FEASIBILITY OF GAME THEORY AND MECHANISM DESIGN TECHNIQUES TO UNDERSTAND GAME BALANCE

A Thesis

Submitted to the Faculty

of

Purdue University

by

Prajwal Balasubramani

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science in Aeronautics and Astronautics

August 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF THESIS APPROVAL

Dr. Daniel A. DeLaurentis, Chair

School of Aeronautics and Astronautics

Dr. Ali Khalid Raz

School of Aeronautics and Astronautics

Dr. Jitesh Panchal

School of Mechanical Engineering

Approved by:

Dr. Gregory A. Blaisdell

Head of the Graduate Program

To my parents and to everyone who strive to make education accessible.

ACKNOWLEDGMENTS

I would like to sincerely thank my advisor, Dr. Daniel DeLaurentis for his mentoring, support and encouragement during my master's program at Purdue. It was his faith in me that made it possible for me to complete and obtain this M.S. degree. The time and resources he has invested in me will always be deeply appreciated.

I would also like to take this opportunity to thank the other members of my committee for their encouragement and support: Dr. Ali Khalid Raz and Dr. Jitesh Panchal. Their expert advice has been invaluable in making this research concise and compelling.

I want to acknowledge the initial contributions by Dr.Santiago Ontanon for developmental work on the microRTS game and making it openly available for research (and/or fun). I also owe a debt of gratitude to Apoorv Maheshwari, Adam Dachowitz, and Chris Brand who were a part of the developmental team of the microRTS framework and also helped me understand the game mechanisms better.

Finally, I would like to thank all my peers and friends within the CISA Lab and Purdue University for all the good times. Especially Andrew Haines, Ashish Patel, Chris Brand, Melanie Grande, and Zhong Thai who made the experience more enjoyable.

TABLE OF CONTENTS

Page

LIST OF TABLES vii

LIST OF FIGURES viii

ABBREVIATIONS x

ABSTRACT xi

1 INTRODUCTION 1

1.1 Motivation 1

1.2 Perspective of Multi-Agents Systems (MAS) 3

1.2.1 Games as MAS 3

1.2.2 Nature of players in games 5

1.3 Research Problem 6

1.4 Thesis Structure 7

2 LITERATURE REVIEW 9

2.1 Game Balance 9

2.2 Prisoners' Dilemma 11

2.3 Game Theory and Mechanism Design 13

2.3.1 Game Theory 13

2.3.2 Mechanism Design 15

3 MULTI-AGENT SYSTEM FRAMEWORK 17

3.1 Multi-Agent System Framework 17

3.1.1 Agents and Characteristics 18

3.1.2 Multi-Agent Systems 20

3.2 Agent Based Models 23

3.3 microRTS 25

3.3.1 Maps 26

Page

3.3.2 Agents, Objects and Actions 27
3.3.3 Bots and Strategies 29
3.3.4 Game Data 30
3.3.5 Mechanisms, Objectives and Perspectives 31
4 APPROACH 34
4.1 Design of Experiments 34
4.1.1 Self-play Experiments 36
4.1.2 Cross-play Experiments 38
4.2 Simulation Setup 38
5 RESULTS 41
5.1 Self-play games 41
5.1.1 Baseline - Results from Native Form 41
5.1.2 Standalone Experiments 42
5.1.3 Asymmetric Experiments 47
5.2 Cross-play games 51
5.3 Discussion 54
6 CONCLUSIONS AND FUTURE WORK 57
6.1 Summary 57
6.2 Future Work 58
REFERENCES 60

LIST OF TABLES

Table Page

3.1 Characteristics of MAS. 22

3.2 MAS Architecture Attributes 23

3.3 Agents, Objects and Actions in microRTS 28

3.4 Bots and Strategies in microRTS 30

4.1 Native form of microRTS (baseline model) 35

4.2 Standalone Self-play Tournaments 37

4.3 Asymmetric Self-play Tournaments 38

4.4 Cross-play Tournaments 38

5.1 Preliminary Baseline Game Results 41

5.2 Tournament results for RQ generations 43

5.3 Tournament results for RP generations 44

5.4 Tournament results for BP generations 45

5.5 Tournament results for SR generations 46

5.6 Tournament results for RQ-RP generations 48

5.7 Tournament results for RQ-BP generations 50

5.8 Tournament results for RQ-SR generations 51

LIST OF FIGURES

Figure		Page
1.1	MAS-Games Methodology [7]	5
2.1	Variation of Prisoners' Dilemma	12
2.2	Classical setting of Prisoners' Dilemma	14
3.1	Agent Classification [28]	19
3.2	Map GUI in microRTS	27
3.3	Agents in microRTS [34]	29
3.4	Example Trace File	31
4.1	Native microRTS - Map	34
4.2	Native microRTS - Map	36
4.3	microRTS GUI	39
5.1	Time taken by Player 2 to win	49
5.2	Win(Payoff) Matrix for Cross-play tournament	52
5.3	Win(Payoff) Matrix for Cross-play tournament	53
5.4	Dominant Strategy Equilibrium	54
1	Native form & Cross-play Results	64
2	Resource Quantity - Gen1 Results	65
3	Resource Quantity - Gen2 Results	66
4	Resource Quantity - Gen3 Results	67
5	Resource Position - Gen1 Results	68
6	Resource Position - Gen2 Results	69
7	Resource Position - Gen3 Results	70
8	Base Position - Gen1 Results	71
9	Base Position - Gen2 Results	72
10	Base Position - Gen3 Results	73

Figure Page

11 Starting Resources - Gen1 Results 74
12 Starting Resources - Gen2 Results 75
13 Starting Resources - Gen3 Results 76
14 RQ-RP - Gen1 Results 77
15 RQ-RP - Gen2 Results 78
16 RQ-BP - Gen1 Results 79
17 RQ-BP - Gen2 Results 80
18 RQ-SR - Gen1 Results 81
19 RQ-SR - Gen2 Results 82

ABBREVIATIONS

ABM	Agent Based Model(ling)
MAS	Multi-Agent Systems
RTS	Real Time Strategy
DOE	Design of Experiments
SOF	Social Objective Function
POF	Player Objective Function

ABSTRACT

Balasubramani, Prajwal M.S.A.A, Purdue University, August 2020. Feasibility of Game Theory and Mechanism Design Techniques to Understand Game Balance. Major Professor: Daniel A. DeLaurentis.

Game balance has been a challenge for game developers since the time games have become more complex. There have been a handful of proposals for game balancing processes outside the manual labor-intensive play testing methods, which most game developers often are forced to use simply due to the lack of better methods. Simple solutions, like *restrictive game play*, are limited because of their inability to provide insight on interdependencies among the mechanisms in the game. Complex techniques framed around the potential of AI algorithms are limited by computational budgets or cognition inability to assess human actions. In order to find a middle ground we investigate Game Theory and Mechanism Design concepts. Both have proven to be effective tools to analyse strategic situations among interacting participants, or in this case 'players'. We test the feasibility of using these techniques in an Real Time Strategy (RTS) game domain to understand game balance. MicroRTS, a small and simple execution of an RTS game is employed as our model. The results provide promising insight on the effectiveness of the method in detecting imbalances and further inspection to find the cause. An additional benefit out of this technique, besides detecting for game imbalances, the approach can be leveraged to create imbalances. This is useful when the designer or player desires to do so.

1. INTRODUCTION

1.1 Motivation

In the last three decades, games have moved from arcade machines to consoles to hand-held mobile devices to cloud operations. With this access to higher and large-scale computational capabilities, games have increasingly become highly detailed but at the same time more complex than before. With this ability to produce high fidelity games, the research potential of games as surrogate models is extensive. Games worldwide are in as many as genres that the human mind can conceive; supply-chain, warfare, math and computation, pattern recognition, machine vision and many more. An interesting result is that these genres directly relate to everyday problems we face around the world. The possibility of using games to test and hypothesize ideas and concepts is very much now in the realm of possibility. We have already experienced this wave occur in the artificial intelligence and machine learning sectors. AlphaGo for GO, AlphaZero and Deep Blue for Chess, and more recently AlphaStar for Starcraft II are some examples that use games as the main source of motivation to build really intelligent decision making AI [1]. While this perspective really shows its potential, game design like modelling also has some inherent flaws. The most concerning of which is understanding game balance.

Simple games with few interacting elements have very few design features that control and impact game play. In games that utilize multiple levels of elements interacting within and across different levels, the complexity in designing the interactions increases exponentially. The technical and gaming term for these interactions is mechanisms. 'Mechanisms', often also referred to as mechanics by professional gamers, and controls the game states and outcomes. There has been a lot of confusion over the semantics of the words 'game mechanisms' and 'game mechanics' but it is now widely

accepted to be the same in the gamer and game design community. For example, in the game of *rock, paper and scissors* the designer is only worried about three mechanisms i.e., the interaction between the three elements. This shows that a game of this scale is easier to understand and design. When we shift to games simulating warfare, real-time strategy (RTS) games such as *Age of Empires* or *Civilisation* or *League of Legends* which involve hundreds, and in some cases thousands, of elements simultaneously interacting, the designer is burdened with establishing every mechanism possible within the game environment fairly for all players. Creating these mechanisms is necessary for a game to be labelled as a 'complete game' or a 'well-defined game'. A 'good/fairly' balanced game needs these defined mechanisms to be just. Going back to the rock, paper and scissors game, hypothetically, if rock beats both scissors and paper then the game would not have been as popular as it is today due the presence of a major imbalance in favor of the player choosing rock as their strategy. This notion of having an equal playing field is to create a more engaging game play and is commonly known as game balance. For smaller games it is easier to interpret the game play in a quantitative sense and arrive at balancing game mechanisms. For complex games like Starcraft II, which is an online multiplayer real time strategy game, achieving game balance is much more complicated than just a single attempt at balancing, quantitatively or analytically.

In this work, our focus is not to prove that game theory and mechanism design techniques are possible methodologies to fix balancing issues in existing games but rather to test the feasibility of using the same to detect and point to the roots of imbalances in a game. The lens of the work is motivated to demonstrate the applicability of this methodology to find imbalances in real world multi-agent system (MAS) scenarios, like warfare, supply chain, and search & rescue-missions etc and one such technique being explored involves the use of game theory and mechanism design.

1.2 Perspective of Multi-Agents Systems (MAS)

The difference between *traditional systems, complex systems* and *multi-agent systems* can be understood by thinking in terms of the constituent elements. In traditional systems, the elements or components, do not have independence in operation or management. Components merely carry out functions defined by the system using rules and protocols. The working definition used at MIT Engineering Systems Division [2] for a system and complex systems are as follow:

"System: A set of interacting components having well-defined (although possibly poorly understood) behavior or purpose; the concept is subjective in that what is a system to one person may not appear to be a system to another."

"Complex Systems: A system with numerous components and interconnections, interactions or interdependencies that are difficult to describe, understand, predict, manage, design, and/or change."

In complex systems, these components are replaced by fully independent system that is motivated to run on its own. An airplane is a perfect example for complex system. However, this is not what a multi-agent system is, even though the term seems to point to the same thing. It is the word *agent* that changes the perspective of the definition. The main difference between MAS and other systems is that, the designed purpose of MAS is not necessarily a common goal. MAS design is more focused on individual agent interaction and the environment. This is mainly why, in this work, game environments will be translated to an MAS setting. Making such translations help to analytically assess the game states and design of experiments. A more detailed discussion and relevant literature on multi-agent systems is presented in Chapter 3.

1.2.1 Games as MAS

There are two ways games and MAS correlate. One from the perspective of the player and the other from the designers'. We will look at both of these perspectives

and choose accordingly what fits our research problem approach. Both are equally important but for the majority of our analysis we use the designers perspective.

From a players perspective the game is a single entity posed as an internal system with which they interact. This makes the players active agents within the system environment. To justify such games as an MAS, the number of players must be really high to contribute to the *multi-agent aspects* of MAS. A well established example for a game being considered as an MAS based on the players perspective is Massive Multiplayer Online Games (MMOGs) [3] such as World of Warcraft. This example also clearly shows the two polar paradigms within the MAS; autonomous and dependent. The autonomous aspects are from individual players and the dependent aspects are from games that involve individual players to collaborate or, more aptly here, form clans.

All, if not most, games inherently are ABMs and this is what drives the designers' perspective. Every component of the game designed is done using the ABM methodology and is built with independence and behavioural protocols. Depending on the game style/type, they are either an MAS or just a single agent system. Games like Tetris can be thought of as a single-agent system, but RTS games on the other hand have a large number of agents present. In this investigation, we will use an RTS environment and hence this perspective becomes really important while conducting the design of experiments. The interactions among the elements of the game and the environment, are essentially what *mechanisms* are in games. This perspective has been successfully used previously to create ABM role playing games (RPGs) for the purposes of research, training and negotiation support in the field of renewable resource management by multiple groups of researchers [4] [5] [6]. This relation between games and MAS is well represented in Figure 1.1.

Based on these correlations it is completely valid to say that games are low fidelity surrogate simulators for real world scenarios represented in the form of an MAS. This translation allows us to use insights about MAS design methodologies & analytics

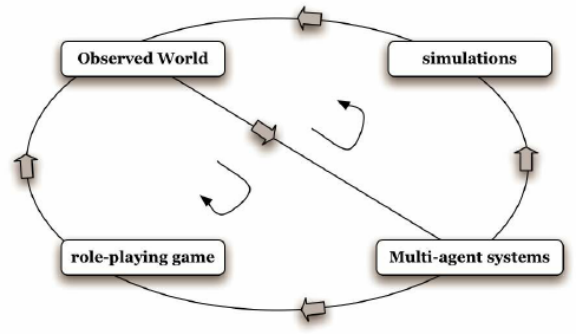

Figure 1.1.: MAS-Games Methodology [7]

and apply them to game settings. Tracking, monitoring and information within the game is also made possible by this Games-to-MAS translation.

1.2.2 Nature of players in games

There two main classifications for players' nature that are generally adopted in games: *Cooperative* and *Non-Cooperative*. The term *nature* here refers to how the player behaves in the game and not the specific actions or action sequences as conveyed by the term *strategy*. Of course, it is directly dependant on the setting of the game whether it allows for players to be able to use either or are restricted to one. Essentially there are games, that run on a predetermined setting of players nature, and there are others that allow the players to decide their nature. This classification is purely of the players behavior but not the classification of the types of games that are defined by game theory, of which there are many.

Cooperative games are ones in which the players are in a collaborative setting and are not competing against each other. The game may have a defined environment that the players are competing against in union, for example single player survival games.

Non-Cooperative or Competitive games are ones in which the players are all functioning independently and are up against one another. Several themes like is multi-

player survival, sports or RTS games are under this category. Sometimes the game environment is also involved in the game design to be competitive against the player.

In our context of investigating game balance using game theory and mechanism design techniques, the cooperative setup becomes infeasible to study as the computational budget is very expensive to completely explore the game states. This is because, in cooperative games the possible game states for a particular scenario are much larger in comparison to competitive settings. In simple terms, the players in competitive settings, no matter what the scenario is, are always forced to take actions that are binary in nature i.e., do or die. In a similar scenario, if the players were cooperative, it allows them to cooperate or co-exist. Just on this level of abstraction the number of cases to be investigated is double in case of cooperative (two decision trees) games over competitive games (one decision tree). Hence the investigation in this research is restricted to a competitive game setting, specifically RTS themed.

1.3 Research Problem

Now that the importance of game balance is understood in general game design terms, the next logical step is to look for its presence in games and evaluate. Using the comparison of MAS to Games, allows us to model real life scenarios into games, and assessing balance there would give us vital insights on the rules and functions in the real domain. The general term used in this work is *game* but it is looked at as a surrogate word to any real world scenario that is competitive in nature. Knowing the balance in any such games also provides an understanding on the *fairness* of the scenario. It further helps in planning actions to make sure the balance is maintained or established. However there may be other cases where the balance is established but it is of our interest to alter it to be imbalanced favoring one side, like that of a war or sports game where both teams strive to be victors.

The possible avenues to explore these competitive games is captured well by the concepts of game theory that are preexisting and widely used. Game theory helps

us understand the game balance from the perspective of a players motivation to do something in the game. in other words, the payoff or strategy matrix. The inclusion of mechanism design is to aid two different purposes. One, to help in the design of experiments for testing the methodology. Two, to use mechanism design analytics to understand the state of balance in a game.

"How do we leverage the concepts of Game Theory and Mechanism Design to understand and assess game balance?, Can game balance be defined by a quantitative measure or parameter within or outside of the game?, Are these methodologies feasible, if so to what extent?" are some of the quantitative questions that this work is trying to address. Questions like: "What are some of the shortcomings in using these methods?, What assumptions are necessary for these methods to be applicable?" are some analytical conclusions we seek.

1.4 Thesis Structure

The remainder of the thesis progresses as follows:

In Chapter 2, the relevant literature to understand and address game theory, mechanism design, and game balance are discussed. Game theory and mechanism design are very large bodies within the economics and mathematics fields of study, hence only the portions of those that are in use in this research are reviewed in depth. Existing methods to detect and quantify game balance are introduced along with current strides made in the field. The commonly known Prisoners' Dilemma game is used as a demonstration tool throughout the section to help in understanding the relevant methodologies.

Chapter 3 provides more context on Multi-Agent System Framework approach utilized. Agent based models (ABM), which RTS games are based on, is also discussed along with its history, benefits, and applicability. microRTS, a simplistic RTS game, used as the model to test our hypothesis is presented. The inner workings of the game and the external setup for the study are also described.

In Chapter 4, the approach to modelling the players, games and design of experiments is introduced. Further, the programming setup to run large number of games and its accompanying analysis setup is presented. The mechanism changes, metrics, and process of collection of data is also explained.

Chapter 5 presents all the results obtained from the different tournament settings and experiments. General discussion on the effectiveness of the methodology to assess game balance is presented. All the research questions are then addressed one by one, with quantitative details from the results supporting them. Specific and interesting games are also elaborated in a case studies subsection.

In Chapter 6, the key findings are summarized and potential applications are presented. Avenues of future research possibilities extending from the findings are also discussed.

2. LITERATURE REVIEW

Game balance is a growing concern for game developers and professional game players. With the increase in games used as surrogate research models, more researchers are also now posing questions about game balance. In this chapter we review the literature for game balancing and introduce the fundamental concepts of game theory and mechanism design which will be key to our approach to detecting and identifying the causes of imbalance in games. To understand the application of these two concepts we use a demonstration game problem, the commonly known *Prisoners' Dilemma*.

2.1 Game Balance

From the time games shifted from being a recreational activity into a household entertainment element, the explosive industry has had game developers tasked to make games engaging and challenging for the masses to invest their time and resources. Engagement is a concept that can be delivered by visuals, sound design, story and/or characters. Player frustration, a counter-intuitive factor contributing success of games, is a very sensitive element that game designers are always worried about. It feeds into both, a game being playable and enjoyable. In order to handle this balance between frustration, which drives one to play more and beat the game, and playability, a designer must make sure the elements and their interactions within the game environment are not biased to any player irrespective of their skill. A very commonly quoted example in this aspect is the game of Chess. It is closest to a well balanced game, the small imbalance comes from the fact that white moves first giving it a small edge over black. There have been attempts to get over this hurdle by making a real-time chess called Kung-Fu Chess where there is no turn based actions. This was critically acclaimed by the audiences, however it removed

the time based frustration that players of chess enjoy killing its popularity slowly [8]. According to Jaffe (2012), questions like "Does either player have any advantage?", "Does every action serve a useful role?" and/or "How much long-term strategy is there?" lie at the centre of assessing game balance [9]. For example, while developing Halo 3 the designers discovered that the sniper rifle was overpowered, reducing effectiveness of several interesting game strategies [10] contributing to an imbalance via advantages to player over game conditions. This was only realised because of a good play-testing setup(games are played by professionals, developers and a small section of casual gamers to test all aspects of the game). Many games are released without such quality work going into balancing it. Most games then receive either updates or patches to fix issues, and some get a complete overhauls. This is a very time consuming and expensive process for the developers to indulge in. This naturally has led teams of researchers, either within the gaming developer community or the AI community, to ask whether it is possible to automate this. This effort was brought to a wider audience by the use of a backend reasoning tool which was not dependent on play-testing [11]. This was however soon halted by the inability of AI to follow human predictability play, since many existing balancing methods relied on observing human-play [12]. Of the many balance related questions, the impact of starting order in turn-based games is the one that is well addressed. Others are still being explored to find ways to control and evaluate. Many efforts have been made to understand game balance and figure out the solution to reduce the need for human play-testing and iterative processes by implementing automation techniques. Jaffe (2013) demonstrated a novel quantitative method using programmed agents instead of human players to assess game balance. He further proposed a restricted game play method, which allows perturbation of game parameters in only one dimension and assess the change to the game state without the effects of the others, and called it restricted gameplay [9] [13]. Beyer et. al., proposed an integrated methods to act as a stepping stone to eventually move away from manual balancing with their integrated balancing process [14]. While Leigh et. al., state that mathematical proof for game

balance may not be possible with the current AI techniques, it can certainly be validated using coevolutionary algorithms [15]. More recently, Wen Xia et. al., (2016) explored the space of multi-agent models intertwined with ecosystem mechanism iin order to provide a more flexible way of game balance control [16]. Our approach takes the work of Jaffe (2012) and Wen Xia (2016) and leverages it to understand game balance in an RTS game.

2.2 Prisoners' Dilemma

Prisoners' Dilemma is a standard example (2.1) for a game analyzed using game theory. It has two individuals who are acting in their own self-interests in a paradoxical decision making scenario. The construct of the game is that both the players are put to the test in front of the justice system to confess or deter for a crime committed of which the two players are suspects. It presents a situation where two players are separated and unable to communicate, must either choose to co-operate with the other or not i.e., confess for the crime or remain silent. The outcomes of the game are structured in such a way that, if the players cooperate then their sentences are the least and, if they choose not to cooperate, the penalty is the highest. The middle case where one cooperates and the other doesn't, the one who remains silent gets no sentence and the one cooperating receives the entire sentence. The problem is setup in such a way that the players are always motivated to protect themselves. This is considered as one of the most well known concepts in modern game theory because it is observed that this form of game occurs very commonly in aspects of economy. There have been multiple variations of the game over the years. Ones where cooperating is rewarded and others where deterring is rewarded. This is completely up to the designer of the game, and this slowly leads into the concepts of mechanism design. The prisoners' dilemma is a balanced game, if the setting has symmetric penalties and rewards for both players. A change in either of those makes it an imbalanced

game in favor of the player with lower penalty or higher rewards because of the biased outcomes even though we cannot predict the actual outcome of the game.

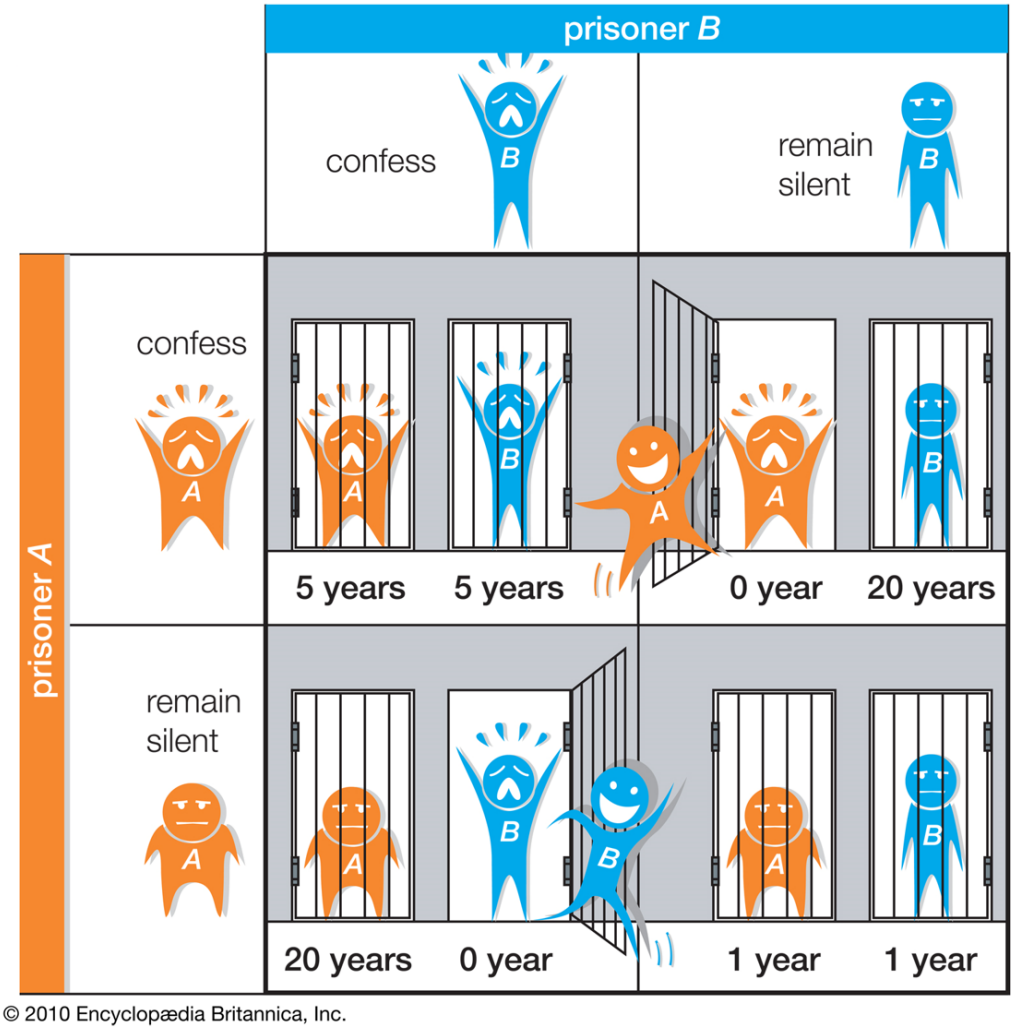

Figure 2.1.: Variation of Prisoners' Dilemma

In the next section we will further explore and understand the different concepts of mechanism design and game theory while using this game as a demonstration problem.

2.3 Game Theory and Mechanism Design

This section presents the traditional concepts of game theory and mechanism design relevant to the game balancing concepts. We begin by introducing game theory which is intimately related to mechanism design. In fact, mechanism design is often referred to as reverse game theory.

2.3.1 Game Theory

Game Theory is the study of the strategic interactions of agents that are self-interested within a system or model. Game theory deals with the preferences, states, information, decisions, and outcomes of the game.

In a given game with more than one agent/player with each player having personal preferences over the outcomes of a game. **Strategy** is a complete plan or decision protocol, that determines what action will be taken by agents in every possible state of the game. Strategy is a single decisions made by the agents/players preference. A vector of such available strategies for an agent to take is called a **strategy profile**. In the example of the prisoners' dilemma from Figure 2.1, player A's strategy profile has two vectors: *Confess* & *Remain silent*. A Strategy in this case would be a decision that player A makes; here that happens when player A assesses what player B may have done and tries to look for the best outcome for themselves. **Best Response** or **Best Reply** is the strategy implemented by a player, conditionally, with respect to the decisions made by the opponent. For every strategy in player A's side, Player B has a best reply based on the reward/penalty. This is relevant only in games where full information is available to the players. From a standpoint of a designer, who has access to all the information, best replies can be determined even for incomplete information games.

In a balanced game, for example lets consider the prisoner's dilemma game as structured in Figure 2.2, where the case is symmetric, and is balanced. A new change

in the penalties compared to the previous game (2.1) is that the new protocol has cooperation and remaining silent together equally rewarding.

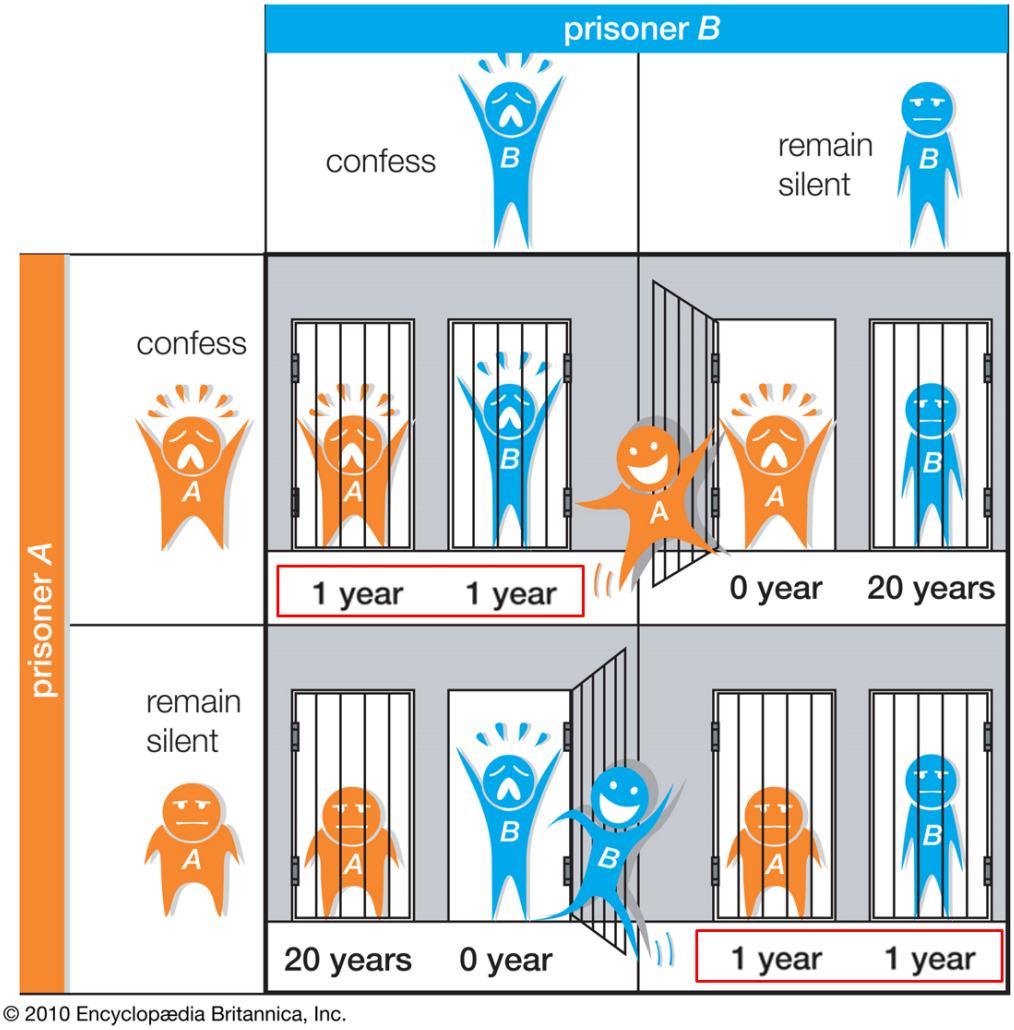

Figure 2.2.: Classical setting of Prisoners' Dilemma

In this case, both the player benefit when they take the same decision and is fair to both, and is completely symmetric in setting. Neither decisions are unbiased to either player. In this case if the designer changed the penalties of only player B to be 5 years instead of 1 year in both cells 1 and 4. Then this is an unfair design where deterministically a previously balanced game now produces different responses from the players (at least player B). A game that was a tie, in a sense, is no longer

deterministically producing similar outcomes and states. The introduction of new changes triggered this effect. This is known as the **Grim Trigger**, a mechanism or strategy to trigger game states from their nature of repeatability. This feature helps us to assess game balance using game theory for our model. Other popular game theory solution metrics are Nash Equilibrium and Dominant Strategy Equilibrium.

Nash is an equilibrium in which, every agent will select a strategy to maximize their own utility/reward when the strategy of the other agents are given. Game theorists use and have been using Nash Equilibrium as one the fundamental solution concepts. Having said that, it does come with some restrictions. Mainly because it assumes that no information is private ans is globally known. It also does not hold good in games with multiple equilibrium i.e., if the game has multiple Nash solutions. This can be overcome if the Nash of a game is being determined by a designer. Dominant Strategy Equilibrium is a stronger metric than Nash which points to a state where all agents will have the same behavior of reward maximizing irrespective of other agents information and strategy. In the prisoners' Dilemma game (Figure 2.1), the Nash equilibrium is when both prisoner's confess. Nash equilibrium is an intersection of the best responses of each player. Here both players are maximizing based on the information of the other players strategies. In terms of Dominant Strategy, both prisoner's remaining silent is the solution. Here the player only thinks about self-maximizing strategy and that is remaining silent to try to avoid any sentence (0 years). It is possible that dominant strategy solution is also in the Nash (in Figure 2.2) although the underlying principles of a dominant strategy leave Nash analysis redundant.

2.3.2 Mechanism Design

Mechanism Design, is a field within the field of microeconomics and game theory. It is the design of mechanisms, protocols, and institutions that are mathematically proven to satisfy system-wide objectives. In our game, it is the the system wide

objective a.k.a social objective is to provide and maintain a fairly balanced game. It also functions under the assumption that agents in the model act in a self-interested manner and may hold private information that is relevant to make internal decisions. Mechanism Design previously has been used in various fields to design protocols and institutions. Gerkey & Mataric (2002) used mechanism design to structure an auction system for autonomous robots for task allocation [17]. Holzman et. al., (2003) propose a new way to apply mechanism design to network routing [18]. In the same category of network capacity allocation Anshelevich et. al., (2004) and Anderson et. al., (2006) also proposed using mechanism design [19], [20]. Both Hinz (2003) and Federico & Rahman (2003) looked at its application in electricity markets [21], [22]. Dang et. al., (2006) utilized these concepts in the field of sensor fusion [23]. In systems engineering, M.Chen et. al., (2004) was one of the first to combine these concepts into the system analytics world to assess peer-to-peer systems [24].

Social Choice Function or Social Objective Function (SOF) is the selection of optimal outcome given agent types and thus satisfies the system wide goal. The SOF selects an alternative from a set of choices given all agents' preferences. The goal of using this in mechanism design is to implement it to satisy certain desired properties. For example, in the prisoners' dilemma game, the SOF for the designer can be that they want the truth to be reported by the players and justice to be served rightly. So the SOF can be structured in a way that the maximum reward is presented to the players for confessing rather than remaining silent. In our game model, we want to identify the imbalances in the game, so we use SOF to structure our tests/games. Similarly, mechanism design also provide the facility to players to have their own objective function called the Player Objective Function. This allows the designer to make sure the players payoff is something that the player actually is willing to play for. If the prison time in all the 4 cases in the game were the same, the player has no utility to satisfy their objective function i.e., minimize penalty (or) maximize reward. With the SOF and POF as the fundamental functionality of mechanism design, this will be leveraged to design and game theory for assessing game balance.

3. MULTI-AGENT SYSTEM FRAMEWORK

In this chapter, an overview of Multi-Agent Systems and the framework utilized to generate a stand-in model for our research problem is presented. Subsequently, agent based modeling is also introduced to draw parallels between MAS, ABM and, microRTS. microRTS is an open source RTS game built on ABM principles that is described in depth to provide context to the research model.

3.1 Multi-Agent System Framework

In the previous sections, we looked at the fundamental concepts of game theory and mechanism design, specifically in the prisoners' dilemma setting. However, to exhibit the practicality and use case of mechanism design and understand the processes involved in the lead up to using MAS framework to interpret game balance, we need a surrogate model to replicate the complexity and behaviour of an MAS. To do so, we consider an RTS game which exhibits classical game theory setting where payoffs are not just dependent on actions of the player but also a result of the opponents actions. Before we jump into the actual surrogate model, we need to understand what MAS is and what are its characteristics. Of those we specifically discuss the ones relevant to our framework.

Since the mid-1990's, the field of MAS has presented very interesting approaches to solving problems and hence a large number of mathematicians, scientists, and engineers are in pursuit of identifying and solving unique problems. Since the advent of intelligent systems and micro-electronics, fields like robotics, artificial intelligence and, swarm technologies are heavily dependent on MAS principles and architectures. Similar to system-of-systems (SoS), the field of MAS is also typically used in cases requiring large scales of individual and independent agents with defined behavior,

communications, beliefs, knowledge, intentions and mechanisms. In SoS however, the architecture also includes a hierarchical management aspect that may or may not be necessary in an MAS setting. Even if a hierarchy exists, the SoS hierarchy addresses layers of varying compositions unlike the homogeneous hierarchies in MAS.

3.1.1 Agents and Characteristics

The concept of multiple systems or large network of systems is well understood, but to draw comparisons and similarities between those and MAS we need to understand what 'agents' mean in the MAS context. Further we also should understand the characteristics of each of these agents. Most commonly "an agent is a computer system that is situated in some environment, and that is capable of autonomous action in the environment in order to meet its design objective" is defined by Wooldridge [25]. An older definition by Russell et. al., states that an agent is "anything that can perceive its environment through sensors and act upon that environment through actuators" [26]. Many authors continued and still continue to provide unique definitions. Feber [27] in his widely popular book on MAS deconstructed the definitions and proposed a subset of defining descriptions which is widely accepted and the most detailed:

- capable of acting in an environment
- can communicate directly with other agents
- driven by a set of tendencies
- possess resources of its own
- capable of perceiving its environment
- only a partial representation of this environment
- possesses skills and can offer services

- may be able to reproduce itself
- behavior tends towards satisfying its objectives, taking account of the resources and skills available to it and depending on its perception, its representations and the communications it receives.

Agents exhibit a host of different characteristics, enabling us to classify them using different schemes. Among many existing schemes, most general is the distinction based on characteristics like autonomy, adaptivity, reactivity, goal orientation, utility orientation, physicality, and social functions. Other common classifications are based on tasks they perform, range and sensitivity of their perception, control structure, internal states and/or environment conditions. Franklin & Grasser provided a hierarchical classification of agents based on properties of agents.

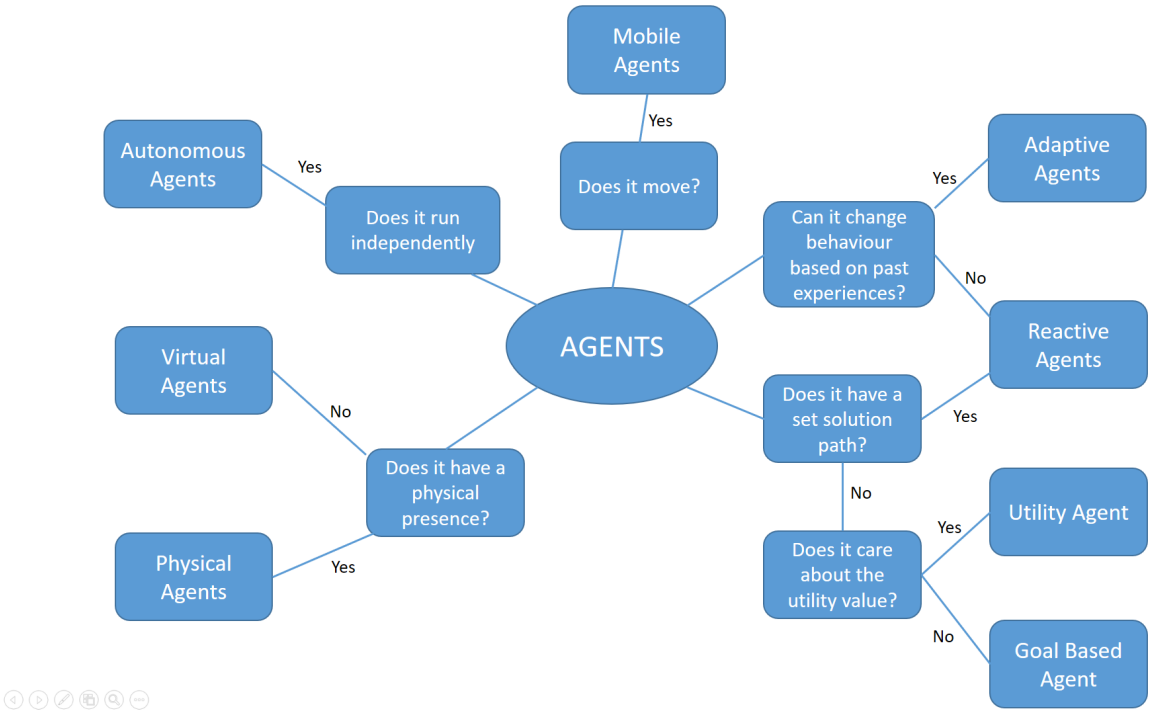

Figure 3.1.: Agent Classification [28]

3.1.2 Multi-Agent Systems

The definition of MAS is more complex than the definition of agents. MAS is a big umbrella term for various kinds of systems. This is due to its branched applicability in many fields, from healthcare to manufacturing to traffic management to autonomous vehicles. Keeping consistency with the definitions, Ferber [27] again provides a set of defining qualities for a system to be classified as MAS:

- An environment or space which generally has volume
- A set of objectives and objects
- An assembly of agents acting as active entities of the system
- An assembly of relations linking objects
- An assembly of operations for agents to perceive, produce, consume, transform and manipulate objects
- Operators with the task of representing the application of these operations and the reaction of the world to this attempt at modification, which we shall call the laws of the universe

The roots of MAS span so wide into multiple host disciplines that are unique, it is difficult to provide a concise and detailed history without making exceptions. However, Ferber 1999 [27] and Bousquet & Le Page 2004 [29] in their work provide to their best abilities, a thorough text discussing in detail about the MAS history. The origin of MAS comes from the field of AI where it first appeared in the second half of the nineteenth century but gained larger attention in the 1980's. Specifically, Distributed Artificial Intelligence is credited to be the root of MAS [29]. This community, like the context in chapter 2, was always focused on single agent systems and their immediate environment. Since the 1980's the focus shifted onto multi agent settings pushing researchers to define and lay down the foundations for a new field.

The applications of MAS are divided into 5 overarching categories by Ferber (1999). These are categorized based on the various application domains, at the time, MAS was being used in. The functionality and objectives of the MAS in each of these categories is unique.

Applications of MAS:

- Problem Solving
- Multi-Agent Simulations
- Building Artificial Worlds
- Collective Robotics
- Program Design

In 2007, six fundamental aspects of MAS that are unique was listed [30] that is widely considered to provide most accurate distinctions from single agent systems. Table 3.1 lists them, defines and also provides context to their meaning in our game balance investigation.

Table 3.1.: Characteristics of MAS.

Characteristic	**Description**	**Context**
Environment	Dynamic and not static environment from an agents point of view due to presence of other agents	Movement of opponent agents around the player makes the environment dynamic
Agent Design	Heterogeneous agents whose behaviors vary in their own extent	Different attacking units in an RTS game have different capabilities and operate with their own protocols
Perception	Agents have partial perception of what is happening in their environment	Player only knows the location of opponents position, but no perception of their resources, units and operations
Control	Control is decentralised for reasons of robustness of model. Coordinating the actions of the agents is a challenge, which is the subject of game theory.	The player (bot) units make decisions on their own based on evaluation functions and protocols
Knowledge	The agents understanding of the world, it varies from one agent to another	What certain units know may or may not be known by other units in the game
Communication	Agents can receive and transmit messages, which allows for coordination and negotiation	Positions of enemy units, resources and bases

With characteristics of MAS clearly laid out, a comprehensive overview of the architecture of MAS with the ranges of their attributes defined by Weiss (1999) is provided in table 3.2.

Table 3.2.: MAS Architecture Attributes

	Attributes	**Range and Type**
	Number	Two or more
	Uniformity	Homogeneous/Heterogeneous
Agents	Goals	Contradictory/Complementary
	Architecture	Reactive/Deliberative
	Abilities	Simple/Complex
	Frequency	Low/High
	Persistence	Short/Long Term
Objects	Pattern	Decentralised/Hierarchical
	Level	Signal Passing/Knowledge intensive
	Variability	Fixed/Dynamic
	Purpose	Competitive/Cooperative
	Predictability	Foreseeable/Unforeseeable
	Accessibility	Limited/Unlimited
Environment	Dynamics	Fixed/Variable
	Diversity	Poor/Rich
	Resource Availability	Restricted/Sufficient

3.2 Agent Based Models

Agent Based Modelling (ABM), also known as Multi-agent Simulations, is a very powerful tool where systems are modeled as a collection of intelligent and/or rule-

based agents. It is relatively a new approach to modelling dynamics of complex systems and MAS. It modelling approach is from the perspective of individual constituents i.e., agents. Generally it follows a bottom-top approach to modelling giving rise to organizational and behavioral characteristics of both the system and environment. Emergence and emergent behavior in ABM is one of the biggest concerns. Even in a simple model with very few interactions, the number of possible complex behavioral patterns are many. Unanticipated and/or non-programmed patterns from agents actually add more information to the state of the ABM. So in a way, ABM helps fix issues within ABMs. Another advantage of ABM is that it is amenable to non-mathematical models too, such as social systems or human systems.

According to Jiang & Gimblett (2002), ABMs can be characterized using four constructs [31]:

- Agents: Set of all simulated heterogeneous behavioral entities.
- Objects: Set of all represented passive entities that do not react to stimuli.
- Environment: Topological space where agents and objects are located and signals propagated.
- Communications: Set of all possible communications between entities.

ABMs are best known for their unique capabilities over other modelling tools, as illustrated by Bonabeau (2002), it captures emergent phenomenon; provides natural description of the system; and its flexible modelling approach [32]. ABM modelling approaches can be divided into three categories based on the type of model being built, and this is described clearly by Axtell (2000) [33].

To solve models that can be completely solved. Mainly used within the traditional space of operations research, where all the processes, operations and interactions are mathematically defined. If any model can be defined completely by mathematical equations, ABM takes the form of Monte-Carlo analysis and if an analytical solution exists then it provides numerical results.

Models that are partially soluble. If models cannot be defined by mathematical and logical equations then the models cannot be completely solved, but can be solved in parts to get some insight on the functioning of the model. It sheds light on possible solutions and properties. On top of these possibilities, these models can be tested using sensitivity analysis to understand the dependencies and stability of the model.

Models that are intractable. If a model is intractable, then it is not possible to define it using mathematical framework. ABM is one of the very few methods with which a systematic analysis can be used as a viable substitute.

Given the similarities in the definitions, approaches and use-cases of ABM and MAS, there exist a plethora of different terms for the similar processes. Some authors refer to such overlapping models as multi agent models and others purely talk about it in the sense of ABM. The terms multi-agent simulation, multi-agent modelling and agent-based modeling can be used interchangeably with some exceptions. Our surrogate model, microRTS (section 3.3), is one such exception. It shows characteristics of both ABM and MAS, and hence the perspective is changed dynamically throughout the study to fit which analytics works well to extract results. From an ABM perspective it allows use to model and control every single agent in the system, and most of this is priori to the analytical processes. However, once we start generating results, it is more fit to assess the data from a MAS perspective,a holistic view, as it is easier to extract and translate the information to work well with the game theory and mechanism design methods. More about this holistic perspective is presented in section 3.3.6

3.3 microRTS

microRTS [34] is a small open-source implementation of an RTS game by Dr. Santiago Ontanon, an associate professor at Drexel University and a research scientist

at Google Research. it is developed in Java and is available with a GPL license for use and distribution [35]. It was primarily designed to perform AI research. The advantage microRTS has over other higher fidelity games like Starcraft or Age of Empires is that microRTS is much simpler, and can be used to quickly test theoretical ideas, before moving on to full-fledged RTS games. Many researchers have used microRTS for testing their hypothesis and ideas, not restricted to AI based topics, successfully. The topics range from hierarchical Task planning in networks to learning action probability models [36] [37]. Application of Convolutional Neural Networks, Monte Carlo Tree Searches and other machine learning methods in RTS games to evaluate game states is also another key research area that microRTS is actively used in [38] [39]. The game's simplistic nature and standard programming procedures, allow us to use it to our own research interests, just like the many before have.

In its default configuration, microRTS is a two-player deterministic and real-time (i.e. players can issue actions simultaneously, and actions are durative) however, our study will involve hard-coded players and not human players. It is possible to experiment both with fully-observable and partially-observable games, as well as with some elements of non-determinism. Thus, it is not adequate for evaluating AI techniques designed to deal with non-determinism (although future versions of microRTS might include non-determinism activated via certain flags). For our testing, we will only be using fully-observable games in the deterministic setting. As part of the source code, the developer has also included a collection of hard-coded, and game-tree search techniques (such as variants of minimax, Monte Carlo search, and Monte Carlo Tree Search) for player bots to utilize. In section 3.3.2 we will discuss more about the player bot strategies involved in our research.

3.3.1 Maps

The map in microRTS is a basic 2D array that stores information as physical cell for each cell in the array. The array forms a GUI grid that the source code already

has an implementation for. The size of the grid is variable according to the users choice, but the code comes with a few standard settings; 8 X 8 (Figure 3.2), 12 X 12, 16 X 16, 24 X 24, and 32 X 32 cells. The game by default opts a dark background for the map and uses high contrast bright colors for agents on the map. The map GUI also has a timer and player statistics indicator at the bottom left corner.

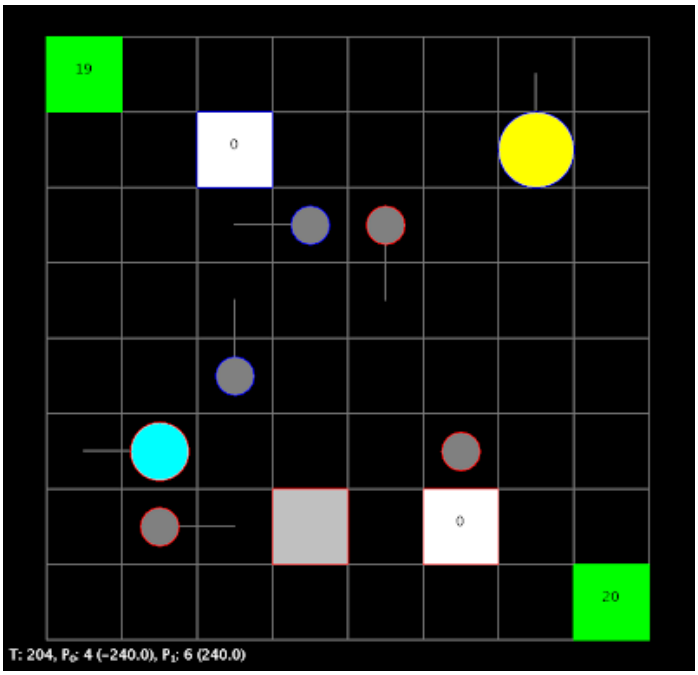

Figure 3.2.: Map GUI in microRTS

3.3.2 Agents, Objects and Actions

Based on the definitions in section 3.1 and 3.2, microRTS has a total of 6 agents. Each of those and their associated functions is summarized in Table 3.3. Figure 3.3 depicts a visual interpretation of Table 3.3. In cases of non-human player, like this study, all agents are provided with a rule-based logic that they follow according to a pre-defined bot strategy. Objects are, as previously defined, entities within ABMs that do not have an active role of the system but make up the functions of the environment. microRTS has only two objects built into the model; Resources (aka

Minerals) & Terrain. Terrain is not involved in the study purely to eliminate the volume of iterative experiments it would add to our Design of Experiments. For example, in an 8x8 grid, the number of useful terrain permutations are little less than 64! (factorial). Actions are a set of available commands that a player can assign to their agents on the map. microRTS has 5 actions that a player may utilize. These are also listed in Table 3.3.

Table 3.3.: Agents, Objects and Actions in microRTS

Agents	**Function**
Base	Stores resources and trains workers
Barracks	Trains attack units (*H,L,R*)
Workers	Harvests minerals, construct buildings and basic defence
Heavy(*H*)	High attack ability, low speed
Light(*L*)	Low attack ability, high speed
Ranged(*R*)	Long range attack ability

Objects	**Purpose**
Minerals	Harvested by workers for resources to train and build units
Terrain	Restrict unit movements (not used)

Actions	**Impact**
Move	Navigate units on the map
Attack	Damage enemy units or buildings
Harvest	Gather resources using Workers
Train	Create attack units or workers
Construct	Build bases or barracks

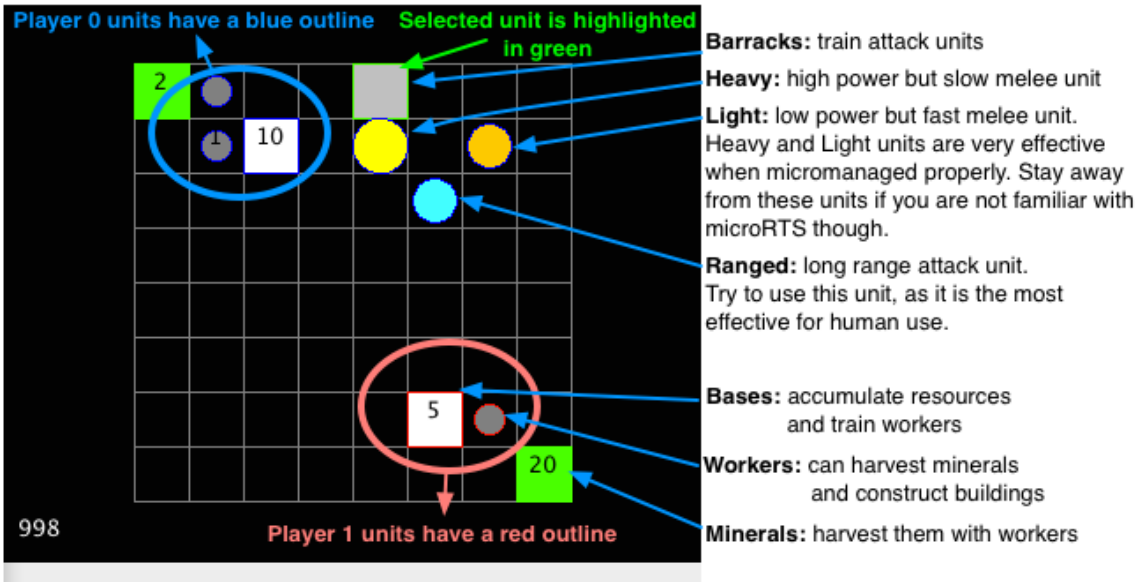

Figure 3.3.: Agents in microRTS [34]

3.3.3 Bots and Strategies

Bots are hard-coded players that perform a given strategy throughout the game. microRTS provides a variety of bots with the open-source package. The options range from simple rule-based abstraction AI to more complex bots using Monte-Carlo or Tree search algorithms to play. In our study we will make use of only the abstraction Bots to limit the computational budget. Games using bots of sophisticated algorithms take somewhere between 2000 to 6000 seconds per game and on the other hand games with abstraction bots take a maximum of 30 seconds. Strategy in general is any plan of action or tactic a player uses to satisfy their objectives. Table 3.4 lists the bots being used in our design of experiments and also their strategies.

Table 3.4.: Bots and Strategies in microRTS

Bots	**Strategy**
LightRush	Rushes opponent with Light attack units
HeavyRush	Rushes opponent with Heavy attack units
RangedRush	Rushes opponent with Ranged attack units
WorkerRush	Rushes opponent with Worker units
LightDefence	Defends using Light attack units
HeavyDefence	Defends using Heavy attack units
RangedDefence	Defends using Ranged attack units
WorkerDefence	Defends using Worker units

3.3.4 Game Data

The microRTS setup allows researchers to store and retrieve game data in two forms. All game results and tournament results are stored in a data file in *.xml* format. This file stores wins, losses, ties, and the average game time for all the three game results. This data the best way to assess each bot strategy either against a similar or different bot strategies. The second form of information about the games are stored in what is called a 'trace' file. This file is a data representation of every timestamp in each game. It stores every action, agent, object and parameters of the game. This can be used to dig deeper into specific games to look for specific instances. This becomes really important when differentiating a balanced game from an imbalanced one. It makes it easier to point to the specific factor contributing to the imbalance and the time at which it occurs. Trace files can also be used to re-visualize a saved game. Figure 3.4 shows a single timestamp record in an example trace file. Somethings that can be seen in the example are resources, player bases and player unit locations, their actions and internal properties.

RandomAIVsRandomBiasedAI-0-0 - Notepad

File Edit Format View Help

```
<rts.TraceEntry time = "0">
  <rts.PhysicalGameState width="8" height="8">
    <terrain>0000000000000000000000000000000000000000000000000000000000000000</terrain>
    <players>
      <rts.Player ID="0" resources="5">
      </rts.Player>
      <rts.Player ID="1" resources="5">
      </rts.Player>
    </players>
    <units>
      <rts.units.Unit type="Resource" ID="4" player="-1" x="0" y="0" resources="20" hitpoints="1" >
      </rts.units.Unit>
      <rts.units.Unit type="Resource" ID="5" player="-1" x="7" y="7" resources="20" hitpoints="1" >
      </rts.units.Unit>
      <rts.units.Unit type="Base" ID="6" player="0" x="2" y="1" resources="0" hitpoints="10" >
      </rts.units.Unit>
      <rts.units.Unit type="Base" ID="7" player="1" x="5" y="6" resources="0" hitpoints="10" >
      </rts.units.Unit>
      <rts.units.Unit type="Worker" ID="8" player="0" x="1" y="1" resources="0" hitpoints="1" >
      </rts.units.Unit>
      <rts.units.Unit type="Worker" ID="9" player="1" x="6" y="6" resources="0" hitpoints="1" >
      </rts.units.Unit>
    </units>
  </rts.PhysicalGameState>
  <actions>
  </actions>
</rts.TraceEntry>
```

Figure 3.4.: Example Trace File

3.3.5 Mechanisms, Objectives and Perspectives

From our experience of game mechanisms defined earlier for the *rock paper and scissors*, and the *prisoners' dilemma* game, we can instantly tell that the mechanisms involved in this game are much larger in number. While the list below contains all mechanisms categories possible in the game, our focus will only be in exploring a few.

- Influence of Strategy
- Influence of Resource Quantity
- Influence of Resources Location
- Influence of Starting Position of Player Base
- Influence of Starting Resources
- Influence of Starting Position of Player Unit (not tested)

- Influence of Unit properties (Light, Range, Heavy, Worker) (not tested)
- Influence of Terrain (not tested)
- Influence of costs (units, buildings) (not tested)
- Influence of Map size

Each of these categories have multiple mechanisms underneath them. Lets consider each of these one by one and discuss in detail the underlying mechanisms.

Resource quantity on a map is predetermined before the game is setup. Most precoded maps use either 25 resources or 50 resources allocated to each player. controlling the number and where they are placed gives us two dimensions of mechanism design at the designer level. Qualitatively there are two ways the influence of starting position of player base can be altered; closer to the resources or away from them. Of course in a quantitative sense, the base can be moved to 64 different positions on a 16x16 map. Each player is given a worker unit and resources (either none or 5) to begin with. The amount of resources that are given can be varied as desired by the game designer. Similarly, in terms of starting position of the worker unit each player is given can also be changed either to be closer to the base, or closer to the resource or equidistant or far from both. Every unit in the game has certain properties that define the impact of their action. For example, all attack units have damage rates that determine the damage they deal to opponent units or buildings in each timestep. There's also unit health, which represents how much damage they can take before being destroyed. Worker units also have harvest rate which influences the amount of resources they can mine in one timestep. The final of the unit properties are the movement speed which defines their rate of movement per timestep across grids. Since this gives rise to too many permutations of mechanism changes, our study does not involve this dimension. The same reason applies to the others explained from this point on. Terrain as already discussed in a previous subsection has the highest changeable permutations for a simple 8x8 map, and gets more complicated as the grid

size increases. Every unit and building is associated with a resource cost to create or build. Altering this is also another dimension the designer can introduce influence on the game result.

From the context of mechanism design, we need to identify the social objective and player objective of the game. Similar to the case in prisoners' dilemma, the social objective and the player objective in this game are interdependent. The players objective is that each player wants to win the game, this can either be done using a superior strategy or by taking advantage of an imbalance in their favour. As the designer, the social objective is to find the imbalances and make sure that the players are on a level playing field. For this we make use of mechanism design as our Design of Experiments tool to test all the strategies and mechanisms.

The influence of strategy is utilized a little differently than the others. It becomes really difficult to interpret such large data and chain-of-events games using game theory to identify the the equilibrium. Changing the perspective of the game helps in reducing the complexity to find the Nash and Dominant Strategies. instead of looking at each bot as a player which would make each timestep in a game a part of the analysis, we consider the bots as a strategy option for a player. So the analysis is done from one level above, at the level of games than time within games. To provide a working example, think of the payoff matrix of the experiment to be between two bots and the actions and instance in the game become the rows and columns of the matrix. This complicates the process and gives rise to large matrices. In our perspective, the view is one level higher, the matrix is built for two players whose options are the bot strategies. This way we do not have to look at what happens within games, but only the final outcomes. This reduces the size of the matrix to the number of bot options available, in our case 10. A detailed description of this process and its nuances are discussed in the DOE section 4.2

4. APPROACH

In chapter 2, we looked at restrictive game balancing technique, where one parameter/factor is the only variable and the rest are made unchangeable to determine the effect of that specific parameter/factor alone. Our design of experiments starts with a similar ideology, where we use mechanism design to account for all such mechanisms and then test them individually. This method of testing is also closely associated to the popular sensitivity analysis. We then further extend this to test with multiple mechanisms as variables at the same time. This is done in order to establish any interdependencies among mechanisms that are not easy to perceive when playing the game at first. Section 4.1 details the different experiments planned using this ideology. Section 4.2 describes the simulation setup. Figure 4.1, shows a schematic of the methodology.

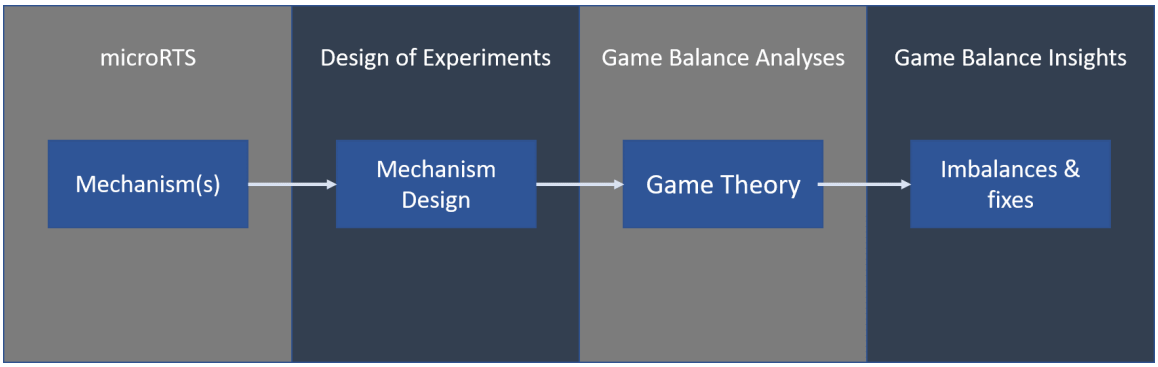

Figure 4.1.: Native microRTS - Map

4.1 Design of Experiments

Each of our experiments in the DOE will involve two layers of testing. The first is what we call the 'self-play games' and the other is 'cross-play games'. Self play games

are used to test the influence of game mechanisms other than strategy. In these games the designer's SOF states that all games must have both players using the same strategy. In cross-play games, the players are allowed to use different strategies. Here we study the influence of strategy. Apart from strategy, we have four other influencing mechanisms that need to be tested. The mechanism design inspired DOE will test each of these individually (similar to restrictive game play methodology [9]) and then have tests where they are combined in pairs. All results are then analysed with the help of solution concepts from game theory. Before the different DOE games are run, a baseline test is conducted to establish control data. In this setting, the game is in its native form and a batch of games are run where both self-play data is recorded. This native form is then altered to produce the variants required by the DOE. With our definition of game mechanisms in microRTS, Table 4.1 and Figure 4.2 describe the native form of the game in terms of mechanisms that are of interest. All games in the DOE will continue to run on the 16 x 16 map and same starting units (since those are not the dimensions of mechanisms being explored).

Table 4.1.: Native form of microRTS (baseline model)

Mechanism	**Setting**
Map	16x16
Resource Quantity	Each player gets 50
Resource Position	Stacked and cornered
Starting Resources	None
Base Positions	Symmetric
Starting Units	One *Worker* unit

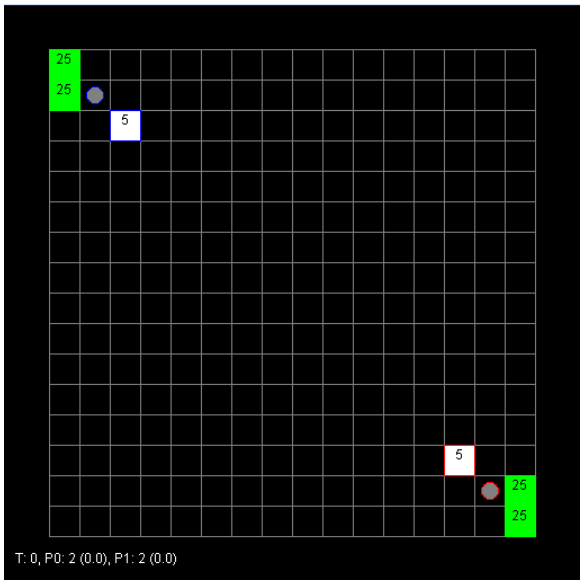

Figure 4.2.: Native microRTS - Map

4.1.1 Self-play Experiments

Using the social objective definition in mechanism design, the designer has the capability to make global rules that all players must follow. For self-play games, as previous mentioned, the players are only allowed to use similar strategies as their opponent, is made as the social objective construct. This then gives rise to four different game batches for each of the four mechanisms planned to be tested. From this point of, these batches will be addressed as tournaments. Each tournament will have eight games, each game representing the eight strategies that the players will use in common (Table 4.4). However for each tournament, there will be three different generations. All generations have the same player strategies and mechanisms of interest but, there is a change in one of the mechanisms of interest. Lets look at an example tournament to understand the structure. If Resource Quantity (RQ) is our first mechanism to be assessed, then that is also points to the tournament type - RQ. Both players will play eight games where they use the eight available strategies. The difference in the three generations is, the resource quantity for player one is changed

from the native form. A total of 3 generations, 24 games for one tournament type, and a total of 96 games for all the different mechanisms tested individually. Table 4.2 shows the standalone self-play experiments as a result of the DOE. Please note that all changes to mechanisms in the standalone self-play tournaments are forced onto player 1 only. Player 2 will resemble the native form.

The next set of experiments are where the designer is making asymmetric changes to the mechanisms i.e., each player has their own unique mechanism of interest. So far we tested each of those individually, now to gauge the interdependencies, the designer makes asymmetric variations of the mechanisms. Here each tournament will have only one generation each but there will be total of 3 tournaments with player one differing the resource quantity but player 2 differing in the other 3 mechanisms. This effort is to show an example correlation between two mechanisms and identify the one with a higher impact. Table 4.3 shows the asymmetric self-play experiments as a result of the DOE.

Table 4.2.: Standalone Self-play Tournaments

Tournament Type	**Generations (3)**
Resource Quantity (RQ)	30, 40 & 60
Resource Position (RP)	Displaced by 1, 2 & 3 cells
Base Position (BP)	Displaced closer, farther & equidistant to resources
Starting Resources (SR)	2, 4 & 5

Table 4.3.: Asymmetric Self-play Tournaments

Tournament Type	**Generation (2)**
(RQ, RP)	(60, closer) & (40, closer)
(RQ, BP)	(60, closer) & (40, closer)
(RQ, SR)	(60, 2) & (60, 5)

4.1.2 Cross-play Experiments

The structure of cross-play experiments is much simpler. The only varying mechanism here are the strategies which the new social objective construct allows players to do. Table 4.4 shows the cross-play experiments as a result of the DOE. This way players play a round-robin tournament with each strategy as a single entry in the tournament. This eventually points to advantages a player can have on their opponents purely based on the strategies in play.

Table 4.4.: Cross-play Tournaments

Tournament Type	**Generation (1)**
Strategy	*LightRush, HeavyRush, RangedRush, WorkerRush, LightDefence, HeavyDefence, RangedDefence, WorkerDefence*

4.2 Simulation Setup

microRTS is a game built on the Java programming platform hence the execution of the tests does not need a GUI interface and can be run using command line. For most of our tournaments and generations this is made use of. Only for games that

produce peculiar results that are anomalous to the data set, are re-run using the inbuilt GUI provided by the developer of the game to visually understand the game progression. Figure shows the GUI to run games. Figure 4.3 is the GUI window of microRTS. To run the tournaments, a new script was written to conduct games in iteration with different settings each time. The script also has the ability to turn off visualization, if needed, to save computation budget. It can be turned on to enable grabbing visual data of games for video demos. The script is built in a way that other mechanisms and bot strategies can be added to increase the extensibility of the tests. This also includes other map sizes and designs. The trace and game data extracted from the each game is stored in their respective tournament folder in a compressed format. The script also allows for the results to be printed onto the command line interface instead of a .xml file.

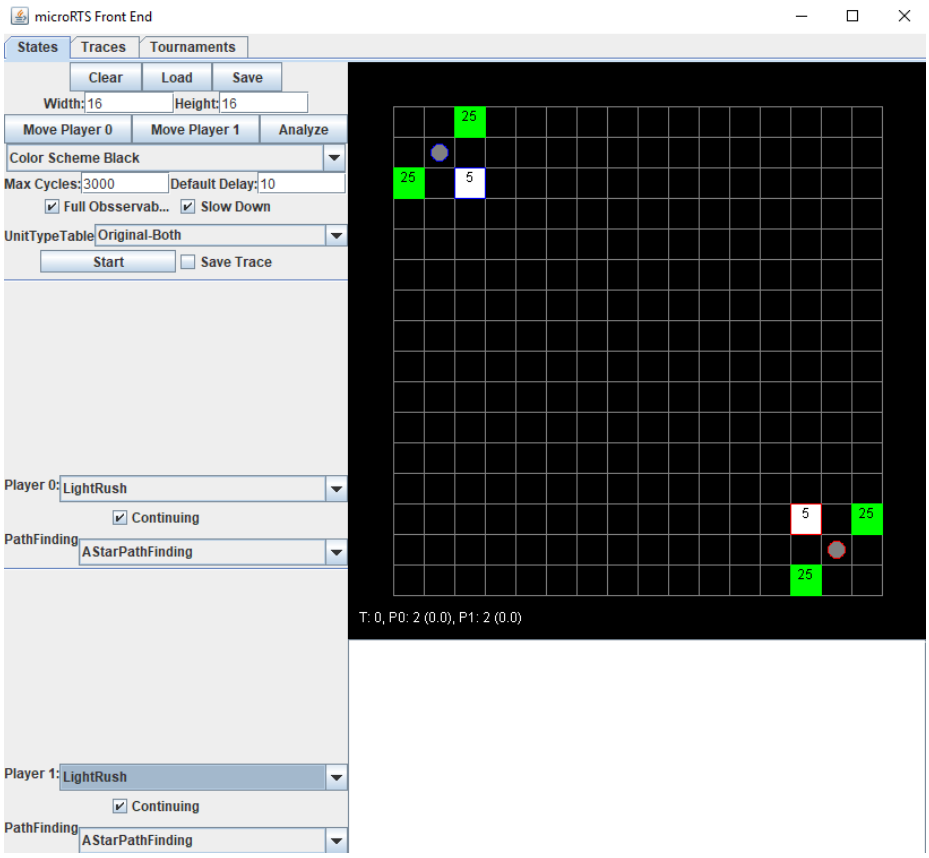

Figure 4.3.: microRTS GUI

Each game is set to run for 5000 time steps which is equal to 5000 action frames in the game. Player 1 starts at the top-left corner of the map and Player 2 in the bottom-right. All games are run twice for a consistency check to validate that the result is deterministic. microRTS has the option to run the game non-deterministically, but we are not using that to allow players to compete in complete information games. The game is decided as a tie/draw when either players have no winnable outcome possible (no units or no resources left) or when the units have a defensive standoff for more than 500 time steps.

5. RESULTS

5.1 Self-play games

5.1.1 Baseline - Results from Native Form

A total of eight matches were run with, 2 iterations for consistency, in the native form (symmetric beginning conditions) of microRTS to register control data to compare the experiments against. The players are forced to use the same strategy by the SOF rule laid out by the designer. Table 5.1 depicts the results from the games. All quantitative data to support the results are in the Appendix.

Table 5.1.: Preliminary Baseline Game Results

Strategy	**Result**
LightRush	Player 2 Wins
HeavyRush	Player 2 Wins
RangedRush	Player 2 Wins
WorkerRush	Player 2 Wins
LightDefence	Player 2 Wins
HeavyDefence	Player 2 Wins
RangedDefence	Tie
WorkerDefence	Tie

Even though the game setup was very symmetrical and the SOF making sure that the players are on equal footing, we see that there is a bias in the native form of the game to favour Player 2, bottom-right corner. This is a starting advantage

that Player 2 has that causes the imbalance. However, a case may be made for the two matches that ended in a tie. Upon closer inspection of the games; The *WorkerDefence* game is a tie because the player units genuinely exhaust the available resources and can no longer create units to continue the game at $3350/5000th$ timestep and the *RangedRush* game ends in a draw at $3590/5000th$ as the units are all killed on both sides and have no more. The starting imbalance in the other cases may be from various factors, collective or singular. Some of the possible reasons are: Path planning algorithm based on the location (even though the positioning is symmetric), lapse in the logic of the programmed strategies, and/or . In the next section, where we present results for the standalone and asymmetric self-play games, we can explore what mechanism in the game could negate this starting imbalance while making sure that that change doesn't cause an imbalance in the mid-game or end-game stages of the game play. Based on the results of these games, Nash and Dominant Strategy Equilibrium solution concepts suggest that Player 1 has only two dominant strategies to force Player 1 into an equilibrium (multiple equilibrium) if they have the ability to do so but players cannot force the game rules. Player 2 has six dominant strategies. microRTS in its native form has no Nash or Dominant Strategy Equilibrium for the given eight player strategies. For it to be completely balanced game, it should generate 8 Dominant Strategy Equilibrium where all strategies are the dominant strategies for both players. Nash equilibrium will not be a solution concept with self-play games as there are no cross-play games that would bring in the concept of best responses, which is key in identifying Nash Equilibrium.

5.1.2 Standalone Experiments

Now that we've established that the native form has some starting imbalances for most player strategies, testing the game mechanisms with our DOE (which is similar to restrictive game play [9]) may help us find the cause of the imbalance.

Resource Quantity

The first of our DOE tournament types is the variation in Resource Quantity (RQ) available to the players. Table 5.2 shows the results of the three generations of testing conducted.

Table 5.2.: Tournament results for RQ generations

Strategy	**RQ-30**	**RQ-40**	**RQ-60**
LightRush	Player 2 Wins	Player 2 Wins	Player 2 Wins
HeavyRush	Player 2 Wins	Player 2 Wins	Player 2 Wins
RangedRush	Player 2 Wins	Player 2 Wins	Player 2 Wins
WorkerRush	Player 2 Wins	Player 2 Wins	Player 2 Wins
LightDefence	Player 1 Wins	Tie	Player 2 Wins
HeavyDefence	Player 2 Wins	Player 2 Wins	Player 2 Wins
RangedDefence	Tie	Tie	Tie
WorkerDefence	Tie	Tie	Tie

The initial reaction is that testing the Resource Quantity mechanism has no effect on majority of the strategies. However when one pays attention the average game time that these games take to complete (please refer Appendix Figure 2, 3 & 4), it shows that Player 2 takes longer to win games in RQ-30 and RQ-40 than the native form (RQ-50) and RQ-60. This shows that RQ does have a significant impact on the balance. In the case of Light defence, it seems that RQ is a very major contributing factor to the balance equation. In RQ-30, using *LightDefence*, Player 1 easily wins and, in RQ-40 the game ties by reaching a balanced state for this setting. For *RangedDefence* and *WorkerDefence*, RQ had no impact to their previously balanced state in the native form. In terms of game theory solution concepts, there is still no Nash or Dominant Strategy Equilibrium but the number of dominant strategies

for both players has shifted. For Player 1, there are now three responses for RQ-3 and RQ-40 generations. Which is an increase compared to the native form. Similarly the number of dominant strategies for Player 2 which was six previously has reduced to five. RQ-60 however is a generation that adds more resource to Player 2 who already had an imbalance favoring them, hence in the effort to balance the game this generation is not useful.

Resource Position

The second of our DOE tournament types is the variation in Resource Position (RP) available to the players. Table 5.3 shows the results of the three generations of testing conducted. Note: the RQ is reset to native form. Both players play with 50 resources available.

Table 5.3.: Tournament results for RP generations

Strategy	**RP-1**	**RP-2**	**RP-3**
LightRush	Player 2 Wins	Player 2 Wins	Player 1 Wins
HeavyRush	Player 2 Wins	Player 2 Wins	Player 1 Wins
RangedRush	Player 2 Wins	Player 2 Wins	Player 1 Wins
WorkerRush	Player 2 Wins	Tie	Player 1 Wins
LightDefence	Player 1 Wins	Player 1 Wins	Player 1 Wins
HeavyDefence	Player 2 Wins	Player 2 Wins	Tie
RangedDefence	Tie	Tie	Player 1 Wins
WorkerDefence	Tie	Tie	Player 1 Wins

Very similar to the effect that resource quantity had on the games, resource position also challenged the biased Player 2 when the resource position was 2 or more cells away. However, the impact that this has is far more significant than the resource

quantity. This mechanism variation generated a higher number of ties and Player 1 favourable results than resource quantity tournaments. Player 1 has eight dominant strategies in RP-3 and player has none. This has completed shifted the balance in favor of Player 1. Although this is not desired, the shift in balance shows the extent of the impact resource position as a mechanism has control over game balance. It seems to be a very sensitive but the changes made in generations are also very small. It might be in the best interest of the designer to not involve resource position as a major game balancing mechanism. Like the previous tournament, no Dominant Strategy Equilibrium exists here too.

Base Position

The third of our DOE tournament types is the variation in Base Position (BP) available to the players. Table 5.4 shows the results of the three generations of testing conducted. Note: the RQ and RP both are reset to native form.

Table 5.4.: Tournament results for BP generations

Strategy	**BP-Closer**	**BP-Farther**	**BP-Equidistant**
LightRush	Player 2 Wins	Player 1 Wins	Player 2 Wins
HeavyRush	Player 2 Wins	Player 1 Wins	Player 2 Wins
RangedRush	Player 2 Wins	Player 1 Wins	Player 2 Wins
WorkerRush	Player 2 Wins	Player 1 Wins	Player 2 Wins
LightDefence	Player 2 Wins	Player 1 Wins	Tie
HeavyDefence	Player 2 Wins	Player 1 Wins	Tie
RangedDefence	Player 2 Wins	Player 1 Wins	Tie
WorkerDefence	Player 2 Wins	Tie	Tie

Compared to the previous two mechanisms, the final outcome has the highest sensitivity to base position. When closer to resources Player 2 has the advantage and when placed two cells away imitating a farther position, the results completely switch over. This could be because of two reason or a combination of both. When placed farther, it takes time to accumulate resource and the base is also more exposed to opponents units making it vulnerable to *Rush* strategies. While the fact that base position is a volatile mechanism, it did bring us very close to finding a Dominant Strategy Equilibrium. In generation 3, nearly half the games are tied and both players have equal number of dominant strategies. The only issue is that they is no intersection and are disjoint sets.

Starting Resources

The third of our DOE tournament types is the variation in Starting Resources (SR) available to the players. Table 5.4 shows the results of the three generations of testing conducted. Note: the RQ, RP, and BP are all reset to native form.

Table 5.5.: Tournament results for SR generations

Strategy	**SR-2**	**SR-4**	**SR-5**
LightRush	Player 2 Wins	Player 2 Wins	Player 2 Wins
HeavyRush	Player 2 Wins	Player 2 Wins	Player 2 Wins
RangedRush	Player 2 Wins	Player 2 Wins	Player 2 Wins
WorkerRush	Player 2 Wins	Player 2 Wins	Player 2 Wins
LightDefence	Player 2 Wins	Player 2 Wins	Player 2 Wins
HeavyDefence	Player 2 Wins	Player 2 Wins	Player 2 Wins
RangedDefence	Tie	Tie	Tie
WorkerDefence	Tie	Tie	Tie

Of all the test conducted so far, the starting resources mechanism tests are the most destabilizing. One, because prior to the DOE, the biased favor for Player 2 in the native form was not known. So testing the model with increased starting resource for Player 2 put them in a much more comfortable position. Two, the resulting outcomes were not helpful in mapping the strategies dependency on the mechanism as the results are very one-sided and provides no perspective for the other player's state. We continue to see that no Dominant Strategy Equilibrium exists.

Next, with the current understanding of the effects that each mechanism has, we test their combined interdependencies on game balance using one example construct - Resource Quantity.

5.1.3 Asymmetric Experiments

The DOE setup for these is described in Section 4.1.1 and depicted in Table 4.3,

RQ-RP tournaments

In this set of experiments, the resource quantity available for Player 2 is varied to 60 and then to 40, and for Player 1 the Resource Position is moved closer compared to the native form. Table 5.6 shows the results of the games.

We have a precursor knowledge that RP has more control over game balance compared to RQ if displaced by more than 2 cells. In this experiment, it is just displaced by 1 cell. Since Player 1 is given a closer resource position than previously, it should have a higher probability to either win or draw (draw - because the game originally has Player 2 winning). However, the game results don't seem to have any conclusive evidence of the opposing mechanism designs that should lead to a negation of the imbalances. No Dominant Strategy Equilibrium exists here too. However, looking at Figure 5.1, which shows the difference in the length of games in terms of time steps, tells that the mechanisms are opposing each others effect on the game balance. What used to be an easier game for Player 2 is no longer easy with reduced

resources (only games that Player 2 won are on the plot to reduce noise of other game results).

Table 5.6.: Tournament results for RQ-RP generations

Strategy	**(60,Closer)**	**(40,Closer)**
LightRush	Player 2 Wins	Player 2 Wins
HeavyRush	Player 2 Wins	Player 2 Wins
RangedRush	Player 2 Wins	Player 2 Wins
WorkerRush	Player 2 Wins	Player 2 Wins
LightDefence	Player 2 Wins	Tie
HeavyDefence	Player 2 Wins	Player 2 Wins
RangedDefence	Tie	Tie
WorkerDefence	Tie	Tie

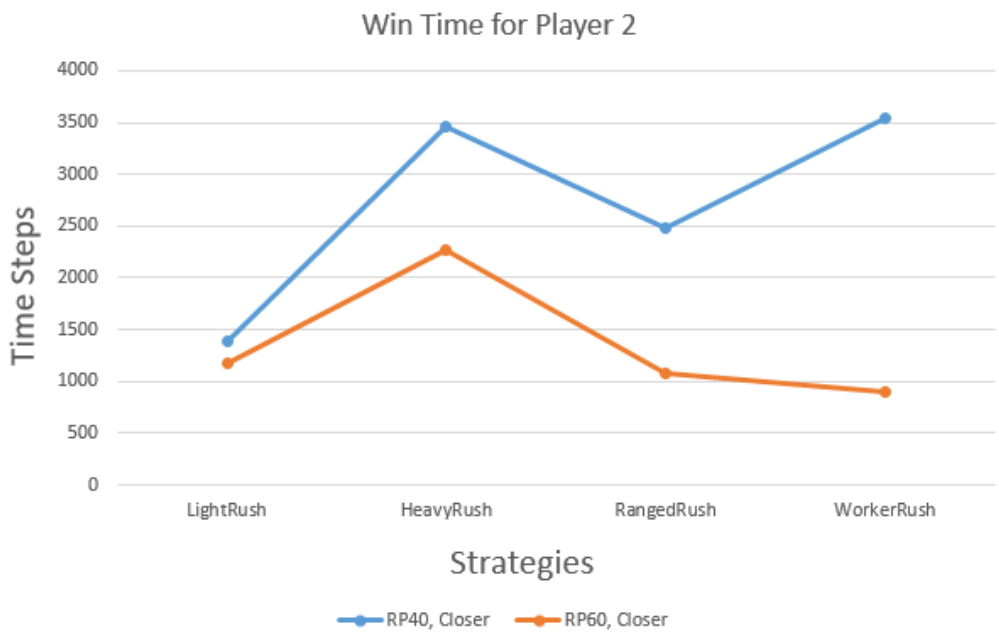

Figure 5.1.: Time taken by Player 2 to win

RQ-BP tournaments

In this set of experiments, the resource quantity available for Player 2 is varied to 60 and then to 40, and for Player 1 the Base Position is moved closer compared to the native form. Table 5.7 shows the results of the games. Not so surprisingly, the impact that Base Position has on a player's win rate is easily observed. As previously discussed, this mechanism is too sensitive and probably is the most influential when changed. However, we continue to observe that for defence oriented strategies that these mechanism changes don't affect their state of balance. This either points to some modelling issues or, more likely, these strategies are independent of these mechanisms and rely on the ones not tested. Even though in both cases Player 1 wins most games, a closer look at the game run times suggest that with the second case the time taken is drastically less (see appendix) as expected with player 2 also having the disadvantage of having lower resources. No Dominant Strategy Equilibrium observed in this case either.

Table 5.7.: Tournament results for RQ-BP generations

Strategy	**(60,Closer)**	**(40,Closer)**
LightRush	Player 1 Wins	Player 1 Wins
HeavyRush	Player 1 Wins	Player 1 Wins
RangedRush	Player 1 Wins	Player 1 Wins
WorkerRush	Player 1 Wins	Player 1 Wins
LightDefence	Player 1 Wins	Player 1 Wins
HeavyDefence	Tie	Tie
RangedDefence	Tie	Tie
WorkerDefence	Tie	Tie

RQ-SR tournaments

In this set of experiments, the resource quantity available for Player 2 is varied to 60 and then to 40, and for Player 1 the Starting Resources is increased first to 2 and then to 4, compared to the native form. Table 5.8 shows the results of the games. In our previous test for Starting Resources, which turned irrelevant because of the direction of the change, the impact could not be assessed. However, with this asymmetric test, ,we see that starting resources do have a major impact to the outcome. In all the cases, having additional resources at the start of the game gave Player 1 the edge to win the games. The defence strategies consistently prove that their modelling has been very well balanced. However, no Dominant Strategy Equilibrium is seen.

Table 5.8.: Tournament results for RQ-SR generations

Strategy	**(60, 2)**	**(40, 5)**
LightRush	Player 1 Wins	Player 1 Wins
HeavyRush	Player 1 Wins	Player 1 Wins
RangedRush	Player 1 Wins	Player 1 Wins
WorkerRush	Player 1 Wins	Player 1 Wins
LightDefence	Player 1 Wins	Player 1 Wins
HeavyDefence	Tie	Tie
RangedDefence	Tie	Tie
WorkerDefence	Tie	Tie

5.2 Cross-play games

In this mode, the SOF allows players to use any strategy against their opponent. Since we've already looked at self game results, we neglect the games where self-play occurs to purely understand the impact of strategy on game balance. Figure 5.2 is a bar chart representing the results for each of the strategies against others. From this we can clearly say that *WorkerDefense* has the highest balance based on the number of games that it has participated in, that has ended up in a tie. This is closely followed by *RangedDefence* for being the second most strategy used in games ending in a tie. This is also very accurate with the nature of results we observed in the self-play games which suggested that *RangedDefence* and *WorkerDefence* are the most balanced strategies in the microRTS game play. *HeavyRush*, *WorkerRush*, *LightRush* and *HeavyDefence* are the least balanced, since they are either too powerful for an opponent or too weak. Since we observed that the game in its native form has players in the bottom right winning even when conditions are identical, to counter that effect, cross-plays were run where Player 1 and Player 2 were given equal number

of chances to start at both ends with all the strategies. Looking at the ties alone will give us more context about the balance in the game. Figure 5.3 shows the percentage of games each strategy has turned into ties.

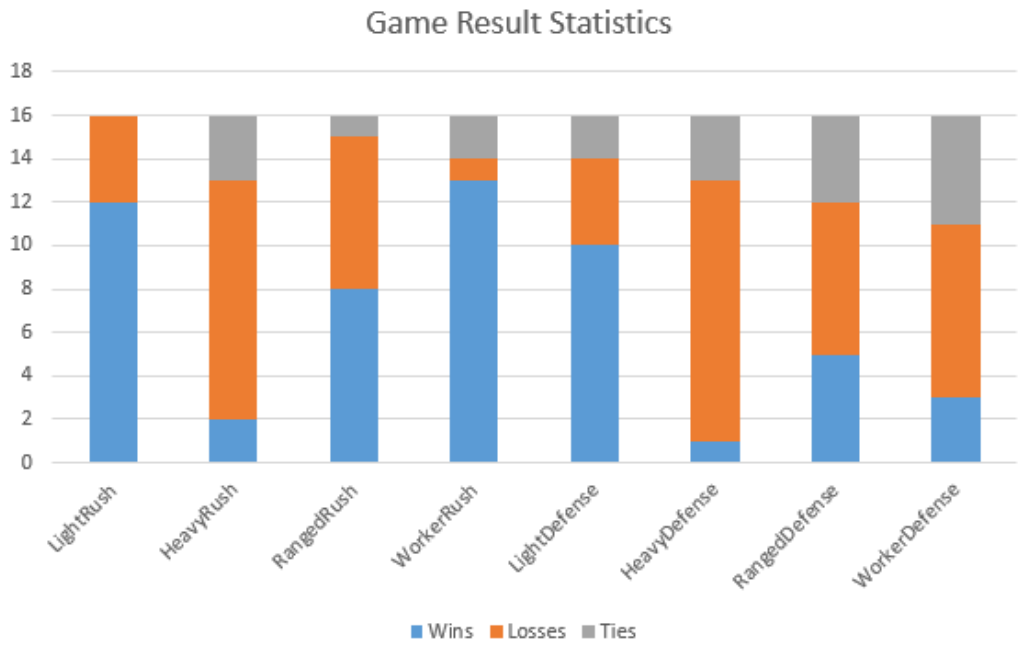

Figure 5.2.: Win(Payoff) Matrix for Cross-play tournament

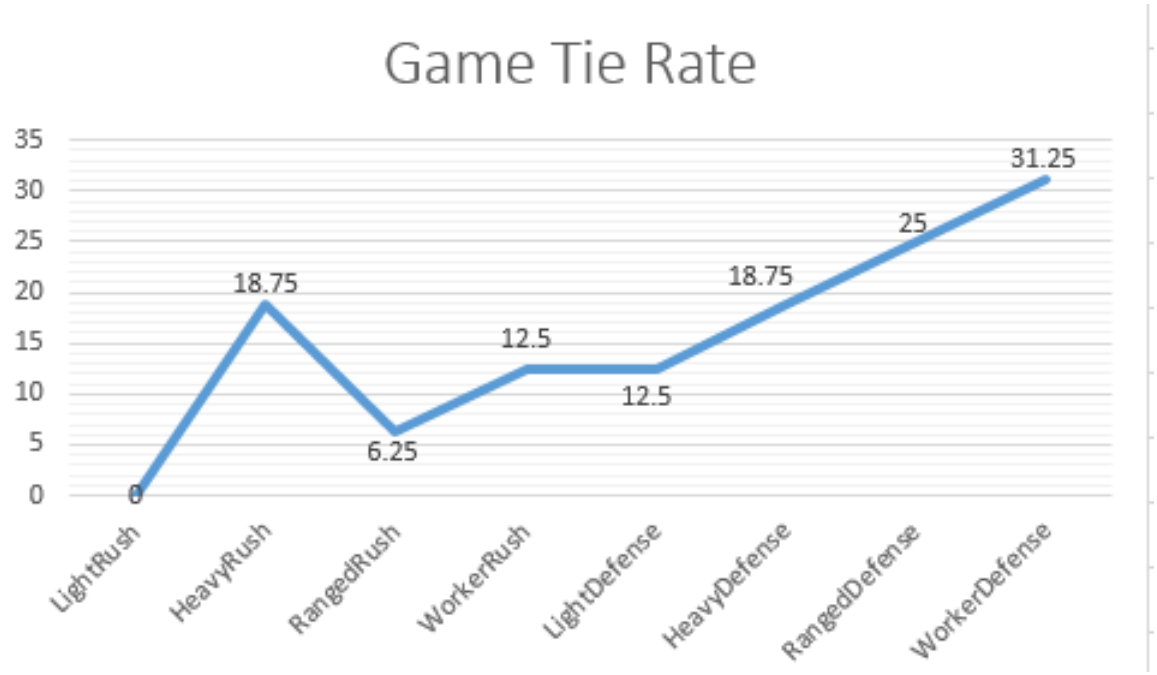

Figure 5.3.: Win(Payoff) Matrix for Cross-play tournament

Consistent with the previous results, defence based strategies produce the highest number of games that end in ties. If an exclusive study of just defence oriented strategies were conducted, then microRTS could easily be termed as a balanced game. Unfortunately, that is not the case here and probably the same even in the world of video games. Developers can't design games expecting all users to play with one type of strategy. Now that we understand where the imbalances and balances lie, it is easy to work on the model to improve it's balance.

From the lens of game theory, with cross-plays we finally are able to identify Dominant Strategy Equilibrium, and there are multiple, two to be precise. Both of these occur when the players use *WorkerRush* and *LightRush* as their options. Of course the instance is repeatable and hence it appears twice in Figure 5.4, but in terms of implementation, it is one pair of strategies.

		Player 2							
	Payoff Matrix	LightRush	HeavyRush	RangedRush	WorkerRush	LightDefense	HeavyDefense	RangedDefense	WorkerDefense
Player 1	LightRush		W	W	L	D	W	W	W
	HeavyRush	L		0	L	L	D	L	0
	RangedRush	L	W		L	L	W	D	W
	WorkerRush	W	W	W		W	W	W	L
	LightDefense	D	D	W	L		D	W	W
	HeavyDefense	L	L	L	L	L		L	L
	RangedDefense	L	D	L	L	L	W		W
	WorkerDefense	L	W	L	D	L	D	L	

Figure 5.4.: Dominant Strategy Equilibrium

5.3 Discussion

In self-play games, for attack based strategies (*Rush*) the change in mechanisms individually made no significant change to the final outcomes of the game. The defence strategies were the most sensitive to mechanism changes and reacted as expected to these changes. This hinted that each player strategy depends on different mechanisms uniquely to achieve a balanced (or favourable) state for themselves. This was further affirmed by the outcomes in the asymmetric self-play and cross-play tournaments. Attaining Dominant Strategy Equilibrium may be close to impossible when all these strategies are involved since their dependencies on the tested mechanisms vary invariably and independently. We may find a Dominant Strategy Equilibrium for a different mechanism (ones that are not tested here) or a different combination of asymmetric testing. From the perspective of *RangedDefence* and *WorkerDefence*, the game is very balanced to begin with. However, with cross play, the established internal balance that these two strategies maintained, fails. The trend that Dominant Strategy Equilibrium has followed in the test cases and the concept of weakly dominant strategies (here, players settling for ties over wins), points to a possibility where game balance can be declared if the game has a dominant strategy equilibrium for all possible game settings or for equilibrium results ending in draws. In other words, either all games played, irrespective of self/cross-play, should end in a tie or if there

exists diversity in the results then the dominant strategy equilibrium of all the games should be games that produce ties.

Our approach detected that microRTS is an imbalanced game but is very close to being a balanced game. We see almost 30-40% of the total number of games ending in ties, inclusive of both self-play and cross-play games. Even though this number is not as high as desired, the test cases and specific strategies analysed suggest that small changes in the game mechanisms will lead to a higher number of tied games. Our test included only a handful of mechanisms available for a designer to tweak, an all inclusive automated mechanism design for multi-agent model would throw more light on the exact balancing needed for the game. However, based on our testing methods and results, a general assessment points to what developers could do to remove the imbalances is discussed below.

Suggested balancing measures for microRTS:

1. Investigate the effects of unit parameters, because from our results it is obvious that most of the imbalances caused are not from the environmental elements of the game. If anything, these show promising results when checked for consistency in impact and extent of impact they have on balance.

2. To remove the starting imbalance, on top providing a symmetric/identical starting game setup, the model further needs to inspected to identify the root cause of this imbalance. One effective avenue to do so is to record and analyse the emergent behavior that the player in the bottom-right displays. This can be done with the help of analysing a large number game states with filtering algorithms like Convolutional Neural Networks.

3. Some issues are caused by the fact that the game runs on a grid system. While that is a limitation of the ABM implementation and not the game design itself, it could be changed to not result in imbalances due to minimalist modelling. One way to do this is use a free roam model (such as Boids Birds) and provide better movement classes to the dynamic agents in the game.

4. While Nash equilibria is a weaker concept compared to Dominant Strategy Equilibria, it can be used to figure out if the game has finite players (which microRTS does) and detect pure strategies. This helps in determining whether strategies used by players are consistent across all games. Since we use rule based strategies this holds true, but when microRTS is used in the context of real world players or complicated AI algorithm based agents, it would be essential to do this in order to fix the balancing issues.

5. Cross-play balancing is the trickier of the two. For example, currently all player strategies use common definitions for its units (agents) and have no personalised definition of their units. In more complex games or real world scenarios this is not true. Every player has a personalised definition of their agents and unique performance indices. This effort to balance games for non-symmetric setting cross-play game makes the process laborious. Most commonly, for strategies whose performance are too weak or too strong, the unit definitions are modified to provide a equal playing field. In the game development industry this is called as 'buffing' and/or 'nerfing' of players, respectively.

6. CONCLUSIONS AND FUTURE WORK

6.1 Summary

To test the feasibility of using Game Theory and Mechanism Design methods in understanding game balance we looked at a simple RTS game, microRTS. This research was mainly driven by the lack of game balancing techniques that provide a middle ground between manual labor intensive methods and AI-based methods that have cognitive inabilities. Our approach uses mechanism design as the design of experiments technique to generate tests where mechanisms, one or more, are changed and game outcomes are recorded. Experiments were designed in a way to also establish the interdependencies among the mechanisms that may impact the state of game balance. microRTS is run in a fully observable, deterministic and two player setting. This makes it a finite, zero sum game. The results from the games are then analysed using game theory lens. This is done to demonstrate any possible correlation between game theory solution concepts and balance of a game.

Although microRTS is a low fidelity ABM, it serves the purpose to test our hypothesis. The results obtained clearly indicate that game theory is a great analytical tool to assess game states. For a zero-sum game, the results obtained still provide sufficient context about the imbalances. The fact that most RTS games when analysed from a perspective of the game end up as zero-sum games. This handicaps the solution concepts of game theory like Nash Equilibrium which already is a limited metric. Hence our research worked on the problem from the perspectives of players instead of the events of the game. Whether that points to game theory's inaptness to game balancing indefinitely is a question that still needs to be further inspected. However, for simple games of the scale similar to microRTS, it is promising and ad-

equate to evaluate the state of the game and understand the individual impact and interdependencies of game mechanisms.

Our objective was also to check the feasibility of mechanism design and its role in designing and implementing institutions to evaluate game balance. Between the simplicity of microRTS and the holistic approach used here, it provides key cognizance on the possible involvement of mechanism design in future game balance testing frameworks. The design of experiments covered only five of all the available mechanisms in microRTS. Even though this is a small number there is sufficient evidence that this is effective for finite game play and non-zero sum games. However, beyond these boundaries the effectiveness is still untested.

microRTS is not a balanced game and from our investigation we know that the most obvious reason is the starting imbalance that gives one of the players an upper hand purely based on their positioning even when the game is setup symmetrically. To reduce the imbalances, microRTS needs to increase its model fidelity, specifically for the map and the interacting agents. Having said that, testing with sophisticated AI-algorithm based player agents used in the annual IEEE competitions [40], would definitely provide additional valuable and accurate understanding of game balance within microRTS.

6.2 Future Work

The first step in extending this work is to include the rest of the mechanisms in the game that were not a part of the tests. Ideally, the ability to automate this process would make it much easier, so that's another major avenue that needs to be explored. Automated Mechanism Design has been established but successfully only for single-agent systems. While the asymmetric self-play tests were just a demonstration for proof of concept and involved only two at a time, this further needs to be expanded to multiple mechanisms and a large number of permutations. With such a strong hold

over all the mechanisms and their relations (internal and external), more specifics on how balance is structured, its issues and nuances in the game can be identified.

Most of the results used in this study are based on the game outcomes (win, loss and ties), but game balance should also be perceived to be more than just the win-loss probability. This argument is supported by our insights on the game run times for the various tests. Involving this into the game theoretical evaluation would throw more light on the state of game balance, especially for self-play games. Particularly since this two player setting produces a zero sum game, using multiple approaches is beneficial to accurately predict the state of balance.

The strategies used by player agents here are all simple rule-based abstraction AI that may not completely interact with all elements of the game. So to include more complicated and sophisticated strategies using AI frameworks or ML algorithms will help in building stronger data sets with higher levels of confidence.

REFERENCES

REFERENCES

[1] G. Robertson and I. Watson, "A Review of Real-Time Strategy Game AI," *AI Magazine*, vol. 35, no. 4, pp. 75–104, Dec. 2014, number: 4. [Online]. Available: https://www.aaai.org/ojs/index.php/aimagazine/article/view/2478

[2] C. L. Magee and O. L. de Weck, "_USAn Attempt at Complex System Classification," Massachusetts Institute of Technology. Engineering Systems Division, Working Paper, May 2002, accepted: 2016-05-31T18:55:16Z. [Online]. Available: https://dspace.mit.edu/handle/1721.1/102730

[3] G. Aranda, C. Carrascosa, and V. Botti, "Characterizing Massively Multiplayer Online Games as Multi-Agent Systems," in *Hybrid Artificial Intelligence Systems*, ser. Lecture Notes in Computer Science, E. Corchado, A. Abraham, and W. Pedrycz, Eds. Berlin, Heidelberg: Springer, 2008, pp. 507–514.

[4] P. Guyot and S. Honiden, "Agent-based participatory simulations: Merging multi-agent systems and role-playing games," *Journal of Artificial Societies and Social Simulation*, vol. 9, no. 4, 2006.

[5] D. F. Adamatti, J. S. Sichman, P. Bommel, R. Ducrot, C. Rabak, and M. Camargo, "JogoMan: A prototype using multi-agent-based simulation and role-playing games in water management," 2005.

[6] O. Barreteau and G. Abrami, "Variable time scales, agent-based models, and role-playing games: The PIEPLUE river basin management game," *Simulation & Gaming*, vol. 38, no. 3, pp. 364–381, Sep. 2007, publisher: SAGE Publications Inc. [Online]. Available: https://doi.org/10.1177/1046878107300668

[7] O. Barreteau, F. Bousquet, and J.-M. Attonaty, "Role-playing games for opening the black box of multi-agent systems: method and lessons of its application to Senegal River Valley irrigated systems," *Journal of artificial societies and social simulation*, vol. 4, no. 2, p. 5, 2001.

[8] "The 12th Annual Independent Games Festival - 2002 Finalists & Winners," Mar. 2014. [Online]. Available: https://web.archive.org/web/20140327125722/http://www.igf.com/2002finalistswinners.html

[9] A. Jaffe, A. Miller, E. Andersen, Y.-E. Liu, A. Karlin, and Z. Popovic, "Evaluating Competitive Game Balance with Restricted Play," in *Eighth Artificial Intelligence and Interactive Digital Entertainment Conference*, Oct. 2012. [Online]. Available: https://www.aaai.org/ocs/index.php/AIIDE/AIIDE12/paper/view/5470

[10] J. Griesemer, "Design in Detail: Changing the Time Between Shots for the Sniper Rifle from 0.5 to 0.7 Seconds for Halo 3," 2010.

[11] M. J. Nelson and M. Mateas, "A requirements analysis for videogame design support tools," 2009, pp. 137–144.

[12] V. Volz, G. Rudolph, and B. Naujoks, "Demonstrating the Feasibility of Automatic Game Balancing," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, ser. GECCO '16. Denver, Colorado, USA: Association for Computing Machinery, Jul. 2016, pp. 269–276. [Online]. Available: https://doi.org/10.1145/2908812.2908913

[13] A. B. Jaffe, "_USUnderstanding Game Balance with Quantitative Methods," Thesis, Jul. 2013, accepted: 2013-07-23T18:28:24Z. [Online]. Available: https://digital.lib.washington.edu:443/researchworks/handle/1773/22797

[14] M. Beyer, A. Agureikin, A. Anokhin, C. Laenger, F. Nolte, J. Winterberg, M. Renka, M. Rieger, N. Pflanzl, M. Preuss, and V. Volz, "An integrated process for game balancing," in *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, Sep. 2016, pp. 1–8, iSSN: 2325-4289.

[15] R. Leigh, J. Schonfeld, and S. J. Louis, "Using coevolution to understand and validate game balance in continuous games," in *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, ser. GECCO '08. Atlanta, GA, USA: Association for Computing Machinery, Jul. 2008, pp. 1563–1570. [Online]. Available: https://doi.org/10.1145/1389095.1389394

[16] W. Xia and B. Anand, "Game balancing with ecosystem mechanism," in *2016 International Conference on Data Mining and Advanced Computing (SAPIENCE)*, Mar. 2016, pp. 317–324.

[17] B. Gerkey and M. Mataric, "Sold!: auction methods for multirobot coordination," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 758–768, Oct. 2002, conference Name: IEEE Transactions on Robotics and Automation.

[18] R. Holzman and N. Law-Yone, "Network structure and strong equilibrium in route selection games," *Mathematical Social Sciences*, vol. 46, no. 2, pp. 193–205, 2003, publisher: Elsevier.

[19] E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, T. Wexler, and T. Roughgarden, "The price of stability for network design with selfish agents," 2004, pp. 295–304.

[20] E. Anderson, F. Kelly, and R. Steinberg, "A contract and balancing mechanism for sharing capacity in a communication network," *Management Science*, vol. 52, no. 1, pp. 39–53, 2006, publisher: INFORMS.

[21] J. Hinz, "Optimal bid strategies for electricity auctions," *Mathematical Methods of Operations Research*, vol. 57, no. 1, pp. 157–171, 2003, publisher: Springer.

[22] G. Federico and D. Rahman, "Bidding in an electricity pay-as-bid auction," *Journal of Regulatory Economics*, vol. 24, no. 2, pp. 175–211, 2003, publisher: Springer.

[23] V. D. Dang, R. K. Dash, A. Rogers, and N. R. Jennings, "Overlapping coalition formation for efficient data fusion in multi-sensor networks," vol. 6, 2006, pp. 635–640.

[24] M. Chen, G. Yang, and X. Liu, "Gridmarket: A practical, efficient market balancing resource for Grid and P2P computing." Springer, 2003, pp. 612–619.

[25] M. Wooldridge, *An introduction to multiagent systems*. John Wiley & Sons, 2009.

[26] S. Russell, P. Norvig, J. Canny, J. Malik, and D. Edwards, "Informed search methods," *Artificial Intelligence: A Modern Approach; Pompili, M., Chavez, S., Eds*, pp. 92–118, 1995.

[27] J. Ferber and G. Weiss, *Multi-agent systems: an introduction to distributed artificial intelligence*. Addison-Wesley Reading, 1999, vol. 1.

[28] S. Franklin and A. Graesser, "Intelligent agents III," *Lecture Notes on Artificial Intelligence (Springer-Verlag, Berlin, 1997) pp*, pp. 21–35, 1997.

[29] F. Bousquet and C. Le Page, "Multi-agent simulations and ecosystem management: a review," *Ecological modelling*, vol. 176, no. 3-4, pp. 313–332, 2004, publisher: Elsevier.

[30] N. Vlassis, "A concise introduction to multiagent systems and distributed artificial intelligence," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 1, no. 1, pp. 1–71, 2007, publisher: Morgan & Claypool Publishers.

[31] B. Jiang and H. R. Gimblett, "An agent-based approach to environmental and urban systems within geographic information systems," *Integrating geographic information systems and agent-based techniques for simulating social and ecological processes. Oxford University Press, New York*, pp. 171–189, 2002.

[32] E. Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems," *Proceedings of the National Academy of Sciences*, vol. 99, no. suppl 3, pp. 7280–7287, May 2002, publisher: National Academy of Sciences Section: Colloquium Paper.

[33] R. Axtell, "Why agents?: on the varied motivations for agent computing in the social sciences," 2000.

[34] "microRTS - microRTS AI Competition," library Catalog: sites.google.com. [Online]. Available: https://sites.google.com/site/micrortsaicompetition/microrts

[35] S. O. Villar, "santiontanon/microrts," Jun. 2020, original-date: 2015-03-12T18:16:53Z. [Online]. Available: https://github.com/santiontanon/microrts

[36] S. Ontanón and M. Buro, "Adversarial hierarchical-task network planning for complex real-time games," 2015.

[37] S. Ontanon, "Experiments on Learning Unit-Action Models from Replay Data from RTS Games," 2016.

[38] N. A. Barriga, M. Stanescu, and M. Buro, "Game tree search based on nondeterministic action scripts in real-time strategy games," *IEEE Transactions on Games*, vol. 10, no. 1, pp. 69–77, 2017, publisher: IEEE.

[39] M. Stanescu, N. A. Barriga, A. Hess, and M. Buro, "Evaluating real-time strategy game states using convolutional neural networks." IEEE, 2016, pp. 1–7.

[40] S. Ontañón, N. A. Barriga, C. R. Silva, R. O. Moraes, and L. H. Lelis, "The first microrts artificial intelligence competition," *AI Magazine*, vol. 39, no. 1, pp. 75–83, 2018.

APPENDIX

```
----------------------
Tournament 1
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 0, 0, 0, 1, 0, 0,
0, 2, 1, 0, 0, 2, 1, 2,
2, 2, 2, 1, 2, 2, 2, 0,
1, 1, 2, 0, 1, 1, 2, 2,
0, 0, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 0, 2, 0, 2,
0, 2, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 1, 1, 0,
0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 2,
0, 1, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 1, 0, 0, 1,
0, 1, 1, 0, 0, 0, 2, 0,
0, 0, 0, 2, 0, 1, 0, 2,
Loses:
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 0, 1, 2,
2, 0, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 1, 2, 1,
2, 0, 1, 2, 2, 0, 0, 0,
2, 0, 2, 0, 2, 0, 2, 0,
Win average time:
1391.0, 1182.0, 989.5, -, 2580.0, 1182.0, 989.5, 1478.5,
-, 3386.0, -, -, -, 1625.0, -, -,
-, 1552.5, 2475.0, -, -, 1642.5, 2845.0, 1595.0,
390.0, 390.0, 390.0, 3205.0, 390.0, 390.0, 390.0, -,
2113.0, 1292.0, 1603.5, -, 2796.0, 2073.0, 1470.0, 1727.5,
-, -, -, -, -, 1693.0, -, -,
-, 2550.0, -, -, -, 2440.0, -, 1747.5,
-, 1625.0, -, -, -, 1135.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, 5000.0, 5000.0, 3574.0, -,
-, -, -, -, -, -, 3640.0, -,
-, -, -, -, -, -, -, 3380.0,
-, 5000.0, -, -, -, 3500.0, -, -,
-, 5000.0, -, -, 3500.0, -, -, 5000.0,
-, 3574.0, 3640.0, -, -, -, 3590.0, -,
-, -, -, 3380.0, -, 5000.0, -, 3350.0,
Lose average time:
1391.0, -, -, 390.0, 2113.0, -, -, -,
1182.0, 3386.0, 1552.5, 390.0, 1292.0, -, 2550.0, 1625.0,
989.5, -, 2475.0, 390.0, 1603.5, -, -, -,
-, -, -, 3205.0, -, -, -, -,
2580.0, -, -, 390.0, 2796.0, -, -, -,
1182.0, 1625.0, 1642.5, 390.0, 2073.0, 1693.0, 2440.0, 1135.0,
989.5, -, 2845.0, 390.0, 1470.0, -, -, -,
1478.5, -, 1595.0, -, 1727.5, -, 1747.5, -,
```

Figure 1.: Native form & Cross-play Results

```
----------------------
Tournament 2
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 0, 0, 0, 1, 1, 0,
0, 2, 1, 0, 0, 2, 1, 2,
2, 2, 2, 1, 2, 2, 2, 1,
1, 1, 2, 0, 1, 1, 2, 2,
0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 2, 0, 1,
0, 1, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 1, 1, 1,
0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 1,
0, 1, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 1, 0, 0, 1,
0, 1, 1, 0, 0, 0, 2, 1,
0, 1, 0, 1, 0, 1, 1, 2,
Loses:
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 0, 0, 1,
2, 0, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 1, 2, 1,
2, 1, 1, 2, 2, 0, 0, 0,
2, 0, 2, 1, 2, 0, 1, 0,
Win average time:
1391.0, 1182.0, 989.5, -, 2223.0, 1182.0, 989.5, 1478.5,
-, 3535.0, -, -, -, 1625.0, 2678.0, -,
-, 1550.0, 2245.0, -, -, 1785.0, 2245.0, 1580.0,
390.0, 390.0, 390.0, 1865.0, 390.0, 390.0, 390.0, 2120.0,
2149.0, 1292.0, 1603.5, -, 2276.0, 2109.0, 1470.0, 1780.5,
-, -, -, -, -, 1693.0, -, -,
-, -, -, -, -, 1977.5, -, 1490.0,
-, 1135.0, -, -, -, 1135.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, 5000.0, 5000.0, 3260.0, 5000.0,
-, -, -, -, -, -, 3260.0, -,
-, -, -, -, -, -, -, 3100.0,
-, 5000.0, -, -, -, 5000.0, -, -,
-, 5000.0, -, -, 5000.0, -, -, 3548.0,
-, 3260.0, 3260.0, -, -, -, 3260.0, 3280.0,
-, 5000.0, -, 3100.0, -, 3548.0, 3280.0, 3170.0,
Lose average time:
1391.0, -, -, 390.0, 2149.0, -, -, -,
1182.0, 3535.0, 1550.0, 390.0, 1292.0, -, -, 1135.0,
989.5, -, 2245.0, 390.0, 1603.5, -, -, -,
-, -, -, 1865.0, -, -, -, -,
2223.0, -, -, 390.0, 2276.0, -, -, -,
1182.0, 1625.0, 1785.0, 390.0, 2109.0, 1693.0, 1977.5, 1135.0,
989.5, 2678.0, 2245.0, 390.0, 1470.0, -, -, -,
1478.5, -, 1580.0, 2120.0, 1780.5, -, 1490.0, -,
```

Figure 2.: Resource Quantity - Gen1 Results

```
----------------------
Tournament 3
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 0, 0, 0, 1, 0, 0,
0, 2, 1, 0, 0, 2, 1, 2,
2, 2, 2, 1, 2, 2, 2, 1,
1, 1, 2, 0, 0, 1, 2, 2,
0, 0, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 0, 1, 0, 2,
0, 2, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 1, 1, 0,
0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 1,
0, 1, 0, 0, 2, 1, 0, 0,
0, 1, 0, 0, 1, 0, 1, 1,
0, 1, 1, 0, 0, 1, 2, 0,
0, 0, 0, 1, 0, 1, 0, 2,
Loses:
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 0, 1, 2,
2, 0, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 2, 0, 0, 0, 0,
2, 1, 2, 2, 1, 1, 1, 1,
2, 0, 1, 2, 2, 0, 0, 0,
2, 0, 2, 1, 2, 0, 2, 0,
Win average time:
1391.0, 1182.0, 989.5, -, 2580.0, 1182.0, 989.5, 1478.5,
-, 2917.0, -, -, -, 1625.0, -, -,
-, 1552.5, 2475.0, -, -, 1642.5, 2775.0, 1595.0,
390.0, 390.0, 390.0, 2510.0, 390.0, 390.0, 390.0, 2820.0,
2113.0, 1292.0, 1603.5, -, -, 2073.0, 1470.0, 1727.5,
-, -, -, -, -, 1693.0, -, -,
-, 2550.0, -, -, -, 1755.0, -, 1747.5,
-, 1625.0, -, -, -, 1135.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, 5000.0, 5000.0, 3260.0, -,
-, -, -, -, -, -, 3260.0, -,
-, -, -, -, -, -, -, 3350.0,
-, 5000.0, -, -, 5000.0, 4680.0, -, -,
-, 5000.0, -, -, 4680.0, -, 3260.0, 5000.0,
-, 3260.0, 3260.0, -, -, 3260.0, 3260.0, -,
-, -, -, 3350.0, -, 5000.0, -, 3350.0,
Lose average time:
1391.0, -, -, 390.0, 2113.0, -, -, -,
1182.0, 2917.0, 1552.5, 390.0, 1292.0, -, 2550.0, 1625.0,
989.5, -, 2475.0, 390.0, 1603.5, -, -, -,
-, -, -, 2510.0, -, -, -, -,
2580.0, -, -, 390.0, -, -, -, -,
1182.0, 1625.0, 1642.5, 390.0, 2073.0, 1693.0, 1755.0, 1135.0,
989.5, -, 2775.0, 390.0, 1470.0, -, -, -,
1478.5, -, 1595.0, 2820.0, 1727.5, -, 1747.5, -,
```

Figure 3.: Resource Quantity - Gen2 Results

```
----------------------
Tournament 4
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 0, 0, 0, 1, 1, 0,
0, 2, 1, 0, 0, 2, 2, 2,
2, 2, 2, 1, 2, 2, 2, 1,
1, 1, 2, 0, 1, 1, 2, 2,
0, 0, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 0, 2, 0, 2,
0, 2, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1,
0, 1, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 1, 0, 0, 1,
0, 0, 0, 0, 0, 0, 2, 0,
0, 0, 0, 1, 0, 1, 0, 2,
Loses:
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 0, 1, 2,
2, 0, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 1, 2, 1,
2, 1, 2, 2, 2, 0, 0, 0,
2, 0, 2, 1, 2, 0, 2, 0,
Win average time:
1391.0, 1182.0, 989.5, -, 2580.0, 1182.0, 989.5, 1478.5,
-, 3462.0, -, -, -, 1625.0, 3958.0, -,
-, 1552.5, 2475.0, -, -, 1642.5, 3220.0, 1595.0,
390.0, 390.0, 390.0, 3550.0, 390.0, 390.0, 390.0, 3515.0,
2113.0, 1292.0, 1603.5, -, 2796.0, 2073.0, 1470.0, 1727.5,
-, -, -, -, -, 1693.0, -, -,
-, 2550.0, -, -, -, 2440.0, -, 1747.5,
-, 1625.0, -, -, -, 1135.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, 6000.0, 6000.0, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, 4070.0,
-, 6000.0, -, -, -, 4200.0, -, -,
-, 6000.0, -, -, 4200.0, -, -, 6000.0,
-, -, -, -, -, -, 4290.0, -,
-, -, -, 4070.0, -, 6000.0, -, 4070.0,
Lose average time:
1391.0, -, -, 390.0, 2113.0, -, -, -,
1182.0, 3462.0, 1552.5, 390.0, 1292.0, -, 2550.0, 1625.0,
989.5, -, 2475.0, 390.0, 1603.5, -, -, -,
-, -, -, 3550.0, -, -, -, -,
2580.0, -, -, 390.0, 2796.0, -, -, -,
1182.0, 1625.0, 1642.5, 390.0, 2073.0, 1693.0, 2440.0, 1135.0,
989.5, 3958.0, 3220.0, 390.0, 1470.0, -, -, -,
1478.5, -, 1595.0, 3515.0, 1727.5, -, 1747.5, -,
```

Figure 4.: Resource Quantity - Gen3 Results

```
----------------------
Tournament 5
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 1, 0, 0, 1, 1, 0,
0, 1, 1, 0, 0, 2, 1, 2,
2, 2, 2, 1, 2, 2, 2, 1,
0, 2, 2, 0, 1, 1, 2, 2,
0, 1, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 0, 2, 0, 2,
0, 1, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 1,
1, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 1, 0, 0, 1,
0, 0, 1, 0, 0, 0, 2, 0,
0, 1, 0, 1, 0, 1, 0, 2,
Loses:
1, 0, 0, 2, 0, 0, 0, 0,
2, 1, 1, 2, 2, 1, 1, 1,
2, 1, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 1, 2, 1,
2, 1, 1, 2, 2, 0, 0, 0,
2, 0, 2, 1, 2, 0, 2, 0,
Win average time:
1134.0, 1147.5, 972.0, -, 1134.0, 1459.5, 976.5, 1437.0,
-, 1042.0, 1313.0, -, -, 1042.0, 1313.0, -,
-, 1065.0, 1175.0, -, -, 2045.0, 1175.0, 1532.5,
377.5, 377.5, 377.5, 3195.0, 377.5, 377.5, 377.5, 3280.0,
-, 1445.5, 1382.0, -, 3243.0, 2257.0, 1908.5, 1372.0,
-, 1302.0, -, -, -, 1526.0, -, -,
-, 1640.0, -, -, -, 2222.5, -, 1482.5,
-, 1015.0, -, -, -, 1015.0, -, -,
Tie average time:
-, -, -, -, 3158.0, -, -, -,
-, -, -, -, -, -, -, 2300.0,
-, -, -, -, -, -, 3170.0, -,
-, -, -, -, -, -, -, 3350.0,
3158.0, -, -, -, -, 5000.0, -, -,
-, -, -, -, 5000.0, -, -, 2850.0,
-, -, 3170.0, -, -, -, 3200.0, -,
-, 2300.0, -, 3350.0, -, 2850.0, -, 3415.0,
Lose average time:
1134.0, -, -, 377.5, -, -, -, -,
1147.5, 1042.0, 1065.0, 377.5, 1445.5, 1302.0, 1640.0, 1015.0,
972.0, 1313.0, 1175.0, 377.5, 1382.0, -, -, -,
-, -, -, 3195.0, -, -, -, -,
1134.0, -, -, 377.5, 3243.0, -, -, -,
1459.5, 1042.0, 2045.0, 377.5, 2257.0, 1526.0, 2222.5, 1015.0,
976.5, 1313.0, 1175.0, 377.5, 1908.5, -, -, -,
1437.0, -, 1532.5, 3280.0, 1372.0, -, 1482.5, -,
```

Figure 5.: Resource Position - Gen1 Results

```
----------------------
Tournament 6
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 0, 0, 0, 1, 0, 1,
0, 2, 1, 0, 0, 2, 1, 2,
2, 2, 2, 0, 2, 2, 2, 0,
1, 2, 2, 0, 1, 1, 2, 2,
0, 0, 0, 0, 0, 1, 0, 1,
0, 1, 0, 0, 0, 1, 0, 2,
0, 1, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 1, 0,
0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 2, 0, 0, 0, 2,
0, 0, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 1, 0, 1, 0,
0, 1, 1, 0, 0, 1, 2, 0,
0, 0, 0, 2, 0, 0, 0, 2,
Loses:
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 2, 0, 1, 1,
2, 0, 1, 2, 2, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 1, 1, 1,
2, 0, 1, 2, 2, 0, 0, 0,
2, 1, 2, 0, 2, 1, 2, 0,
Win average time:
1387.0, 1163.5, 975.0, -, 2459.0, 1163.5, 975.0, 1520.0,
-, 3446.0, -, -, -, 1625.0, -, 2538.0,
-, 1552.5, 2545.0, -, -, 1642.5, 2885.0, 1427.5,
395.0, 395.0, 395.0, -, 395.0, 395.0, 395.0, -,
2518.0, 1616.5, 1415.0, -, 2791.0, 2094.0, 1338.5, 1547.0,
-, -, -, -, -, 1693.0, -, 2392.0,
-, 2550.0, -, -, -, 1755.0, -, 1607.5,
-, 1075.0, -, -, -, 1075.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, -, 5000.0, 3625.0, -,
-, -, -, -, -, -, 3630.0, -,
-, -, -, 5000.0, -, -, -, 4200.0,
-, -, -, -, -, 3480.0, -, -,
-, 5000.0, -, -, 3480.0, -, 3610.0, -,
-, 3625.0, 3630.0, -, -, 3610.0, 3570.0, -,
-, -, -, 4200.0, -, -, -, 5000.0,
Lose average time:
1387.0, -, -, 395.0, 2518.0, -, -, -,
1163.5, 3446.0, 1552.5, 395.0, 1616.5, -, 2550.0, 1075.0,
975.0, -, 2545.0, 395.0, 1415.0, -, -, -,
-, -, -, -, -, -, -, -,
2459.0, -, -, 395.0, 2791.0, -, -, -,
1163.5, 1625.0, 1642.5, 395.0, 2094.0, 1693.0, 1755.0, 1075.0,
975.0, -, 2885.0, 395.0, 1338.5, -, -, -,
1520.0, 2538.0, 1427.5, -, 1547.0, 2392.0, 1607.5, -,
```

Figure 6.: Resource Position - Gen2 Results

```
----------------------
Tournament 7
Wins:
1, 1, 2, 0, 1, 1, 2, 1,
0, 1, 1, 0, 0, 0, 1, 0,
0, 1, 1, 0, 0, 2, 1, 1,
2, 2, 2, 1, 2, 2, 2, 1,
1, 2, 1, 0, 1, 1, 2, 1,
1, 2, 0, 0, 0, 0, 0, 0,
0, 1, 1, 0, 0, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1,
Ties:
0, 1, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 1, 0, 0,
0, 0, 0, 0, 1, 2, 1, 1,
0, 0, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 0, 1, 0, 0,
Loses:
1, 0, 0, 2, 1, 1, 0, 1,
1, 1, 1, 2, 2, 2, 1, 1,
2, 1, 1, 2, 1, 0, 1, 1,
0, 0, 0, 1, 0, 0, 0, 1,
1, 0, 0, 2, 1, 0, 0, 1,
1, 0, 2, 2, 1, 0, 1, 1,
2, 1, 1, 2, 2, 0, 1, 1,
1, 0, 1, 1, 1, 0, 1, 1,
Win average time:
1170.0, 976.0, 1231.0, -, 1170.0, 976.0, 1231.0, 1139.0,
-, 2203.0, 1816.0, -, -, -, 2035.0, -,
-, 2360.0, 1810.0, -, -, 1907.5, 1810.0, 1250.0,
387.5, 387.5, 387.5, 905.0, 387.5, 387.5, 387.5, 905.0,
1430.0, 1727.0, 1470.0, -, 2924.0, 2325.0, 1427.5, 1374.0,
1615.0, 1718.5, -, -, -, -, -, -,
-, 2205.0, 2765.0, -, -, 1940.0, 2765.0, 1775.0,
1325.0, 1325.0, 1925.0, 800.0, 1325.0, 1325.0, 1925.0, 800.0,
Tie average time:
-, 5000.0, -, -, -, -, -, -,
5000.0, -, -, -, -, -, -, 3320.0,
-, -, -, -, 3980.0, -, -, -,
-, -, -, -, -, -, -, -,
-, -, 3980.0, -, -, 3722.0, -, -,
-, -, -, -, 3722.0, 5000.0, 3980.0, 3890.0,
-, -, -, -, -, 3980.0, -, -,
-, 3320.0, -, -, -, 3890.0, -, -,
Lose average time:
1170.0, -, -, 387.5, 1430.0, 1615.0, -, 1325.0,
976.0, 2203.0, 2360.0, 387.5, 1727.0, 1718.5, 2205.0, 1325.0,
1231.0, 1816.0, 1810.0, 387.5, 1470.0, -, 2765.0, 1925.0,
-, -, -, 905.0, -, -, -, 800.0,
1170.0, -, -, 387.5, 2924.0, -, -, 1325.0,
976.0, -, 1907.5, 387.5, 2325.0, -, 1940.0, 1325.0,
1231.0, 2035.0, 1810.0, 387.5, 1427.5, -, 2765.0, 1925.0,
1139.0, -, 1250.0, 905.0, 1374.0, -, 1775.0, 800.0,
```

Figure 7.: Resource Position - Gen3 Results

```
----------------------
Tournament 8
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 1, 0, 0, 1, 1, 0,
0, 1, 1, 0, 0, 2, 1, 2,
2, 2, 2, 1, 2, 2, 2, 1,
1, 2, 1, 0, 0, 1, 1, 2,
0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 1, 0, 2,
0, 2, 0, 1, 0, 2, 0, 1,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 1, 0,
0, 0, 0, 0, 1, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 2, 1, 1, 0,
0, 1, 0, 0, 1, 0, 1, 0,
0, 1, 1, 0, 1, 1, 2, 0,
0, 0, 0, 0, 0, 0, 0, 0,
Loses:
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 1, 2, 2, 0, 0, 2,
2, 1, 1, 2, 1, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 1,
1, 0, 0, 2, 0, 0, 0, 0,
2, 1, 2, 2, 1, 1, 1, 2,
2, 1, 1, 2, 1, 0, 0, 0,
2, 0, 2, 1, 2, 0, 2, 1,
Win average time:
1042.0, 1052.0, 993.5, -, 1042.0, 1405.0, 988.5, 1363.5,
-, 1884.0, 1887.0, -, -, 1884.0, 1887.0, -,
-, 1045.0, 1065.0, -, -, 1452.5, 1065.0, 1737.5,
390.0, 390.0, 390.0, 2170.0, 390.0, 390.0, 390.0, 3045.0,
1568.0, 1965.0, 1425.0, -, -, 1855.0, 1828.0, 1534.0,
-, -, -, -, -, 2198.0, -, -,
-, -, -, -, -, 3090.0, -, 1515.0,
-, 1925.0, -, 2715.0, -, 2282.5, -, 2655.0,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, -, 3130.0, 3480.0, -,
-, -, -, -, 2510.0, -, 3340.0, -,
-, -, -, -, -, -, -, -,
-, -, 2510.0, -, 2872.0, 3620.0, 2510.0, -,
-, 3130.0, -, -, 3620.0, -, 4440.0, -,
-, 3480.0, 3340.0, -, 2510.0, 4440.0, 3220.0, -,
-, -, -, -, -, -, -, -,
Lose average time:
1042.0, -, -, 390.0, 1568.0, -, -, -,
1052.0, 1884.0, 1045.0, 390.0, 1965.0, -, -, 1925.0,
993.5, 1887.0, 1065.0, 390.0, 1425.0, -, -, -,
-, -, -, 2170.0, -, -, -, 2715.0,
1042.0, -, -, 390.0, -, -, -, -,
1405.0, 1884.0, 1452.5, 390.0, 1855.0, 2198.0, 3090.0, 2282.5,
988.5, 1887.0, 1065.0, 390.0, 1828.0, -, -, -,
1363.5, -, 1737.5, 3045.0, 1534.0, -, 1515.0, 2655.0,
```

Figure 8.: Base Position - Gen1 Results

```
Tournament 9
Wins:
1, 1, 2, 0, 1, 1, 1, 1,
1, 1, 1, 0, 0, 0, 1, 1,
0, 1, 1, 0, 0, 2, 1, 1,
2, 2, 2, 1, 2, 2, 2, 1,
1, 1, 2, 0, 1, 1, 2, 1,
1, 2, 0, 0, 1, 1, 1, 1,
1, 1, 1, 0, 0, 1, 1, 1,
1, 1, 1, 0, 1, 1, 1, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1,
0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 2,
Loses:
1, 1, 0, 2, 1, 1, 1, 1,
1, 1, 1, 2, 1, 2, 1, 1,
2, 1, 1, 2, 2, 0, 1, 1,
0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 2, 1, 1, 0, 1,
1, 0, 2, 2, 1, 1, 1, 1,
1, 1, 1, 2, 2, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 0,
Win average time:
995.0, 1056.0, 1071.0, -, 995.0, 1056.0, 906.0, 1244.0,
2389.0, 1898.0, 1763.0, -, -, -, 1409.0, 3089.0,
-, 1680.0, 1665.0, -, -, 1642.5, 1665.0, 1890.0,
370.0, 370.0, 370.0, 1105.0, 370.0, 370.0, 370.0, 1105.0,
1426.0, 1276.0, 1497.5, -, 1426.0, 1276.0, 1540.0, 1369.0,
2376.0, 1890.0, -, -, 3462.0, 1424.0, 2619.0, 2373.0,
2670.0, 2195.0, 2760.0, -, -, 2195.0, 3295.0, 1240.0,
1815.0, 1250.0, 1035.0, -, 1815.0, 1250.0, 1035.0, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, 5000.0, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, 5000.0,
-, 5000.0, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, 5000.0, -, -, -, 5000.0,
Lose average time:
995.0, 2389.0, -, 370.0, 1426.0, 2376.0, 2670.0, 1815.0,
1056.0, 1898.0, 1680.0, 370.0, 1276.0, 1890.0, 2195.0, 1250.0,
1071.0, 1763.0, 1665.0, 370.0, 1497.5, -, 2760.0, 1035.0,
-, -, -, 1105.0, -, -, -, -,
995.0, -, -, 370.0, 1426.0, 3462.0, -, 1815.0,
1056.0, -, 1642.5, 370.0, 1276.0, 1424.0, 2195.0, 1250.0,
906.0, 1409.0, 1665.0, 370.0, 1540.0, 2619.0, 3295.0, 1035.0,
1244.0, 3089.0, 1890.0, 1105.0, 1369.0, 2373.0, 1240.0, -,
```

Figure 9.: Base Position - Gen2 Results

```
----------------------
Tournament 10
Wins:
1, 2, 2, 0, 2, 2, 2, 2,
0, 1, 0, 0, 1, 0, 0, 0,
0, 2, 1, 0, 0, 2, 1, 2,
2, 2, 2, 1, 2, 2, 2, 2,
0, 1, 1, 0, 0, 1, 1, 2,
0, 1, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 2,
0, 0, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 1, 2,
0, 0, 0, 0, 1, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 2, 1, 1, 0,
0, 1, 0, 0, 1, 2, 2, 1,
0, 1, 1, 0, 1, 2, 2, 0,
0, 2, 0, 0, 0, 1, 0, 2,
Loses:
1, 0, 0, 2, 0, 0, 0, 0,
2, 1, 2, 2, 1, 1, 1, 0,
2, 0, 1, 2, 1, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
2, 1, 0, 2, 0, 0, 0, 0,
2, 0, 2, 2, 1, 0, 0, 1,
2, 0, 1, 2, 1, 0, 0, 0,
2, 0, 2, 2, 2, 0, 2, 0,
Win average time:
975.0, 1011.0, 987.0, -, 1824.0, 1484.5, 1056.0, 1639.0,
-, 1845.0, -, -, 1936.0, -, -, -,
-, 1472.5, 1175.0, -, -, 1535.0, 1175.0, 1650.0,
390.0, 390.0, 390.0, 2945.0, 390.0, 390.0, 390.0, 2485.0,
-, 1973.0, 1482.0, -, -, 2516.0, 1308.0, 1575.5,
-, 1893.0, -, -, -, -, -, -,
-, 3310.0, -, -, -, -, -, 1707.5,
-, -, -, -, -, 1415.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, -, 3150.0, 3300.0, 4402.5,
-, -, -, -, 2530.0, -, 3250.0, -,
-, -, -, -, -, -, -, -,
-, -, 2530.0, -, 3008.0, 5000.0, 2530.0, -,
-, 3150.0, -, -, 5000.0, 5000.0, 3209.5, 3755.0,
-, 3300.0, 3250.0, -, 2530.0, 3209.5, 3260.0, -,
-, 4402.5, -, -, -, 3755.0, -, 3210.0,
Lose average time:
975.0, -, -, 390.0, -, -, -, -,
1011.0, 1845.0, 1472.5, 390.0, 1973.0, 1893.0, 3310.0, -,
987.0, -, 1175.0, 390.0, 1482.0, -, -, -,
-, -, -, 2945.0, -, -, -, -,
1824.0, 1936.0, -, 390.0, -, -, -, -,
1484.5, -, 1535.0, 390.0, 2516.0, -, -, 1415.0,
1056.0, -, 1175.0, 390.0, 1308.0, -, -, -,
1639.0, -, 1650.0, 2485.0, 1575.5, -, 1707.5, -,
```

Figure 10.: Base Position - Gen3 Results

```
----------------------
Tournament 11
Wins:
1, 2, 2, 0, 0, 2, 2, 2,
0, 1, 1, 0, 1, 1, 1, 1,
0, 1, 1, 0, 0, 1, 2, 2,
2, 2, 2, 1, 2, 2, 2, 1,
2, 1, 1, 0, 1, 2, 2, 2,
0, 1, 0, 0, 0, 1, 0, 1,
0, 1, 0, 0, 0, 1, 0, 2,
0, 1, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 1, 2, 0,
0, 0, 0, 1, 0, 0, 0, 2,
Loses:
1, 0, 0, 2, 2, 0, 0, 0,
2, 1, 1, 2, 1, 1, 1, 1,
2, 1, 1, 2, 1, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
0, 1, 0, 2, 1, 0, 0, 0,
2, 1, 1, 2, 2, 1, 1, 1,
2, 1, 2, 2, 2, 0, 0, 0,
2, 1, 2, 1, 2, 1, 2, 0,
Win average time:
1430.0, 1195.0, 1001.0, -, -, 1498.5, 1029.5, 1335.0,
-, 1042.0, 1313.0, -, 2279.0, 1042.0, 1313.0, 3363.0,
-, 1065.0, 1175.0, -, -, 1065.0, 2122.5, 1542.5,
370.0, 370.0, 370.0, 3325.0, 370.0, 370.0, 370.0, 3280.0,
1516.0, 1542.0, 1390.0, -, 2754.0, 1977.0, 1993.5, 1534.5,
-, 1302.0, -, -, -, 1526.0, -, 3547.0,
-, 1640.0, -, -, -, 1750.0, -, 1747.5,
-, 1035.0, -, -, -, 1035.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, 3500.0, 3710.0, -, -,
-, -, -, -, -, -, -, 3560.0,
-, -, 3500.0, -, -, -, -, -,
-, -, 3710.0, -, -, -, 3855.0, -,
-, -, -, -, -, 3855.0, 3590.0, -,
-, -, -, 3560.0, -, -, -, 3470.0,
Lose average time:
1430.0, -, -, 370.0, 1516.0, -, -, -,
1195.0, 1042.0, 1065.0, 370.0, 1542.0, 1302.0, 1640.0, 1035.0,
1001.0, 1313.0, 1175.0, 370.0, 1390.0, -, -, -,
-, -, -, 3325.0, -, -, -, -,
-, 2279.0, -, 370.0, 2754.0, -, -, -,
1498.5, 1042.0, 1065.0, 370.0, 1977.0, 1526.0, 1750.0, 1035.0,
1029.5, 1313.0, 2122.5, 370.0, 1993.5, -, -, -,
1335.0, 3363.0, 1542.5, 3280.0, 1534.5, 3547.0, 1747.5, -,
```

Figure 11.: Starting Resources - Gen1 Results

```
----------------------
Tournament 12
Wins:
1, 1, 2, 0, 1, 1, 2, 2,
1, 1, 1, 0, 1, 1, 1, 1,
0, 1, 1, 0, 0, 2, 2, 2,
2, 2, 2, 1, 2, 2, 2, 1,
1, 1, 2, 0, 1, 1, 2, 2,
0, 0, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 0, 1, 0, 2,
0, 1, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 1, 0, 0,
1, 1, 0, 0, 1, 0, 1, 1,
0, 0, 0, 0, 0, 1, 2, 0,
0, 0, 0, 1, 0, 1, 0, 2,
Loses:
1, 1, 0, 2, 1, 0, 0, 0,
1, 1, 1, 2, 1, 0, 1, 1,
2, 1, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 1, 0, 2, 1, 0, 0, 0,
1, 1, 2, 2, 1, 1, 1, 1,
2, 1, 2, 2, 2, 0, 0, 0,
2, 1, 2, 1, 2, 0, 2, 0,
Win average time:
928.0, 656.0, 868.0, -, 928.0, 656.0, 868.0, 1241.5,
1589.0, 1363.0, 1103.0, -, 1589.0, 1363.0, 1103.0, 2495.0,
-, 865.0, 865.0, -, -, 1232.5, 1967.5, 1280.0,
365.0, 365.0, 365.0, 3325.0, 365.0, 365.0, 365.0, 3250.0,
1409.0, 1813.0, 1310.5, -, 2063.0, 1684.0, 1718.0, 1482.5,
-, -, -, -, -, 1526.0, -, -,
-, 1570.0, -, -, -, 1750.0, -, 1527.5,
-, 1255.0, -, -, -, 1255.0, -, -,
Tie average time:
-, -, -, -, -, 3480.0, -, -,
-, -, -, -, -, 5000.0, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, 3590.0,
-, -, -, -, -, 5000.0, -, -,
3480.0, 5000.0, -, -, 5000.0, -, 3825.0, 3820.0,
-, -, -, -, -, 3825.0, 3590.0, -,
-, -, -, 3590.0, -, 3820.0, -, 3520.0,
Lose average time:
928.0, 1589.0, -, 365.0, 1409.0, -, -, -,
656.0, 1363.0, 865.0, 365.0, 1813.0, -, 1570.0, 1255.0,
868.0, 1103.0, 865.0, 365.0, 1310.5, -, -, -,
-, -, -, 3325.0, -, -, -, -,
928.0, 1589.0, -, 365.0, 2063.0, -, -, -,
656.0, 1363.0, 1232.5, 365.0, 1684.0, 1526.0, 1750.0, 1255.0,
868.0, 1103.0, 1967.5, 365.0, 1718.0, -, -, -,
1241.5, 2495.0, 1280.0, 3250.0, 1482.5, -, 1527.5, -,
```

Figure 12.: Starting Resources - Gen2 Results

```
----------------------
Tournament 13
Wins:
1, 1, 1, 0, 1, 2, 2, 2,
1, 1, 1, 0, 1, 1, 1, 1,
1, 1, 1, 0, 1, 2, 1, 2,
2, 2, 2, 1, 2, 2, 2, 1,
0, 1, 1, 0, 0, 2, 1, 2,
0, 0, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 0, 1, 0, 2,
0, 1, 0, 0, 0, 2, 0, 0,
Ties:
0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 1,
1, 0, 0, 0, 2, 0, 1, 0,
0, 1, 0, 0, 0, 0, 1, 0,
0, 0, 1, 0, 1, 1, 2, 0,
0, 0, 0, 1, 0, 0, 0, 2,
Loses:
1, 1, 1, 2, 0, 0, 0, 0,
1, 1, 1, 2, 1, 0, 1, 1,
1, 1, 1, 2, 1, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 1, 1, 2, 0, 0, 0, 0,
2, 1, 2, 2, 2, 1, 1, 2,
2, 1, 1, 2, 1, 0, 0, 0,
2, 1, 2, 1, 2, 0, 2, 0,
Win average time:
602.0, 602.0, 602.0, -, 602.0, 1308.5, 856.5, 1208.5,
935.0, 874.0, 943.0, -, 935.0, 874.0, 943.0, 3838.0,
835.0, 675.0, 745.0, -, 835.0, 1582.5, 745.0, 1457.5,
559.5, 365.0, 365.0, 3435.0, 559.5, 365.0, 365.0, 3435.0,
-, 1750.0, 1262.0, -, -, 1917.0, 2049.0, 1414.5,
-, -, -, -, -, 1604.0, -, -,
-, 3150.0, -, -, -, 1750.0, -, 1570.0,
-, 3750.0, -, -, -, 2975.0, -, -,
Tie average time:
-, -, -, -, 5000.0, -, -, -,
-, -, -, -, -, 3490.0, -, -,
-, -, -, -, -, -, 3520.0, -,
-, -, -, -, -, -, -, 3630.0,
5000.0, -, -, -, 3490.0, -, 2950.0, -,
-, 3490.0, -, -, -, -, 3540.0, -,
-, -, 3520.0, -, 2950.0, 3540.0, 3520.0, -,
-, -, -, 3630.0, -, -, -, 3610.0,
Lose average time:
602.0, 935.0, 835.0, 559.5, -, -, -, -,
602.0, 874.0, 675.0, 365.0, 1750.0, -, 3150.0, 3750.0,
602.0, 943.0, 745.0, 365.0, 1262.0, -, -, -,
-, -, -, 3435.0, -, -, -, -,
602.0, 935.0, 835.0, 559.5, -, -, -, -,
1308.5, 874.0, 1582.5, 365.0, 1917.0, 1604.0, 1750.0, 2975.0,
856.5, 943.0, 745.0, 365.0, 2049.0, -, -, -,
1208.5, 3838.0, 1457.5, 3435.0, 1414.5, -, 1570.0, -,
```

Figure 13.: Starting Resources - Gen3 Results

```
----------------------
Tournament 14
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 0, 0, 0, 1, 1, 0,
0, 2, 1, 0, 0, 2, 2, 2,
2, 2, 2, 1, 2, 2, 2, 1,
1, 1, 2, 0, 1, 1, 2, 2,
0, 0, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 0, 2, 0, 2,
0, 2, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1,
0, 1, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 1, 0, 0, 1,
0, 0, 0, 0, 0, 0, 2, 0,
0, 0, 0, 1, 0, 1, 0, 2,
Loses:
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 0, 1, 2,
2, 0, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 1, 2, 1,
2, 1, 2, 2, 2, 0, 0, 0,
2, 0, 2, 1, 2, 0, 2, 0,
Win average time:
1391.0, 1182.0, 989.5, -, 2580.0, 1182.0, 989.5, 1478.5,
-, 3462.0, -, -, -, 1625.0, 3958.0, -,
-, 1552.5, 2475.0, -, -, 1642.5, 3220.0, 1595.0,
390.0, 390.0, 390.0, 3550.0, 390.0, 390.0, 390.0, 3515.0,
2113.0, 1292.0, 1603.5, -, 2796.0, 2073.0, 1470.0, 1727.5,
-, -, -, -, -, 1693.0, -, -,
-, 2550.0, -, -, -, 2440.0, -, 1747.5,
-, 1625.0, -, -, -, 1135.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, 5000.0, 5000.0, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, 4070.0,
-, 5000.0, -, -, -, 4200.0, -, -,
-, 5000.0, -, -, 4200.0, -, -, 5000.0,
-, -, -, -, -, -, 4290.0, -,
-, -, -, 4070.0, -, 5000.0, -, 4070.0,
Lose average time:
1391.0, -, -, 390.0, 2113.0, -, -, -,
1182.0, 3462.0, 1552.5, 390.0, 1292.0, -, 2550.0, 1625.0,
989.5, -, 2475.0, 390.0, 1603.5, -, -, -,
-, -, -, 3550.0, -, -, -, -,
2580.0, -, -, 390.0, 2796.0, -, -, -,
1182.0, 1625.0, 1642.5, 390.0, 2073.0, 1693.0, 2440.0, 1135.0,
989.5, 3958.0, 3220.0, 390.0, 1470.0, -, -, -,
1478.5, -, 1595.0, 3515.0, 1727.5, -, 1747.5, -,
```

Figure 14.: RQ-RP - Gen1 Results

```
----------------------
Tournament 15
Wins:
1, 2, 2, 0, 1, 1, 2, 2,
0, 1, 1, 0, 0, 0, 1, 2,
0, 1, 1, 0, 0, 2, 2, 2,
2, 2, 2, 1, 2, 2, 2, 1,
0, 1, 2, 0, 0, 1, 2, 2,
1, 1, 0, 0, 0, 1, 0, 1,
0, 0, 0, 0, 0, 0, 0, 2,
0, 0, 0, 1, 0, 1, 0, 1,
Ties:
0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 1, 1, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
1, 1, 0, 0, 2, 1, 0, 0,
0, 1, 0, 0, 1, 0, 2, 0,
0, 1, 0, 0, 0, 2, 2, 0,
0, 0, 0, 0, 0, 0, 0, 0,
Loses:
1, 0, 0, 2, 0, 1, 0, 0,
2, 1, 1, 2, 1, 1, 0, 0,
2, 1, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 1,
1, 0, 0, 2, 0, 0, 0, 0,
1, 0, 2, 2, 1, 1, 0, 1,
2, 1, 2, 2, 2, 0, 0, 0,
2, 2, 2, 1, 2, 1, 2, 1,
Win average time:
1179.0, 1625.5, 1059.5, -, 1179.0, 945.0, 1083.5, 1945.5,
-, 2262.0, 1487.0, -, -, -, 1487.0, 2846.0,
-, 1080.0, 1085.0, -, -, 1327.5, 2110.0, 1382.5,
390.0, 390.0, 390.0, 905.0, 390.0, 390.0, 390.0, 905.0,
-, 2618.0, 1534.5, -, -, 1822.0, 1892.5, 2017.0,
1875.0, 2262.0, -, -, -, 1894.0, -, 2453.0,
-, -, -, -, -, -, -, 1550.0,
-, -, -, 1565.0, -, 2385.0, -, 1565.0,
Tie average time:
-, -, -, -, 5000.0, -, -, -,
-, -, -, -, 5000.0, 5000.0, 5000.0, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
5000.0, 5000.0, -, -, 5000.0, 5000.0, -, -,
-, 5000.0, -, -, 5000.0, -, 3635.0, -,
-, 5000.0, -, -, -, 3635.0, 3650.0, -,
-, -, -, -, -, -, -, -,
Lose average time:
1179.0, -, -, 390.0, -, 1875.0, -, -,
1625.5, 2262.0, 1080.0, 390.0, 2618.0, 2262.0, -, -,
1059.5, 1487.0, 1085.0, 390.0, 1534.5, -, -, -,
-, -, -, 905.0, -, -, -, 1565.0,
1179.0, -, -, 390.0, -, -, -, -,
945.0, -, 1327.5, 390.0, 1822.0, 1894.0, -, 2385.0,
1083.5, 1487.0, 2110.0, 390.0, 1892.5, -, -, -,
1945.5, 2846.0, 1382.5, 905.0, 2017.0, 2453.0, 1550.0, 1565.0,
```

Figure 15.: RQ-RP - Gen2 Results

```
----------------------
Tournament 16
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 1, 0, 0, 1, 1, 0,
0, 1, 1, 0, 0, 2, 2, 2,
2, 2, 2, 1, 2, 2, 2, 2,
1, 2, 2, 0, 1, 1, 2, 2,
0, 1, 0, 0, 0, 0, 0, 1,
0, 1, 0, 0, 0, 2, 0, 2,
0, 2, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 1, 2, 0, 0,
0, 0, 0, 0, 0, 0, 2, 0,
0, 0, 0, 0, 0, 0, 0, 2,
Loses:
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 1, 2, 2, 1, 1, 2,
2, 1, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 0, 2, 1,
2, 1, 2, 2, 2, 0, 0, 0,
2, 0, 2, 2, 2, 1, 2, 0,
Win average time:
1450.0, 1128.5, 882.5, -, 1450.0, 1285.0, 1147.0, 1376.0,
-, 1158.0, 1757.0, -, -, 1158.0, 1757.0, -,
-, 1305.0, 2305.0, -, -, 1502.5, 2907.5, 1557.5,
387.5, 387.5, 387.5, 2205.0, 387.5, 387.5, 387.5, 3032.5,
1695.0, 1674.5, 1282.5, -, 1404.0, 1244.0, 1590.0, 1388.0,
-, 1312.0, -, -, -, -, -, 3188.0,
-, 1835.0, -, -, -, 2237.5, -, 1655.0,
-, 2550.0, -, -, -, 1915.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, 5000.0, -, -,
-, -, -, -, 5000.0, 5000.0, -, -,
-, -, -, -, -, -, 4290.0, -,
-, -, -, -, -, -, -, 4000.0,
Lose average time:
1450.0, -, -, 387.5, 1695.0, -, -, -,
1128.5, 1158.0, 1305.0, 387.5, 1674.5, 1312.0, 1835.0, 2550.0,
882.5, 1757.0, 2305.0, 387.5, 1282.5, -, -, -,
-, -, -, 2205.0, -, -, -, -,
1450.0, -, -, 387.5, 1404.0, -, -, -,
1285.0, 1158.0, 1502.5, 387.5, 1244.0, -, 2237.5, 1915.0,
1147.0, 1757.0, 2907.5, 387.5, 1590.0, -, -, -,
1376.0, -, 1557.5, 3032.5, 1388.0, 3188.0, 1655.0, -,
```

Figure 16.: RQ-BP - Gen1 Results

```
----------------------
Tournament 17
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 1, 0, 0, 1, 1, 1,
0, 1, 1, 0, 0, 2, 1, 2,
2, 2, 2, 1, 2, 2, 2, 1,
1, 2, 2, 0, 1, 1, 2, 2,
0, 1, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 1, 0, 2,
0, 1, 0, 1, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 1, 2, 1, 1,
0, 0, 1, 0, 0, 1, 2, 0,
0, 0, 0, 0, 0, 1, 0, 2,
Loses:
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 1, 2, 2, 1, 1, 1,
2, 1, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 1,
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 0, 1, 1,
2, 1, 1, 2, 2, 0, 0, 0,
2, 1, 2, 1, 2, 0, 2, 0,
Win average time:
1450.0, 1128.5, 882.5, -, 1450.0, 1285.0, 1147.0, 1376.0,
-, 1158.0, 1757.0, -, -, 1158.0, 1757.0, 3109.0,
-, 1305.0, 2305.0, -, -, 1502.5, 2305.0, 1557.5,
387.5, 387.5, 387.5, 2205.0, 387.5, 387.5, 387.5, 2640.0,
1695.0, 1674.5, 1282.5, -, 1404.0, 1244.0, 1590.0, 1388.0,
-, 1312.0, -, -, -, -, -, -,
-, 1835.0, -, -, -, 1420.0, -, 1655.0,
-, 1915.0, -, 2420.0, -, 1915.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, 3080.0, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, 3558.0, -, -,
-, -, -, -, 3558.0, 5000.0, 3100.0, 3558.0,
-, -, 3080.0, -, -, 3100.0, 3100.0, -,
-, -, -, -, -, 3558.0, -, 2920.0,
Lose average time:
1450.0, -, -, 387.5, 1695.0, -, -, -,
1128.5, 1158.0, 1305.0, 387.5, 1674.5, 1312.0, 1835.0, 1915.0,
882.5, 1757.0, 2305.0, 387.5, 1282.5, -, -, -,
-, -, -, 2205.0, -, -, -, 2420.0,
1450.0, -, -, 387.5, 1404.0, -, -, -,
1285.0, 1158.0, 1502.5, 387.5, 1244.0, -, 1420.0, 1915.0,
1147.0, 1757.0, 2305.0, 387.5, 1590.0, -, -, -,
1376.0, 3109.0, 1557.5, 2640.0, 1388.0, -, 1655.0, -,
```

Figure 17.: RQ-BP - Gen2 Results

```
----------------------
Tournament 18
Wins:
1, 2, 2, 0, 1, 2, 2, 2,
0, 1, 0, 0, 0, 1, 2, 1,
0, 2, 1, 0, 0, 2, 2, 2,
2, 2, 2, 1, 2, 2, 2, 2,
1, 2, 2, 0, 1, 1, 2, 2,
0, 1, 0, 0, 0, 1, 0, 1,
0, 0, 0, 0, 0, 1, 0, 2,
0, 1, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 1, 0, 1, 0,
0, 0, 0, 0, 0, 1, 2, 0,
0, 0, 0, 0, 0, 0, 0, 2,
Loses:
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 2, 1, 0, 1,
2, 0, 1, 2, 2, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 2, 1, 0, 0, 0,
2, 1, 2, 2, 1, 1, 1, 1,
2, 2, 2, 2, 2, 0, 0, 0,
2, 1, 2, 2, 2, 1, 2, 0,
Win average time:
1135.0, 1614.0, 972.5, -, 1135.0, 1669.5, 876.5, 1273.0,
-, 1952.0, -, -, -, 1952.0, 2728.0, 2988.0,
-, 1480.0, 2475.0, -, -, 1572.5, 3057.5, 1315.0,
367.5, 367.5, 367.5, 2120.0, 367.5, 367.5, 367.5, 2587.5,
1988.0, 1428.5, 1925.5, -, 2047.0, 1237.0, 1566.5, 1576.5,
-, 1848.0, -, -, -, 2055.0, -, 2363.0,
-, -, -, -, -, 2785.0, -, 1662.5,
-, 1060.0, -, -, -, 1060.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, 5000.0, -, -,
-, -, -, -, 5000.0, -, 4200.0, -,
-, -, -, -, -, 4200.0, 4290.0, -,
-, -, -, -, -, -, -, 4030.0,
Lose average time:
1135.0, -, -, 367.5, 1988.0, -, -, -,
1614.0, 1952.0, 1480.0, 367.5, 1428.5, 1848.0, -, 1060.0,
972.5, -, 2475.0, 367.5, 1925.5, -, -, -,
-, -, -, 2120.0, -, -, -, -,
1135.0, -, -, 367.5, 2047.0, -, -, -,
1669.5, 1952.0, 1572.5, 367.5, 1237.0, 2055.0, 2785.0, 1060.0,
876.5, 2728.0, 3057.5, 367.5, 1566.5, -, -, -,
1273.0, 2988.0, 1315.0, 2587.5, 1576.5, 2363.0, 1662.5, -,
```

Figure 18.: RQ-SR - Gen1 Results

```
----------------------
Tournament 19
Wins:
1, 2, 1, 0, 1, 2, 2, 2,
0, 1, 1, 0, 0, 1, 1, 1,
1, 1, 1, 0, 1, 2, 2, 2,
2, 2, 2, 1, 2, 2, 2, 2,
1, 2, 1, 0, 1, 2, 2, 2,
0, 0, 0, 0, 0, 0, 0, 1,
0, 1, 0, 0, 0, 2, 0, 2,
0, 1, 0, 0, 0, 1, 0, 0,
Ties:
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 2, 0, 0,
0, 0, 0, 0, 0, 0, 2, 0,
0, 0, 0, 0, 0, 0, 0, 2,
Loses:
1, 0, 1, 2, 1, 0, 0, 0,
2, 1, 1, 2, 2, 0, 1, 1,
1, 1, 1, 2, 1, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 1, 2, 1, 0, 0, 0,
2, 1, 2, 2, 2, 0, 2, 1,
2, 1, 2, 2, 2, 0, 0, 0,
2, 1, 2, 2, 2, 1, 2, 0,
Win average time:
638.0, 1104.5, 638.0, -, 638.0, 1148.5, 989.5, 1213.0,
-, 875.0, 760.0, -, -, 875.0, 760.0, 2510.0,
705.0, 705.0, 705.0, -, 705.0, 1410.0, 2317.5, 1290.0,
496.0, 465.0, 475.0, 2120.0, 496.0, 465.0, 475.0, 2677.5,
2503.0, 1467.0, 988.0, -, 2649.0, 1420.0, 1401.5, 1321.0,
-, -, -, -, -, -, -, 3799.0,
-, 1415.0, -, -, -, 2117.5, -, 1377.5,
-, 1060.0, -, -, -, 1060.0, -, -,
Tie average time:
-, -, -, -, -, -, -, -,
-, -, -, -, -, 3170.0, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, -, -, -, -, -, -, -,
-, 3170.0, -, -, -, 4200.0, -, -,
-, -, -, -, -, -, 4290.0, -,
-, -, -, -, -, -, -, 4060.0,
Lose average time:
638.0, -, 705.0, 496.0, 2503.0, -, -, -,
1104.5, 875.0, 705.0, 465.0, 1467.0, -, 1415.0, 1060.0,
638.0, 760.0, 705.0, 475.0, 988.0, -, -, -,
-, -, -, 2120.0, -, -, -, -,
638.0, -, 705.0, 496.0, 2649.0, -, -, -,
1148.5, 875.0, 1410.0, 465.0, 1420.0, -, 2117.5, 1060.0,
989.5, 760.0, 2317.5, 475.0, 1401.5, -, -, -,
1213.0, 2510.0, 1290.0, 2677.5, 1321.0, 3799.0, 1377.5, -,
```

Figure 19.: RQ-SR - Gen2 Results