount applied after each play out.
- *Epsilon* represents the probability of using MAST instead of picking a random move.
- $ChargeDefaults$ is a couple of default values to be applied if a charge is stopped in early stages.
- $Defensiveness$ is a pair of values (one per each player) that bias the goal over valuing defensive moves.
- *Aggressiveness* is a pair of values (one per each player) that bias the goal towards aggressive moves.
- *RandomError* represents the probability of picking a random move instead of the best one.
- $NiceThreshold$ biases the move selection, forcing the AI to not always pick the best move and becoming a too strong opponent.

The parameters $Defensiveness$, *Aggressiveness*, and $ChargeDefaults$ are represented in the search as arrays of 2 items each, one per player. That explains the

discrepancy between the number of parameters mentioned earlier, and the number of items in the description list here above.

3.6 Probability Model

In this section the probabilistic model will be described, applying a Bayesian approach. Recapitulating, a set of parameters is plugged into a MCTS algorithm, whose outcome changes the game actions. A dataset of several games where players can be distinguished depending on a personality is given as input. Let a set of uniformly distributed parameters - within different ranges, as described in section 3.7 - be defined as $\vec{\theta} = \theta_1, \ldots, \theta_L$, where we know $L = 17$; $\vec{\rho} = \rho_1, \ldots, \rho_4$ indicates any among the people's personalities; and D^k_{mcts} with $k = 1, \ldots, M$, is the match data retrieved from the MCTS, where M is the number of matches played. The number of matches associated to a specific personality ρ will be indicated as M_ρ. It should also be distinguished from D^j_h, which refers to the human gameplay data that has been collected, with $j = 1, \ldots, M$. As a simplification, we will refer to ρ as any of the four personalities, as it is not necessary to singularly specify them in this context, and to $\vec{\rho}$ as the set of all personalities in the model. We refer to Appendix A for a brief summary of the probability rules and definitions referred to in this section.
Overall, it is in the interest of this project to confirm the thesis where

$$P(\vec{\theta}) \neq P(\vec{\theta}|\rho), \qquad \forall \rho \in \vec{\rho} \tag{3.1}$$

as otherwise we would be proving that the set of parameters $\vec{\theta}$ used will not be relevant to distinguish a common pattern in the behaviour of a user with personality ρ. Additionally, it would be favorable that $D^j_h = D^k_{mcts}$, $\forall j, k \in [1, \cdots, M]$, or in other words, it would be optimal to have a set of moves resulting from the MCTS that would match human actions.
In figure 3.6 is the Bayesian Network representation of the problem, to help clarify the following statements.

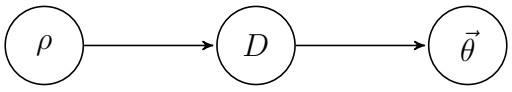


Figure 3.6: Bayesian Network representation.

The goal is to find the probability of $P(\vec{\theta}|\rho)$ over all the set of data, where

$$P(\vec{\theta}|\rho) = \sum_{j=1}^{M_\rho} P(\vec{\theta}|D^j_h) P(D^j_h|\rho). \tag{3.2}$$

Considering all matches in each set, and defining D_ρ as the set of games $[1, M_\rho]$ for a specific ρ, we can rewrite equation 3.2 as

$$P(\vec{\theta}|\rho) = P(\vec{\theta}|D_\rho, \rho) P(D_\rho|\rho) = P(\vec{\theta}|D_\rho) P(D_\rho|\rho). \tag{3.3}$$

Knowing that $\vec{\theta}$ is conditionally independent of ρ given D_ρ, then $P(\vec{\theta}|D_\rho, \rho) = P(\vec{\theta}|D_\rho)$, and in order to maximise it we rewrite it as

$$P(\vec{\theta}|D_\rho) = \frac{P(D_\rho|\vec{\theta})P(\vec{\theta})}{\sum_{\vec{\theta}_j} P(D_\rho|\vec{\theta}_j)P(\vec{\theta}_j)}. \tag{3.4}$$

To break down equation 3.2, considering D_h as the set of all D_h^j, and starting from the last term of the chain, let

$$P(D_h|\rho) = \frac{P(\rho|D_h)P(D_h)}{P(\rho)}.$$

As we are considering the datasets divided by personality ρ, we can assume for our purpose that $P(\rho|D_h^j) = 1$ if the personality ρ is associated to the data D_h^j, or $P(\rho|D_h^j) = 0$ otherwise.
Considering uniform probability over the personalities ρ, it is given that $P(\rho) = \frac{1}{4}$, concluding that

$$P(\vec{\theta}|\rho) = 4\sum_{j=1}^{M_\rho} P(\vec{\theta}|D_h^j)P(D_h^j) \tag{3.5}$$

where the matches taken into account are the one corresponding to the personality ρ considered. Rewriting equation (3.5) for a specific ρ it is then obtained that

$$P(\vec{\theta}|\rho) = 4P(\vec{\theta}|D_\rho)P(D_\rho). \tag{3.6}$$

From equation (3.6) the *posterior* is defined as $P(\vec{\theta}|D_\rho)$, as the probability of having a specific set of parameters $\vec{\theta}$ matching a set of matches $D_\rho = D_h^1, D_h^2, \cdots, D_h^{M_\rho}$.
The problem of maximising the posterior can be reduced to maximising the numerator of 3.4, as the denominator is the sum over all the possible data and parameters' sets and is constant. Furthermore, $P(\vec{\theta})$ is the *prior* of our model, constant as well, and defined as

$$P(\vec{\theta}) = \frac{1}{W}$$

where W is the cardinality of the set of all possible parameters $\vec{\theta}$. Hence, the maximisation problem has scaled down to

$$\underset{\vec{\theta}}{\operatorname{argmax}}\, P(D_\rho|\vec{\theta}) \tag{3.7}$$

which happens to be the *likelihood* of the model, where D_ρ indicates all the matches belonging to the same set, defined over a specific personality ρ. In other words, it is the probability of selecting a list of certain actions given a set of parameters.
Being the games defined as Markov Decision Processes, the likelihood for each match D_h can then be calculated as

$$P(D_h|\vec{\theta}) = P_{\vec{\theta}}(a_1, ..., a_T|s_1, ..., s_T) = \prod_{t=1}^{T} P_{\vec{\theta}}(a_t|s_t) \tag{3.8}$$

where T is the number of turns in the match, and $P_{\vec{\theta}}(a_t|s_t)$ is the probability of choosing action a in state s, in turn t. It is to be noted that, on a higher level,

multiple actions could be available for selection in any state s, due to the non-deterministic policy of the games.
The probability of a human player selecting an action a_{human} in a certain state s is calculated using the Boltzmann-Gibbs distribution - which is often associated to multi-armed bandit problems[31], hence applicable to MCTS - as

$$P_{\vec{\theta}}(a_{human}|s_t) = \frac{e^{Q_{\vec{\theta}}(a_{human},s_t)\kappa}}{\sum_{j=1}^{A} e^{Q_{\vec{\theta}}(a_j,s_j)\kappa}}$$

where $\kappa = 0.1$ is a constant, $Q_{\vec{\theta}}(a_{human}, s_t)$ is the Q-value the search with parameters $\vec{\theta}$ has associated to action a_{human} in turn t, A is the total of all legal actions at current state, and $Q_{\vec{\theta}}$ is the Q-value of said actions, as calculated from equation (2.1).
The optimal outcome of the chosen set of parameters for a specific personality would be to have

$$P_{\vec{\theta}}(a_{mcts}|s_t) = P_{\vec{\theta}}(a_{human}|s_t)$$

where in the same state s_t, for any state $t \in [1, \ldots, T]$, it is true that $a_{human} = a_{mcts}$.
To maximise equation 3.7, we can calculate $P(D_\rho|\vec{\theta})$ as the product of $P(D_h|\vec{\theta})$ over all the matches D_h in D_ρ. That is defined as the fitness function $F(\vec{\theta})$ for the genetic algorithm, where $\vec{\theta}$ will - in the GA case - be an individual i. The fitness is then defined as:

$$F(\vec{\theta}) = \sqrt[M]{\prod_{m=1}^{M} \sqrt[T]{\prod_{t=1}^{T} P_{\vec{\theta}}(a_{human}^{t,m}|s_{t,m}))}} \tag{3.9}$$

where T is the number of turns in each match, M is the number of matches gone through, and, for each state s, $a_{human}^{t,m}$ is the action chosen by the human player. It is easy to notice that the fitness function is the geometric mean over all the matches of the probability of selecting the action a_{human} picked by the human player in state s. The fitness function uses the geometric mean over the matches in order to normalise the fitness over matches with different number of turns, and the different cardinality of the sets M_ρ, for each $\rho \in \vec{\rho}$.
The set of parameters $\vec{\theta}$ that derives the likelihood for each set of data D_i is, however, unknown. The optimization method used to find the values applicable to the $\vec{\theta}$ is a Genetic Algorithm with a population of individuals $\vec{\theta}$ and the likelihood (3.8) is utilised as fitness function that needs to be maximised.

3.7 Genetic Algorithm

We have tried to solve the optimization problem with the implementation of a Genetic Algorithm, as we previously mentioned in section 2.6.
We have taken a population of $\Psi = 100$ individuals, where each individual has a set of chromosomes c_i, with $i = 1, \ldots, 17$, representing the parameters of the Monte Carlo Tree Search, described in subsection 3.5.1, and initialized uniformly at random, in a given interval. Each chromosome's value is allowed to be in a meaningful range, and will be better described at the end of this section.

Each individual is used as parameters to run the search and, once the moves are selected, the fitness $F(i)$ is evaluated as the geometric mean of the Boltzmann-Gibbs distribution function over every turn - or state s - of each match, using equation (3.9) proposed earlier. For the GA purposes, $\vec{\theta}$ will be referred to as individual i.
Different fitness functions could have been applied in order to possibly find a better fitting one, however, only equation (3.9) has been considered due to time limitations.
The selection uses a roulette wheel approach, where each individual has probability

$$P(i) = \frac{F(i)}{\sum_{j=1}^{\Psi} F(j)}$$

of being selected, where $F(i)$ is the fitness value of individual i. The cumulative probability distribution is then calculated over all the individuals as

$$C = \sum_{j=1}^{\Psi} F(j)$$

and a value r is picked uniformly at random in $[0, 1]$. If

$$\frac{\sum_{j=1}^{i-1} F(j)}{C} < r < \frac{\sum_{j=1}^{i} F(j)}{C}$$

then r sets the minimum threshold for selecting a specific individual, and individual i is picked. The roulette-wheel method has been preferred over the other selection algorithms, because of its non-biased approach[29].
Taking two selected individuals at the time, we apply crossover with probability $p_c = 0.05$. In case the crossover will not be carried out, the individuals remain unchanged. We will be using averaging crossover, because of its closest analogy with actual genetics[66]. Recapitulating, the gene g of the offsprings o_i is derived from the genes of parents p_i, and is defined as:

$$g_{o_1} = \alpha g_{p1} + (1 - \alpha) g_{p2}$$

and

$$g_{o_2} = (1 - \alpha) g_{p_1} + \alpha g_{p_2}$$

where α is chosen uniformly at random in $[0, 1]$ for each gene.
Mutation has probability $p_{mut} = 0.1$ to happen, with a creep rate $C_r = 0.4$, which represents the width of the distribution. We are adopting a uniform distribution to apply the creep mutation, so the new gene would be obtained as

$$g_m = g - \frac{C_r}{2} + C_r \gamma$$

where γ is uniformly distributed between $[0, 1]$.
The values of p_c, p_{mut} and C_r have been tuned with few testing over a set of the available games, and have been selected depending on the resulting number of generations, in order to try to avoid premature convergence.
The algorithm continues evolving the population until it becomes stagnant, or until one of the individuals reaches an acceptable fitness $f_a = 0.9$. To define a population

as stagnant, we first calculate the variance of the chromosomes in all the individuals of the current population, as

$$\sigma_k^2 = \frac{\sum_{j=1}^{\Psi} \left(\frac{g_j^k - g_\mu^k}{\sup(g^k) - \inf(g^k)} \right)^2}{(\Psi - 1)}$$

where g_j^k is the kth gene in individual j, g_μ^k is the average value for gene k, with $k = [1...C]$ and $j = [1...\Psi]$, where C is the number of chromosomes and Ψ is the population size. The population is considered then stagnant when

$$max_{k,j}^{C,\Psi}(\sigma_k) <= \sigma_a$$

where $\sigma_a = 0.005$ is what we can consider an acceptable standard deviation for our gene's values pool.
When the algorithm stops, the fittest individual will be the best set of parameters to represent the personality of that gameplay set, being it the one with the highest likelihood.

3.7.1 Parameters Intervals

This paragraph contains details about the values each gene in the chromosome can possibly take. It is to be reminded that each gene represents a MCTS parameter, as have been described in section 3.5. The genes are then said to be:

- long *Rave*, where $Rave \in [0, (y * Limit)]$, with y a small integer constant, and *limit* the number of simulations the search runs for (for us, $y = 5$ and $Limit = 1000$);
- long *Grave*, where $Grave \in [0, 120]$;
- long *ChargeDepth*, where $ChargeDepth \in [0, 100]$;
- long *Horizon*, where $Horizon \in [1, 20]$;
- long *ExplorationFactor*, where $ExplorationFactor \in [1, UCT_{limit}]$, with $UCT_{limit} = \overline{\Lambda_J} + \sqrt{\frac{2 \ln n}{n_i}}$, where $\overline{\Lambda_j}$ is the average reward for the state j, n is the number of simulation so far, and n_i is the number of simulations after the ith move. We can approximate $UCT_{limit} = 140$;
- double *TreeDiscount*, where $TreeDiscount \in [0.9, 1]$;
- double *ChargeDiscount*, where $ChargeDiscount \in [0.9, 1]$;
- double *Epsilon*, where $Epsilon \in [0, 1]$;
- double[2] *ChargeDefaults*, where $ChargeDefaults[i] \in [35.00, 45.00]$;
- double[2] *Defensiveness*, where $Defensiveness[i] \in [0, 3]$;
- double[2] *Aggressiveness*, where $Aggressiveness[i] \in [0, 3]$;
- double *RandomError*, where $RandomError \in [0, 1]$;
- float *NiceThreshold*, where $NiceThreshold \in [0, 100]$.

The types of the variables refer to Java primitive data types[83], where *long* is a 64-bit two's complement integer, *double* is a double-precision 64-bit IEEE 754 floating point, *float* is a single-precision 32-bit IEEE 754 floating point, and with the notation *[2]* we indicate arrays of 2 elements, or the element of index i if indicated as *[i]*.

3.8 Evaluation

The evaluation process involves the Skirmish games not used for training, of which we have user-associated personalities, and a set of games from the 2013 and 2014 match archives available on the GGP researchers' website[11]. The archives have been filtered, leaving us with human-played games, with no errors in the match, and that are either completed, or with an acceptable number of moves - to avoid considering games that have been aborted only after a couple of turns.
Once recovered the winning individuals from the Genetic Algorithm, the evaluation matches are run through both with the correct parameters, and with random ones, calculating the Action Agreement Ratio (AAR)[56], which is the percentage of matching moves between the users' results and the search's. We also calculate $\Delta_Q = Q(a_{human}) - Q(a_{search})$, where $Q(a)$ is the Q-value associated to action a in state s. The magnitude of Δ will be significant to evaluate if the parameters we plugged in are having any influence.
The results of the evaluation process will be further discussed in the results chapter 4.4.

4
Results

This chapter summarises the results collected along the stages of this thesis project. In section 4.1 is a comparison between the personality model of choice and the base project's, so the interested user can translate the personality-based results to either works. In section 4.2 are collected the results from the data collection. The output from the Genetic Algorithm will be discussed in section 4.3. Lastly, the results from the evaluation process will be discussed in section 4.4.

4.1 Comparison of Personality Models

It has been mentioned in section 2.2.7 that the original project this thesis is basing itself on adopted the Big Five as personality model. Due to its complexity, it is harder to be applied in this context, therefore a comparison of the two models, the Big Five and the Four Temperaments, had to be searched upon. Unfortunately, such direct comparison does not seem to be directly available. We then compare Galen-Hippocrates' four types to Keirsey's and then to Myers-Briggs.
It is easy to find association between Keirsey's types, Galen-Hippocrates' and Myer-Briggs', as for example in Keirsey's book[61]. In table 4.1 we can see a summary of those types, and how they are equated to each other.

Table 4.1: Comparison of Galen-Hippocrates' Four Temperaments with Keirsey's Four Temperaments and Myer-Briggs personality types, as from [61].

Models	Types			
Galen-Hippocrates	Sanguine	Choleric	Melancholic	Phlegmatic
Keirsey	Artisan	Idealist	Guardian	Rational
Myer-Briggs	E**STP** I**STP** E**SFP** I**SFP**	E**NFJ** I**NFJ** E**NFP** I**NFP**	E**NTJ** I**NTJ** E**NTP** I**NTP**	E**STJ** I**STJ** E**SFJ** I**SFJ**
Common traits Myers-Bryggs	SP	NF	NT	SJ

There has been a study to correlate the Myers-Briggs types to the Big 5, and the results are described in [71], and shown in table 4.2. The results were collected by a population of 267 men and 201 women, and it was found that the correlation values for men and women have negligible difference. Hence, we can just as well consider the results gathered from the men population, and consider it valid. Given table

4.1, we can easily define a table with the probability of being any of the Myer-Briggs types, given the Galen-Hippocrates', and it can be seen in table 4.3.

Table 4.2: Correlation of Big 5 factors with Myer-Briggs (data from [71]).

Big 5	Myer-Briggs			
	E-I	S-N	T-F	J-P
Neuroticism	0.16	-0.06	0.06	0.11
Extraversion	-0.74	0.10	0.19	0.15
Openness	0.03	0.72	0.02	0.30
Agreeableness	-0.03	0.04	0.44	-0.06
Conscientiousness	0.08	-0.15	-0.15	-0.49

Table 4.3: Probability of each Myer-Briggs preferences given the Galen-Hippocrates' type.

	E	I	N	S	F	T	J	P
Choleric	0,5	0,5	1	0	1	0	0,5	0,5
Sanguine	0,5	0,5	0	1	0,5	0,5	0	1
Phlegmatic	0,5	0,5	0	1	0,5	0,5	1	0
Melancholic	0,5	0,5	1	0	0	1	0,5	0,5

Table 4.4: Extension of table 4.2, with clarification of the preferences scale.

	E	I	N	S	F	T	J	P
Openness	0	0,03	0,72	0	0,02	0	0	0,3
Conscientiousness	0	0,08	0	0,15	0	0,15	0,49	0
Extraversion	0,74	0	0,1	0	0,19	0	0	0,15
Agreeableness	0,03	0	0,04	0	0,44	0	0,06	0
Neuroticism	0	0,16	0	0,06	0,06	0	0	0,11

Table 4.4 shows the same values as in table 4.2, just written in a way that emphasises the correlation values of each of the Myer-Briggs preferences, given the Big 5 types. Finally, the relation between tables 4.3 and 4.4 can be calculated. Let J be a matrix containing the values in table 4.3, and K be a matrix containing the values in table 4.4. Then it can be said that

$$R = JK^T$$

where R shows the correlation between the Four Temperaments and the Big 5, and which values can be seen in table 4.5.

Table 4.5: Correlation between the Four Temperaments and Big 5.

	Openness	Conscientiousness	Extraversion	Agreeableness	Neuroticism
Choleric	0,905	0,285	0,735	0,525	0,195
Sanguine	0,325	0,265	0,615	0,235	0,28
Phlegmatic	0,025	0,755	0,465	0,295	0,17
Melancholic	0,885	0,435	0,545	0,085	0,135

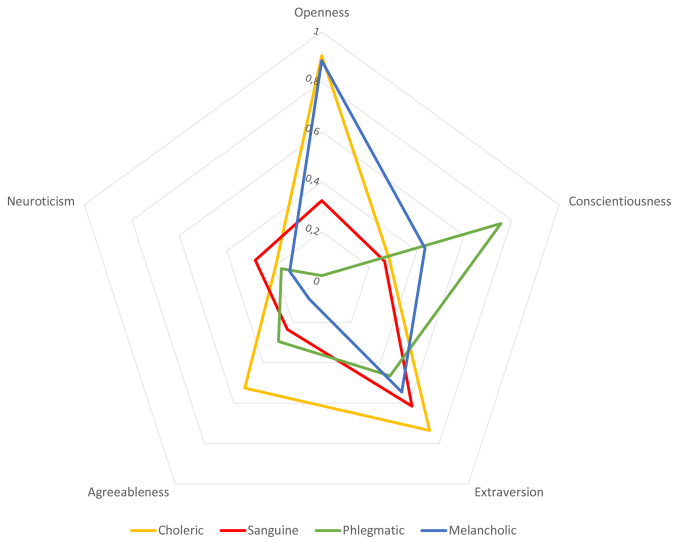


Figure 4.1: Correlation between the Four Temperaments and Big 5 factors.

In figure 4.1 we can see the comparison between all the temperaments, while in figure 4.2 we can see the temperaments singularly. Since the old project allows only three settings for each of the Big 5 traits (low, medium, high), we thought it could be useful to normalize the results in 4.5 in a scale 0-2, and consequently round up at the closest integer. The results are shown in table 4.6.

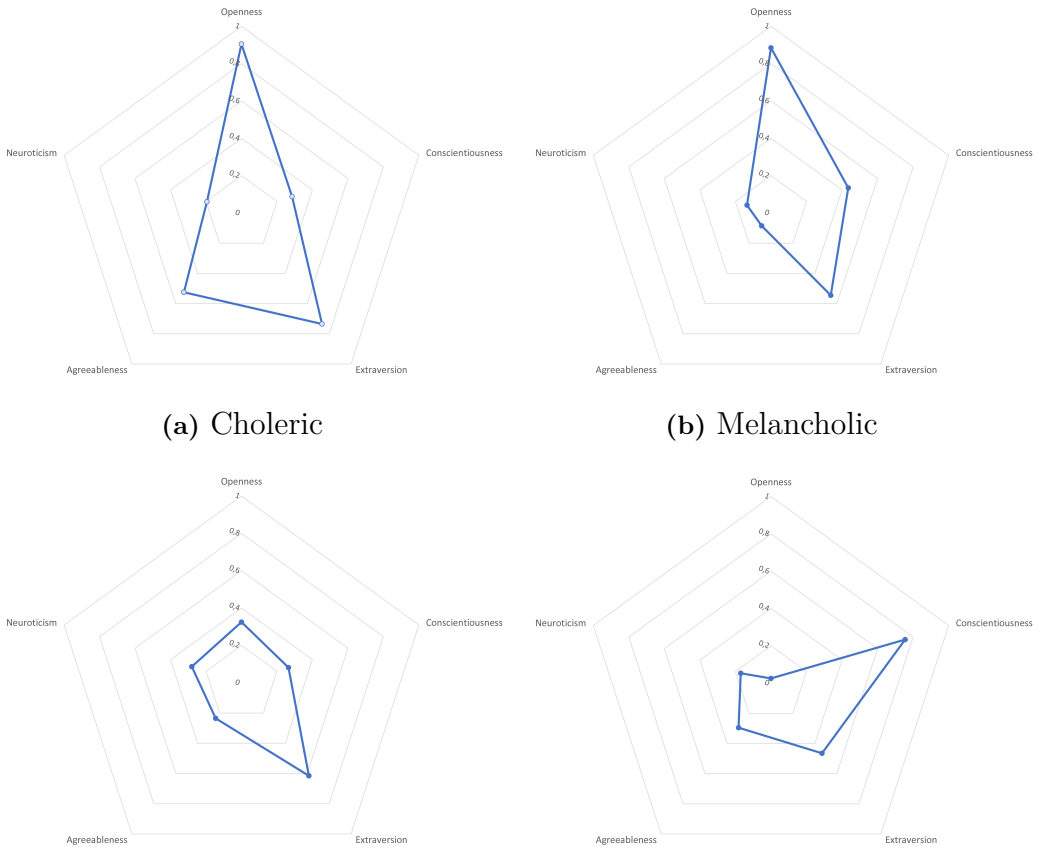
Table 4.6: Correlation of Big 5 factors with Myer-Briggs (table 4.5) normalised to 2.

	Openness	Conscientiousness	Extraversion	Agreeableness	Neuroticism
Choleric	2	1	1	1	0
Sanguine	1	1	1	0	1
Phlegmatic	0	2	1	1	0
Melancholic	2	1	1	0	0

The conclusion that can be drawn from looking at the results, is that the Neuroticism value is mostly irrelevant for the Four Temperaments model, nonetheless, we believe it might influence the game play, proving in this way that the chosen personality model is not complete to represent human behaviour properly in the general game play field. However, the magnitude of data collected for this project reduced the option of utilising any other complicated model, which would maybe be more suitable for modeling human behaviour in games.

4.2 Data Collection

Trying to counterbalance the lack of user-friendliness of the Tiltyard server, and the time-consumption of some of the games that have been chosen, the data collection process has been ongoing from the beginning of this project to the very last weeks. However, once the data training had started, only the collection of Skirmish games



(a) Choleric **(b)** Melancholic

(c) Sanguine **(d)** Phlegmatic

Figure 4.2: The four graphs representing the correlation between each of the Four Temperaments and the factors that define the Big 5 model.

was allowed, namely the matches required for the evaluation process.
Overall, data has been collected from 84 matches, in 46 different games, with 17 different users. The games involving two human players have been considered as two different matches by the genetic algorithm training, being associated to two different personalities and gaming styles. All users had different background and board games experience, and the most experienced players have been asked to try multiple games changing their playing style, trying to associate it to each of the different procedural personæ. Thanks to this, although the amount of games collected has been lower than expected, the distribution of games and personalities has been tendentially homogeneous in both personality and games distribution, as it can be seen from table 4.7.

In the questionnaire given to the users, aside their own personality, they also have been asked to specify which personality they were perceiving from their opponent. Considering some bias toward the Phlegmatic personality due to Tiltyard's slow response, we can generally see a tendency in associating the opponents to the Melancholic and the Phlegmatic personalities, while the self-definition results in a more homogeneous spread. In table 4.8 can be seen the summary of the perceived personalities of the opponents.

Table 4.7: Summary of the games collected and the personality the users self-associated to themselves.

	Checkers	Skirmish	Connect Four	Nine Board Tic-Tac-Toe	Tot.
Melancholic	1	3	4	17	25
Choleric	5	1	8	8	22
Phlegmatic	0	1	3	9	13
Sanguine	4	3	11	6	23
Tot.	10	8	26	40	84

As mentioned above, the Phlegmatic's numbers might be slightly skewed, inferring some false positive when the Tiltyard server was being unresponsive, and some false negative when the users realised that the long turns might have been caused by the system (ignoring completely - in such case - the possible phlegmatic tendency of the opponent). However, considering that a good part of our testers were in communication with each other - either in the same room, or in the IRC channel set up for them - we still think that, overall, the collected results could be meaningful. Nonetheless, this is the reason why only the self-described personality is taken into consideration for the training of our genetic algorithm. It is interesting to see that among the 84 matches collected, in 16 of them (almost 20%) either of the players perceived the personality that was self defined by the opponent. The distribution of the success in matching the opponents personality can be found in table 4.9. Checkers and Skirmish were unknown games to most of the users before participating in the data collection, and that would excuse the inability to recognize any pattern, or strategy, in the opponent's game. Whilst Nine Boards Tic-Tac-Toe was a new game as well, the simple correlation to basic Tic-Tac-Toe, could have improved the process of following the gameplay. On the other hand, it can be assumed that the

Table 4.8: Summary of the games collected and the perceived personality the users associated to their opponents.

	Checkers	Skirmish	Connect Four	Nine Board Tic-Tac-Toe	Tot.
Melancholic	5	3	5	22	35
Choleric	1	2	6	11	20
Phlegmatic	4	2	10	5	20
Sanguine	0	1	5	2	8
Tot.	10	8	26	40	84

Table 4.9: Summary of the games where the users recognized successfully the opponent's personality.

	Checkers	Skirmish	Connect Four	Nine Board Tic-Tac-Toe	Tot.
Melancholic	1	0	0	5	7
Choleric	0	0	1	3	4
Phlegmatic	0	1	2	1	3
Sanguine	0	0	1	1	2
Tot.	1	1	4	10	16

Melancholic type has been the most recognized as it might be easier to understand, by the end of the game, if the opponent has had a long term strategy ongoing during the whole match.

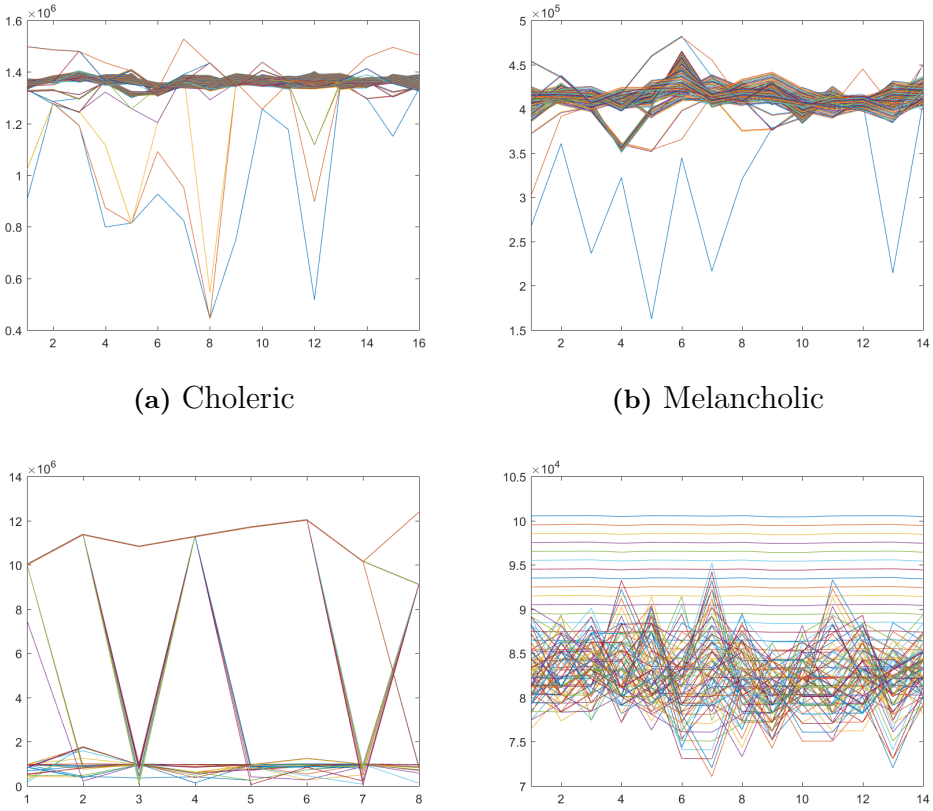
4.3 Genetic Algorithm

The Genetic Algorithm has been executed a few times, over different sized sets of data, and the first thing to be noticed it is that the general fitness was much lower than hoped for, causing the terminating test depending on acceptable fitness value $f_a = 0.9$ to be unused. Only stagnancy of the individuals was then defining whether the algorithm would have terminated or not.
It has been mentioned in 3.7 that a few test runs have been carried out in order to define which values to assign to the probabilities of crossover and mutation, p_c and p_{mut}, and creep rate C_r. Those values have been tuned after several tests over the Phlegmatic's batch of games, which was the quickest to execute. The execution time of each block of games depended heavily on the games played, rather than on the parameters selection themselves. It is found that the Choleric and Sanguine batch, containing a higher number of Checkers games, are noticeably slower than the Phlegmatic group. In figure 4.3 it can be seen how the Phlegmatic individuals have running time on the scale of $10^4 ms$, while Choleric and Sanguine have it on the scale of $10^6 ms$. The graphs also show how much impact the different parameters have on the execution times, which is generally consistent in the same personality data group, if we exclude a few noticeable peeks that reduce in number and intensity with the increasing of the generations.

4.3.1 Results Overview

In Appendix B can be found the plots of the single individuals over three runs of the genetic algorithm. Each page contains the plots of the single genes, for each individual in each generation. Of those three runs, the first one has a different set of "algorithm" values p_c, p_{mut}, and C_r than the ones described in section 3.7. Specifically, the first run adopts $p_c = 0.8$, $p_{mut} = 0.015$, and $C_r = 0.2$. The first thing to be noticed from those plots is the number of generations. Although different "algorithm" values have been tested over a set of the data in order to maximise the number of generations - as mentioned earlier -, after the first tests, the results have become inconsistent. Several runs took long to stagnate (over 40 generations), whilst took other only few (8, for example). It has finally been decided to keep the same set of values, given the unpredictability of the number of evolutions.

It is to be noticed that different personalities tend to different values for the same gene. However, it is also to be pointed out that different runs aim at different results as well. That can be excused as the optimal individual that is picked does not look the same for each execution, leading to slightly different values combinations. Looking in more detail to some of the graphs in Appendix B it can be seen when, for each personality, different executions return very diverse values for the same



(a) Choleric

(b) Melancholic

(c) Sanguine

(d) Phlegmatic

Figure 4.3: The four graphs represent the running times (in ms) for all the individuals in each generation, on y and x axis respectively.

parameter. What it means is that each parameter alone is not really relevant in the actions-matching for the personality in exam, as for example, the *ChargeDefault* (see figure 4.5). This statement can be clarified looking at figure 4.4.

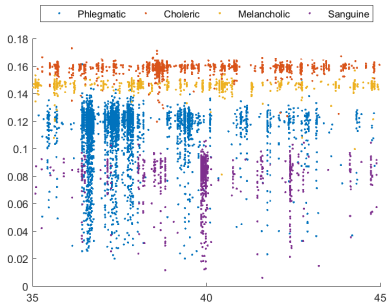
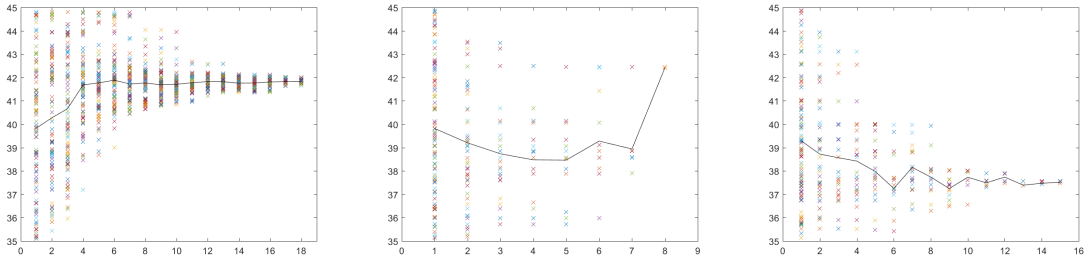


Figure 4.4: Correlation between specific Charge Default (x-axis) and fitness (y-axis).

It can be seen that each genes clusters around some specific values, although not in a meaningful distribution. Further correlations need to be investigated, to understand

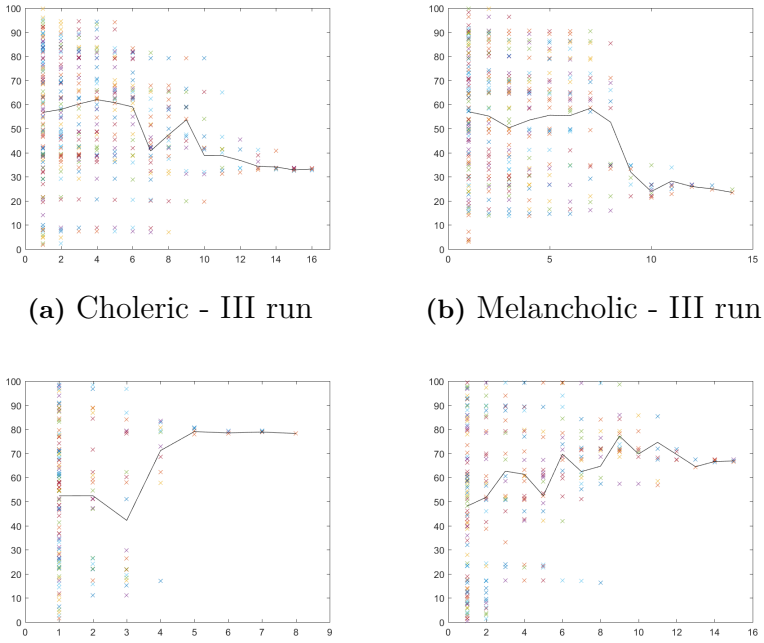
which combination of parameters is indeed useful to influence the agent's game, and which are not. It can also be noted from figure B.7 that the *Exploration Factor* gene



(a) Phlegmatic - I run **(b)** Phlegmatic - II run **(c)** Phlegmatic - III run

Figure 4.5: Charge Default gene for Phlegmatic personality, where the generations are on the x-axis, and Charge Default values are on the y-axis.

seems to tend to the same value independently of the personality.
From the *horizon* graphs in figure B.9 it can be seen that the Melancholic type has generally a higher valued outcome, as it would be expected, being it the personality with the higher correlation to game strategy. In the same way, the parameter *NiceThreshold* seems to generally be higher for Phlegmatic and Sanguine, and our model confirms them to be the two personalities with lower competitiveness. Figure 4.6 collects the results for the gene, after the third run. Contrarily to our expec-

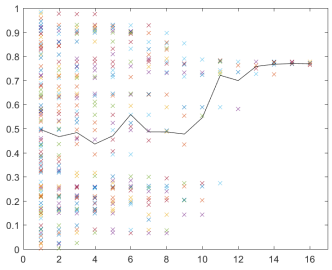


(a) Choleric - III run **(b)** Melancholic - III run

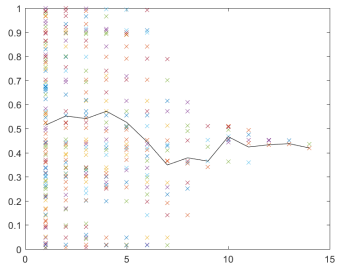
(c) Sanguine - III run **(d)** Phlegmatic - III run

Figure 4.6: Nice Threshold gene's results of the III run, the generations are on the x-axis, while the values for Nice Threshold are on the y-axis.

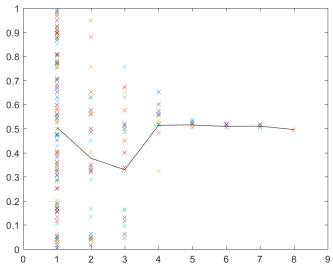
tations, the *RandomError* variable tends to a probability $P(RandomError) = 0.5$, or higher in some cases (see figure 4.7 for a clearer picture). We recall that this parameter controls the probability of picking a random move instead of the one suggested by the search. This means that half of the actions the MCTS outputs are aleatory, and that influences the fitness of the individuals, decreasing the helpfulness of elitism in keeping the individual with the highest fitness for the following generations.



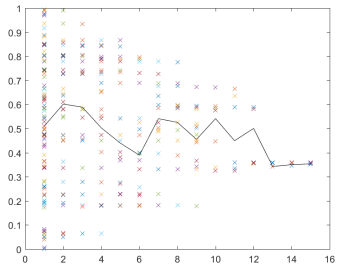
(a) Choleric - III run



(b) Melancholic - III run



(c) Sanguine - III run



(d) Phlegmatic - III run

Figure 4.7: Random Error gene's results of the III run, where the x-axis represents the number of generations, and the y-axis the value of Random Error.

4.3.2 Fittest Individuals

After a few executions of the Genetic Algorithm, the fittest individuals for each of them have been picked to feed the evaluation process. The values are collected in table 4.10. The Genetic Algorithm returns all doubles; the parameters are casted to the supposed type when plugged into the MCTS algorithm. The table aforementioned contains values that are partially already rounded (the ones with type *long*, to be exact), while the floating point ones are rounded to the third decimal place. In figure 4.8 can be seen the evolution of the fitness function, generation after generation. The red line represents the fitness of the fittest individual, calculated with equation (3.9); the blue line represents the average fitness of the individuals in that generation, while the blue errorbars are the standard deviation. Lastly, the black line represents the probability of picking a random move, referred here as "random-fitness". It was calculated as the geometric mean over all the matches of the probability $P(AnyMove) = \frac{1}{AvailableMoves}$ for each state. Considering that

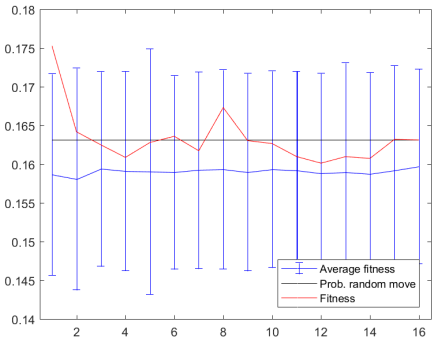
Table 4.10: Fittest individuals picked for evaluation.

Parameter	Phlegmatic	Sanguine	Choleric	Melancholic
epsilon	0.264	0.507	0.435	0.045
rave	2328	2744	4102	832
grave	65	71	84	55
chargeDiscount	0.937	0.947	0.985	0.930
treeDiscount	0.952	0.943	0.972	0.969
aggression	1.259	1.533	1.796	2.622
defensiveness	1.161	1.878	0.208	0.725
chargeDepth	53	54	47	61
horizon	12	12	11	19
randErr	0.503	0.451	0.398	0.380
chargeDefault	41.768	40.014	39.984	35.782
explorationFactor	68	81	71	105
niceThreshold	33.903	55.308	24.593	12.885

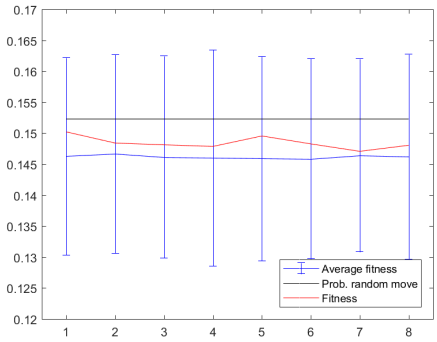
the MCTS is run always from the same state, it is expected that the probability $P(AnyMove)$ is constant throughout the generations. The optimal outcome would have been having the fitness of the fittest individual topping the random-fitness, but that is not true for most of the personalities. Figure 4.8 presents data from two different runs, as it is expected to be equally meaningful, the presented data has been chosen only on the basis of being clearer, although the results between executions are slightly different, they are equivalent nonetheless. It is to be noted that, while the fittest individual is -averagely- not much better than only picking random moves at each turn, it is still above the average fitness among the individuals in the same generation, as expected.
For completeness, it is to be pointed out that only at report composing time it has been found a discrepancy between the geometric mean calculated by MATLAB (plotted in the graphs), and the one returned in the genetic algorithm logs. This is indeed a problem to be fixed, but considered the demanding computation required by the algorithm, it has been decided that this difference is irrelevant for the conclusions to be drawn.

4.4 Results Game Play

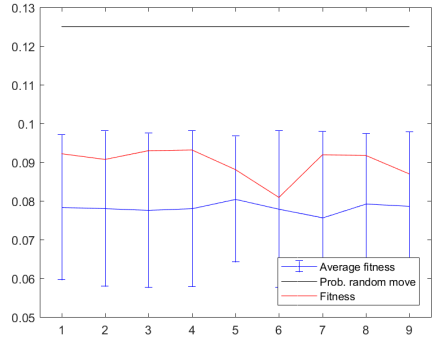
Among the games collected, only the Skirmish ones are used for evaluation purposes. Together with the skimming over the games recovered from the Tiltyard's bulks, it lead us to a total of 8 Skirmish games from the data collection and 54 games from the archives, divided in 22 Connect Four, 24 Nine Boards Tic-Tac-Toe, and 6 Checkers matches. The evaluation algorithm has been running with the fittest individuals outputted by the genetic algorithm (see table 4.10), and a set of random ones. The results are described below, while a selection of the gameplays' graphs can be found in appendix B.



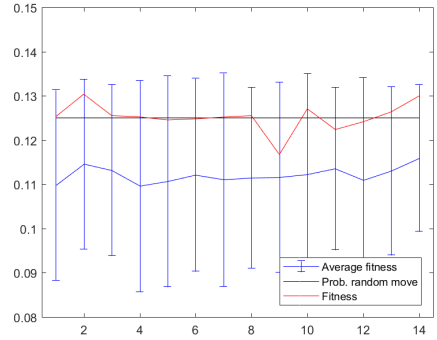
(a) Choleric - III run



(b) Melancholic - IV run



(c) Sanguine - IV run



(d) Phlegmatic - III run

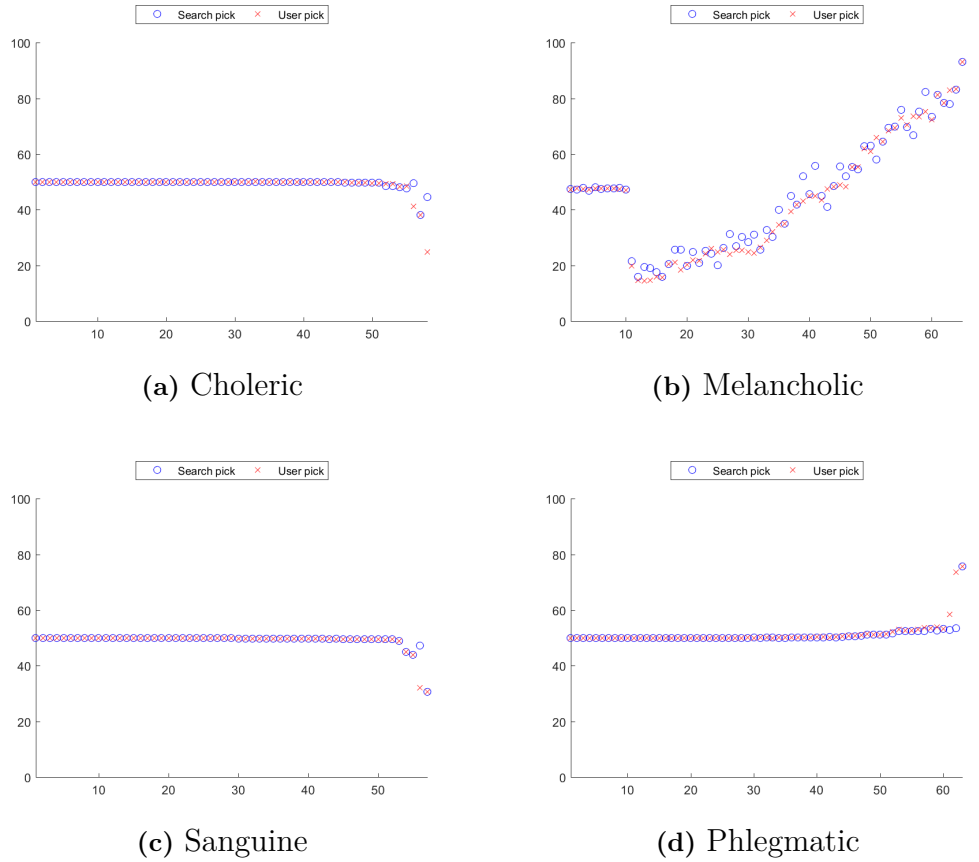
Figure 4.8: Fitness evolution (y-axis) throughout the generations (x-axis), for each personality.

4.4.1 Skirmish Games with Fittest Individuals

The first evaluation process has taken the fittest individuals resulting from the genetic algorithm. Figure 4.9 shows the results of one Skirmish game for each personality, played with the "correct" parameters outputted by the GA. Although the logs confirm that search moves are not actually the same as the user's, figure 4.9 shows that the selected moves had an extremely similar QValue. It is necessary to point out that Skirmish is a game with a lot of legal moves at each state, averaging then the values of all of them. Table 4.11 contains the list of last moves in the Choleric game, to show that the actions selected are, as matter of fact, different, but essentially with the same weight. It is to be noticed that the Melancholic game seems to be the least moderate, however the search is still following pretty closely the trend set by the user.

4.4.2 Random Games with Fittest Individuals

Plugging in the "correct" individuals in a set of games we have no knowledge about returned some interesting information. Appendix B contains plots of a few representative games, while here we would be describing only the most relevant ones.



(a) Choleric **(b)** Melancholic

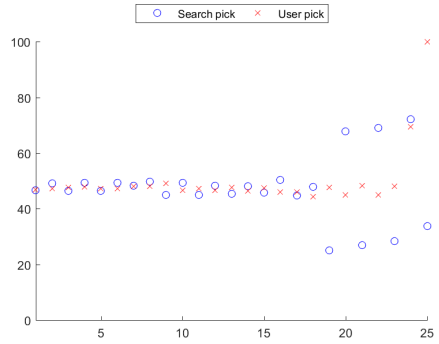
(c) Sanguine **(d)** Phlegmatic

Figure 4.9: Evaluation of Skirmish matches, where the moves are represented on the x-axis, and their QValue on the y-axis.

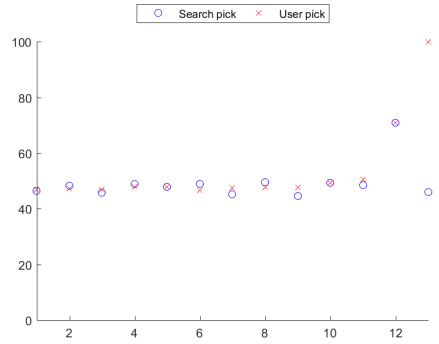
In figure 4.10 can be seen three examples of tendency of game development, for the Sanguine personality, in games of Nine Boards Tic-Tac-Toe (NBTTT): figure (a) represents a mostly matching game, with divergence in the last states; figure (b) indicates more correlation between the search and the user's selection; finally, figure (c) shows a totally scattered selection. This can indeed mean that a few of those games were played by users with Sanguine personality, while others are related to other sets of parameters, therefore, another personality type.
Looking at the single matches, for both Nine Boards Tic-Tac-Toe and Connect Four, in figures B.17, B.18, B.19, B.20, B.21, B.22, B.23, and B.24, it can be noticed that Choleric and Sanguine tend to be a better fit. This implies that those two personalities might be easier to mimic, as well as that the personality model used in this thesis is not truly representative.
From figure B.25, presenting the results for the Checkers game, it can be seen that it has some of the Skirmish behaviour. Due to Checkers extensive state space, the difference in value between them is minimal. However, the Melancholic personality's search seems to have more difficulties matching the users', then the other personalities, confirming once more what said earlier about personalities that are easier to represent.

Table 4.11: Last moves in the Choleric-Skirmish game.

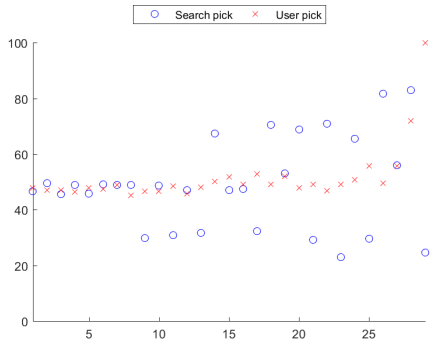
Search Action	QValue	User Action	QValue
[(move 1 5 4 8), noop]	49.495	[(move 6 7 4 5), noop]	49.720
[noop, (move 5 4 6 6)]	49.791	[noop, (move 5 4 4 6)]	49.773
[(move 4 5 3 4), noop]	49.953	[(move 3 8 4 7), noop]	49.271
[noop, (move 4 6 5 8)]	49.217	[noop, (move 4 6 6 5)]	49.045
[(move 4 7 6 5), noop]	42.226	[(move 8 4 6 5), noop]	42.222
[noop, (move 2 7 2 6)]	39.652	[noop, (move 2 7 2 5)]	39.413
[(move 3 1 2 2), noop]	47.195	[(move 1 5 2 5), noop]	26.864



(a) Sanguine - NBTTT - Match 2



(b) Sanguine - NBTTT - Match 9



(c) Sanguine - NBTTT - Match 19

Figure 4.10: Evaluation of Nine Boards Tic-Tac-Toe for Sanguine personality, where the moves are represented on the x-axis, and their QValue on the y-axis.

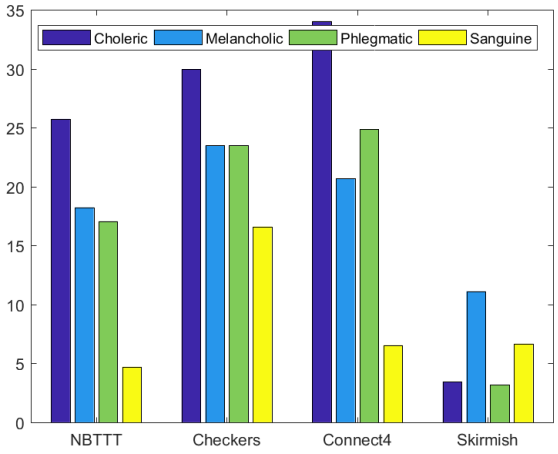


Figure 4.11: Evaluation of AAR with Fittest Individuals (percent).

Recapitulating, in figure 4.11 is the summary of the Action Agreement Ratio for each game, once the search has been fed with the parameters from the genetic algorithm. It has to be pointed out that figure 4.11 only includes the perfectly matching actions chosen by both the search and the users, while the graph in figure 4.9 only compares the moves' QValues. It is to be noticed that the Choleric parameters obtained from 25% to 35% matching moves, while Sanguine only 15% in the best case. However, looking at figure 4.9, it would look like more. That means that the actions chosen by the search is extremely similar in QValue to the one picked by the human player. Therefore, that might imply that the choice between the two moves is equivalent, and still achieve a human-like gaming style.

4.4.3 Skirmish Games with Random Individuals

The skirmish games that had been collected, and associated to a personality, have also been tested with 3 random sets of parameters. In figure 4.12 is an extract of the graphs collected, showing how the behaviour of the search changes, on the same match, depending on the parameters plugged in. We found irrelevant to show the graphs relative the rest of the matches for Melancholic and Sanguine, as the behaviour is tendentiously the same as the one showed in figure 4.12. The individuals used have the values shown in table 4.12.

Table 4.12: Random individuals picked for evaluation.

Parameter	Ind. 1	Ind. 2	Ind. 3	Ind. 4	Ind. 5	Ind. 6	Ind. 7	Ind. 8	Ind. 9	Ind. 10	Ind. 11	Ind. 12
eps.	0.299	0.540	0.205	0.206	0.617	0.191	0.757	0.778	0.292	0.308	0.155	0.394
rave	4508	1464	4904	4798	285	3867	3223	3778	3455	3818	3332	4852
grave	88	9	69	117	90	97	67	29	73	15	45	8
chargeDis.	0.928	0.945	0.950	0.989	0.996	0.984	0.980	0.901	0.923	0.931	0.964	0.995
treeDis.	0.971	0.958	0.925	0.987	0.945	0.965	0.934	0.964	0.930	0.972	0.902	0.978
aggr.	0.784	2.612	2.927	2.044	0.430	1.634	2.684	0.139	1.936	0.701	0.907	2.172
defen.	1.500	1.866	1.052	1.316	0.946	0.645	1.956	1.503	0.995	1.075	0.123	2.998
ch.Depth	65	20	94	60	19	67	34.248	21.756	62.990	29.706	75.676	30.799
horizon	9	7	2	6	17	17	15	19	8	3	3	13
randErr	0.290	0.485	0.248	0.754	0.075	0.392	0.427	0.624	0.450	0.095	0.566	0.024
ch.Default	39.706	37.803	39.729	42.191	41.370	37.540	42.251	39.168	44.157	38.380	38.980	44.756
expl.Fac.	125	81	120	49	36	35	5	13	121	47	137	12
niceTh.	38.448	49.630	45.040	8.273	46.950	28.877	18.790	61.309	19.282	45.644	41.199	35.793

Approximately, it can be seen the same behaviour as with the "correct" parameter, excused by the magnitude of the state space of the game, as discussed earlier. However, it is possible to see some sloppiness in the search selection, compared to the user's. Figure 4.13 shows the percentage of matching moves having random individuals playing over the matches that were associated to personalities from the data collection.

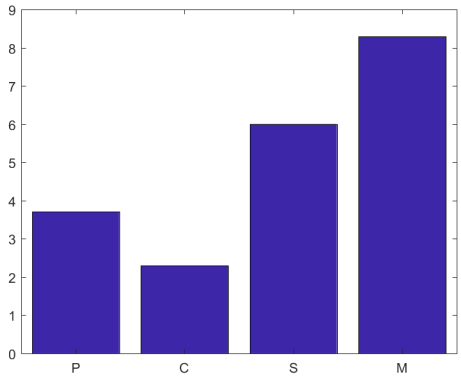


Figure 4.13: Actions Agreement Ratio for Skirmish games with random individuals, divided by personality (percent).

This result shows an higher average of agreed moves, confirming the hypothesis of the genetic algorithm hitting a local maxima and not returning the optimal individuals, as well as that the personality model adopted might not be correctly modeled.

4.4.4 Random Games with Random Individuals

Lastly, four random individuals (A, B, C, D) have been plugged in the games we have no knowledge about, and the summary of it can be seen in figure 4.14. Comparing it with figure 4.11, it can be seen that individual D has more matching moves than the Choleric individual from the genetic algorithm (which was the one with the highest frequency of match, as seen in section 4.4.2).

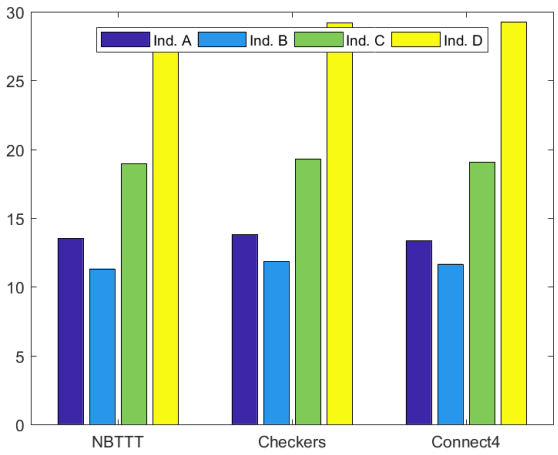


Figure 4.14: Evaluation of AAR with Random Individuals (percentage).

The meaning behind this result, in addition to what can be interpreted from the graphs in appendix B, can once more be that the personality model used for this thesis is not a true fit, as well as the genetic algorithm hit a local maxima, and

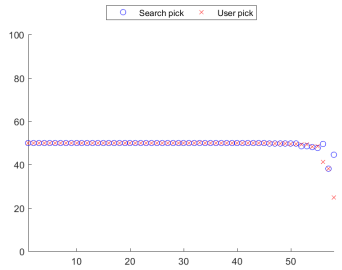
would require some more parameters' tweaking to return improved results.

4.5 Limitations and Improvements

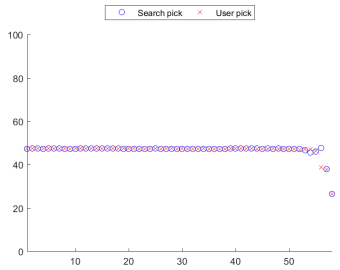
The project has been subject to limitations due to both time, and power constraints. The remarkable time complexity of the Genetic Algorithm, together with time constraints, forced us to restrain the number of tests for parameters' tweaking, increasing the chances of hitting a local maxima or having premature stagnancy. More tweaking would lead to better results, overall, but we have to bound this thesis to a few choices of Genetic Algorithm parameters. It is to be recalled also that the geometric mean plotted on the graphs turned out to be slightly different than the one calculated in the genetic algorithm. Further investigations about this matter will have to be performed. In the case this would be fixed, it would be natural to suppose that the average fitness for each generation would be closer to the probability of picking one move at random, at least at initial states.
Mostly, however, the limitation lies on the data collection. Due to non-optimal resources, the data this thesis bases itself on, for both training and evaluation, has not been enough to infer a proper state-action distribution and get properly defined results overall. If a larger dataset would be available, it would be interesting to notice how the genetic algorithm would react to a more demanding training, and if the fittest individuals would be a better match for the personality model adopted here. In the case of more training data, it could also be possible to adopt a more detailed personality model, which could help reproducing gameplay more closely. The data to feed the genetic algorithm with will have also to be from more reliable sources, as self-assigning personality might not be accurate. Furthermore, some of our players have been acting other personalities, which could have affected the data in a negative way, considering how hard it could be to suppress emotions in some cases. Lastly, understanding how the MCTS parameters can be changed, and how they are correlated to each other would supposedly help having more precise individuals as output from the GA.

4.6 Evaluation Results

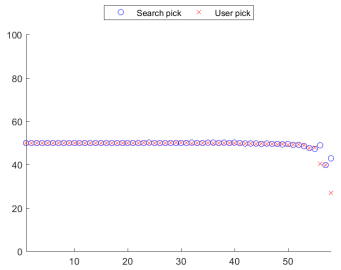
Recapitulating the results collected during the evaluation process, it can be said that the individuals returned from the genetic algorithm do not have a relevantly better match than random parameters, over games that have been divided on the basis of our personality model. However, applied to a different set of games, random individuals seem to have a better fit (as seen in section 4.4.4), leading to the idea that a different personality model might be more suitable for gaming styles representation.
It is then confirmed from section 4.4.3 that equation (3.1) is true, however a more polarised result would be ideal.



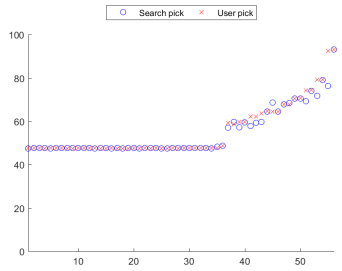
(a) Choleric Match 1 - Random Individual 1



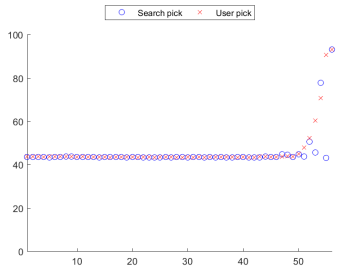
(b) Choleric Match 1 - Random Individual 2



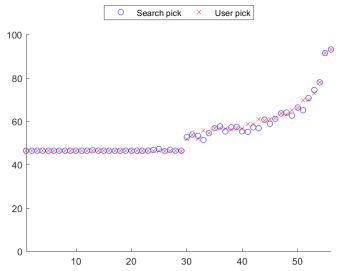
(c) Choleric Match 1 - Random Individual 3



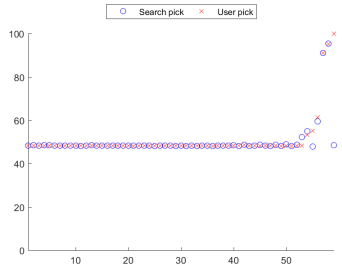
(d) Melancholic Match 1 - Random Individual 4



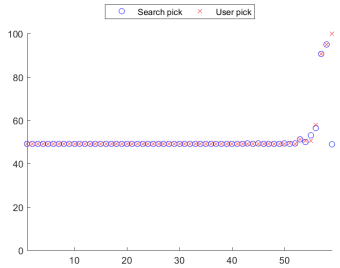
(e) Melancholic Match 1 - Random Individual 5



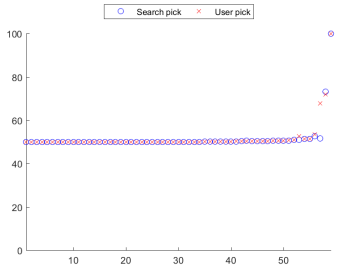
(f) Melancholic Match 1 - Random Individual 6



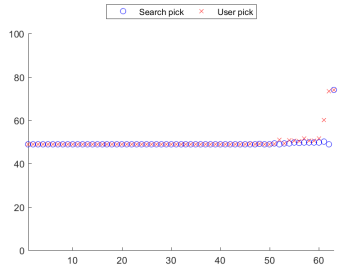
(g) Sanguine Match 1 - Random Individual 7



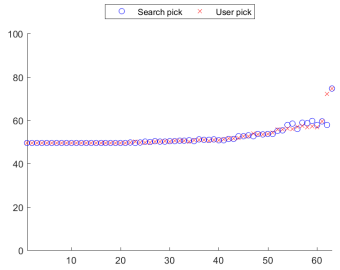
(h) Sanguine Match 1 - Random Individual 8



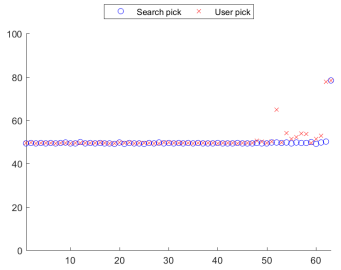
(i) Sanguine Match 1 - Random Individual 9



(j) Phlegmatic Match 1 - Random Individual 10



(k) Phlegmatic Match 1 - Random Individual 11



(l) Phlegmatic Match 1 - Random Individual 12

Figure 4.12: Evaluation of the Skirmish games with random individuals, where the moves are represented on the x-axis, and their QValue on the y-axis.

5

Conclusion

In order to infer a personality from human playing styles, a genetic algorithm has been trained with the aid of collected human-played games. The data has been gathered through the Tiltyard server[8], which is a platform for General Game Playing where to test and develop General Game Players. General Game Playing is also the framework used in this project (see section 2.3), where no knowledge of the game is priorly required. This encouraged having a relatively wide selection of games, allowing the users to select any among the games of Checkers, Connect Four, Nine-Boards Tic-Tac-Toe, and Skirmish. The users also provided information about their personality, based on the Galen-Hippocrates' Four Temperaments model (see section 2.2.8), which has purposely been chosen as model for this thesis due to its simplicity. The games collected have then been partially used to feed a genetic algorithm (see section 2.6), where each individual represented a set of parameters for a Monte Carlo Tree Search algorithm (see section 2.4). The Monte Carlo Tree Search is the main algorithm behind the artificial player used in this thesis, and has been supplied with a set of enhancements, each of which is depending on a control variable. The set of those control variables is the aforementioned individual from the genetic algorithm. Once the GA has been running on a set of game representing a specific personality, the fittest individual is selected for evaluation 3.8. This reduced to 4 different sets of parameters, one for each personality in the model, that have been plugged in the artificial player, and checked with games that have been left out from the training process.
The evaluation process consisted in calculating the Action Agreement Ratio (AAR), together with confronting the QValues of the chosen moves as Δ_Q. The results of this evaluation, collected in chapter 4, proved that many search-selected moves have an imperceptible Δ_Q, whilst still not reaching perfect matching. This could mean that the gaming style achieved is comparable to human's, although not taking the exact same decisions. However, it is still to be confirmed if it is indeed personality biased.
The results obtained in this project are promising enough to believe that a probability model could be inferred from games data. Although the dataset collected and utilised as testbed was not adequate to be considered fully statistically relevant, therefore not confirming nor disproving the thesis where a personality model can be inferred from games data, we are confident that a final conclusion can be drawn with further research.
Additional focus would be applicable in determining the best MCTS enhancements and how they affect the search; tweaking the GA parameters so to find a better environment for games data training; applying a different personality model that

could be a better representative of the players' behaviours; and implement some machine learning strategies to capture patterns in actions' selections for different players. However, none of the above would have meaningful results if the dataset would not be remarkably increased.

Bibliography

[1] GGP.org | View | Checkers. [Online] Available: http://www.ggp.org/view/tiltyard/games/base/checkers/v0/.

[2] GGP.org | View | Connect Four. [Online] Available: http://www.ggp.org/view/tiltyard/games/base/connectFour/v0/.

[3] GGP.org | View | Nine-Board Tic-Tac-Toe. [Online] Available: http://www.ggp.org/view/tiltyard/games/base/nineBoardTicTacToe/v0/.

[4] GGP.org | View | Skirmish. [Online] Available: http://www.ggp.org/view/tiltyard/games/base/skirmish/v0/.

[5] The Resistance | Board Game | BoardGameGeek. [Online] Available: https://boardgamegeek.com/boardgame/41114/resistance, Jan. 2010.

[6] CadiaPlayer. [Online] Available: http://cadia.ru.is/wiki/public:cadiaplayer:main, Dec. 2012.

[7] The Cardboard Republic | Gamers Archetypes. [Online] Available: http://www.cardboardrepublic.com/gamer-archetypes, Nov. 2012.

[8] GGP.org | Tiltyard. [Online] Available: http://tiltyard.ggp.org/, Mar. 2012.

[9] What Gamer are You? | The Cardboard Republic. [Online] Available: http://www.cardboardrepublic.com/quiz/questions.php, Nov. 2013.

[10] What kind of board game player are you? Is there a quiz for this question? | BoardGameGeek | BoardGameGeek. [Online] Available: https://boardgamegeek.com/thread/1051071/what-kind-board-game-player-are-you-there-quiz-que, Oct. 2013.

[11] GGP.org - for Researchers. [Online] Available: http://www.ggp.org/researchers, Apr. 2016.

[12] How to Play - Werewolf. [Online] Available: https://www.playwerewolf.co/rules/, Oct. 2016.

[13] Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. *Machine learning 47*, 2-3 (2002), 235–256.

[14] Baker, J. E. Adaptive selection methods for genetic algorithms. In *Proceedings of an International Conference on Genetic Algorithms and their applications* (1985), Hillsdale, New Jersey, pp. 101–111.

[15] Baluja, S., and Caruana, R. Removing the genetics from the standard genetic algorithm. In *Machine Learning: Proceedings of the Twelfth International Conference* (1995), pp. 38–46.

[16] Bartle, R. Hearts, clubs, diamonds, spades: Players who suit MUDs. *Journal of MUD research 1*, 1 (1996), 19.

[17] Bartle, R. Player Type Theory: Uses and Abuses | Richard BARTLE. [Online] Available: https://www.youtube.com/watch?v=ZIzLbE-93nc, Mar. 2012.

[18] Bellman, R. A markovian decision process. *Journal of Mathematics and Mechanics* (1957), 679–684.

[19] Bjornsson, Y., and Finnsson, H. Cadiaplayer: A simulation-based general game player. *IEEE Transactions on Computational Intelligence and AI in Games 1*, 1 (2009), 4–15.

[20] Blickle, T., and Thiele, L. A comparison of selection schemes used in genetic algorithms, 1995.

[21] Boyle, G. J. Myers-Briggs Type Indicator (MBTI): Some Psychometric Limitations. *Australian Psychologist 30*, 1 (1995), 71–74.

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

[23] Brügmann, B. Monte carlo go. Tech. rep., Technical report, Physics Department, Syracuse University Syracuse, NY, 1993.

[24] Campbella, M., Hoane Jr, A. J., and Hsuc, F.-h. Deep Blue. *Artificial Intelligence 134* (2002), 57–83.

[25] Cazenave, T. Generalized rapid action value estimation. In *IJCAI* (2015), vol. 15, pp. 754–760.

[26] Chaslot, G. Monte-carlo tree search. *Maastricht: Universiteit Maastricht* (2010).

[27] Chaslot, G. M. J., Winands, M. H., HERIK, H. J. V. D., Uiterwijk, J. W., and Bouzy, B. Progressive strategies for monte-carlo tree search. *New Mathematics and Natural Computation 4*, 03 (2008), 343–357.

[28] Chiappelli, F., Prolo, P., and Cajulis, O. S. Evidence-based research in complementary and alternative medicine in history. *Evidence-based Complementary and Alternative Medicine 2*, 4 (2005), 453–458.

[29] Chipperfield, A. Introduction to genetic algorithms. *IEE CONTROL ENGINEERING SERIES* (1997), 1–2.

[30] Cho, B. H., Jung, S. H., Seong, Y. R., and Oh, H. R. Exploiting intelligence in fighting action games using neural networks. *IEICE transactions on information and systems 89*, 3 (2006), 1249–1256.

[31] Coulom, R. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games* (2006), Springer, pp. 72–83.

[32] Crotti, S. Human general game play tournament - Instructions. [Online] Available: https://goo.gl/ifUhRz, May 2017.

[33] Devlin, S., Anspoka, A., Sephton, N., Cowling, P. I., and Rollason, J. Combining gameplay data with monte carlo tree search to emulate human play.

[34] Durupinar, F., Allbeck, J., Pelechano, N., and Badler, N. Creating crowd variation with the ocean personality model. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent*

systems-Volume 3 (2008), International Foundation for Autonomous Agents and Multiagent Systems, pp. 1217–1220.

[35] Enzenberger, M., Muller, M., Arneson, B., and Segal, R. Fuego - an open-source framework for board games and go engine based on monte carlo tree search. *IEEE Transactions on Computational Intelligence and AI in Games 2*, 4 (2010), 259–270.

[36] Finnsson, H. CADIA-Player: A General Game Playing Agent.

[37] Finnsson, H. Generalized monte-carlo tree search extensions for general game playing. In *AAAI* (2012).

[38] Finnsson, H., and Björnsson, Y. Simulation-based approach to general game playing. In *AAAI* (2008), vol. 8, pp. 259–264.

[39] Finnsson, H., and Björnsson, Y. Simulation control in general game playing agents. In *Proc. Int. Joint Conf. Artif. Intell. Workshop Gen. Game Playing, Pasadena, California* (2009), pp. 21–26.

[40] Finnsson, H., and Björnsson, Y. Learning simulation control in general game-playing agents. In *AAAI* (2010), vol. 10, pp. 954–959.

[41] Foundry, Q. Example of Board Game Motivation Profile report. [Online] Available: https://apps.quanticfoundry.com/tabletop/demo/1/, Sept. 2016.

[42] Freed, M., Bear, T., Goldman, H., Hyatt, G., Reber, P., and Tauber, J. Towards more human-like computer opponents. In *Working Notes of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment* (2000), no. 1, pp. 22–26.

[43] Gelly, S., and Silver, D. Combining online and offline knowledge in uct. In *Proceedings of the 24th international conference on Machine learning* (2007), ACM, pp. 273–280.

[44] Gelly, S., and Silver, D. Monte-carlo tree search and rapid action value estimation in computer go. *Artificial Intelligence 175*, 11 (2011), 1856–1875.

[45] Gelly, S., Wang, Y., Teytaud, O., Patterns, M. U., and Tao, P. Modification of uct with patterns in monte-carlo go.

[46] Genesereth, M., and Björnsson, Y. The international general game playing competition. *AI Magazine 34*, 2 (2013), 107.

[47] Genesereth, M., Love, N., and Pell, B. General game playing: Overview of the aaai competition. *AI magazine 26*, 2 (2005), 62.

[48] Genesereth, M., and Thielscher, M. General game playing. *Synthesis Lectures on Artificial Intelligence and Machine Learning 8*, 2 (2014), 1–229.

[49] Gold, K. Trigram Timmies and Bayesian Johnnies: Probabilistic Models of Personality in Dominion. In *AIIDE* (2011).

[50] Goldberg, L. R. The structure of phenotypic personality traits. *American psychologist 48*, 1 (1993), 26.

[51] Gorman, B., Thurau, C., Bauckhage, C., and Humphrys, M. Believability testing and bayesian imitation in interactive computer games. In *International Conference on Simulation of Adaptive Behavior* (2006), Springer, pp. 655–666.

[52] Helgadóttir, H. E., Jónsdóttir, S., Sigurdsson, A. M., Schiffel, S., and Vilhjálmsson, H. H. Virtual General Game Playing Agent. In *In-*

ternational Conference on Intelligent Virtual Agents (2016), Springer, pp. 464–469.

[53] Helmbold, D. P., and Parker-Wood, A. All-moves-as-first heuristics in monte-carlo go. In *IC-AI* (2009), pp. 605–610.

[54] Holland, J. H. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[55] Holmgård, C. Procedural personas for player decision modeling and procedural content generation.

[56] Holmgård, C., Liapis, A., Togelius, J., and Yannakakis, G. N. Monte-carlo tree search for persona based player modeling. In *Eleventh Artificial Intelligence and Interactive Digital Entertainment Conference* (2015).

[57] Hopcroft, J. E., Motwani, R., and Ullman, J. D. Automata theory, languages, and computation. *International Edition 24* (2006).

[58] Hosu, I.-A., and Rebedea, T. Playing atari games with deep reinforcement learning and human checkpoint replay. *arXiv preprint arXiv:1607.05077* (2016).

[59] Isaksen, A., Gopstein, D., and Nealen, A. Exploring game space using survival analysis. In *FDG* (2015).

[60] Kahneman, D. *Thinking, fast and slow*. Macmillan, 2011.

[61] Keirsey, D. *Please understand me 2*. Prometheus Nemesis Book Company, 1998.

[62] Khalifa, A., Isaksen, A., Togelius, J., and Nealen, A. Modifying mcts for human-like general video game playing.

[63] Kleinberg, J., and Tardos, E. *Algorithm design*. Pearson Education India, 2006.

[64] Kocsis, L., and Szepesvári, C. Bandit based monte-carlo planning. In *ECML* (2006), vol. 6, Springer, pp. 282–293.

[65] Kocsis, L., Szepesvári, C., and Willemson, J. Improved monte-carlo search. *Univ. Tartu, Estonia, Tech. Rep 1* (2006).

[66] Ladkany, G. S., and Trabia, M. B. A genetic algorithm with weighted average normally-distributed arithmetic crossover and twinkling. *Applied Mathematics 3*, 10 (2012), 1220.

[67] Lai, T. L., and Robbins, H. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics 6*, 1 (1985), 4–22.

[68] Lorentz, R. Amazons discover monte-carlo. *Computers and games* (2008), 13–24.

[69] Love, N., Hinrichs, T., Haley, D., Schkufza, E., and Genesereth, M. General game playing: Game description language specification, 2008.

[70] Martinez-Arellano, G., Cant, R., and Woods, D. Creating ai characters for fighting games using genetic programming. *IEEE Transactions on Computational Intelligence and AI in Games* (2016).

[71] McCrae, R. R., and Costa, P. T. Reinterpreting the Myers-Briggs type indicator from the perspective of the five-factor model of personality. *Journal of personality 57*, 1 (1989), 17–40.

[72] Merenda, P. F. Toward a four-factor theory of temperament and/or personality. *Journal of personality assessment 51*, 3 (1987), 367–374.

[73] Miller, B. L., Goldberg, D. E., et al. Genetic algorithms, tournament selection, and the effects of noise. *Complex systems 9*, 3 (1995), 193–212.

[74] Miranda, M., Sánchez-Ruiz, A. A., and Peinado, F. A neuroevolution approach to imitating human-like play in ms. pac-man video game.

[75] Mitchell, M. *An introduction to genetic algorithms*. MIT press, 1998.

[76] Murphy, K. P. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[77] Myers, I., and Myers, P. *Gifts differing: Understanding personality type*. Nicholas Brealey Publishing, 2010.

[78] Myers-Briggs Foundation, T. MBTI basics. [Online] Available: http://www.myersbriggs.org/my-mbti-personality-type/mbti-basics/, Dec. 2004.

[79] Nacke, L. E., Bateman, C., and Mandryk, R. L. BrainHex: preliminary results from a neurobiological gamer typology survey. In *International Conference on Entertainment Computing* (2011), Springer, pp. 288–293.

[80] Ng, A. Y., Russell, S. J., et al. Algorithms for inverse reinforcement learning. In *Icml* (2000), pp. 663–670.

[81] Nick Yee, N. D. The Quantic Lab. [Online] Available: https://apps.quanticfoundry.com/lab/, July 2015.

[82] Norris, J. R. *Markov chains*. No. 2. Cambridge university press, 1998.

[83] Oracle. Primitive data types (the java™ tutorials > learning the java language > language basics). [Online] Available: https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html, Aug. 2006.

[84] Pelikan, M., Goldberg, D. E., and Tsutsui, S. Hierarchical bayesian optimization algorithm: Toward a new generation of evolutionary algorithms. In *SICE 2003 Annual Conference* (2003), vol. 3, IEEE, pp. 2738–2743.

[85] Pell, B. *Strategy generation and evaluation for meta-game playing*. PhD thesis, University of Cambridge Cambridge, England, 1993.

[86] Perez-Liebana, D., Samothrakis, S., Togelius, J., Schaul, T., Lucas, S. M., Couëtoux, A., Lee, J., Lim, C.-U., and Thompson, T. The 2014 general video game playing competition. *IEEE Transactions on Computational Intelligence and AI in Games 8*, 3 (2016), 229–243.

[87] Pittenger, D. J. Measuring the MBTI... and coming up short. *Journal of Career Planning and Employment 54*, 1 (1993), 48–52.

[88] Puterman, M. L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[89] Rosewater, M. Timmy, Johnny, and Spike. *Wizards of the Coast 11* (2002).

[90] Rosewater, M. Timmy, Johnny, and Spike Revisited. *Wizards of the Coast* (2006).

[91] Rothkopf, C. A., and Dimitrakakis, C. Preference elicitation and inverse reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (2011), Springer, pp. 34–48.

[92] Russell, S., Norvig, P., and Intelligence, A. A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs 25* (1995), 27.

[93] Santo, W. L., and Wong, A. Evaluating mcts in a new ai framework for hearts.

[94] Schaeffer, J. The history heuristic and alpha-beta search enhancements in practice. *IEEE transactions on pattern analysis and machine intelligence 11*, 11 (1989), 1203–1212.

[95] Schaeffer, J., and Van den Herik, H. J. Games, computers, and artificial intelligence. *Artificial Intelligence 134*, 1-2 (2002), 1–7.

[96] Schiffel, S., and Thielscher, M. Reasoning About General Games Described in GDL-II. In *AAAI* (2011), Citeseer.

[97] Scholz, R. W. Decision making under uncertainty. *Decision Making under Uncertainty: Cognitive Decision Research, Social Interaction, Development and Epistemology 16* (1983), 3.

[98] Schreiber, S., and Landau, A. "general Game Playing Base Package. `https://github.com/ggp-org/ggp-base`, 2017.

[99] Shivaswamy, P., and Joachims, T. Multi-armed bandit problems with history. In *Artificial Intelligence and Statistics* (2012), pp. 1046–1054.

[100] Silver, D., and Hassabis, D. Research Blog: AlphaGo: Mastering the ancient game of Go with Machine Learning. *Google Research Blog*.

[101] Simon, H. A., and Schaeffer, J. The game of chess. *Handbook of game theory with economic applications 1* (1992), 1–17.

[102] Soni, B., and Hingston, P. Bots trained to play like a human are more fun. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* (2008), IEEE, pp. 363–369.

[103] Spronck, P., Sprinkhuizen-Kuyper, I., and Postma, E. Improving opponent intelligence through offline evolutionary learning. *International Journal of Intelligent Games and Simulation 2*, 1 (2003), 20–27.

[104] Spronck, P., Sprinkhuizen-Kuyper, I., and Postma, E. Online adaptation of game opponent ai in simulation and in practice. In *Proceedings of the 4th International Conference on Intelligent Games and Simulation* (2003), pp. 93–100.

[105] Stewart, B. Personality and play styles: A unified model. *Gamasutra, September 1* (2011).

[106] Szita, I., Chaslot, G., and Spronck, P. Monte-carlo tree search in settlers of catan. *ACG 6048* (2009), 21–32.

[107] Thielscher, M. A General Game Description Language for Incomplete Information Games. In *AAAI* (2010), vol. 10, Citeseer, pp. 994–999.

[108] Thurau, C., Bauckhage, C., and Sagerer, G. Learning human-like movement behavior for computer games. In *Proc. Int. Conf. on the Simulation of Adaptive Behavior* (2004), pp. 315–323.

[109] Togelius, J., Yannakakis, G. N., Stanley, K. O., and Browne, C. Search-based procedural content generation: A taxonomy and survey. *IEEE Transactions on Computational Intelligence and AI in Games 3*, 3 (2011), 172–186.

[110] Turner, B. M., and Sederberg, P. B. Approximate bayesian computation with differential evolution. *Journal of Mathematical Psychology 56*, 5 (2012), 375 – 385.

[111] Umbarkar, A., and Sheth, P. Crossover operators in genetic algorithms: A review. *ICTACT journal on soft computing 6*, 1 (2015).

[112] Wahde, M. *Biologically inspired optimization methods: an introduction*. WIT press, 2008.

[113] Wawrzynczak, A., Jaroszynski, M., and Borysiewicz, M. Data-driven genetic algorithm in bayesian estimation of the abrupt atmospheric contamination source. In *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on* (2014), IEEE, pp. 519–527.

[114] West, D. B., et al. *Introduction to graph theory*, vol. 2. Prentice hall Upper Saddle River, 2001.

[115] Winands, M. H., and Björnsson, Y. Evaluation function based monte-carlo loa. In *Advances in Computer Games* (2009), Springer, pp. 33–44.

[116] Winands, M. H., Bjornsson, Y., and Saito, J.-T. Monte carlo tree search in lines of action. *IEEE Transactions on Computational Intelligence and AI in Games 2*, 4 (2010), 239–250.

[117] Yee, N. The Gamer Motivation Profile: What We Learned From 250,000 Gamers. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play* (2016), ACM, pp. 2–2.

[118] Zanetti, S., and Rhalibi, A. E. Machine learning techniques for fps in q3. In *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology* (2004), ACM, pp. 239–244.

A

Preliminaries

In this appendix we have decided to add a summary of the basic notions needed to fully understand the development of this project.

A.1 Data Structures

We will define here the data structures that will be come across during this thesis. The definitions are adapted from [63],[92] and [114].

A.1.1 Graphs

Definition A.1 *A **graph** G is a triple consisting of a collection V of **nodes**, or **vertex**; a set E of **edges**, and a relation that associates with each edge two vertices (not necessarily distinct) called its **endpoints**. In other words, we represent an edge $e \in E$ as a two-element subset of $V : e = u, v$ for some $u, v \in V$, where u and v are the **endpoints** of e.*

A.1.2 Directed Graphs

Definition A.2 *A **directed graph** G' is a graph with a set of nodes V, and a set of **directed edges** E'. Each $e' \in E$ is an **ordered pair (u,v)**. The roles of the nodes are not interchangeable, and we call u the **tail** of the edge, and v the **head**. We also say that an edge is an edge **from** its tail **to** its head.*

A.1.3 Paths and cycles

Definition A.3 *A **path** in a graph $G = (V, E)$ is a sequence H of nodes $v_1, v_2, ..., v_{k-1}, v_k$ with the property that each consecutive pair v_i, v_{i+1} is joined by an edge in G.*
Definition A.4 *A **cycle** is a path $v_1, v_2, ..., v_{k-1}, v_k$ in which $k > 2$, the first $k - 1$ nodes are all distinct, and $v_k = v_1$. A graph with no cycle is **acyclic**.*

A.1.4 Connected Graphs

Definition A.5 *A graph G is **connected** if each pair of vertices in G belongs to a path; otherwise, G is **disconnected**.*
Definition A.6 *A directed graph is **strongly connected** if, for every two nodes u and v, there is a path from u to v and a path from v to u.*

A.1.5 Trees

Definition A.7 *A **tree** T is a connected acyclic graph. Picked a node r as **root**, each other edge is oriented away from it. For each node v, a **parent** (or **ancestor**) of v is defined as the node u that directly precedes v on its path from r. A node w is defined **child** (or **descendant**) of v if v is the parent of w. A **leaf** is a vertex with no descendants.*

A.1.6 State Space

Definition A.8 *The **state space** of a problem is the set of all the states reachable from the initial state by any sequence of actions.*

A.2 Finite State-Machines

State machines are defined as sets of states connected through transitions, which are controlled by inputs. Here we will define them differentiating whether they are deterministic, an example of which can be seen in figure A.1, or non-deterministic, as in figure A.2. Definitions are adapted from [57]; we will be using the terms "state machine" and "automaton" interchangeably.

A.2.1 Deterministic Finite Automata (DFA)

Definition A.9 *A **Deterministic Finite Automata** is a tuple $< S, \Sigma, \delta, s_0, \Gamma >$, where:*

- *A finite set of **states** S.*
- *A finite set of **input symbols**, or **alphabet**, Σ.*
- *A **transition function** that takes as argument a state and an input symbol, and returns a state. The transition function will commonly be denoted δ, and $\delta : S \times \Sigma \to S$. In informal graph representation of automata, δ was represented by arcs between states and the labels on the arcs. If s is a state, and σ is an input symbol, then $\delta(s, \sigma)$ is the state q such that there is an arc labeled σ from s to q.*
- *A **start state** s_0, one of the states in S.*
- *A set of **final** or **accepting states** Γ, where $\Gamma \subseteq S$.*

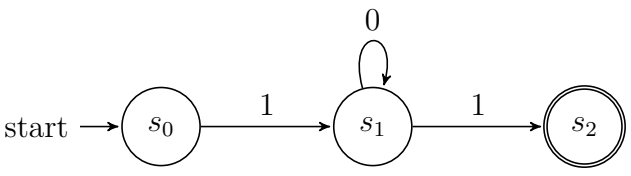


Figure A.1: Example of simple DFA.

A.2.2 Non-Deterministic Finite Automata (NFA)

Definition A.10 *A **Non-Deterministic Finite Automata** is a tuple* $< S, \Sigma, \delta, s_0, \Gamma >$*, where:*

- *A finite set of **states** S.*
- *A finite set of **input symbols**, or **alphabet**, Σ.*
- *A **transition function** δ that takes as argument a state and an input symbol, and returns a subset of states, $\delta : S \times \Sigma \to \mathcal{P}(S)$.*
- *A **start state** s_0, one of the states in S.*
- *A set of **final** or **accepting states** Γ, where $\Gamma \subseteq S$.*

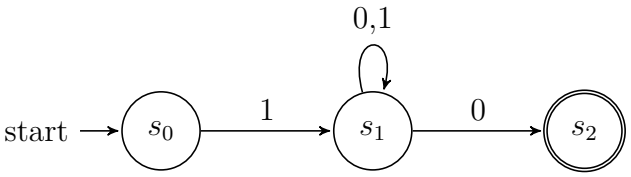


Figure A.2: Example of simple NFA.[1]

A.3 Probability

Here is a small recap on the probability basics needed to understand the logic behind the probability model in section 3.6. This appendix will not contain more formulas and definition than strictly needed, so we refer to [76] for further information. In this thesis, $P(\mathcal{E})$ indicates the probability of event $\mathcal{E}$ being true. It is known that $0 \geq P(\mathcal{E}) \geq 1$, where $P(\mathcal{E}) = 1$ means the event will certainly happen.

A.3.1 Elementary rules

Definition A.11 (Joint Probability) *Given two events $\mathcal{A}$ and $\mathcal{B}$, the probability of both happening jointly is given by the product rule:*

$$P(\mathcal{A} \wedge \mathcal{B}) = P(\mathcal{A}, \mathcal{B}) = P(\mathcal{A}|\mathcal{B})P(\mathcal{B}).$$

Definition A.12 (Sum Rule (Marginal Distribution)) *Given a joint distribution of two events $\mathcal{A}, \mathcal{B}$, the marginal distribution is defined as:*

$$P(\mathcal{A}) = \sum_{\mathcal{b}} P(\mathcal{A}, \mathcal{B}) = \sum_{\mathcal{b}} P(\mathcal{A}|\mathcal{B} = \mathcal{b})P(\mathcal{B} = \mathcal{b}).$$

Definition A.13 (Chain Rule) *Given a set of $\mathcal{N}$ events $\mathcal{A}_{[1,\ldots,\mathcal{N}]}$ the probability of them happening jointly is given by the chain rule:*

$$P(\mathcal{A}_{[1,\ldots,\mathcal{N}]}) = P(\mathcal{A}_1)P(\mathcal{A}_2|\mathcal{A}_1)\ldots P(\mathcal{A}_\mathcal{N}|\mathcal{A}_{\mathcal{N}-1}, \mathcal{A}_{\mathcal{N}-2}, \ldots, \mathcal{A}_1).$$

[1] Note that for state s_1 the input 0 can lead indistinctly to either state s_1 or s_2.

Definition A.14 (Conditional Probability) *Given two events $\mathcal{A}$ and $\mathcal{B}$, the probability of event $\mathcal{A}$, given that $\mathcal{B}$ is true, is:*

$$P(\mathcal{A}|\mathcal{B}) = \frac{P(\mathcal{A}, \mathcal{B})}{P(\mathcal{B})} \qquad \text{if } P(\mathcal{B} > 0).$$

Definition A.15 (Bayes' Rule) *Given two events $\mathcal{A}$ and $\mathcal{B}$,*

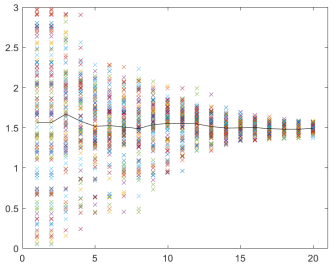
$$P(\mathcal{A} = a|\mathcal{B} = b) = \frac{P(\mathcal{A} = a, \mathcal{B} = b)}{P(\mathcal{B} = b)} = \frac{P(\mathcal{A} = a)P(\mathcal{B} = b|\mathcal{A} = a)}{\sum_a' P(\mathcal{A} = a')P(\mathcal{B} = b|\mathcal{A} = a')}.$$

B

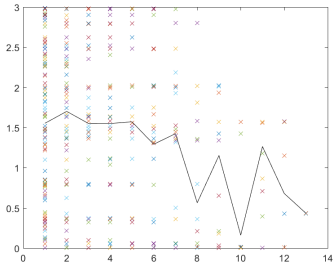
Collection of Graphs

In this appendix chapter it is possible to find a collection of graphs showing the development of the single genes in each individual in the population, throughout the generations. On the y-axis it will be the values accepted for the specific gene, while the x-axis will indicate the generations. The blue line in each of those graphs represents the average value. The discussion over the following graphs can be found in section 4.3.
Hereafter, there are scatter plots of the single genes in each individuals correlated to their fitness value. X-axis represents the interval the gene's values are in, while Y-axis represents a variation of the fitness value. We refer once more to section 4.3 for further information.
The graphs relative to the evaluation process can be found in this appendix as well, where the x-axis indicates the turns of the game, and the y-axis the Q-value of the actions selected by the search and the users. The graphs will be further discussed in section 4.4.

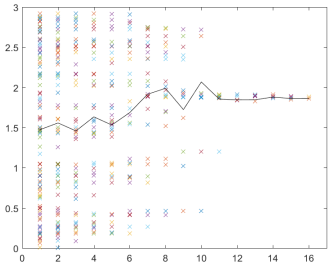
B.1 Single Individuals



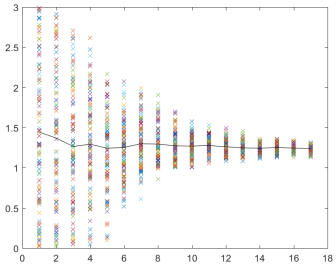
(a) Choleric - I run



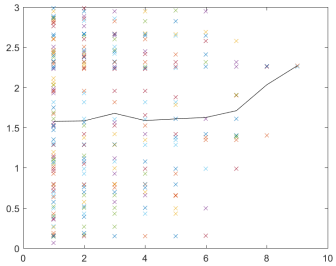
(b) Choleric - II run



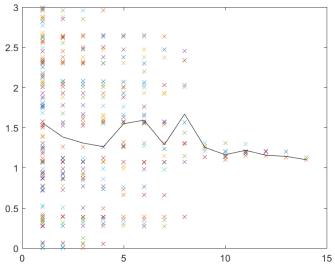
(c) Choleric - III run



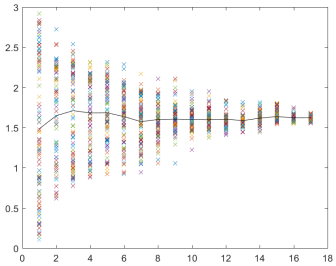
(d) Melancholic - I run



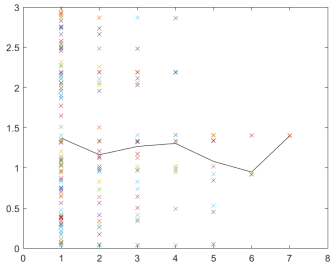
(e) Melancholic - II run



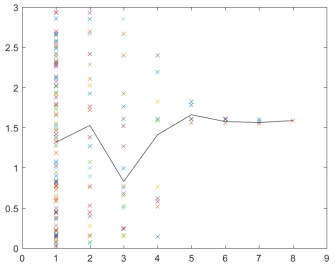
(f) Melancholic - III run



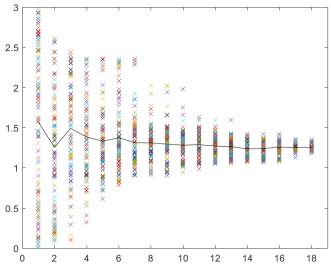
(g) Sanguine - I run



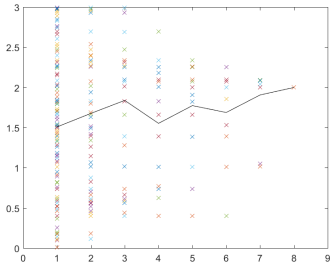
(h) Sanguine - II run



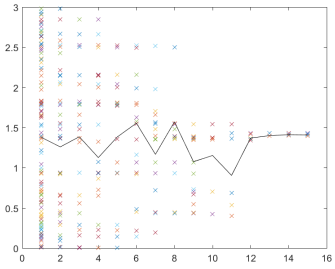
(i) Sanguine - III run



(j) Phlegmatic - I run

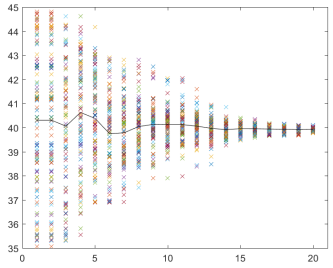


(k) Phlegmatic - II run

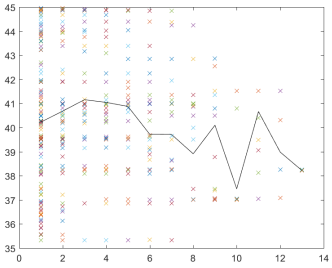


(l) Phlegmatic - III run

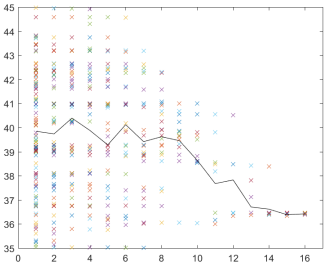
Figure B.1: Aggressiveness gene, with generations on x-axis, and value of each individual on y-axis.



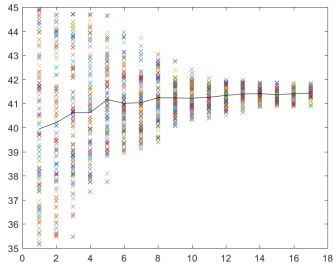
(a) Choleric - I run



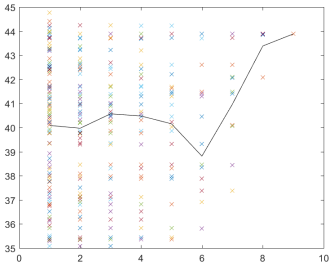
(b) Choleric - II run



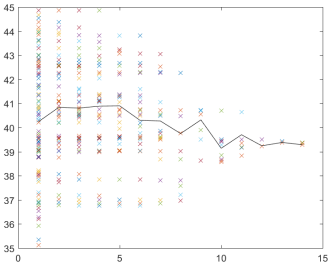
(c) Choleric - III run



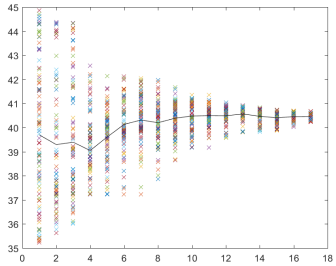
(d) Melancholic - I run



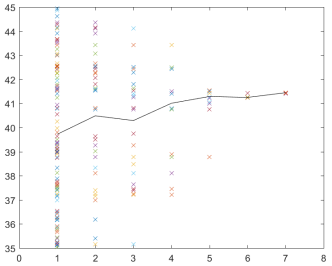
(e) Melancholic - II run



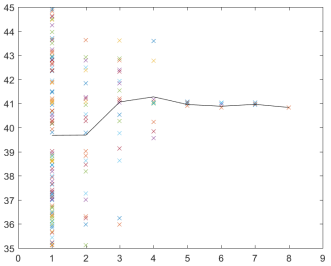
(f) Melancholic - III run



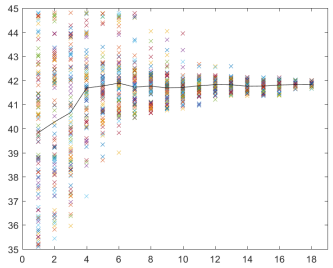
(g) Sanguine - I run



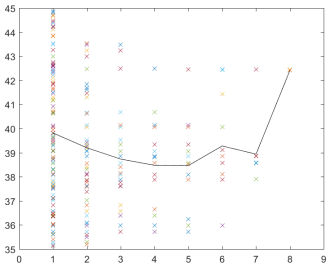
(h) Sanguine - II run



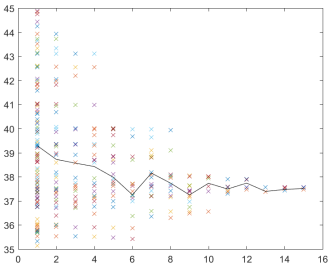
(i) Sanguine - III run



(j) Phlegmatic - I run

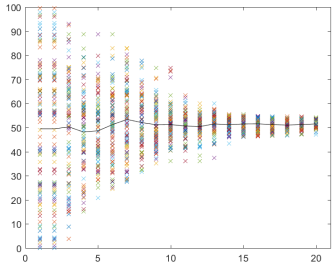


(k) Phlegmatic - II run

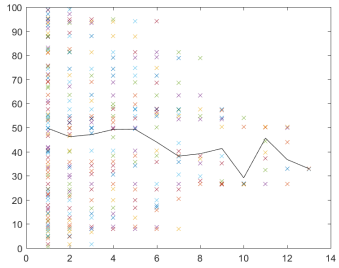


(l) Phlegmatic - III run

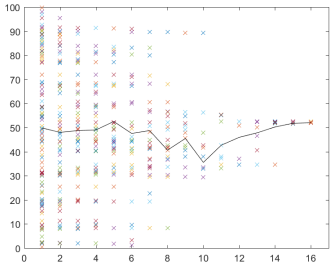
Figure B.2: Charge Default gene, with generations on x-axis, and value of each individual on y-axis.



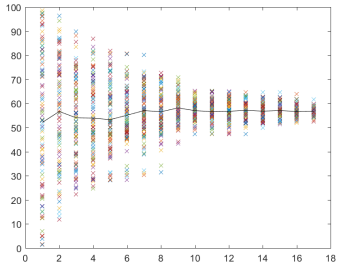
(a) Choleric - I run



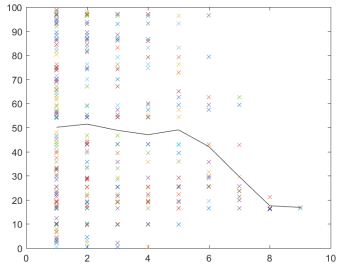
(b) Choleric - II run



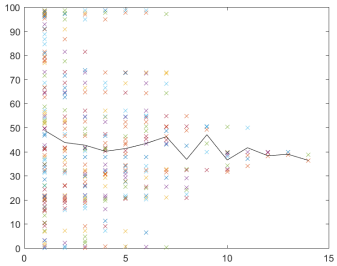
(c) Choleric - III run



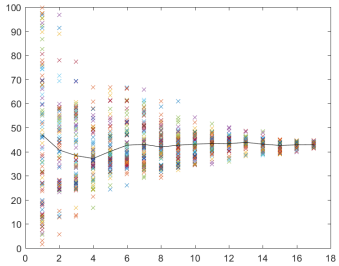
(d) Melancholic - I run



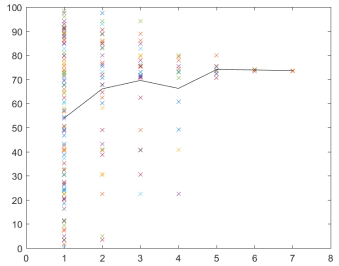
(e) Melancholic - II run



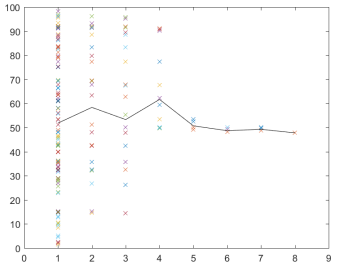
(f) Melancholic - III run



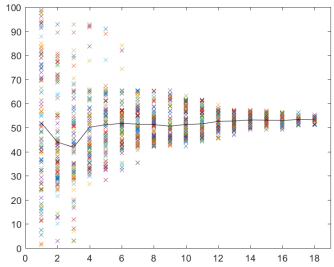
(g) Sanguine - I run



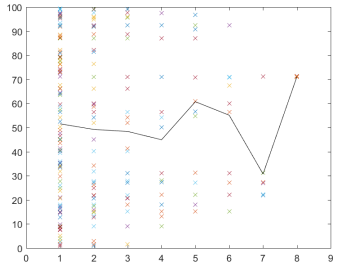
(h) Sanguine - II run



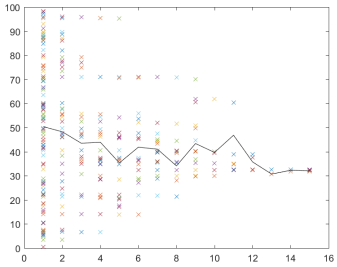
(i) Sanguine - III run



(j) Phlegmatic - I run

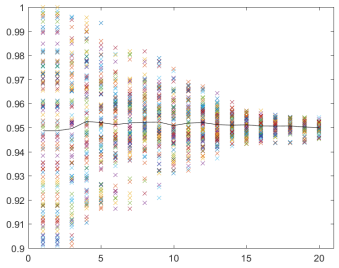


(k) Phlegmatic - II run

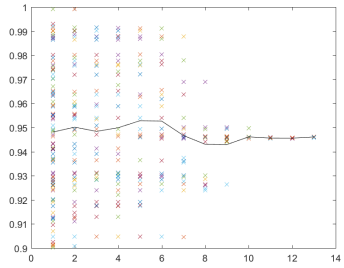


(l) Phlegmatic - III run

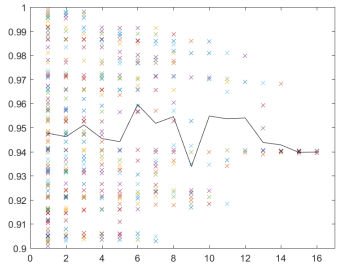
Figure B.3: Charge Depth gene, with generations on x-axis, and value of each individual on y-axis.



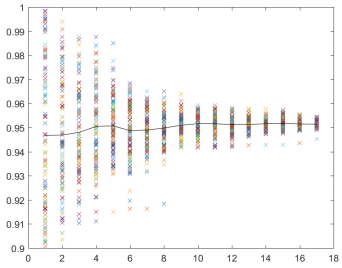
(a) Choleric - I run



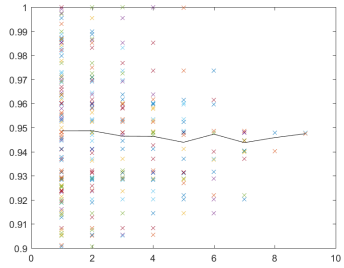
(b) Choleric - II run



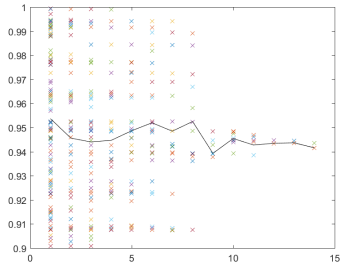
(c) Choleric - III run



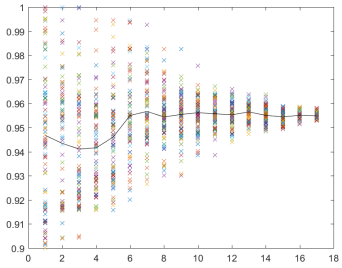
(d) Melancholic - I run



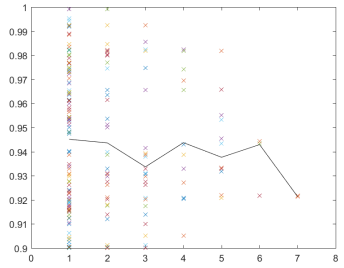
(e) Melancholic - II run



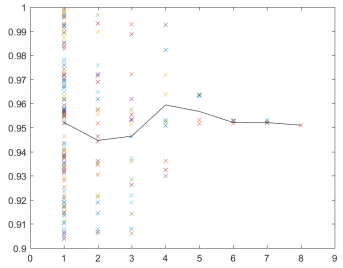
(f) Melancholic - III run



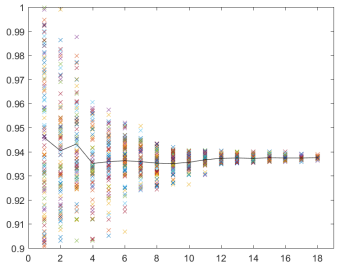
(g) Sanguine - I run



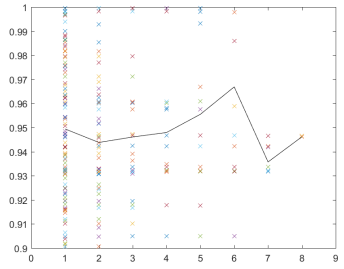
(h) Sanguine - II run



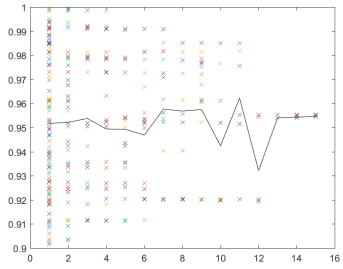
(i) Sanguine - III run



(j) Phlegmatic - I run

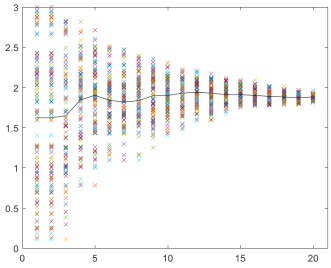


(k) Phlegmatic - II run

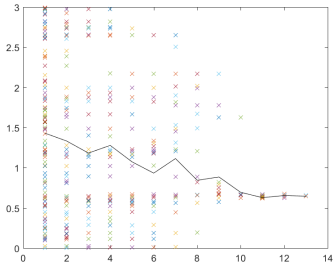


(l) Phlegmatic - III run

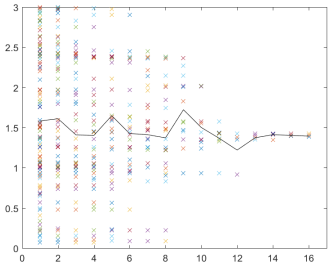
Figure B.4: Charge Discount gene, with generations on x-axis, and value of each individual on y-axis.



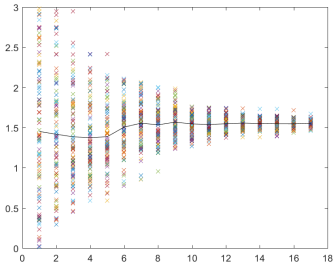
(a) Choleric - I run



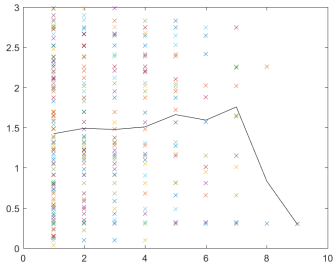
(b) Choleric - II run



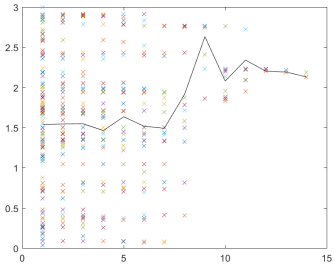
(c) Choleric - III run



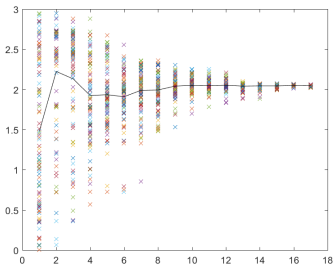
(d) Melancholic - I run



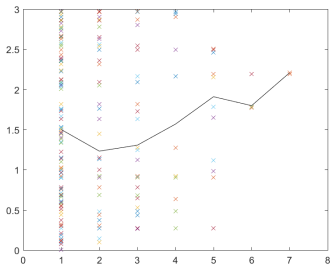
(e) Melancholic - II run



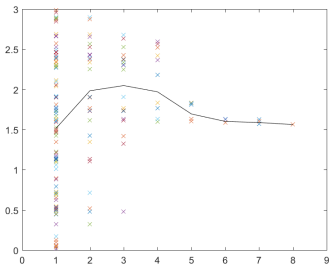
(f) Melancholic - III run



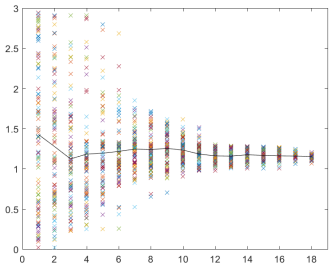
(g) Sanguine - I run



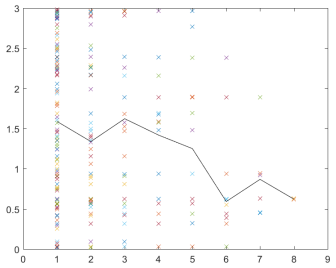
(h) Sanguine - II run



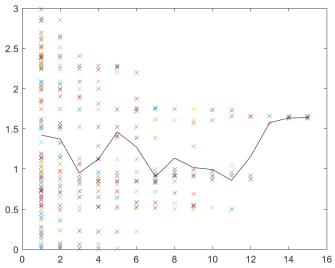
(i) Sanguine - III run



(j) Phlegmatic - I run

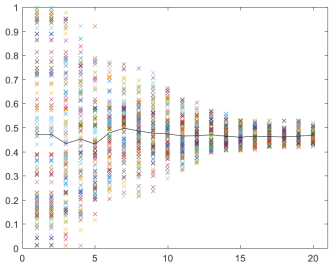


(k) Phlegmatic - II run

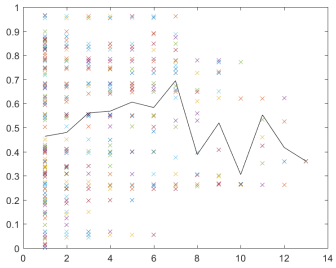


(l) Phlegmatic - III run

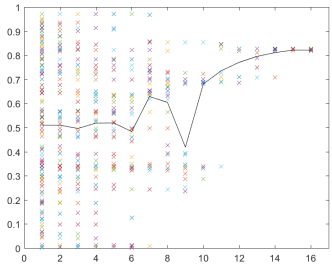
Figure B.5: Defensiveness gene, with generations on x-axis, and value of each individual on y-axis.



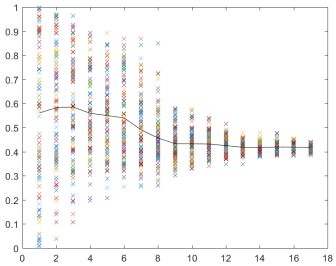
(a) Choleric - I run



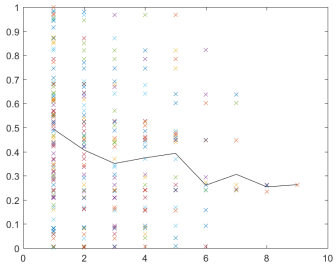
(b) Choleric - II run



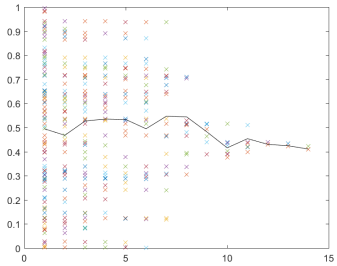
(c) Choleric - III run



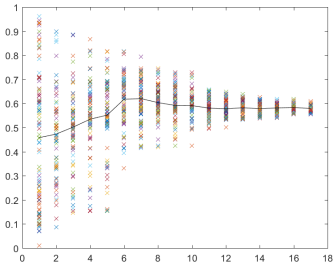
(d) Melancholic - I run



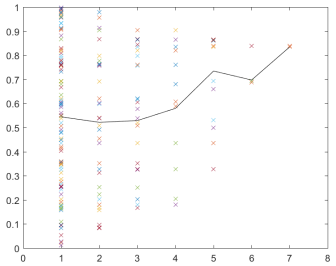
(e) Melancholic - II run



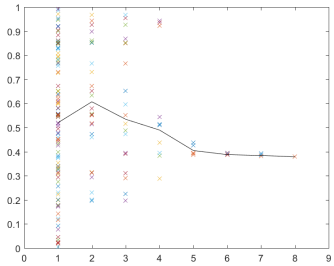
(f) Melancholic - III run



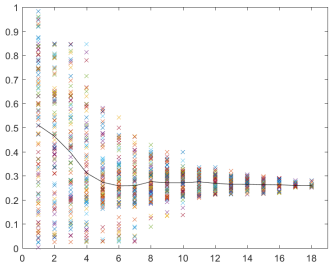
(g) Sanguine - I run



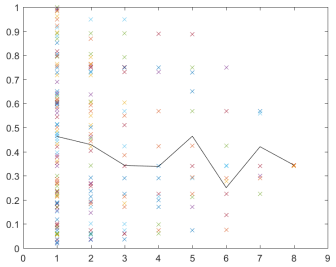
(h) Sanguine - II run



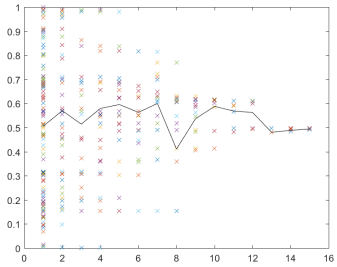
(i) Sanguine - III run



(j) Phlegmatic - I run

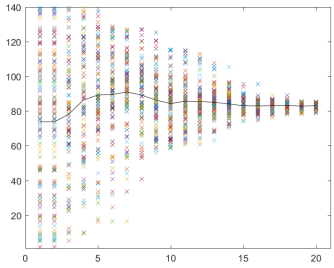


(k) Phlegmatic - II run

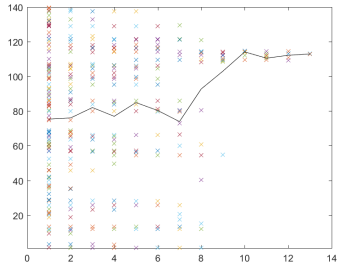


(l) Phlegmatic - III run

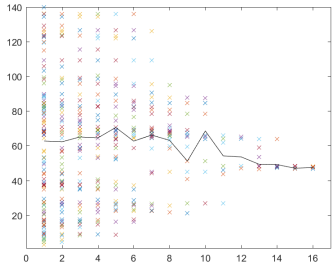
Figure B.6: Epsilon gene, with generations on x-axis, and value of each individual on y-axis.



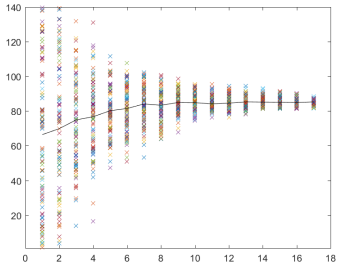
(a) Choleric - I run



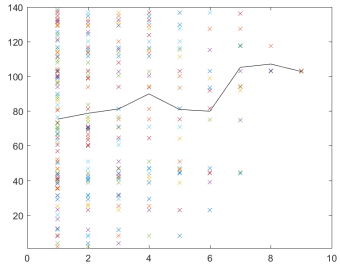
(b) Choleric - II run



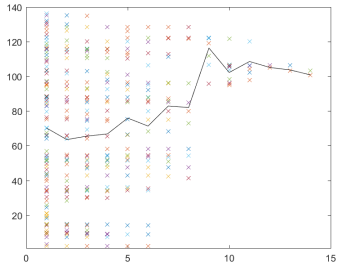
(c) Choleric - III run



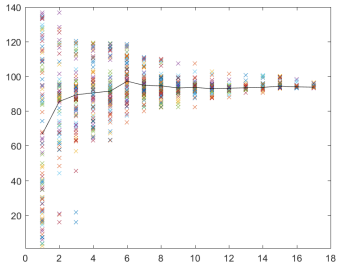
(d) Melancholic - I run



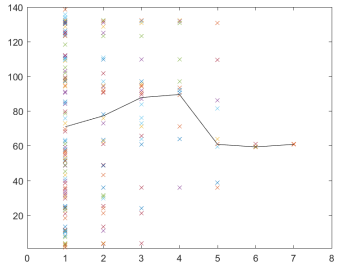
(e) Melancholic - II run



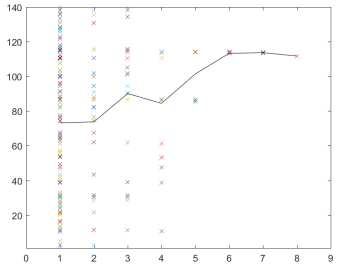
(f) Melancholic - III run



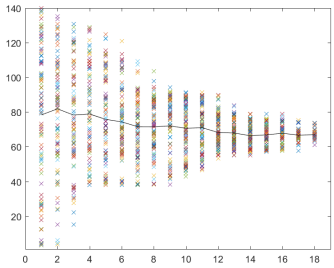
(g) Sanguine - I run



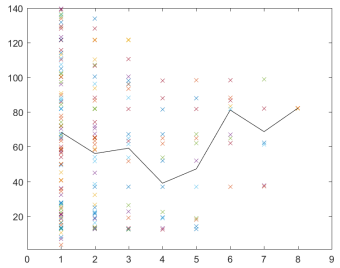
(h) Sanguine - II run



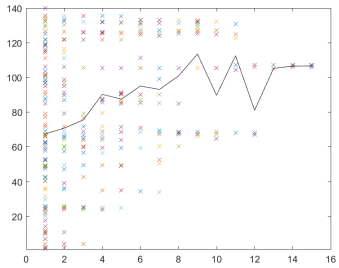
(i) Sanguine - III run



(j) Phlegmatic - I run

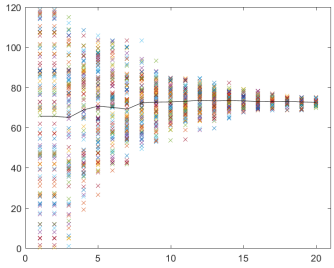


(k) Phlegmatic - II run

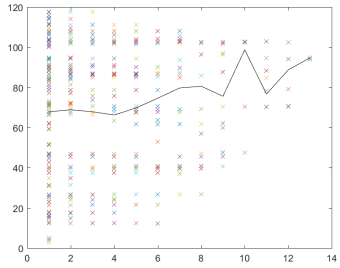


(l) Phlegmatic - III run

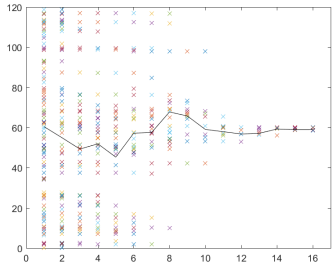
Figure B.7: Exploration Factor gene, with generations on x-axis, and value of each individual on y-axis.



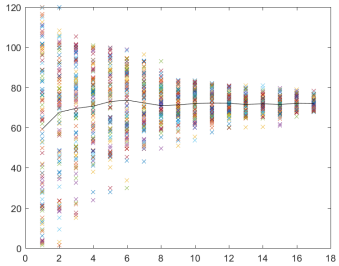
(a) Choleric - I run



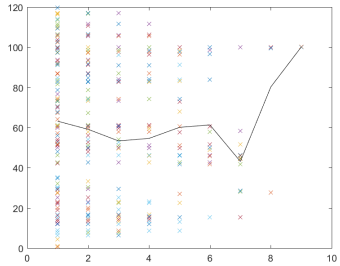
(b) Choleric - II run



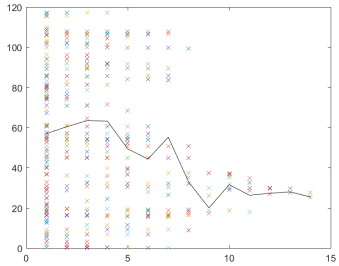
(c) Choleric - III run



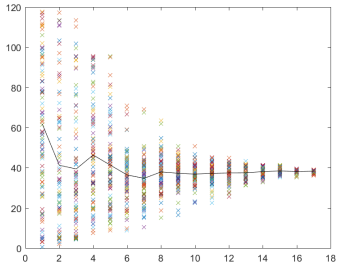
(d) Melancholic - I run



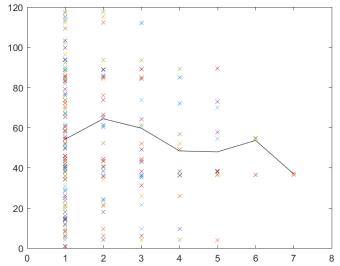
(e) Melancholic - II run



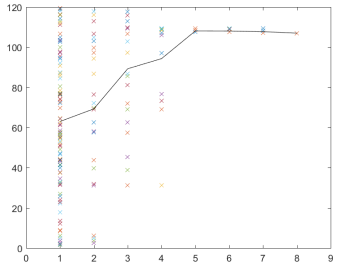
(f) Melancholic - III run



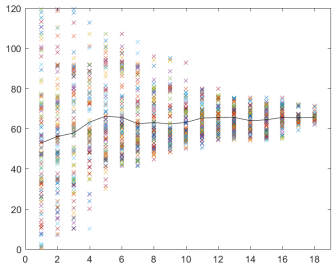
(g) Sanguine - I run



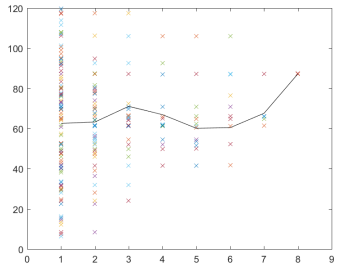
(h) Sanguine - II run



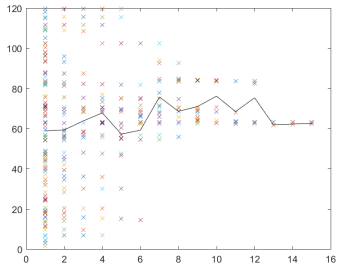
(i) Sanguine - III run



(j) Phlegmatic - I run

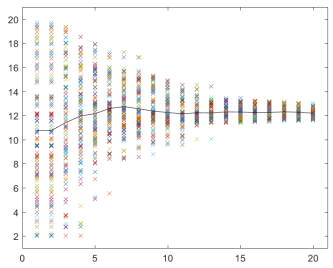


(k) Phlegmatic - II run

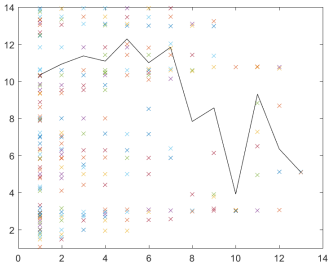


(l) Phlegmatic - III run

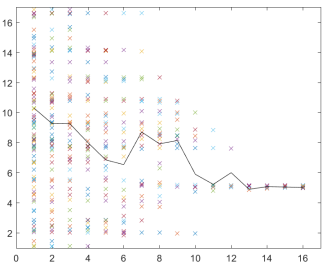
Figure B.8: Grave gene, with generations on x-axis, and value of each individual on y-axis.



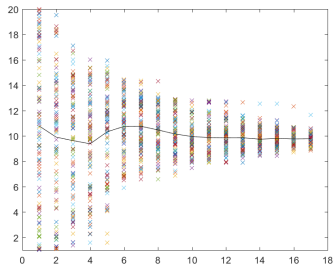
(a) Choleric - I run



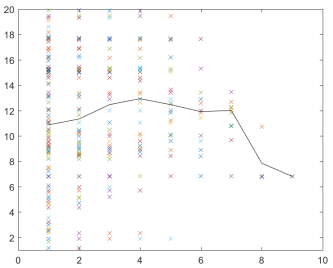
(b) Choleric - II run



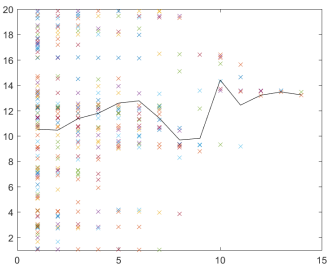
(c) Choleric - III run



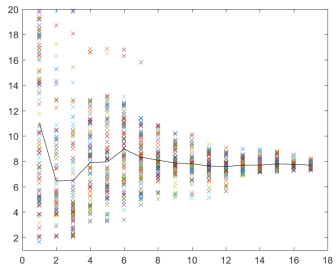
(d) Melancholic - I run



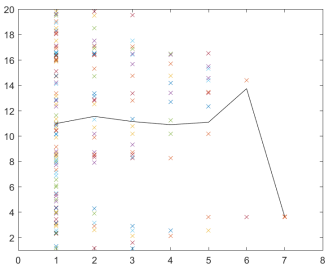
(e) Melancholic - II run



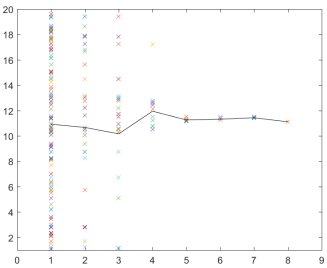
(f) Melancholic - III run



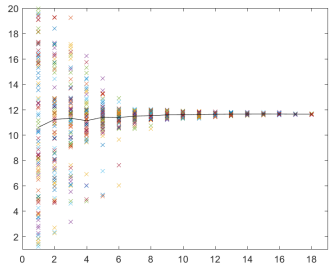
(g) Sanguine - I run



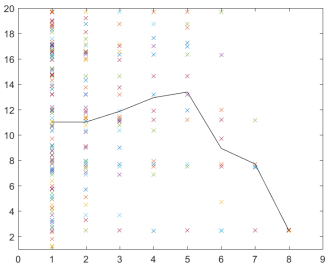
(h) Sanguine - II run



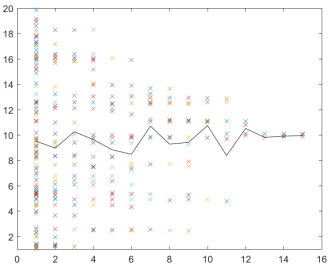
(i) Sanguine - III run



(j) Phlegmatic - I run

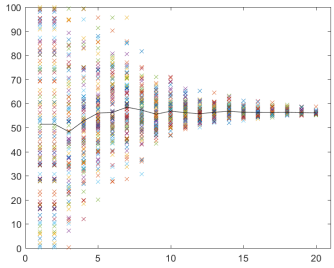


(k) Phlegmatic - II run

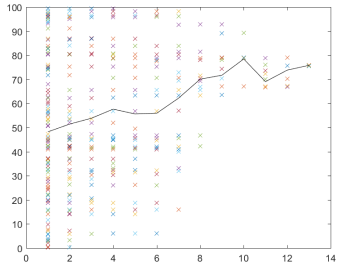


(l) Phlegmatic - III run

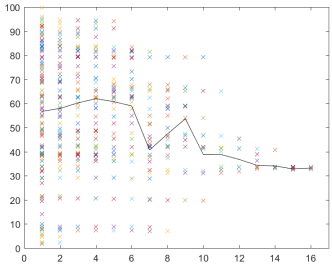
Figure B.9: Horizon gene, with generations on x-axis, and value of each individual on y-axis.



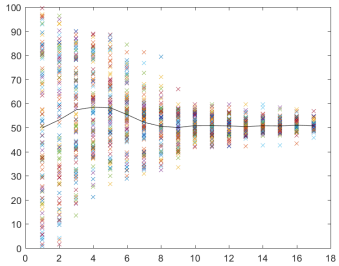
(a) Choleric - I run



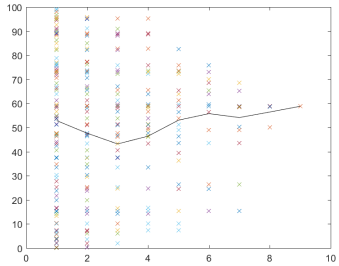
(b) Choleric - II run



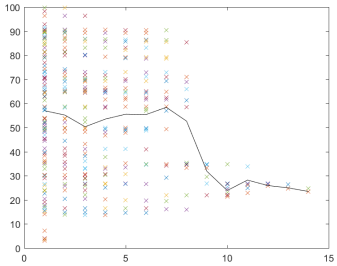
(c) Choleric - III run



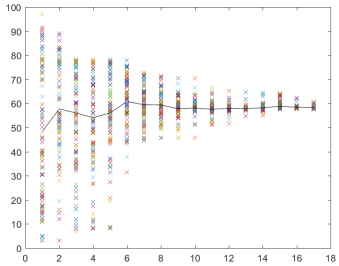
(d) Melancholic - I run



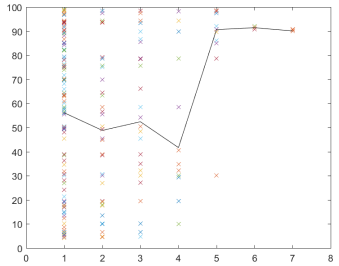
(e) Melancholic - II run



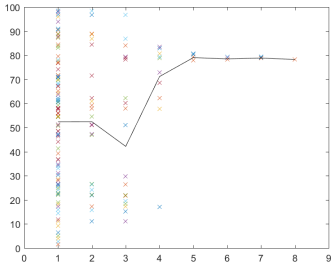
(f) Melancholic - III run



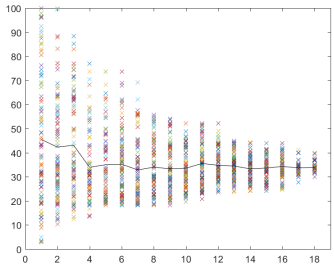
(g) Sanguine - I run



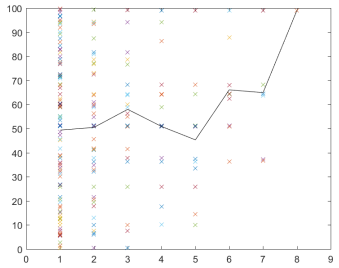
(h) Sanguine - II run



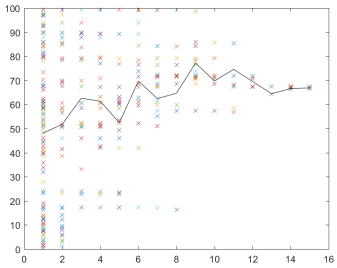
(i) Sanguine - III run



(j) Phlegmatic - I run

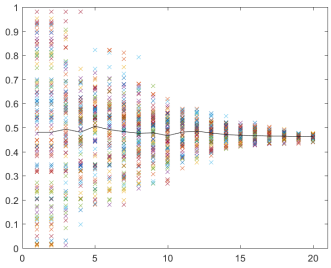


(k) Phlegmatic - II run

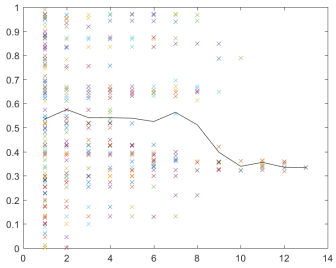


(l) Phlegmatic - III run

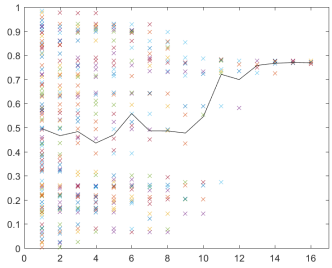
Figure B.10: Nice Threshold gene, with generations on x-axis, and value of each individual on y-axis.



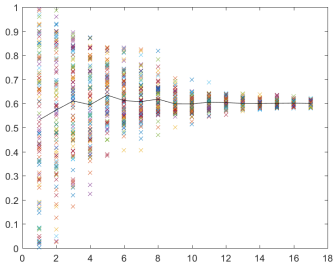
(a) Choleric - I run



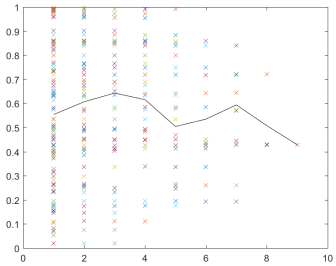
(b) Choleric - II run



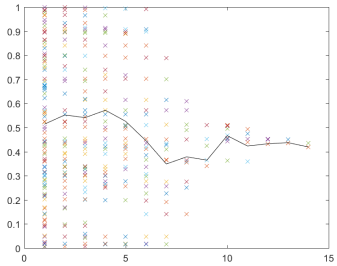
(c) Choleric - III run



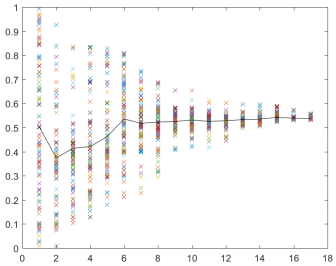
(d) Melancholic - I run



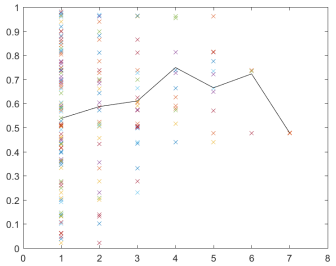
(e) Melancholic - II run



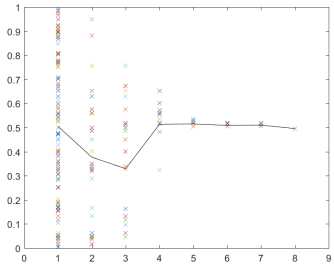
(f) Melancholic - III run



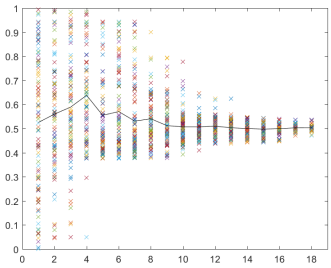
(g) Sanguine - I run



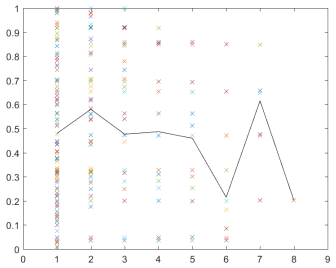
(h) Sanguine - II run



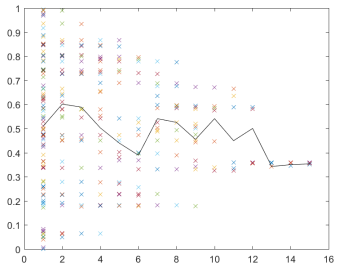
(i) Sanguine - III run



(j) Phlegmatic - I run

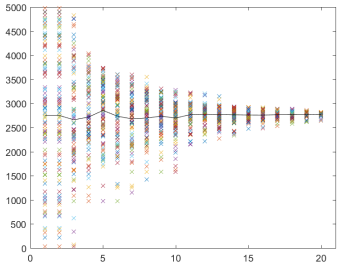


(k) Phlegmatic - II run

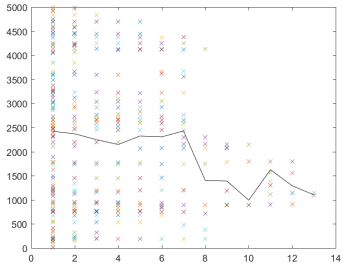


(l) Phlegmatic - III run

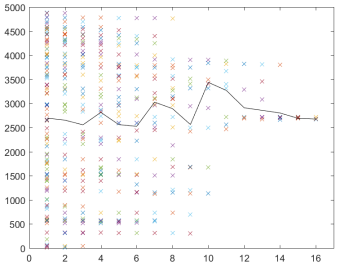
Figure B.11: Random Error gene, with generations on x-axis, and value of each individual on y-axis.



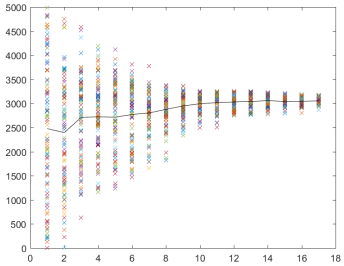
(a) Choleric - I run



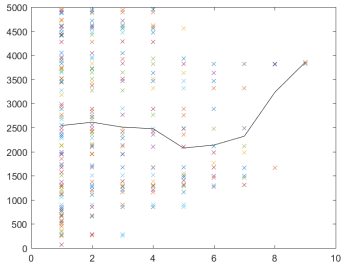
(b) Choleric - II run



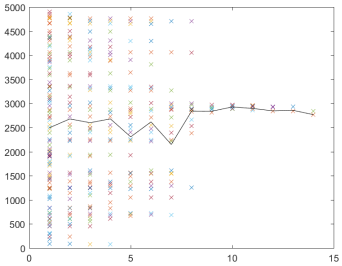
(c) Choleric - III run



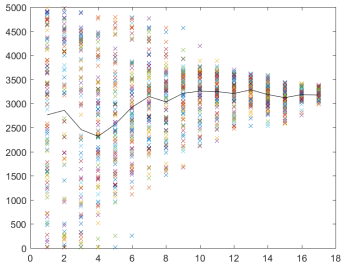
(d) Melancholic - I run



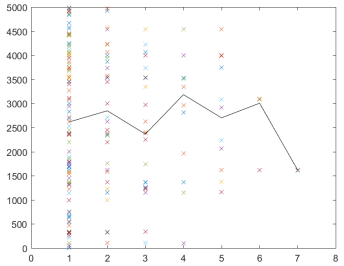
(e) Melancholic - II run



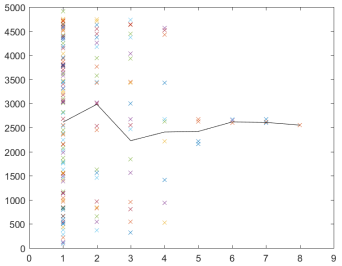
(f) Melancholic - III run



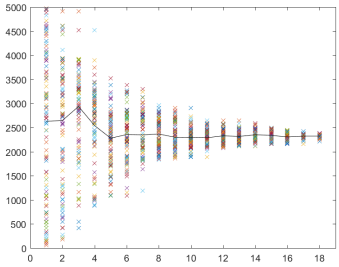
(g) Sanguine - I run



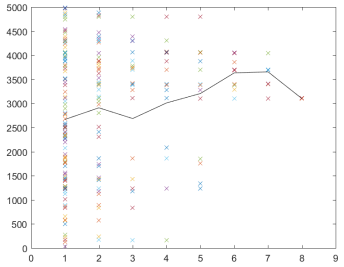
(h) Sanguine - II run



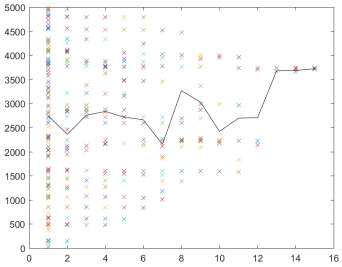
(i) Sanguine - III run



(j) Phlegmatic - I run

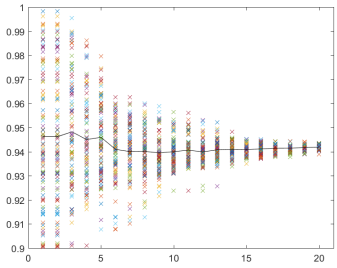


(k) Phlegmatic - II run

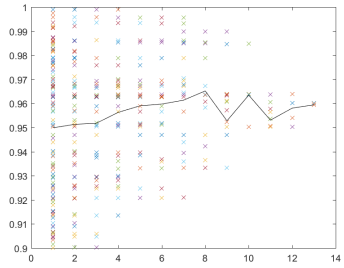


(l) Phlegmatic - III run

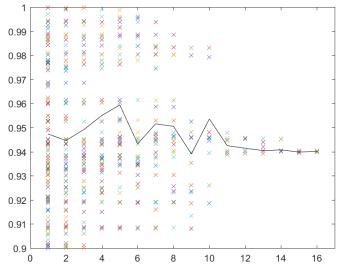
Figure B.12: Rave gene, with generations on x-axis, and value of each individual on y-axis.



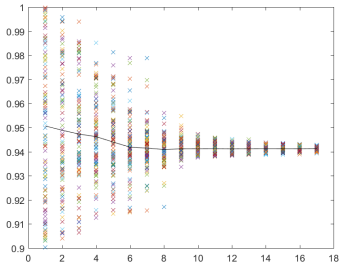
(a) Choleric - I run



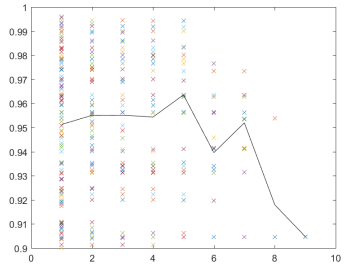
(b) Choleric - II run



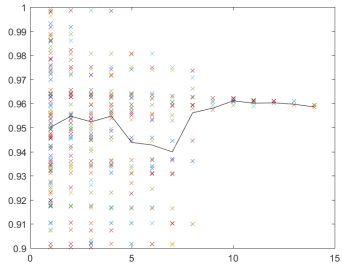
(c) Choleric - III run



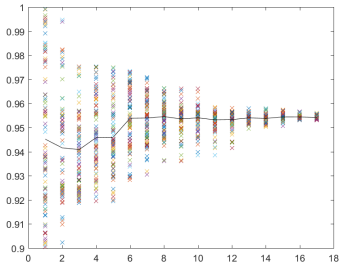
(d) Melancholic - I run



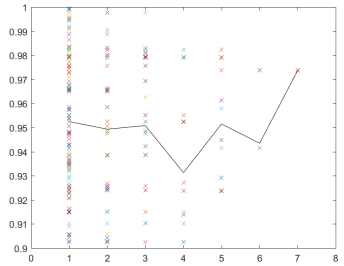
(e) Melancholic - II run



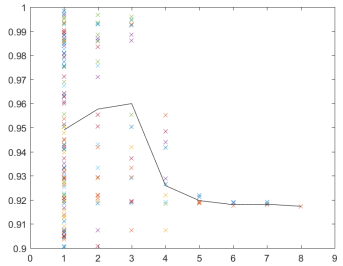
(f) Melancholic - III run



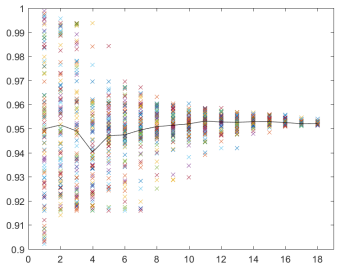
(g) Sanguine - I run



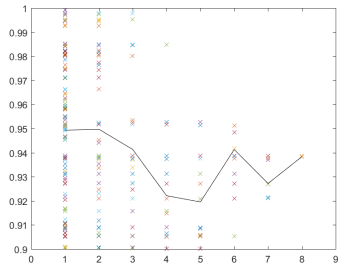
(h) Sanguine - II run



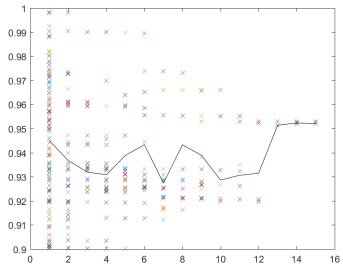
(i) Sanguine - III run



(j) Phlegmatic - I run

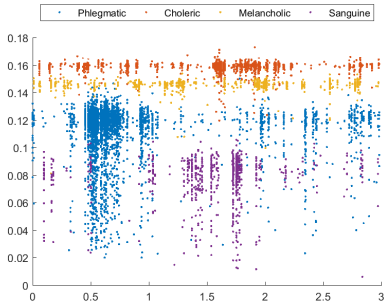


(k) Phlegmatic - II run

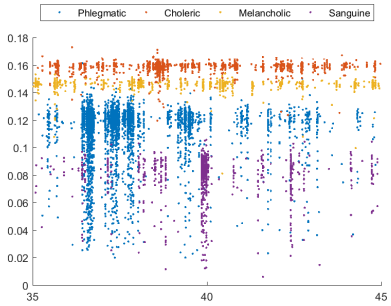


(l) Phlegmatic - III run

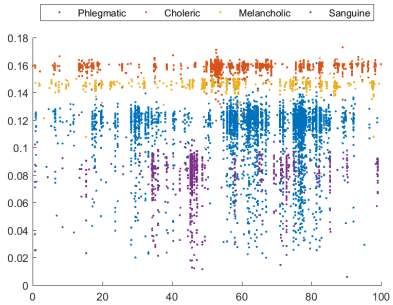
Figure B.13: Tree Discount gene, with generations on x-axis, and value of each individual on y-axis.



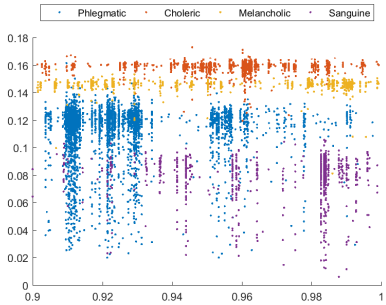
(a) Aggressiveness



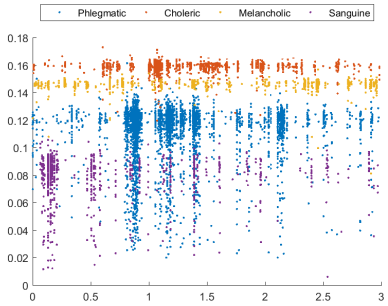
(b) Charge Default



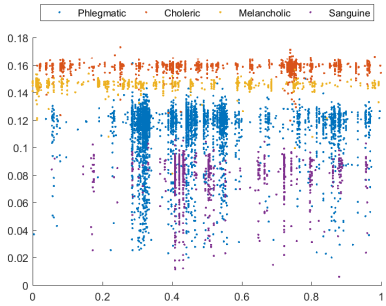
(c) Charge Depth



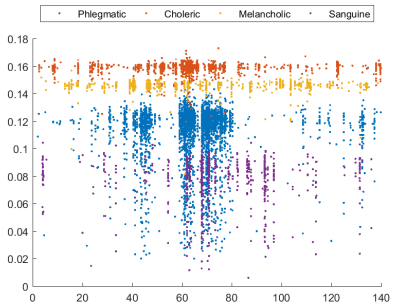
(d) Charge Discount



(e) Defensiveness

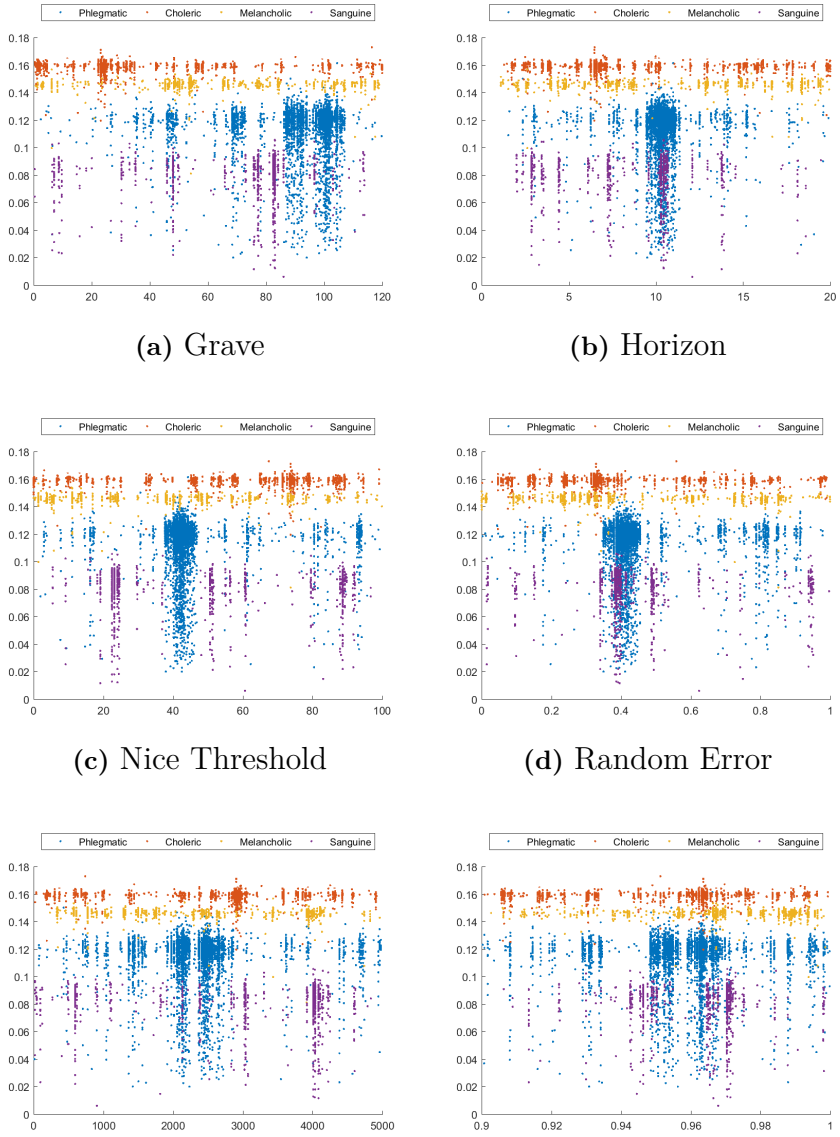


(f) Epsilon



(g) Exploration Factor

Figure B.14: Correlation between specific genes (values of which are on x-axis) and fitness (y-axis) (continued on the next page).



(a) Grave

(b) Horizon

(c) Nice Threshold

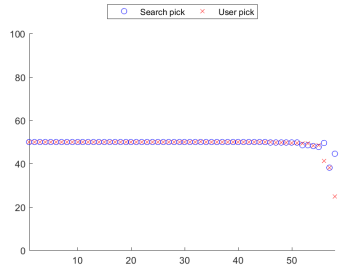
(d) Random Error

(e) Rave

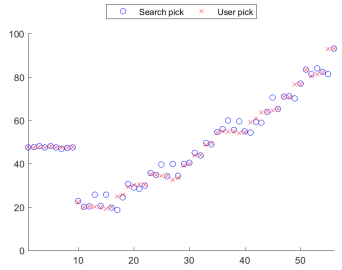
(f) Tree Discount

Figure B.15: Correlation between specific genes (values of which are on x-axis) and fitness (y-axis).

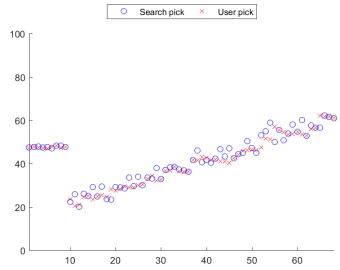
B.2 Evaluation



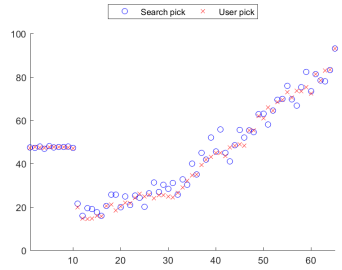
(a) Choleric - Skirmish - Match 1



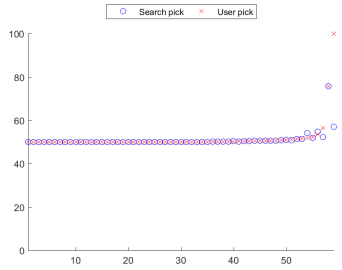
(b) Melancholic - Skirmish - Match 1



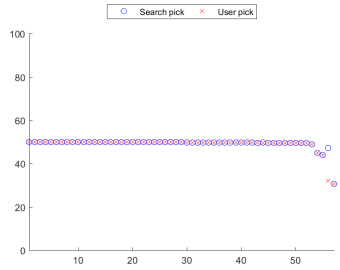
(c) Melancholic - Skirmish - Match 2



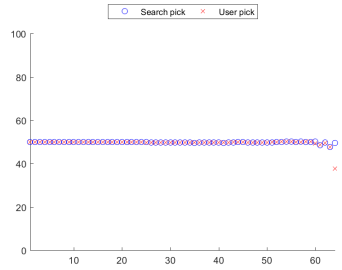
(d) Melancholic - Skirmish - Match 3



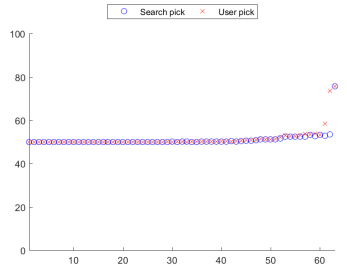
(e) Sanguine - Skirmish - Match 1



(f) Sanguine - Skirmish - Match 2

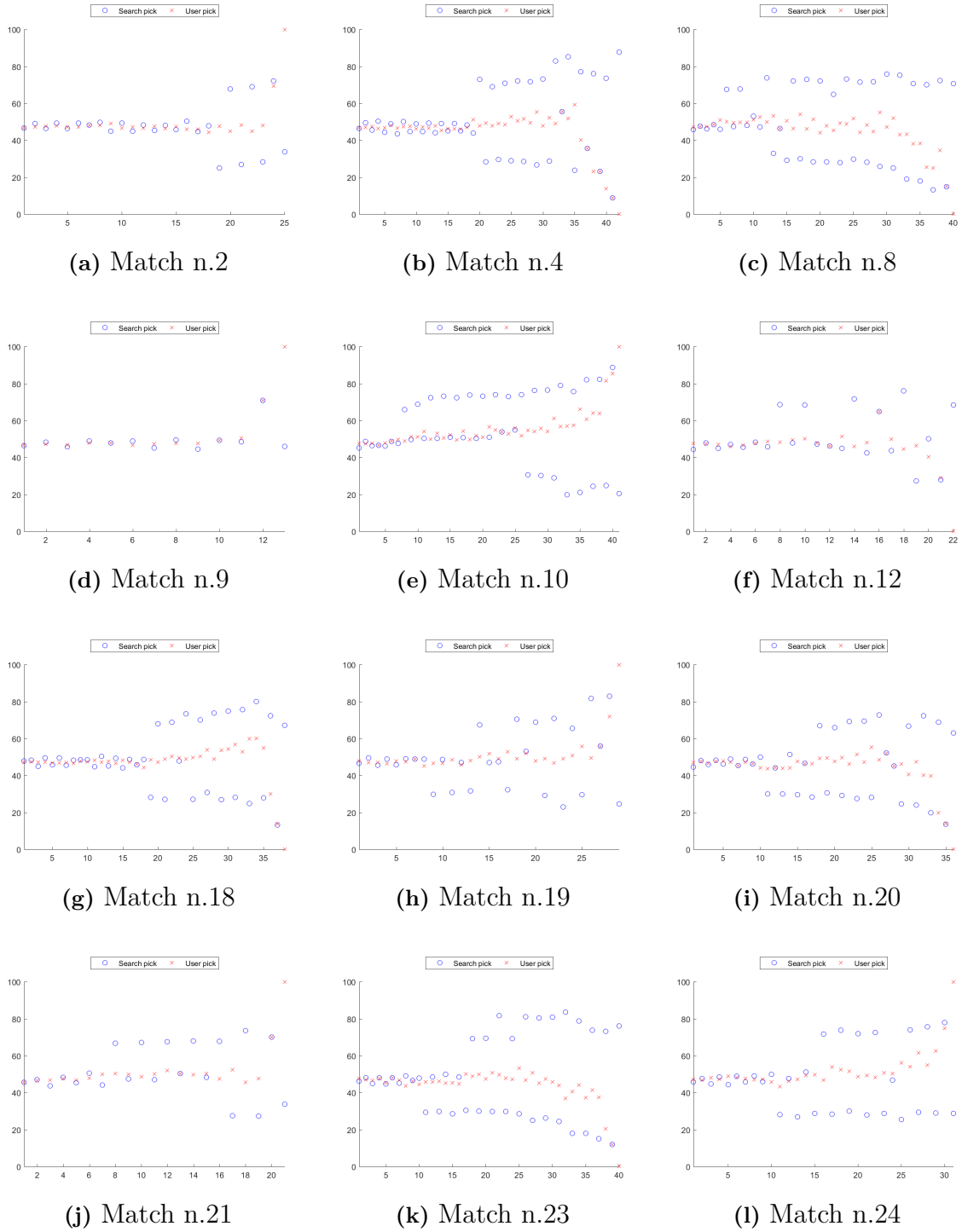


(g) Sanguine - Skirmish - Match 3



(h) Phlegmatic - Skirmish - Match 1

Figure B.16: Evaluation of the Skirmish games with fittest individuals, with moves on x-axis and QValues on y-axis.



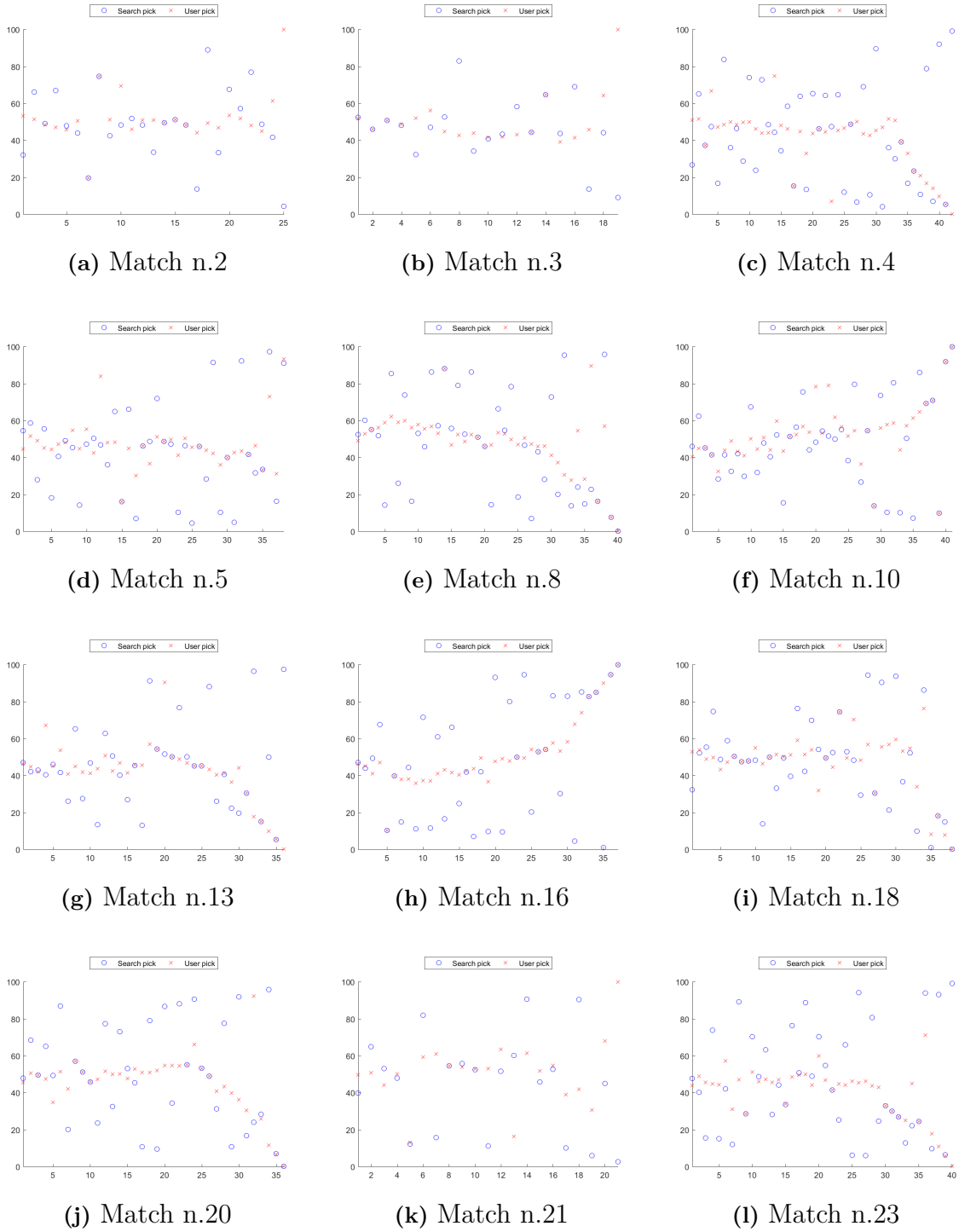
(a) Match n.2 **(b)** Match n.4 **(c)** Match n.8

(d) Match n.9 **(e)** Match n.10 **(f)** Match n.12

(g) Match n.18 **(h)** Match n.19 **(i)** Match n.20

(j) Match n.21 **(k)** Match n.23 **(l)** Match n.24

Figure B.17: Evaluation of Nine Boards Tic-Tac-Toe for Sanguine Personality, with moves on x-axis and QValues on y-axis.



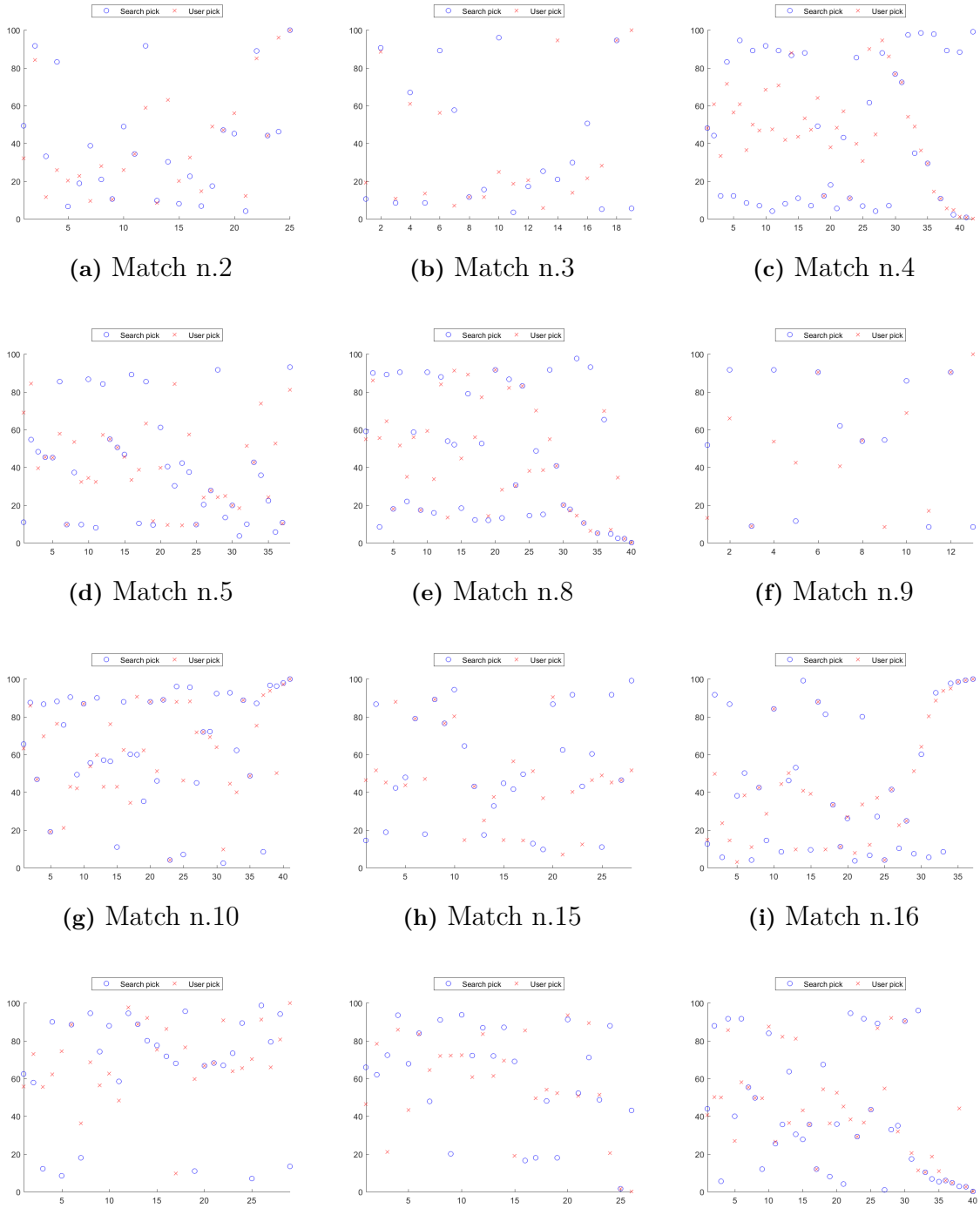
(a) Match n.2 **(b)** Match n.3 **(c)** Match n.4

(d) Match n.5 **(e)** Match n.8 **(f)** Match n.10

(g) Match n.13 **(h)** Match n.16 **(i)** Match n.18

(j) Match n.20 **(k)** Match n.21 **(l)** Match n.23

Figure B.18: Evaluation of Nine Boards Tic-Tac-Toe for Phlegmatic Personality, with moves on x-axis and QValues on y-axis.



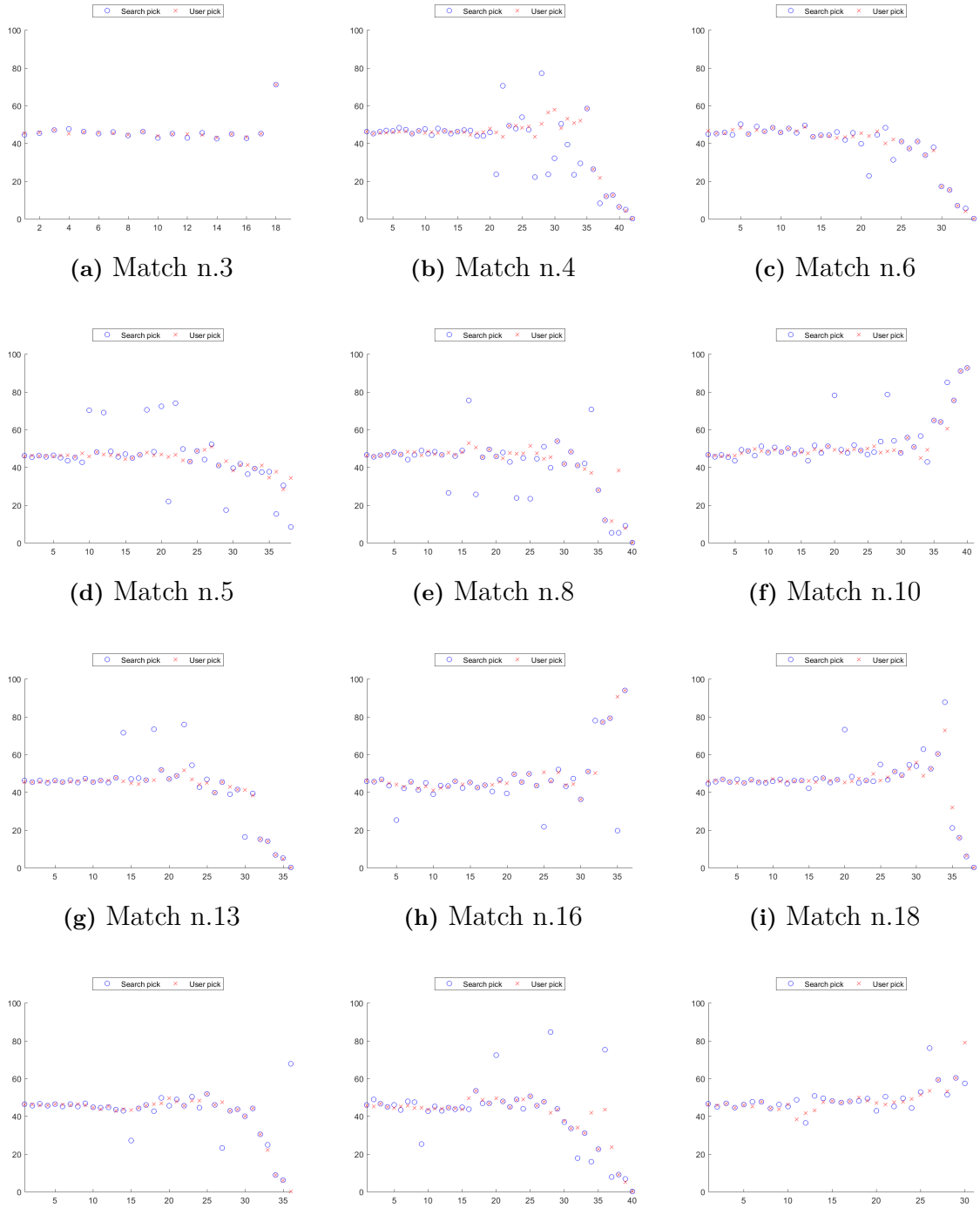
(a) Match n.2 **(b)** Match n.3 **(c)** Match n.4

(d) Match n.5 **(e)** Match n.8 **(f)** Match n.9

(g) Match n.10 **(h)** Match n.15 **(i)** Match n.16

(j) Match n.19 **(k)** Match n.22 **(l)** Match n.23

Figure B.19: Evaluation of Nine Boards Tic-Tac-Toe for Melancholic Personality, with moves on x-axis and QValues on y-axis.



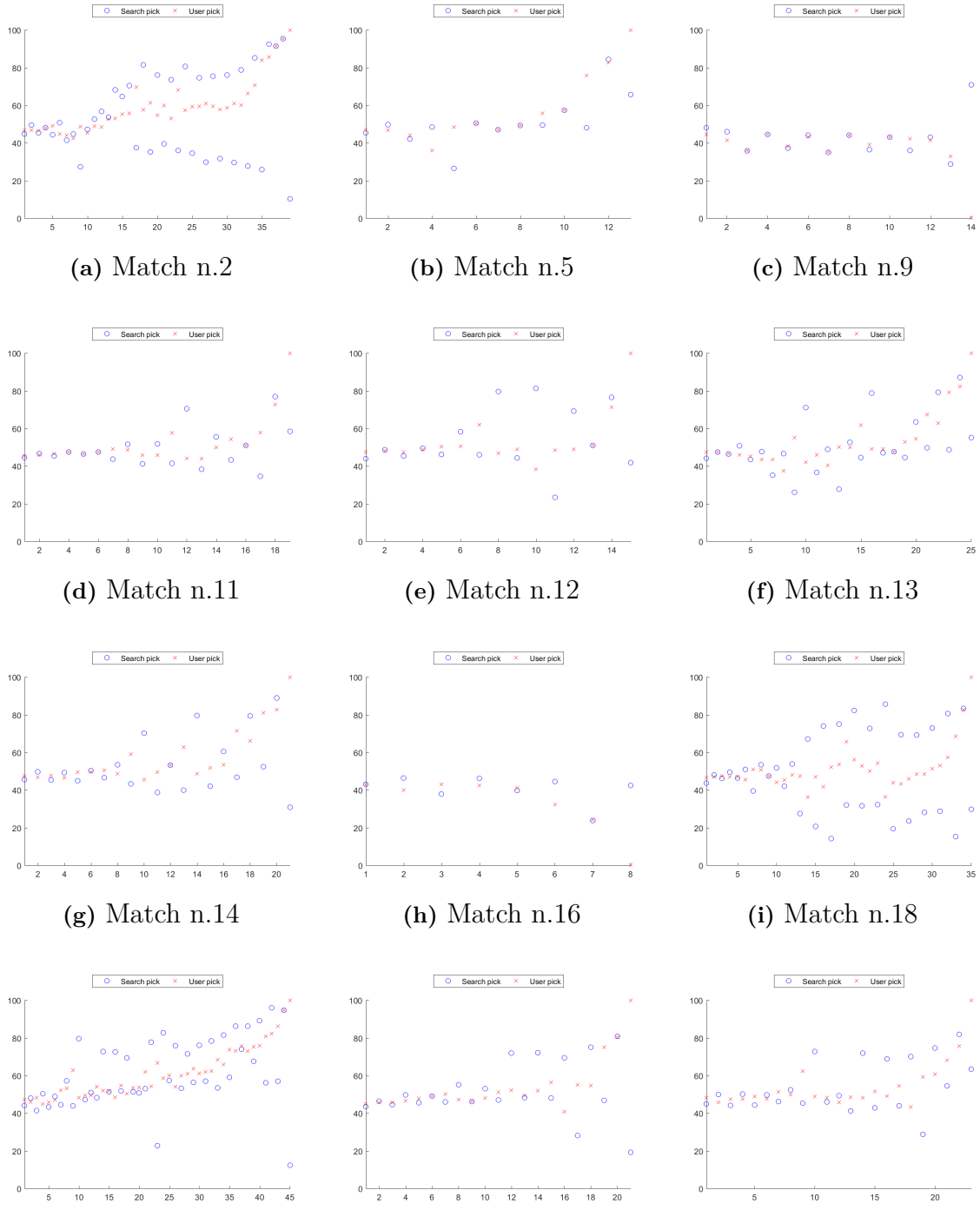
(a) Match n.3 **(b)** Match n.4 **(c)** Match n.6

(d) Match n.5 **(e)** Match n.8 **(f)** Match n.10

(g) Match n.13 **(h)** Match n.16 **(i)** Match n.18

(j) Match n.20 **(k)** Match n.23 **(l)** Match n.24

Figure B.20: Evaluation of Nine Boards Tic-Tac-Toe for Choleric Personality, with moves on x-axis and QValues on y-axis.



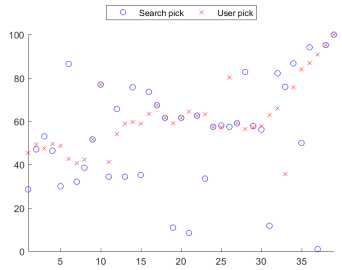
(a) Match n.2 **(b)** Match n.5 **(c)** Match n.9

(d) Match n.11 **(e)** Match n.12 **(f)** Match n.13

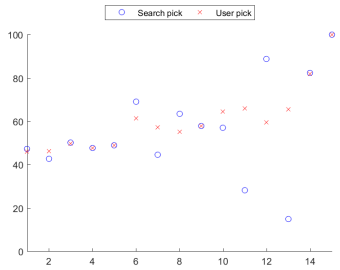
(g) Match n.14 **(h)** Match n.16 **(i)** Match n.18

(j) Match n.19 **(k)** Match n.20 **(l)** Match n.22

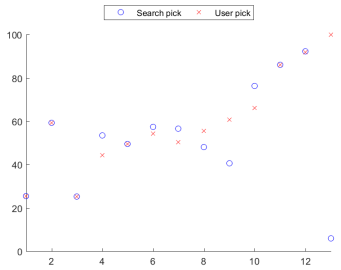
Figure B.21: Evaluation of Connect Four for Sanguine Personality, with moves on x-axis and QValues on y-axis.



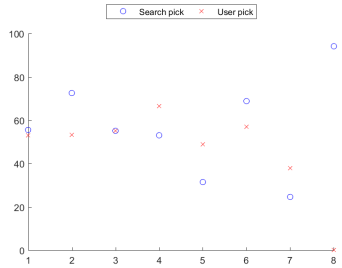
(a) Match n.2



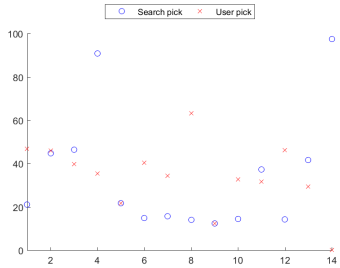
(b) Match n.4



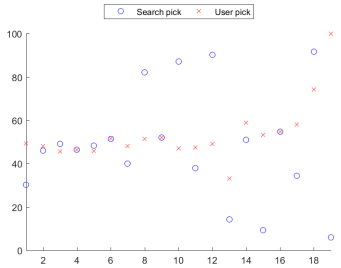
(c) Match n.5



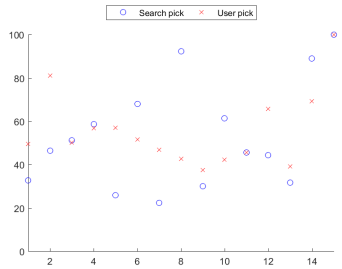
(d) Match n.8



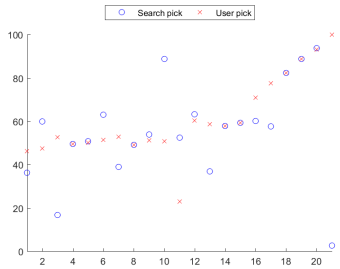
(e) Match n.9



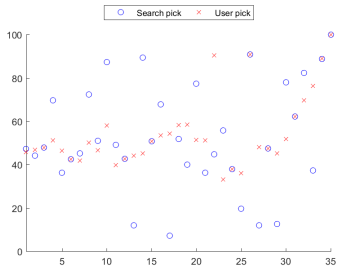
(f) Match n.11



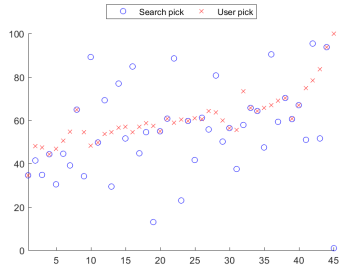
(g) Match n.12



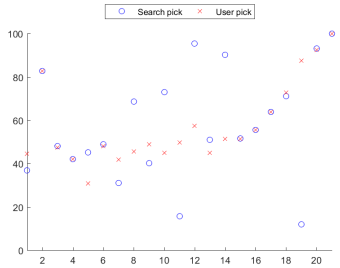
(h) Match n.14



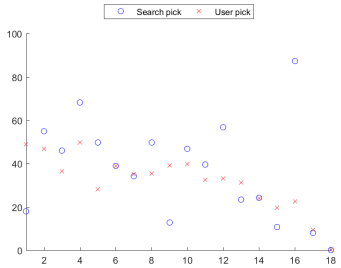
(i) Match n.18



(j) Match n.19

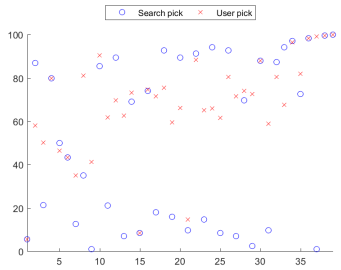


(k) Match n.20

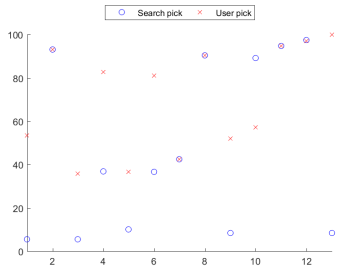


(l) Match n.21

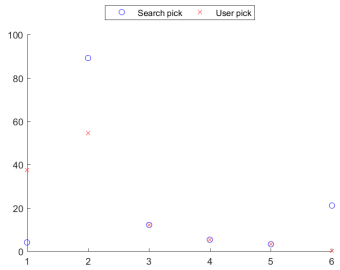
Figure B.22: Evaluation of Connect Four for Phlegmatic Personality, with moves on x-axis and QValues on y-axis.



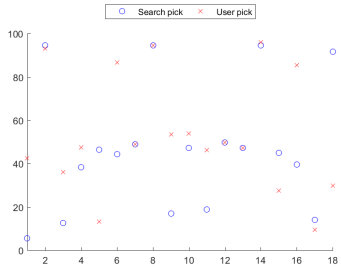
(a) Match n.2



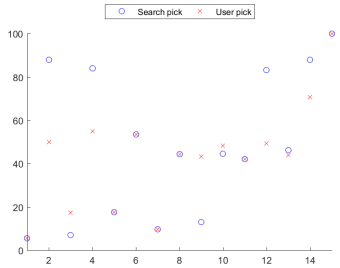
(b) Match n.5



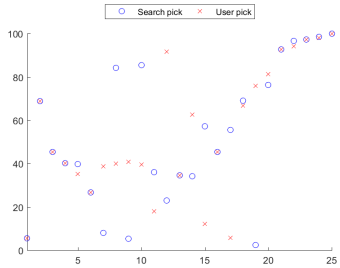
(c) Match n.7



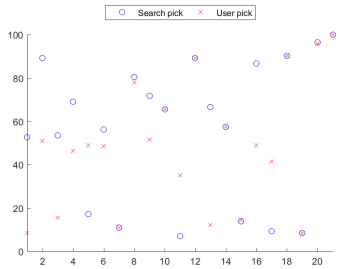
(d) Match n.10



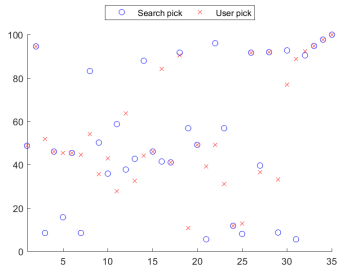
(e) Match n.12



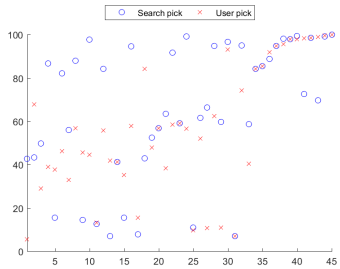
(f) Match n.13



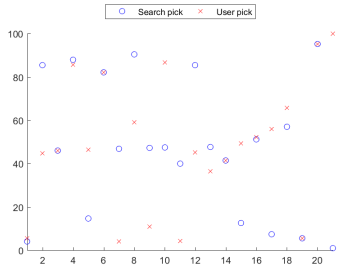
(g) Match n.14



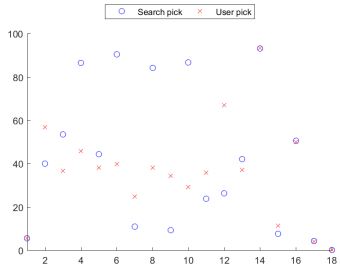
(h) Match n.18



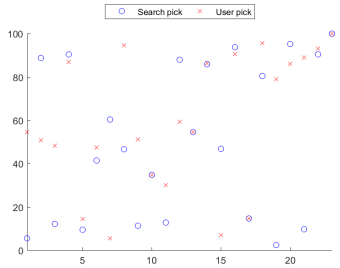
(i) Match n.19



(j) Match n.20

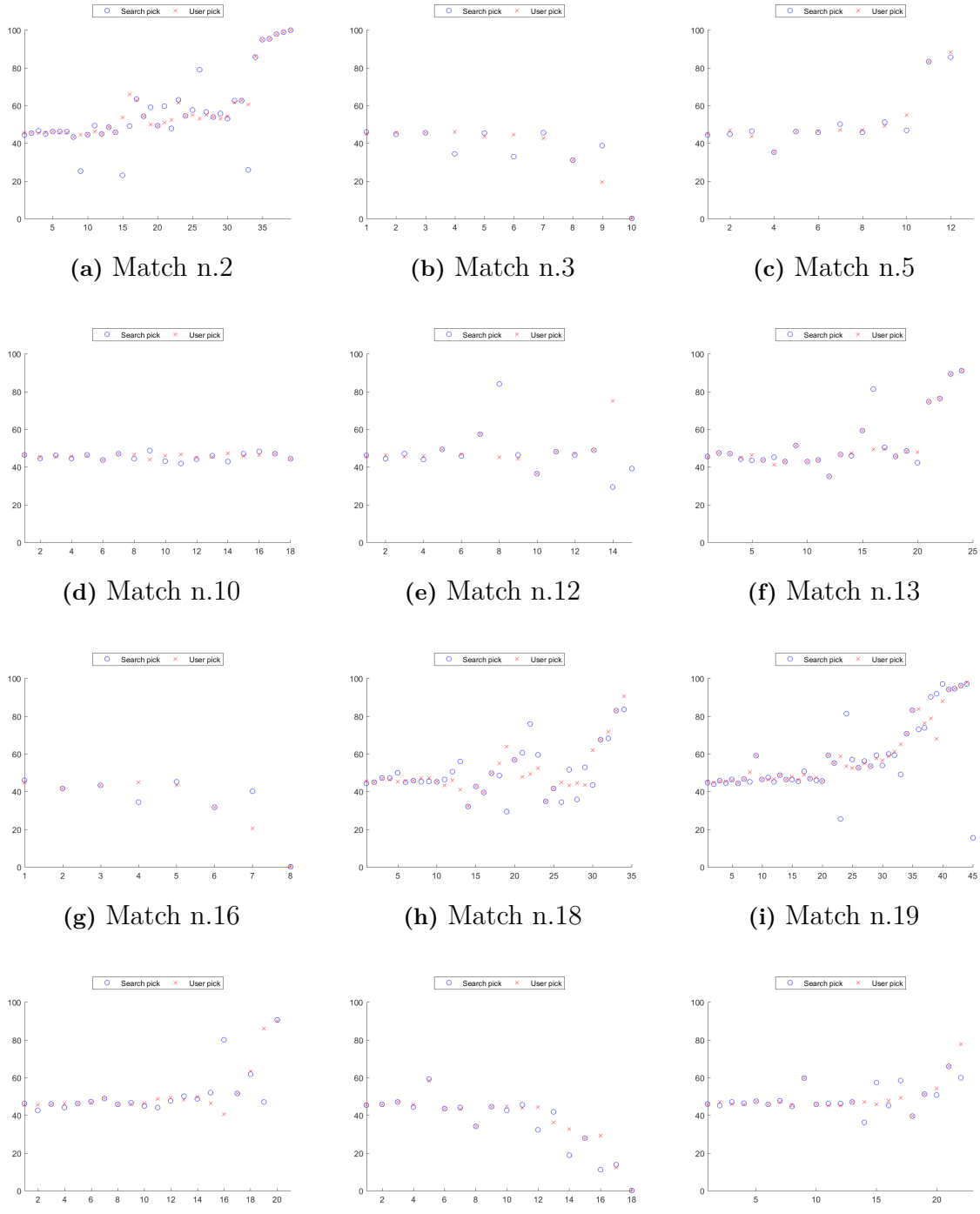


(k) Match n.21



(l) Match n.22

Figure B.23: Evaluation of Connect Four for Melancholic Personality, with moves on x-axis and QValues on y-axis.



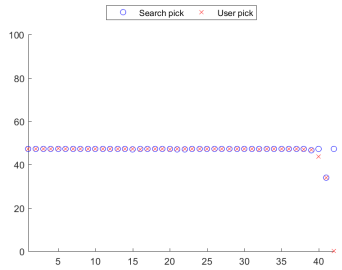
(a) Match n.2 **(b)** Match n.3 **(c)** Match n.5

(d) Match n.10 **(e)** Match n.12 **(f)** Match n.13

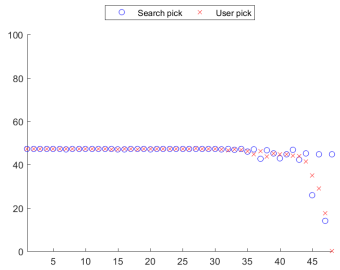
(g) Match n.16 **(h)** Match n.18 **(i)** Match n.19

(j) Match n.20 **(k)** Match n.21 **(l)** Match n.22

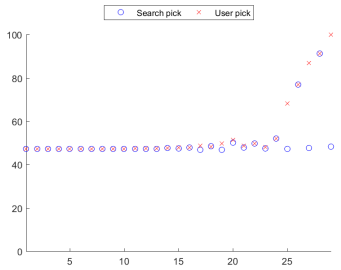
Figure B.24: Evaluation of Connect Four for Choleric Personality, with moves on x-axis and QValues on y-axis.



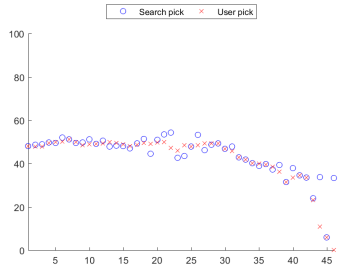
(a) Sanguine - Match 1



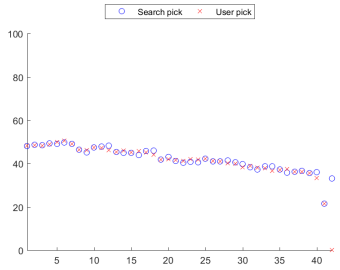
(b) Sanguine - Match 2



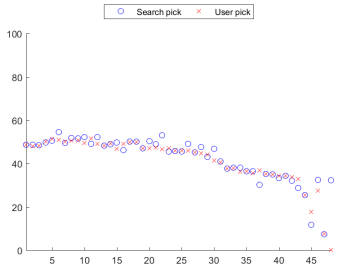
(c) Sanguine - Match 3



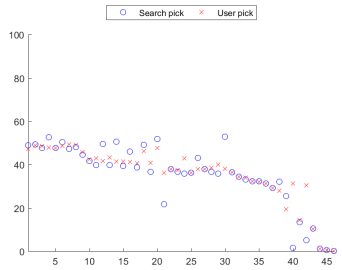
(d) Phlegmatic - Match 1



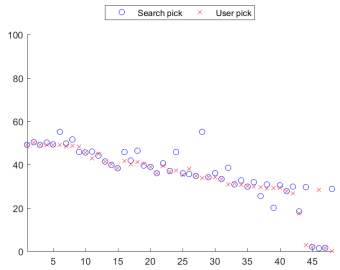
(e) Phlegmatic - Match 2



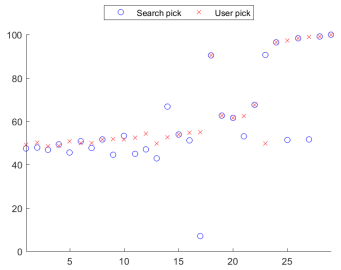
(f) Phlegmatic - Match 4



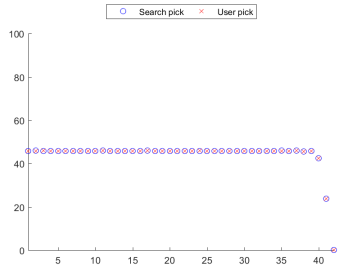
(g) Melancholic - Match 1



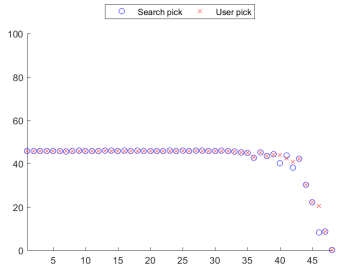
(h) Melancholic - Match 4



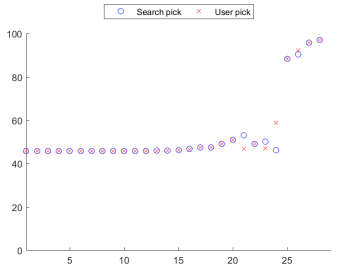
(i) Melancholic - Match 6



(j) Choleric - Match 2

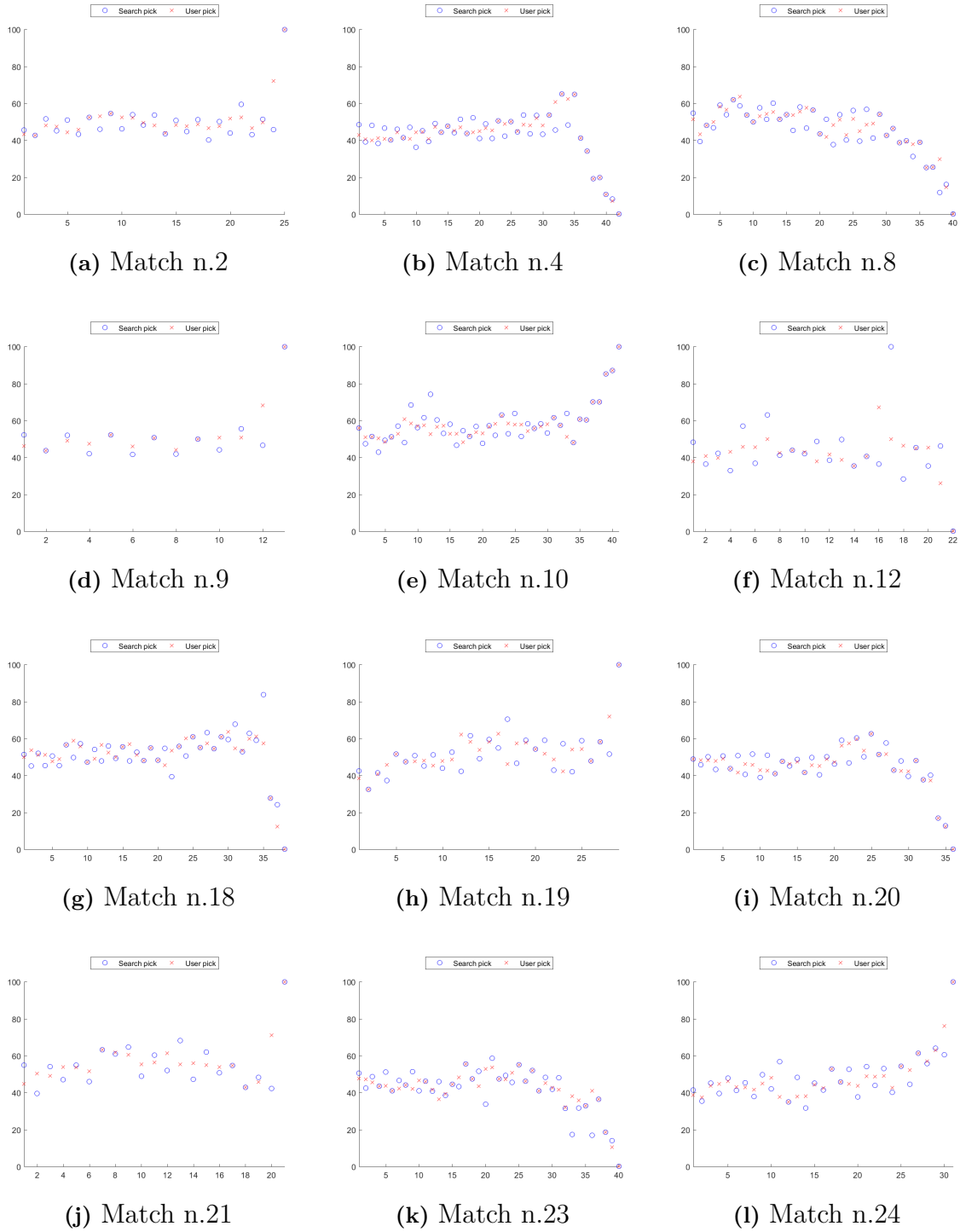


(k) Choleric - Match 4



(l) Choleric - Match 6

Figure B.25: Evaluation of Checkers Games, with moves on x-axis and QValues on y-axis.



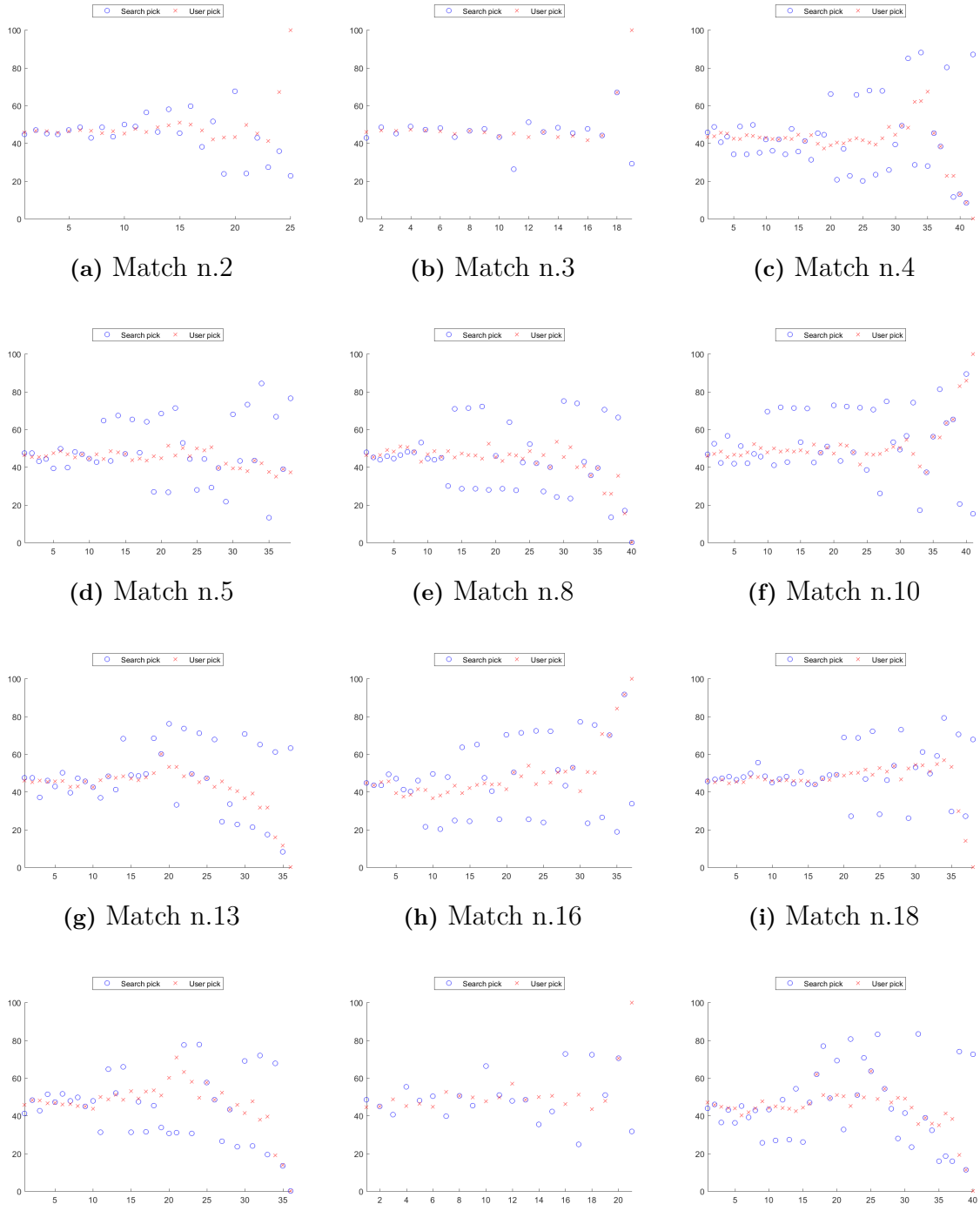
(a) Match n.2 **(b)** Match n.4 **(c)** Match n.8

(d) Match n.9 **(e)** Match n.10 **(f)** Match n.12

(g) Match n.18 **(h)** Match n.19 **(i)** Match n.20

(j) Match n.21 **(k)** Match n.23 **(l)** Match n.24

Figure B.26: Evaluation of Nine Boards Tic-Tac-Toe for Random Individual A, with moves on x-axis and QValues on y-axis.



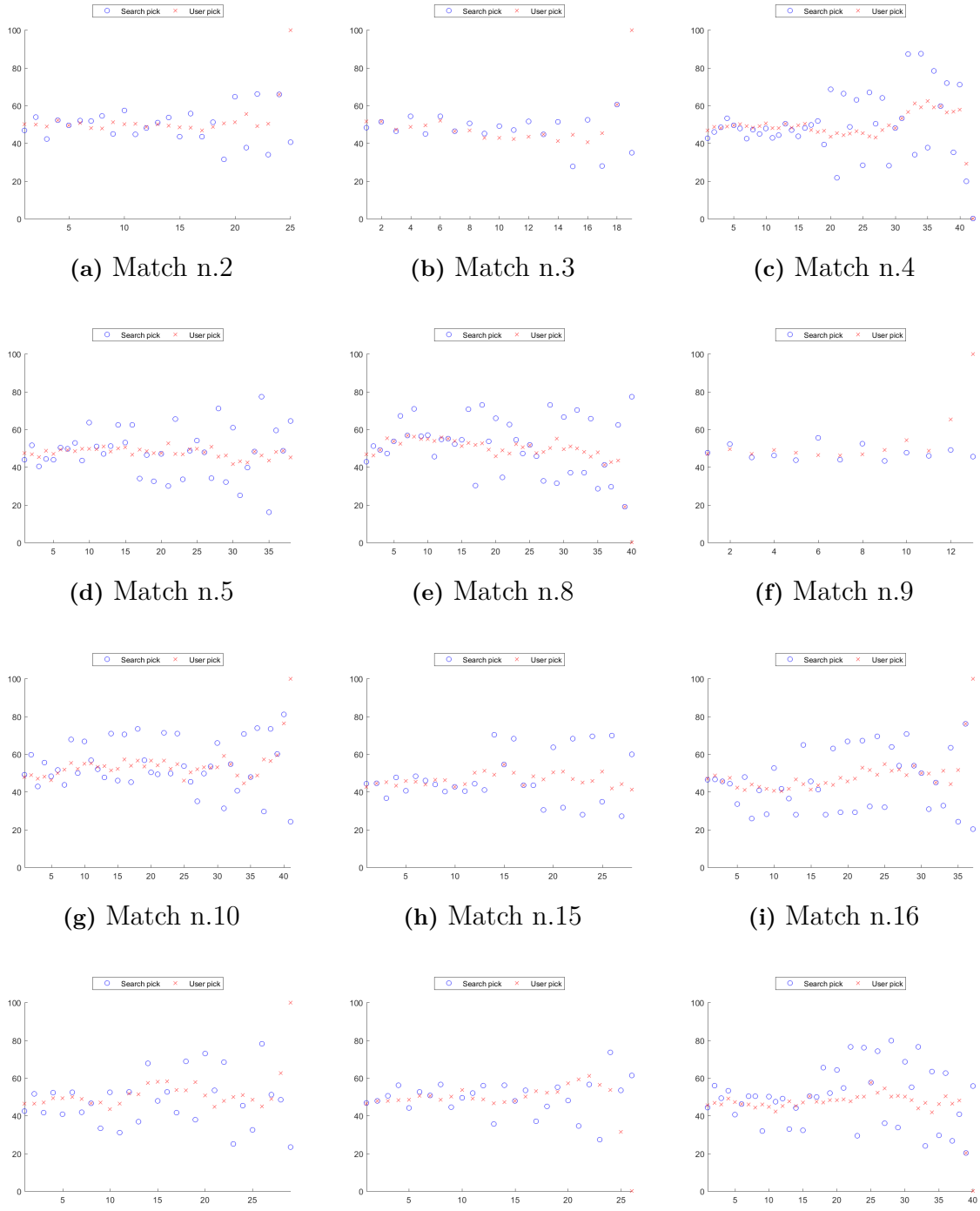
(a) Match n.2 **(b)** Match n.3 **(c)** Match n.4

(d) Match n.5 **(e)** Match n.8 **(f)** Match n.10

(g) Match n.13 **(h)** Match n.16 **(i)** Match n.18

(j) Match n.20 **(k)** Match n.21 **(l)** Match n.23

Figure B.27: Evaluation of Nine Boards Tic-Tac-Toe for Random Individual B, with moves on x-axis and QValues on y-axis.



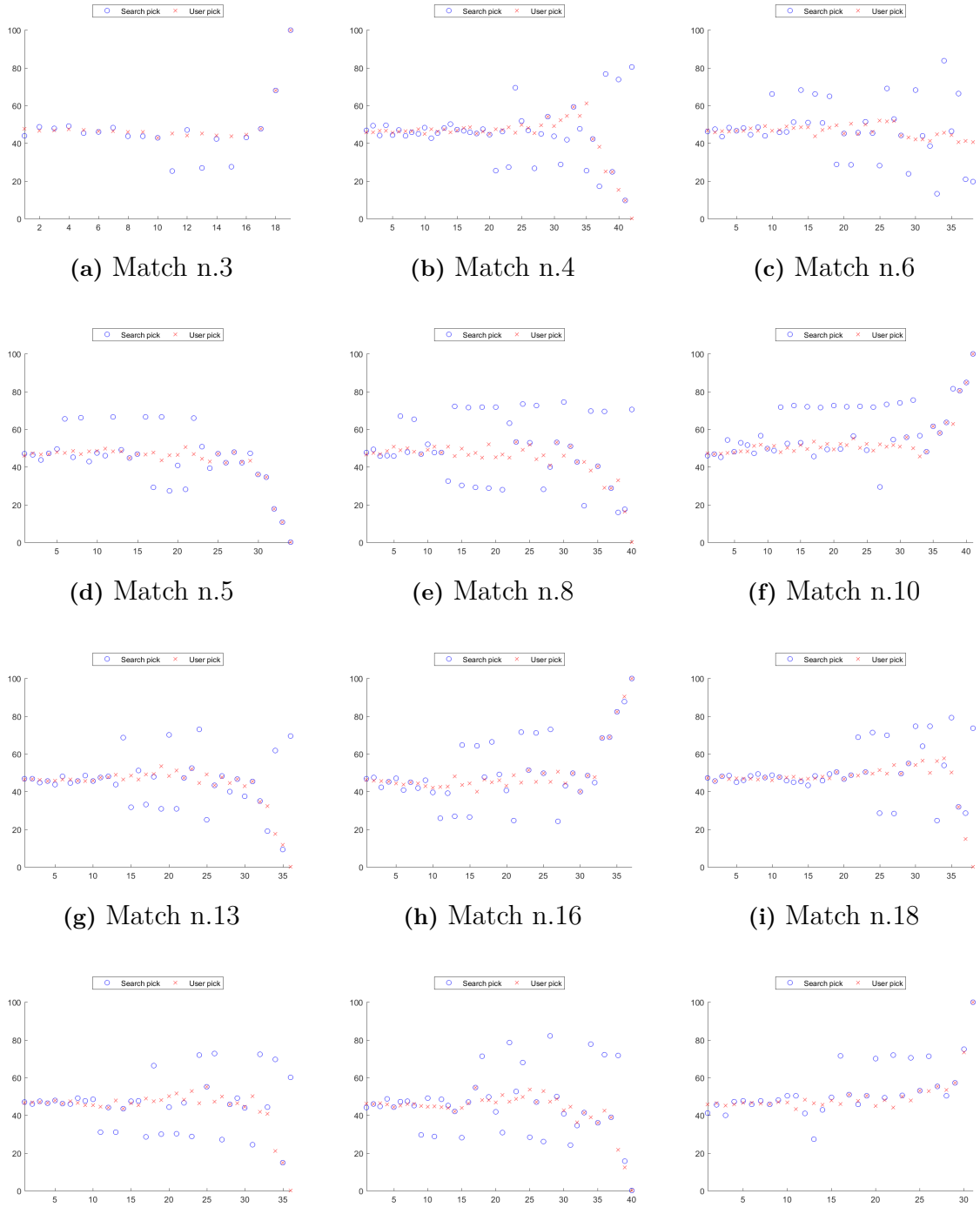
(a) Match n.2 **(b)** Match n.3 **(c)** Match n.4

(d) Match n.5 **(e)** Match n.8 **(f)** Match n.9

(g) Match n.10 **(h)** Match n.15 **(i)** Match n.16

(j) Match n.19 **(k)** Match n.22 **(l)** Match n.23

Figure B.28: Evaluation of Nine Boards Tic-Tac-Toe for Random Individual C, with moves on x-axis and QValues on y-axis.



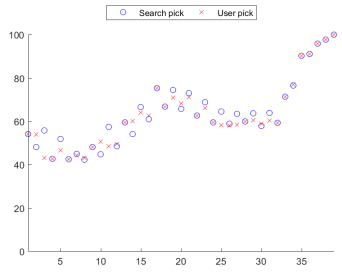
(a) Match n.3 **(b)** Match n.4 **(c)** Match n.6

(d) Match n.5 **(e)** Match n.8 **(f)** Match n.10

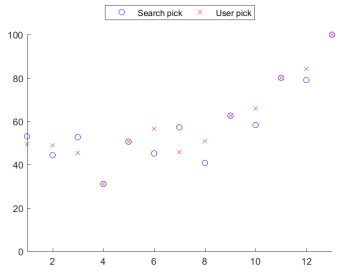
(g) Match n.13 **(h)** Match n.16 **(i)** Match n.18

(j) Match n.20 **(k)** Match n.23 **(l)** Match n.24

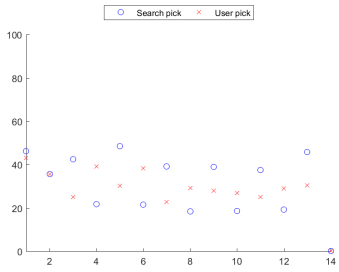
Figure B.29: Evaluation of Nine Boards Tic-Tac-Toe for Random Individual D, with moves on x-axis and QValues on y-axis.



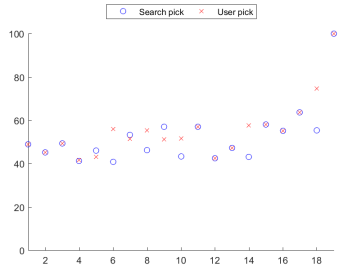
(a) Match n.2



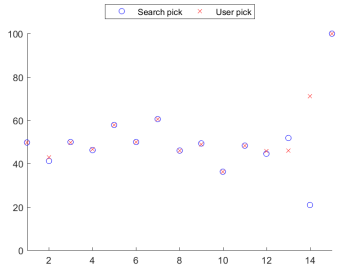
(b) Match n.5



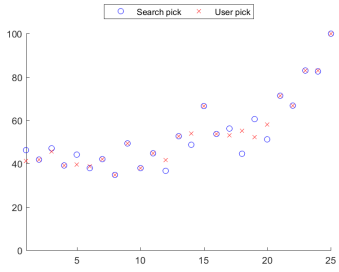
(c) Match n.9



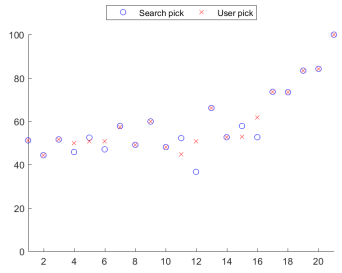
(d) Match n.11



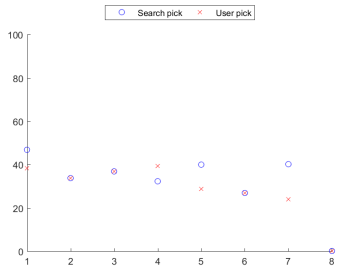
(e) Match n.12



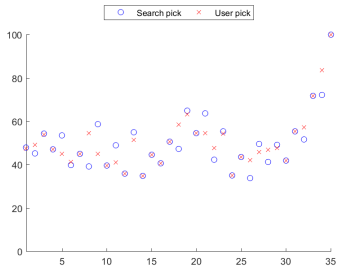
(f) Match n.13



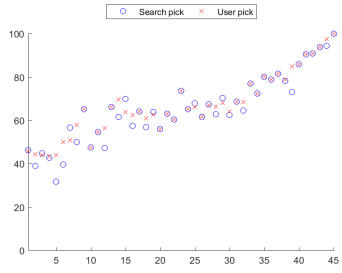
(g) Match n.14



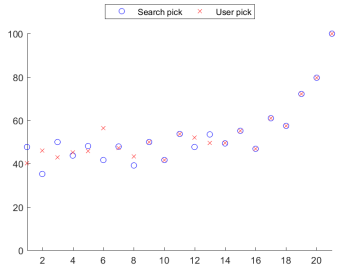
(h) Match n.16



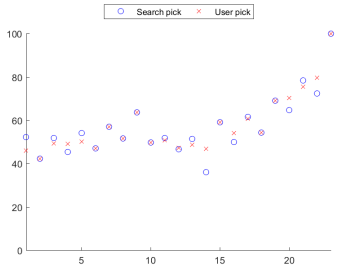
(i) Match n.18



(j) Match n.19

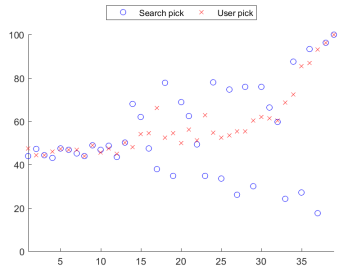


(k) Match n.20

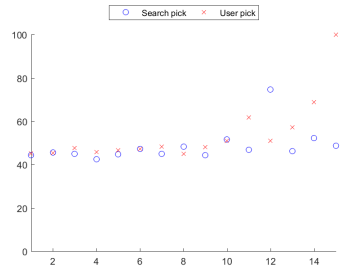


(l) Match n.22

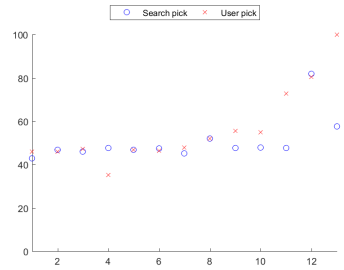
Figure B.30: Evaluation of Connect Four for Random Individual A, with moves on x-axis and QValues on y-axis.



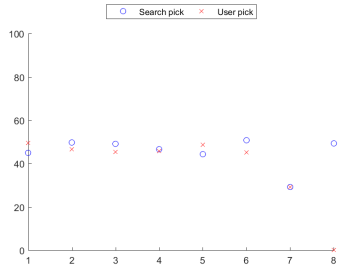
(a) Match n.2



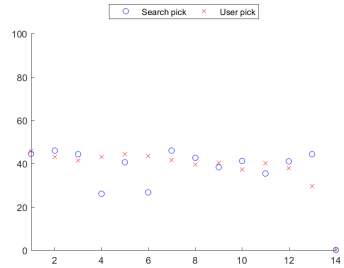
(b) Match n.4



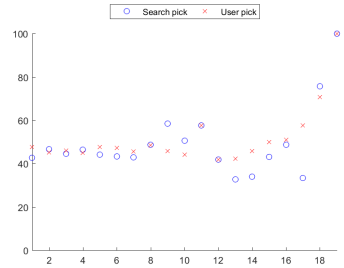
(c) Match n.5



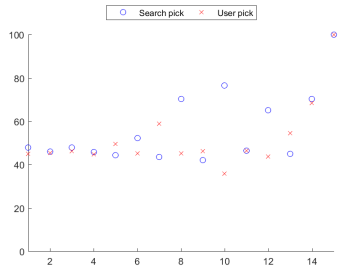
(d) Match n.8



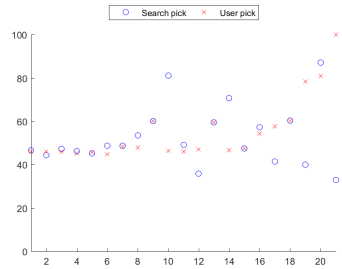
(e) Match n.9



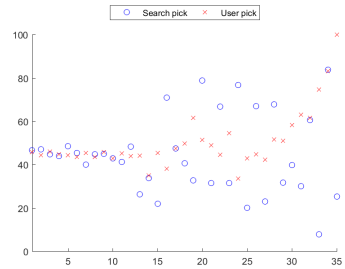
(f) 11



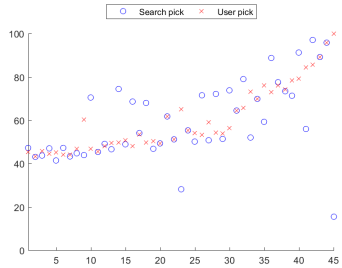
(g) Match n.12



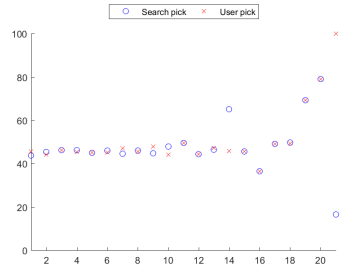
(h) Match n.14



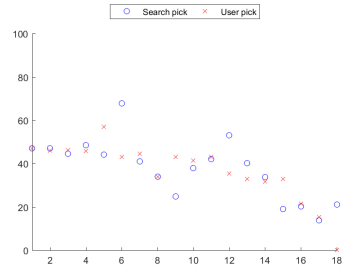
(i) Match n.18



(j) Match n.19

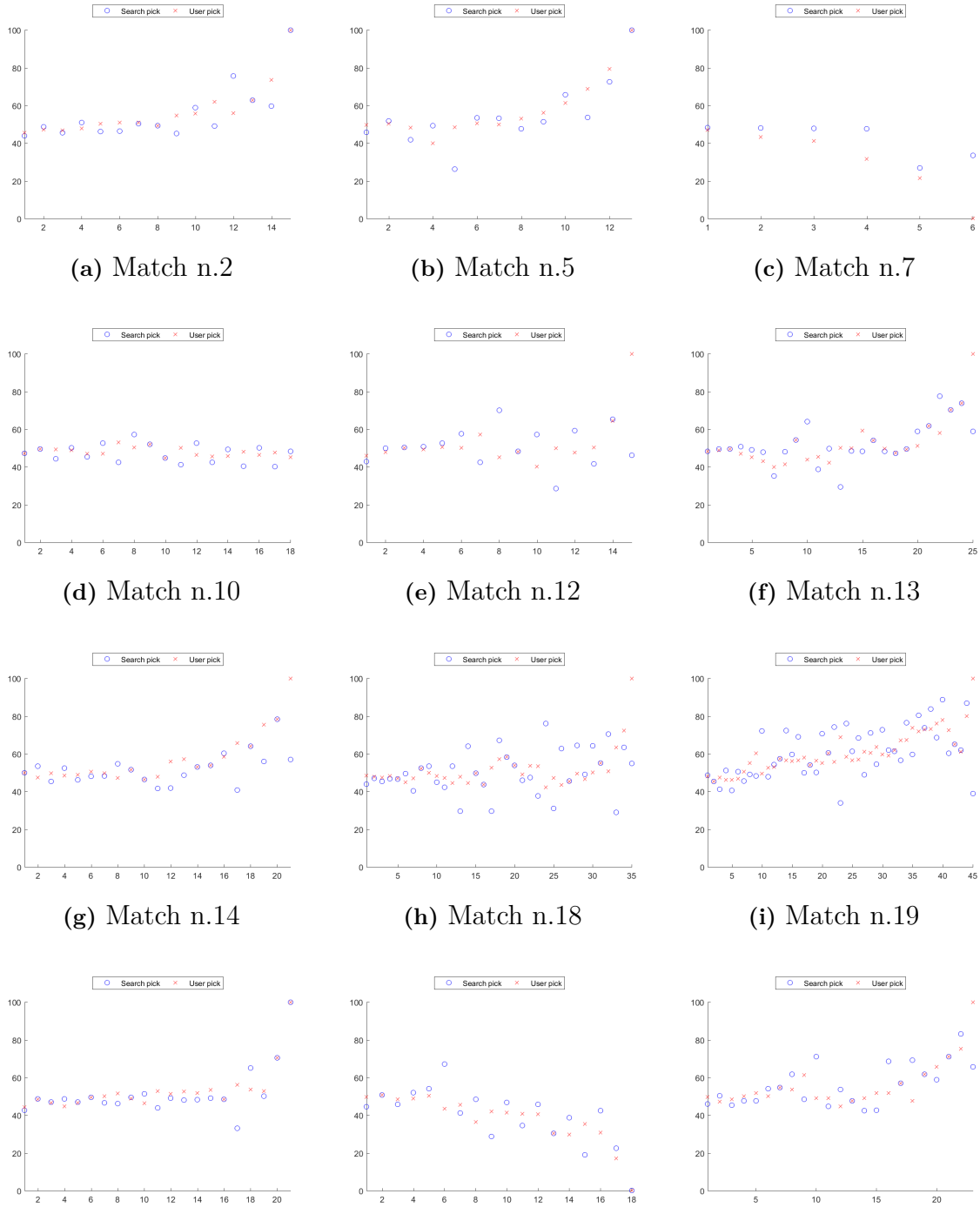


(k) Match n.20



(l) Match n.21

Figure B.31: Evaluation of Connect Four for Random Individual B, with moves on x-axis and QValues on y-axis.



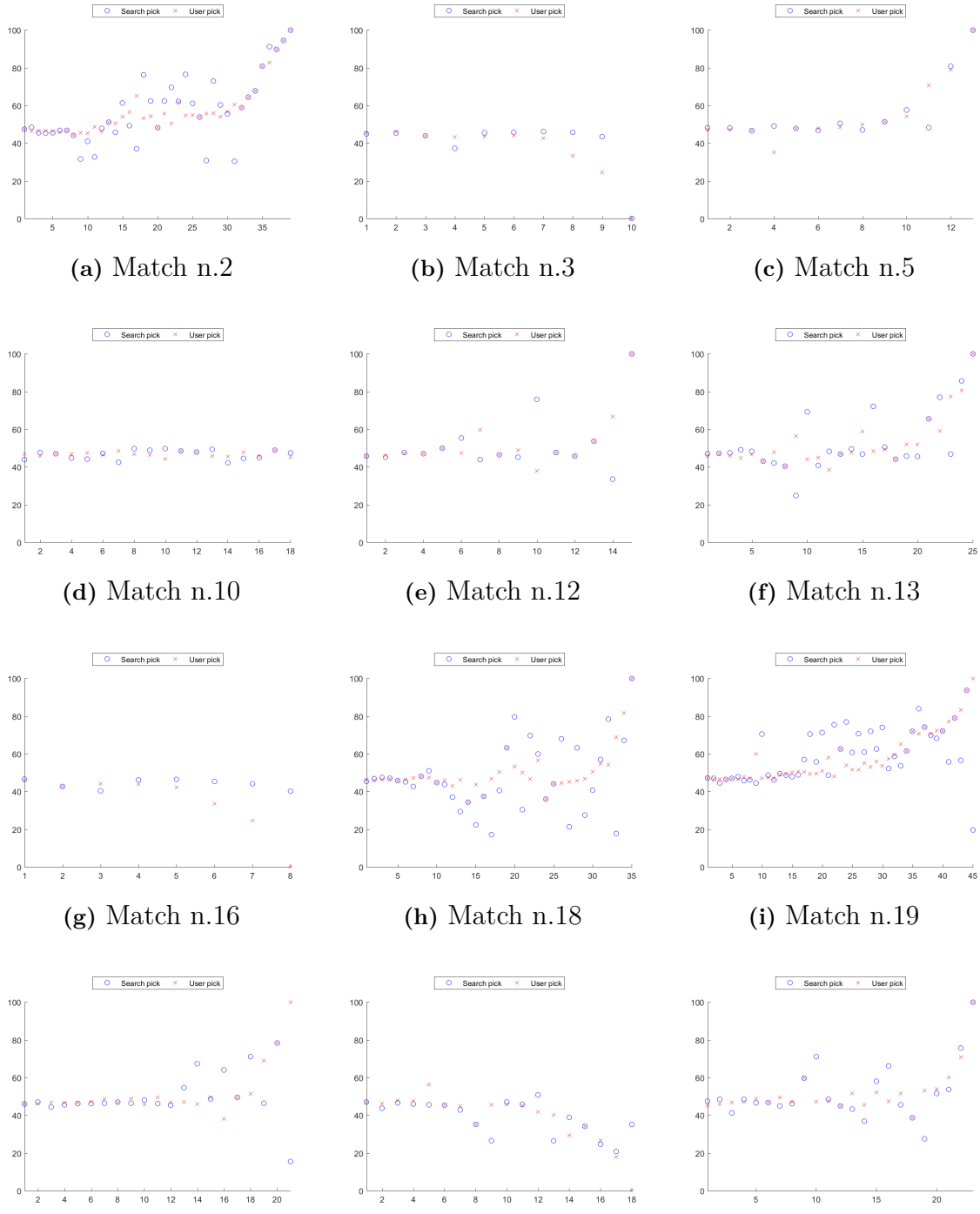
(a) Match n.2 **(b)** Match n.5 **(c)** Match n.7

(d) Match n.10 **(e)** Match n.12 **(f)** Match n.13

(g) Match n.14 **(h)** Match n.18 **(i)** Match n.19

(j) Match n.20 **(k)** Match n.21 **(l)** Match n.22

Figure B.32: Evaluation of Connect Four for Random Individual C, with moves on x-axis and QValues on y-axis.



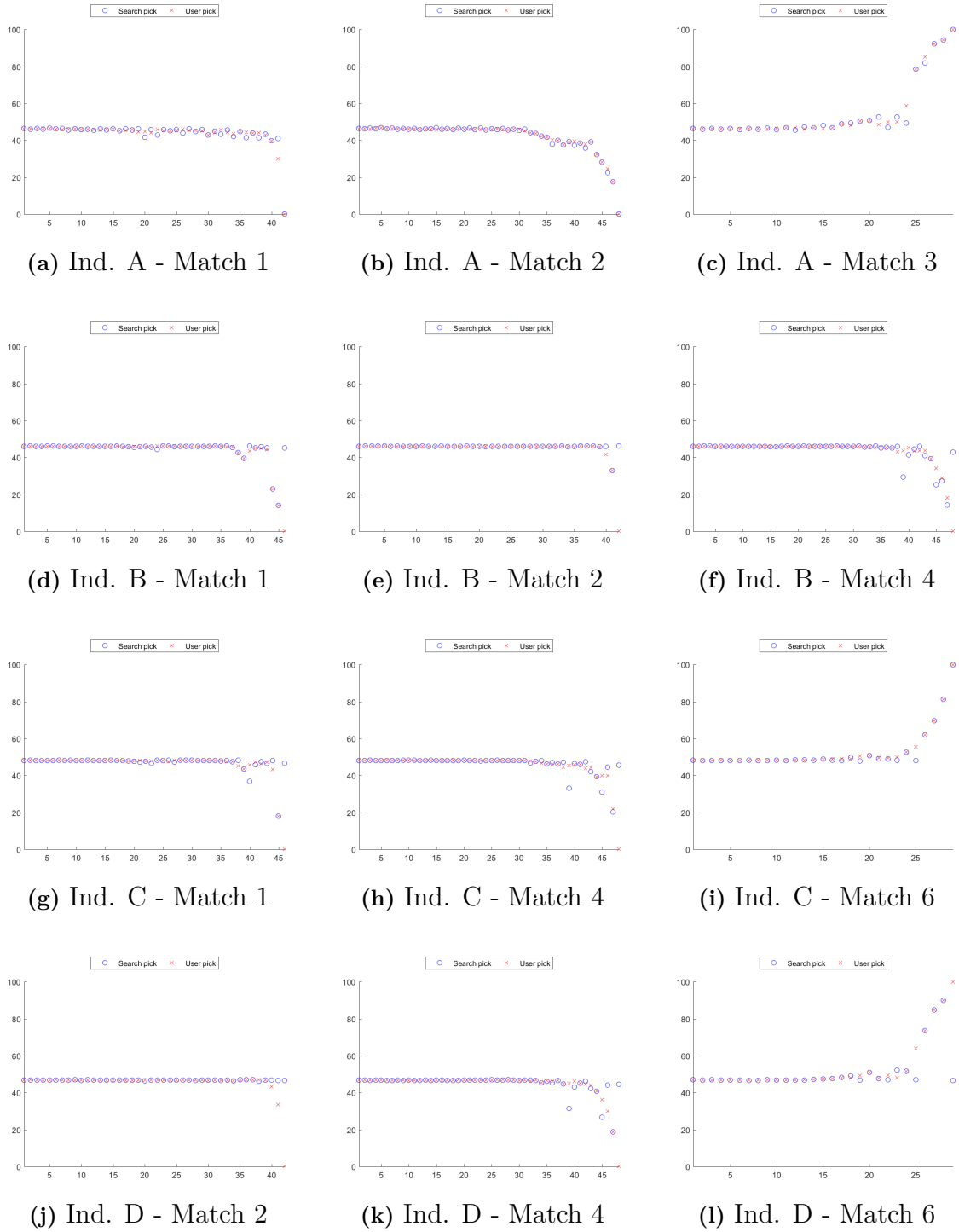
(a) Match n.2 **(b)** Match n.3 **(c)** Match n.5

(d) Match n.10 **(e)** Match n.12 **(f)** Match n.13

(g) Match n.16 **(h)** Match n.18 **(i)** Match n.19

(j) Match n.20 **(k)** Match n.21 **(l)** Match n.22

Figure B.33: Evaluation of Connect Four for Random Individual D, with moves on x-axis and QValues on y-axis.



(a) Ind. A - Match 1 (b) Ind. A - Match 2 (c) Ind. A - Match 3

(d) Ind. B - Match 1 (e) Ind. B - Match 2 (f) Ind. B - Match 4

(g) Ind. C - Match 1 (h) Ind. C - Match 4 (i) Ind. C - Match 6

(j) Ind. D - Match 2 (k) Ind. D - Match 4 (l) Ind. D - Match 6

Figure B.34: Evaluation of Checkers games for Random Individuals, with moves on x-axis and QValues on y-axis.