5-2017

A Study on the Effects of Mutation on Populations Using Strategies while Playing Iterative Prisoner's Dilemma

Ramses Romulus De Guzman Reyes
The University of Texas Rio Grande Valley

A STUDY ON THE EFFECTS OF MUTATION ON POPULATIONS USING STRATEGIES WHILE PLAYING ITERATIVE PRISONER'S DILEMMA

A Thesis

by

RAMSES ROMULUS DE GUZMAN REYES

Submitted to the Graduate College of
The University of Texas Rio Grande Valley
In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2017

Major Subject: Computer Science

A STUDY ON THE EFFECTS OF MUTATION ON POPULATIONS USING STRATEGIES WHILE PLAYING ITERATIVE PRISONER'S DILEMMA

A Thesis
by
RAMSES ROMULUS DE GUZMAN REYES

COMMITTEE MEMBERS

Dr. Wendy Lawrence-Fowler
Chair of Committee

Dr. Richard Fowler
Committee Member

Dr. Emmett Tomai
Committee Member

May 2017

ABSTRACT

Reyes, Ramses Romulus De, A Study on the Effects of Mutation on Populations Using Strategies While Playing Iterative Prisoner’s Dilemma. Master of Science (MS), May, 2017, 58 pp., 25 tables, 14 figures, reference, 13 titles.

This thesis examines the effects different types of mutation and mutation rates have on populations using strategies while playing the Iterative Prisoners Dilemma (IPD). The system used in order to conduct this study was used in Leas et al. (2016), which uses genetic algorithms as a means of studying memory and its impact on populations playing IPD. For this study, experiments are organized into three different environments: Control, Static and Dynamic. The Control Environment focuses on analyzing the system and forming initial results. The Static Environment focuses on studying the effects of different rates on populations using strategies while playing IPD, while the Dynamic Environment analyzes the effects of different mutations have on populations using strategies while playing IPD.

DEDICATION

The completion of this thesis would not have been possible without the support of my advisors, previous advisor, Dr. Laura Grabowski, and by current advisor, Dr. Wendy Laurence Fowler. My friends Richard Cantu, Javier Marroquin, Aaron Chiriku, Corey Muniz, Jesus Aguerro, Amber Cavasos and all the people who helped me get where I am today. My parents Rodolfo Villaluz Reyes and Edna De Guzman Reyes for their love and support and my sisters, Sheena Valrie Reyes and Sidney Charm Reyes for being there for me. Thank you for all you have done. It’s something I will always remember.

ACKNOWLEDGMENTS

I will always be grateful for my previous advisor Dr. Laura Grabowski, for her kindness and inspiration and my current advisor, Dr. Wendy Laurence Fowler for her wisdom and organization. Both were excellent mentors that I am proud to have worked with. My thanks go to Dr. Michael Wiser for the foundation of this project, Mr. David Kurtley and Jesus Garcia their knowledge and assistance in reference to the system and finally, my friend, Mikeala Leas for her support and help throughout this thesis.

My thanks go to Dr. Robert Schweller, Dr. Emmett Tomai and the rest of the faculty and staff in the University of Texas Rio Grande Valley Computer Science Program. Without them I would have never become the person I am today.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Page

CHAPTER I

INTRODUCTION AND BACKGROUND

Motivation

An organism's strategy is defined as "a behavioral phenotype, i.e. a specification of what an organism will do in any situation it finds itself in" [1]. As such, strategies tend to change over the course of time and circumstance. Mutations on the strategies used within different types of behavior, such as hunting, displaying or breeding, can either be beneficial or detrimental over the course of an organism's lifetime. These changes can affect future generations [2].

Game theory holds potential insight to the subject of mutations dealing with strategies and the effect they have on a population of organisms. The subject of game theory, in tandem with virtual environments using evolutionary methods such as genetic algorithms, gives us the ability to study the subject of mutation and its effects on populations with the ability to implement strategies.

Background on Evolution

The Evolutionary Process and the Importance of Mutation

The process of evolution is composed of three fundamental components. The first is reproduction. In order for organisms to evolve, they must be able to create offspring which can inherit traits of the previous generation. Reproduction can be organized into two different categories, sexual and asexual. Sexual reproduction is a process which uses the genetic material of two different individuals (parents) in order to create offspring different from the previous generation. Asexual reproduction, alternatively, involves using the genetic material of only one individual.

The second component of evolution is selection. In order for evolution to occur, a way of selecting the organisms within a generation (fitness function) and allowing these organisms to breed is needed. Examples of selection include mating rituals, such as contests of strength as seen in walruses and lions and the use of display as seen in peacocks.

The third component of evolution is variation. Variation allows organisms to change through successive generations. Examples of variation can be seen in environments such as the Galapagos where birds evolved different species, each specializing in a different food source. Mutations are important as it allows for variation to occur, fueling the evolutionary process. As Darwin stated in his work, *The Origin of Species*, if it could be proven that complex organs in nature could be formed without “numerous, successive, slight modifications”, the theory of evolution would not exist [3].

Problems with Natural Evolution

Over the course of time, evolution has produced a number of complex structures. For example, eyes and ears are a result of the evolutionary process [3].Although information about the evolutionary process of different structures is important; gaining this sort of information is an arduous task. In order to understand details about the evolution of a structure, one must have either a record of the feature as it goes through the evolution or the ability to observe the structure as it evolves to its current state. Perfect records of organisms are difficult to find and therefore observe. Many organisms also have lives long enough that observing a number of generations would take a significant amount of time.

Although the evolution of physical features occurs over a long period of time, many of these changes could be observed through remnants and remains, unlike behavioral changes which are studied through the observation of live specimens. The challenge becomes more difficult when looking at the natural evolution of strategies. Strategies do not leave a tangible record to study. The artifacts we use to study biological evolution (fossil records, prints) are unsuitable for data analysis in reference to strategies. To address this challenge, researchers employ methods prominent in the field of Evolutionary Computation.

Evolutionary Computation

The field of evolutionary computation is an important subfield under the umbrella of artificial intelligence and includes fields of research such as digital evolution and evolutionary methods such as genetic algorithms. The subject of evolutionary computation involves the use of algorithms with characteristics based on Darwinian evolution in order to solve complex problems. Evolutionary algorithms, in tandem with computer simulations, give rise to evolutionary computational systems that allow us to emulate and observe evolutionary processes in a measurable timescale. This is in contrast to the timescale for biological systems.

Evolutionary algorithms involve using a set of potential solutions to a specific problem. These solutions are put through an iterative process of evaluation and selection. Through each iteration of this process, mutations can occur. Over the course of successive generations, constant selection of the best evaluated individuals lead to potential solutions that are different than what was set originally. This process of constant selection of the best gives rise to a system that is flexible, adaptable, and robust [3].

Virtualization/Simulation

Though observing natural evolution has its problems, there is a potential solution to this issue which is through the use of virtual environments to simulate evolutionary phenomena. Through these virtual environments, we can make observations and glean insight into evolutionary processes. Using a simulated environment allows control over multiple attributes/factors dealing with the evolutionary process including population size, the environment, the individual genotype and phenotype, mutation rate, number of generations allowed, and amount of time involved over the course of the simulation. Virtual environments give us the ability to study evolution from many perspectives, giving us information that would be more difficult to observe in a living system. The ability to modify variables and attributes in an experiment allows for a deeper look into aspects of evolution.

Changes in variables and attributes an experiment can generate large amounts of information [4]. The processing and analyzing of this large quantity of data is simplified given the speed a computer has and its ability to manipulate and record information. This makes data much more accessible in comparison to natural experiments, as calculations done by a computer can be order of magnitudes faster than calculations done physically.

Virtualization allows for a platform in order to study evolution. Evolution computation provides the algorithms used in order to study evolution. Together both virtualization and evolutionary computation provide a way to study evolution in time scales faster than those that can be observed in a natural environment.

Background on Game Theory

Importance of Game Theory

Game theory is the study of decision making between two or more organisms, is discussed in a number of fields including Economics [5], Psychology [6], Biology [7], and Computer Science [4]. Game theory is important because as it provides an explanation to the behavior of many organisms [2]. Research in the field of game theory potentially holds answers to questions in social, economic and scientific contexts as the study of behavior is an important part of these subjects.

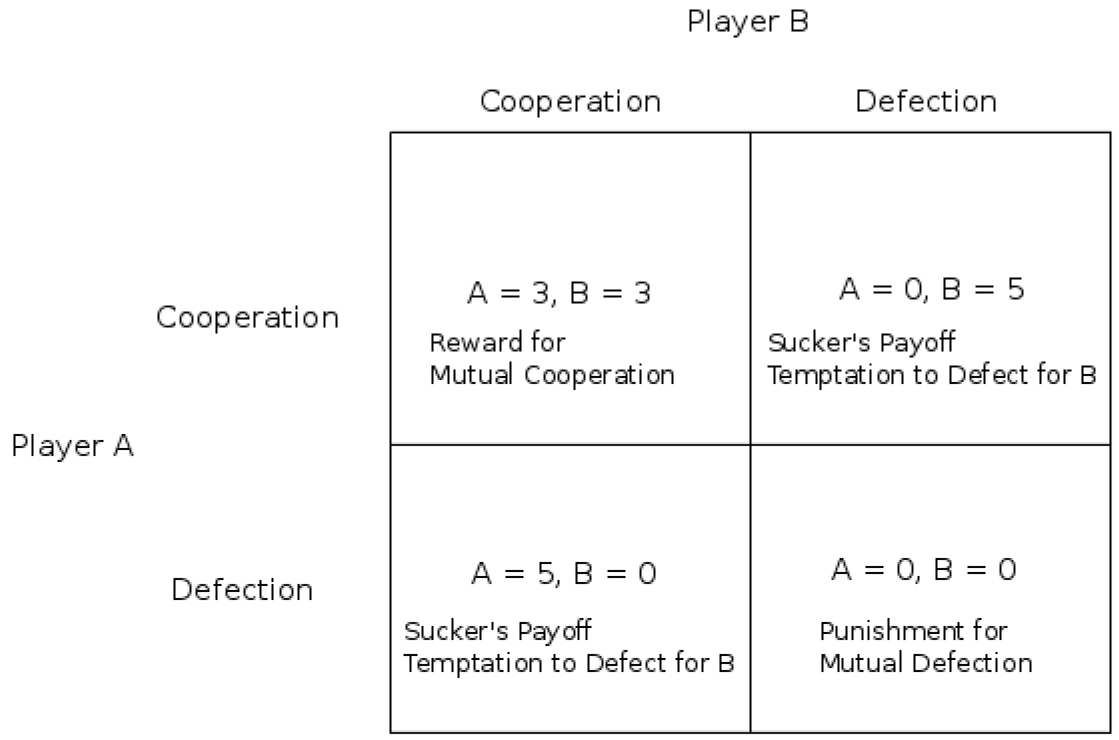
Strategies are often connected with games, as they affect the way players interact with a game. Strategies play a vital role in the field of game theory. Mutations within an individual's strategy can change not just the outcome of present events, but also its overall fitness which, over a number of generations, can affect entire populations.

Prisoner's Dilemma

One of the most well-known games studied in the field of game theory is the Prisoner's Dilemma(PD) formalized by Albert Tucker [8] . Since its creation, papers spanning various fields have been published such as the *Rational Cooperation in Finitely Repeated Prisoner's Dilemma* by Andreoni and Miller [9], which focuses on the aspect of cooperation, and *Effective Choice in the Prisoner's Dilemma* by Axelrod [10], which focus on the strategies used by individuals playing the game. An example of the payout matrix for PD is presented in figure 1. where rewards range from 0 to 5.

The Prisoner's Dilemma is played as follows:

- There are two players in this game.
- Each player is given a choice to Cooperate or Defect. (Note: Each player's choice remains private throughout the game)
- The reward each player receives depends on the choice made by each player.
    - Mutual cooperation gives the best total reward (split equally to both players).
    - Mutual defection gives the worse total reward (split equally to both players).
    - If one player cooperates and the other player defects,
        - The player who cooperates gets the worse individual reward, while
        - The player who defects gets the best individual reward.



		Player B	
		Cooperation	Defection
Player A	Cooperation	A = 3, B = 3 Reward for Mutual Cooperation	A = 0, B = 5 Sucker's Payoff Temptation to Defect for B
	Defection	A = 5, B = 0 Sucker's Payoff Temptation to Defect for B	A = 0, B = 0 Punishment for Mutual Defection

FIGURE 1. THE PAYOUT MATRIX FOR THE PRISONER'S DILEMMA

Iterative Prisoner's Dilemma

The Iterative Prisoner's Dilemma Model (IPD) is a concept in which two players play the Prisoner’s Dilemma (PD) more than once. This was introduced by Axelrod in the 1980s [10]. This study involved a tournament consisting of 14 different programs, which competed against each other through the PD. This competition was run in round robin order ensuring that each program would compete against all others. After each program finishes its round, the points of each of the programs are tallied and the game begins again. In this study, each program has a strategy it uses in order to deal with its opponents. Some of these programs used simple strategies like always defecting (Always Defect) in the presence of its opponent, while others do the opposite and always cooperate (always cooperate). Others were more complex and try to predict their opponent’s actions.

Multiple iterations add a layer of complexity not found in the original PD. Memory in the IPD is a big factor as it allows a player to remember its opponent’s actions. In addition, strategies require varying amounts of memory. For example, organisms using strategies that purely execute one action such as Always Cooperate or Always Defect do not require any amount of memory as no matter what its opponent does, its opponent’s actions will not affect the future decisions of these organisms. On the other hand individuals with strategies that base their decisions on their opponent's previous actions require memory.

The results in Axelrod’s experiment showed that most successful strategy was a simple strategy called Tit for Tat [10]. This strategy required only enough memory to remember its opponent’s previous action. Tit for Tat starts cooperating with its opponent .After its initial cooperation it then bases its next action on the opponent’s previous action. If its opponent previously defected, Tit for Tat will defect, otherwise it cooperates. With the results of this experiment Axelrod concluded that in order for strategies to become successful in IPD, it must have four characteristics: nice, punishing, forgiving, and consistent.

Axelrod [10] proposed that first, a strategy must be “nice”. A nice strategy does not defect before its opponent does. Most of the high scoring strategies in this Axelrod’s experiment were nice strategies. The next characteristic is “punishing”. A successful strategy must be able to retaliate against defection. Strategies that are nice, but not punishing are taken advantage of by strategies that tend to defect. A good example of a strategy that is nice, but not punishing is the strategy Always Cooperate. This strategy never defects and therefore is susceptible to strategies like Always Defect, which never cooperates. The third characteristic is “forgiving”. A successful strategy must have the ability to forgive. Tit for Tat is a good example of a strategy that forgives. It punishes when its opponent defects, but cooperates again if the opponent cooperates. The aspect of forgiveness stops long runs of revenge and counter revenge, which can reduce points for both players. The final characteristic is “consistent”. Strategies must remain consistent because randomness in a strategy can trigger retaliatory behavior from other players.

Methodology

System Overview

The system I will be using for my experiments was created by Leas, Edolson, Annis and Nahum [4]. This system uses genetic algorithms in order analyze populations that use strategies as they play IPD. Each experiment in this thesis consists of a population of organisms using strategies as they compete against one another.

Each organism is composed of two components: the organism’s bits of memory, and its decision list. In this system, an organism’s genome is defined as a binary representation of the organism, specified by the memory and the decision list. In turn, an organism’s size is defined by the length of its genome. The strategy an organism uses is defined by its genome, i.e., the decisions it makes based on its memory. Memory is used to record an opponent's actions while the decision list is used to determine a course of action using data the organism has collected and stored in its memory.

In this system, each organism plays Iterated Prisoner's Dilemma (IPD) against its competitors in a predefined number of rounds. An organism's payout (total reward) is calculated by the running total number of points it scores during each round. Each organism is then ranked based on its fitness as defined as the total number of points won during the competition. A select percentage of the fittest organisms are allowed to breed the next generation. Mutations can be applied to these organisms with a goal of increasing diversity of the next generation, allowing for evolution.

Organism Structure

An organism's genome is composed of two different sections: memory and its decision list. Memory is used by the organism in order its opponents previous actions. The decision list is used by the organism to determine a course of action using the memory an organism has collected.

Memory in this system is binary, with True (Cooperation) as 1 and False (Defection) as 0. Memory also dictates the size of an organism's decision list. An organism's decision list is 2^m where m is the number of bits of memory an organism has.

An organism makes decisions by calculating the binary representation of its memory and mapping that value to the decision list. It then saves new information by deleting its oldest memory bit and shifting all bits of memory to the left by one space, if it has more than one bit of memory, and then it registers its opponent's action as the least significant bit of memory. In this system, organism's strategy is synonymous to its genome as it represents the actions an organism will take (decision list) given different circumstances (memory).

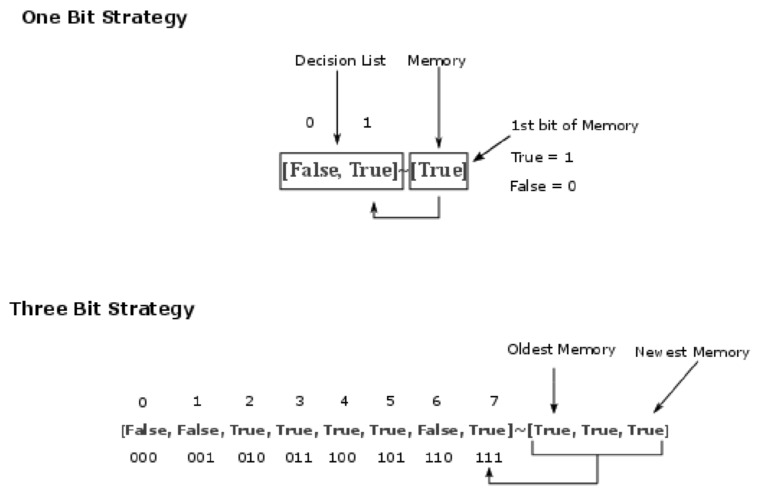


FIGURE 2.A DESCRIPTION OF STRUCTURE OF AN ORGANISM UNDER THE SYSTEM USED BY LEAS.

Purpose

In this thesis, I explore the effects of different types of mutation and the effects of mutation rates on populations using strategies in virtual environments as presented by Leas [4].

Hypothesis

Given the results specified in Axelrod's Tournament [10], it appears that the Tit for Tat strategy would do well regardless of any changes in mutation rate associated with memory, especially if a population was seeded with Tit for Tat as its initial strategy. On the other hand, the possibility of other strategies exerting their dominance in the same population cannot be ignored. However, given the cost of memory has a potential relationship with the strategies an organism uses (Leas et al. 2016), if the cost per bit of memory is low, then organisms incline to evolve more complicated strategies, allowing for multiple bits of memory.

For populations under the influence of low mutation rates, it will be difficult it will be difficult to observe an emergence of new strategies as low mutation rates decrease rate of evolution within a population. Though there is a possibility of an emergence of various strategies within a population under low mutation rates.

For populations under the influence high mutation rates a number of new strategies should emerge. The possibility exists that a population of organism under high mutation rates could contain a small amount of new strategies.

To test this hypothesis, I defined a set of experiments and created three environments control, static, and dynamic.

Control Environment Experiments

The focus of the experiments in the control environment is to analyze the effects of zero mutation rates on populations using strategies while playing Iterative Prisoner's Dilemma (IPD). This environment acts as a testing ground for the system and as a baseline for later experiments. There are five experiments run in the control environment. The first two experiments involve setting all mutations to the rate, zero. The first experiment is randomly seeded with strategies, while the second experiment is seeded with one strategy. The last three experiments involve testing populations where one of the mutations used by the system is set to the rate, zero, while the rest of the mutations are set the rate, .05.

Static Environment Experiments

The focus of static environment experiments is to analyze the effects of different mutational thresholds on populations using strategies while playing IPD. Each experiment under this environment will test a specific rate of mutation. All mutation types under this environment will subject to the same rate that is constant throughout each experiment. The rates that are tested in this environment are .01, .025, .05, .75, and .1 .

Dynamic Environment Experiments

The experiments in the dynamic environment are designed to analyze the effects of different types of mutation on populations using strategies while playing IPD. For each experiment in this environment only one type of mutation is changed while the rest stay at a constant rate. There are six experiments under this environment; two for each type of mutation which are low (.01) and high (.1) mutation rates.

CHAPTER II

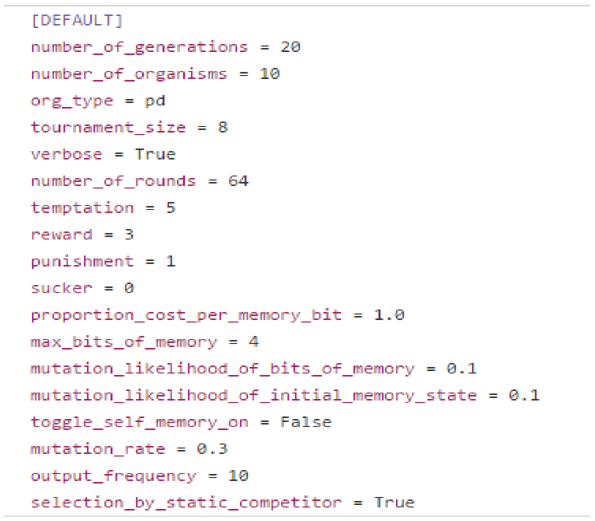
METHODS

System Design

This system is composed of different files which help in implementing various tests used to analyze memory under IPD. For this thesis, files that compose the system are located a Hyper Performance Compute Cluster (HPCC) user account located in Michigan State University

Files used in this thesis include:

The pd_config.init is the configuration file used to set up the experiments for this thesis. The file contains the variables needed to calibrate an experiment. This system uses the pd_config.init file as input for the pd_qsub_generator.py file which was used in order to run experiments. A sample pd_config.init file is included in figure 2.



```
[DEFAULT]
number_of_generations = 20
number_of_organisms = 10
org_type = pd
tournament_size = 8
verbose = True
number_of_rounds = 64
temptation = 5
reward = 3
punishment = 1
sucker = 0
proportion_cost_per_memory_bit = 1.0
max_bits_of_memory = 4
mutation_likelihood_of_bits_of_memory = 0.1
mutation_likelihood_of_initial_memory_state = 0.1
toggle_self_memory_on = False
mutation_rate = 0.3
output_frequency = 10
selection_by_static_competitor = True
```

FIGURE 3: AN EXAMPLE OF THE PD.CONFIG.INIT FILE AND ITS CONFIGURATIONS

For the series of experiments in this documented in this thesis, the following settings were modified.

- number_of_generations – sets the number of generations for each experimental run.
- number_of_organisms – sets the number of organisms for each run
- org_type – sets the type of organism for experiment. For these experiments, organisms are set to pd.
- number_of_rounds – the number of rounds that each organism participates in
- temptation – the amount of points gained by an organism when it defects and its opponent cooperates
- reward – the amount of reward points by an organism when both it and its opponent mutually cooperate
- punishment – the amount of points gained by an organism when both it and its opponent mutually defect
- sucker – the amount of reward points by an organism when it cooperates and its opponent defects
- proportion_cost_per_memory_bit – amount of cost per memory bit
- max_bits_of_memory – the number of memory bits allowed for each organism
- mutation_likelihood_of_bits_of_memory – sets the mutation rate for an organism's amount of memory
- mutation_likelihood_of_initial_memory_state – set the mutation rate for an organism's memory
- toggle_self_memory_on – toggles whether an organism can remember their own choices

- mutation_rate – sets the general mutation rate for the genetic algorithm. This also affects decision list mutations.
- selection_by_static_competitor – if set to True allows for organisms to compete with set strategies.

The pd_qsub_generator.py is the python script that allows the system to use the Hyper Performance Compute Cluster (HPCC) for calculations.. Execution of this file sends an instance of an experiment to the HPCC in order to execute. When the execution of an experiment is finished, data is sent back to the user in the form of a directory.

The main.py file contains the critical functions needed in order to run the system, including the methods and specifications for the genetic algorithm.

The pd_beaker_one_strategies_df.py is the first of many analysis files that are used to extract data from the raw information given by the system. Output data from the execution of this file is used as input for other pd_beaker files.

The pd_beaker_two_strategies_df_r.r file is an "r" file that uses data from the previous pd_beaker files and with it, creates a graphical representation that users can utilize to better understand experimental data. The output of this file is a pdf file that shows a line graph containing the frequency of strategies over the course of generations. Using this file helps give insight into strategies, their rise and fall over time.

Focus

The central purpose of these experiments is to gain insight into the behavior of strategies under various mutation thresholds and the effect that different mutational rates have on populations using strategies while playing IPD.

Setup

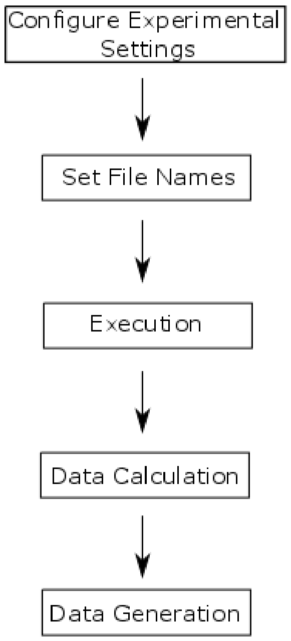


FIGURE 4.A FLOWCHART SHOWING THE PROCESS OF RUNNING EXPERIMENTS UNDER THE SYSTEM USED IN BY LEAS [4].

Note: Descriptions for each state of this flowchart is described in the appendix.

Experimental Design

The experiments for this thesis are organized into three different types called environments which are control, static, and dynamic. The control environment act as a testing ground for the system and as reference for measuring the effects of mutation rates have on populations using strategies under Static and Dynamic environments. The Static environment focuses on analyzing how different mutational thresholds affect populations using strategies while under the influence of IPD. The Dynamic environment, on the other hand, focuses on the different mutations themselves and the role they play in the experimental environment.

For all environments, mutation rates ranging from low (.01) to high (.1) are tested. For each environment, the number of organisms and generations are set to 500. Nine variables (org_type, tournament_size, verbose, number_of_rounds, temptation, reward, punishment, sucker, and output frequency respectively) were kept standard in reference to the experiments done by Leas et al. (2016) [4]. The variable proportion_cost_per_memory_bit was kept at zero, allowing for the evolution of more complex strategies, and max_bits_of_memory was set to four, allowing for the evolution of strategies using up to four bits of memory. The variable toggle_self_memory_on was set to "False" for each experiment. This "False" setting did not allow an organism to remember its previous actions.

All experiments in this thesis use the default settings specified by table 1, with the exception of the first control experiment started with a random initial population. Note that for all experiments in this thesis, each experiment consists of five runs, which are used for data analysis.

TABLE 1: DEFAULT SETTINGS FOR ENVIRONMENT EXPERIMENTS

Environment Experiments	Default Settings
number_of generations	**500**
number of organisms	**500**
org_type	**pd**
tournament_size	**8**
verbose	**True**
number_of_rounds	**64**
temptation	**5**
reward	**3**
punishment	**1**
sucker	**0**
proportion_cost_per_memory_bit	**0**
max_bits_of_memory	**4**
toggle_self_memory_on	**False**
toggle_self_memory_on	**False**
selection_by_static_competitor	**False**
output_frequency	**10**
Initial Population	**[False,True]~[True]**

Control Environment

The focus of the Control experiments is to gain a better understanding of strategies under conditions where mutation is not allowed. For the first experiment, all three types of mutation are set to the rate zero, while the initial population is seeded with random strategies. This allows for better analysis in reference to strategies in a population considering the fact that the only changes occurring within this experiment involves the initial random seeding of strategies within the initial population of the experiment. This exclusion limits the number of strategies generated by the system simplifying experimental results.

For the second experiment, all three types of mutation are set to the rate zero, and the initial population is set to the Tit for Tat ([False,True]~[True]) strategy. For this experiment organisms compete with a set of preset competitors specified by Leas et al. (2016). This experiment was the simplest and most predictable of the control experiments. There was no mutation involved in any generation and with the initial population set as one strategy (Tit for Tat) allows for the growth of a populations using only one strategy throughout each generation.

For each of the last three experiments, one of the three types of mutation is set to rate zero, while the other two are set to .05. Like the second experiment, the initial population is seeded with the Tit for Tat strategy. The purpose of these three experiments is to observe the effects on populations using strategies in an environment where one of three types of mutation (Bit, Memory, or General) is not applicable. The information gained from each of the Control Environment Experiments is used as reference for the Static and Dynamic Environments.

TABLE 2. THE CONFIGURATION SETTINGS USED IN THE CONTROL ENVIRONMENT – 1ST AND 2ND EXPERIMENTS

Control Environment Experiments	1st Experiment	2nd Experiment
Initial Population	**Random**	**[False,True]~[True]**
mutation_likelihood_of_bits_of_memory	**0**	**0**
mutation_likelihood_of_initial_memory_state	**0**	**0**
mutation_rate	**0**	**0**

TABLE 3: THE CONFIGURATIONS SETTINGS USED FOR CONTROL ENVIRONMENT EXPERIMENTS FOR BIT, MEMORY AND GENERAL MUTATION RATES

Control Enviroment Experiments	Bit	Memory	General
mutation_likelihood_of_bits_of_memory	**0**	**0.05**	**0.05**
mutation_likelihood_of_initial_memory_state	**0.05**	**0**	**0.05**
mutation_rate	**0.05**	**0.05**	**0**

Static Environment

In the Static Environment Experiment, all mutation types are at the same value. The focus of this part of the environment is to test mutational thresholds and the effect they have on strategic population playing IPD. There are five experiments in this environment; one for each rate. The rates of mutation tested in this environment are low (.01, .025), medium (.05) and high (.075, .1)

TABLE 4. THE CONFIGURATION SETTINGS USED IN THE STATIC ENVIRONMENT EXPERIMENTS .01 AND .1

Static Environment (Low)	0.01	0.025
mutation_likelihood_of_bits_of_memory	**0.01**	**0.025**
mutation_likelihood_of_initial_memory_state	**0.01**	**0.025**
mutation_rate	**0.01**	**0.025**

TABLE 5. THE CONFIGURATION SETTINGS USED IN THE STATIC ENVIRONMENT EXPERIMENTS .025 AND .075

Static Environment (High)	0.075	0.1
mutation_likelihood_of_bits_of_memory	**0.075**	**0.1**
mutation_likelihood_of_initial_memory_state	**0.075**	**0.1**
mutation_rate	**0.075**	**0.1**

TABLE 6. THE CONFIGURATION SETTINGS USED IN THE STATIC ENVIRONMENT EXPERIMENT .05

Static Environment (Medium)	0.05
mutation_likelihood_of_bits_of_memory	**0.05**
mutation_likelihood_of_initial_memory_state	**0.05**
mutation_rate	**0.05**

Dynamic Environment

In the Dynamic Environment experiment, unlike the Static Environments which keep mutation rates constant with each other, variation is allowed within the range of the mutation rates. The focus for this environment is to study the how different mutations affect populations using strategies while playing the IPD. This environment analyzes the effects of the three mutation types used in this system: Bit, Memory and General. For all mutation types two experiments are run: low (.01) and high (.1). Configurations for each experiment are shown by the tables below.

Bit Mutation

TABLE 7. THE CONFIGURATION SETTINGS USED IN THE DYNAMIC ENVIRONMENT FOCUSING ON THE VARIABLE MUTATION_LIKELIHOOD_OF_BITS_OF_MEMORY

Dynamic Environment (Bit Mutation)	Low(.01)	High(.1)
mutation_likelihood_of_bits_of_memory	**0.01**	**0.1**
mutation_likelihood_of_initial_memory_state	**0.05**	**0.05**
mutation_rate	**0.05**	**0.05**

Memory Mutation

TABLE 8. THE CONFIGURATION SETTINGS USED IN THE DYNAMIC ENVIRONMENT FOCUSING ON THE VARIABLE MUTATION_LIKELIHOOD_OF_INITIAL_MEMORY_STATE

Dynamic Environment (Memory Mutation)	Low(.01)	High(.1)
mutation_likelihood_of_bits_of_memory	**0.05**	**0.05**
mutation_likelihood_of_initial_memory_state	**0.01**	**0.1**
mutation_rate	**0.05**	**0.05**

General Mutational Rate

TABLE 9. THE CONFIGURATION SETTINGS USED IN THE DYNAMIC ENVIRONMENT FOCUSING ON THE VARIABLE MUTATION_RATE

Dynamic Environment (General Mutation)	Low(.01)	High(.1)
mutation_likelihood_of_bits_of_memory	**0.05**	**0.05**
mutation_likelihood_of_initial_memory_state	**0.05**	**0.05**
mutation_rate	**0.01**	**0.1**

CHAPTER III

RESULTS

Control Experiments

1st Control Experiment

Table 10 shows the results of most common strategies observed from five runs under conditions specified by the first control experiment. Under these conditions we observe that Tit for Tat ([False, True]~[True]) is the most common strategy as it is prominent in all five runs. The second most common strategy is also a version of Tit for Tat (False, True, False, True]~[False/True, True]) with the difference of an extra bit. The third was a strategy with four bits of memory, which was observed on one of the runs. In this experiment the random seeding of strategies in the initial population of the experiment adds the only instance of diversity within a population in an experiment with no mutation.

TABLE 10. THE MOST COMMON STRATEGIES OBSERVED UNDER THE 1ST EXPERIMENT. USING UNFIXED COMPETITION SELECTION AND RANDOM POPULATION

Control - 1st Experiment: Unfixed Selection/Random Population	# of Runs
[False, True]~[True]	**5**
[False, True, False, True]~[True, True]	**2**
[False, True, True, False, False, False, True, False, True, True, True, True, False, True, False, True]~[True, True, True, True]	**1**

Figure 6 shows the results of the 1st experiment's 4th run. In this run, Tit for Tat (one bit) was the most prominent strategy in the first few generations of each run. As time passes the number of organisms adopting the Tit for Tat (one bit) strategy dwindled, while organisms adopting the Tit for Tat (two bit) and the four-bit strategy increased in number. After 500 generations, the most prominent strategy in the population was a four-bit strategy followed by Tit for Tat (two bit), while Tit for Tat (one bit) was close to extinction. Through these observations, I can infer as strategies increase in complexity, the chance of gaining advantages over other strategies increases.

Given that mutation is zero under this experiment, the number of strategies created during the initial generation sets the diversity of strategies within a population. Strategies that reach extinction have no chance of appearing in future generations because of the lack of mutation. Strategies that survive the initial few generations have less competition. As time goes on the amount of strategies available decrease until the population stabilizes, leaving organisms to adopt strategies that do not conflict with one another.

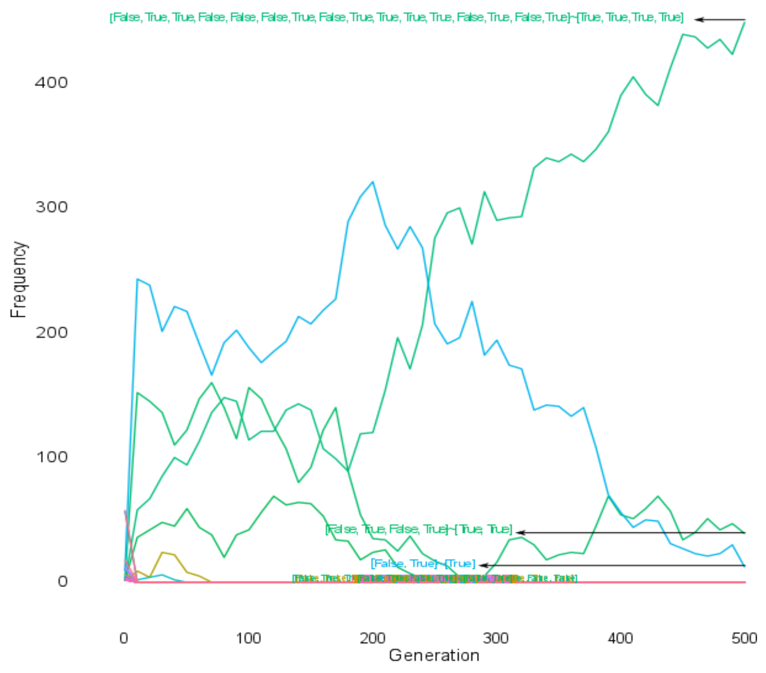


FIGURE 5. A GRAPH OF THE MOST COMMON STRATEGIES OBSERVED UNDER THE 1ST EXPERIMENT'S 4TH RUN

2nd Control Experiment

In the 2nd control experiment, table 11 shows the results of most common strategies observed from five runs under conditions specified by the first control experiment. As shown in Figure 7, the only existing strategy observed under these conditions is the initial seeded strategy of Tit for Tat. Results in this experiment were predictable as there was no mutation and only one strategy was seeded in the initial population. Therefore, there is no competition between any organisms because they all used the same strategy throughout each generation.

TABLE 11. THE MOST COMMON STRATEGIES OBSERVED UNDER THE 2ND EXPERIMENT. USING STATIC COMPETITION SELECTION AND RANDOM POPULATION

Control – 2nd Experiment: Static Selection/Static Population	# of Runs
[False, True]~[True]	5

3rd Control Experiment – 0 Bit Mutation

In the 3rd control experiment, as shown in table 12, the lack of Bit Mutation caused the organisms on this experiment to keep the same number of bits throughout each generation. The only changes that occurred throughout this experiment were differences in the decision list. From the most common strategies observed under this experiment, Tit for Tat ([False, True]~[True]), Always Cooperate ([True, True]~[True]), Always Defect ([False, False]~[True]), and ([True, False]~[True]), the first three strategies were observed in all five runs while the fourth was only observed in two runs.

TABLE 12. THE MOST COMMON STRATEGIES OBSERVED UNDER THE CONTROL EXPERIMENT TESTING BIT MUTATION

Control – Bit Mutation Rate: 0 - Unfixed Selection/ Static Population	# of Runs
[False, True]~[True]	5
[True, True]~[True]	5
[False, False]~[True]	5
[True, False`]~[True]	2

In the 3rd control experiment as shown figure 8, Tit for Tat tends to be the dominant strategy throughout each run, which is followed by Always Cooperate and Always Defect. In IPD a relationship exists between these three strategies [11]. Tit for Tat is resistant to invasion against Always Defect, but neutral to Always Cooperate. Always Cooperate is neutral against Tit for Tat, but through genetic drift, Always Cooperate can invade a population of organisms using Tit for Tat. On the other hand, Always Cooperate is susceptible to invasion by organisms using Always Defect. In this experiment, Tit for Tat was the most prominent strategy overall and Always Cooperate was the second most prominent. A sudden drop in organisms using Tit for Tat is followed by a spike in organisms using Always Defect. This growing population of organisms using Always Defect is attacked by users of Tit for Tat. The population stabilizes as Tit for Tat takes dominance again and is followed by a smaller population of Always Cooperate. Always Defect present within a few organisms in the population.

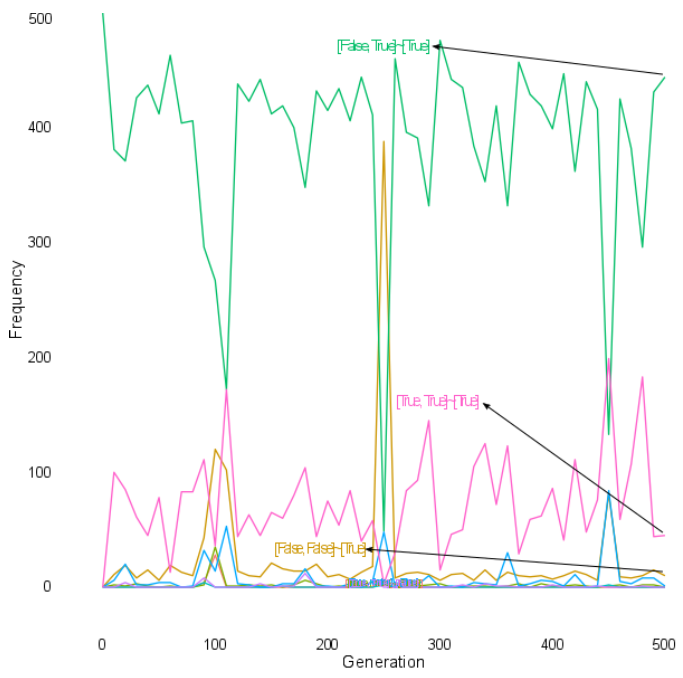


FIGURE 6: A GRAPH OF THE MOST COMMON STRATEGIES OBSERVED IN THE CONTROL EXPERIMENT TESTING BIT MUTATION USING THE 4TH RUN

4th Control Experiment – 0 Memory Mutation .

In the 4th control experiment, as shown in table 13, one bit strategies are the most common types of strategies observed, followed by the two and three bit strategies respectively. Because the rates for the Bit Mutation and General Mutation are not low, more complex strategies have a greater chance of evolving. The lack of Memory Mutation allows for a population that remembers past events with 100 percent accuracy. Small changes in memory can potentially affect the outcome of a strategy, and under these experimental conditions, that variable is removed.

The seeding of Tit for Tat within each initial population of sets the memory for the entire experiment because this strategy always starts by cooperating. Strategies that defect can still evolve through decision list mutations, but because the population contains Tit for Tat, these strategies have difficulty lasting a few generations after evolving. Organisms adapting strongly defecting strategies tend to go extinct under this circumstance. Organisms that mostly defect in their decision list can exist if they cooperate initially. Initial cooperation discourages defection in an environment, where defectors are punished. Strategies that defect initially do not last long as the first defection can cause retaliation, which slows down the growth of organisms adopting this type of strategy. Overall this leads to an environment of cooperators, which helps explain success of this experiment.

TABLE 13. THE MOST COMMON STRATEGIES OBSERVED UNDER THE CONTROL EXPERIMENT TESTING MEMORY MUTATION

Control – Memory Mutation Rate: 0 - Unfixed Selection/ Static Population	# of Runs
[False, True]~[True]	3
[True, True]~[True]	3
[False, False]~[True]	2
[True, False`]~[True]	1
[False, True, True, True]~[True, True]	1
[False, True, False, True]~[True, True]	3
[False, False, True, True]~[True, True]	1
[False, False, False, True]~[True, True]	1
[False, False, True, True, True, True, False, True]~[True, True, True]	1
[False, False, False, True, True, True, False, True]~[True, True, True]	1
[False, False, True, True, False, False, False, True]~[True, True, True]	1

5th Control Experiment – 0 General Mutation

In the 5th control experiment as shown in table14, the results of this experiment were the same as the results specified by the 2ndcontrol experiment. In the system, general mutation acts as the primary mutation check that specifies if an component of the organism can mutate. General Mutation also dictates when decision list mutations can occur. Bit Mutation and memory mutation are only allowed to occur after general mutation and cannot happen otherwise. This explains why this experiment only has one strategy.

TABLE 14. THE MOST COMMON STRATEGIES OBSERVED UNDER THE CONTROL EXPERIMENT TESTING GENERAL MUTATION

Control – General Mutation 0 - Unfixed Selection/ Static Population	# of Runs
[False, True]~[True]	5

Static Environment Experiments

Low Mutation Rate - .01

In the .01 static environment experiment as shown in table 15, the most common strategies observed are one bit strategies which are Tit for Tat and Always Cooperate, with Reverse Tit for Tat and two bit Tit for Tat occurring in one of the runs. Although Always Defect did evolve in some of the runs, data showed that the number of organisms that adopted this strategy were few. Contrary to results in the previous Control Experiments, Reverse Tit for Tat (**[True, False]~[True]**) did well in one of the runs. As observed in the prior results, this strategy occurred in less runs than Always Defect. It was also surprising that Tit for Tat (two bit) was able to evolve as this experiment had the lowest set of mutational rates outside of the Control experiments.

TABLE 15. THE MOST COMMON STRATEGIES OBSERVED UNDER THE STATIC EXPERIMENT USING .01 MUTATIONAL RATES

Static Environment (Low) - .01 Mutational Rate	# of Runs
[False, True]~[True]	**5**
[True, True]~[True]	**5**
[True, False]~[True]	**1**
[False, True, False, True]~[True, True]	**1**

Low Mutation Rate - .025

In the .025 static environment experiment as shown in Table 16, the most common strategies observed where similar to the previous experiment, i.e. Static Environment using Low Mutational Rate (.01). In this experiment, Tit for Tat, Always Cooperate, and Always Defect were the most prominent strategies occurring in all runs, while Reverse Tit for Tat occurred in three runs. Unlike the prior Static Environment experiment with lower mutation rates, two bit strategies did not evolve under current experimental conditions, though with higher mutation rates more complex strategies are more likely to evolve.

TABLE 16. THE MOST COMMON STRATEGIES OBSERVED UNDER THE STATIC EXPERIMENT USING .025 MUTATIONAL RATES

Static Environment (Low) - .025 Mutational Rate	# of Runs
[False, True]~[True]	**5**
[True, True]~[True]	**5**
[True, False]~[True]	**3**
[False, False]~[True]	**5**

In Figure 9, as observed in the previous Control experiments, we see the relationship between the three common strategies, Tit for Tat, Always Cooperate and Always Defect. According to these results, over a span of generations the less common strategy [True, False]~[True] tends to occur with a similar frequency to Always Defect. This could be explained by its nature, unlike Tit for Tat, [True, False]~[True] initially defects and takes advantage of cooperators, similar to Always Defect.[True, False]~[True] also cooperates on remembrance of defection, which entices forgiveness in organisms using strategies similar to Tit for Tat. This strategy invites retaliation from organisms using strategies similar to Always Defect. Its smaller number of occurrences can be related to its relationship with Always Defect, while its score can be linked to its similarity to Always Defect.

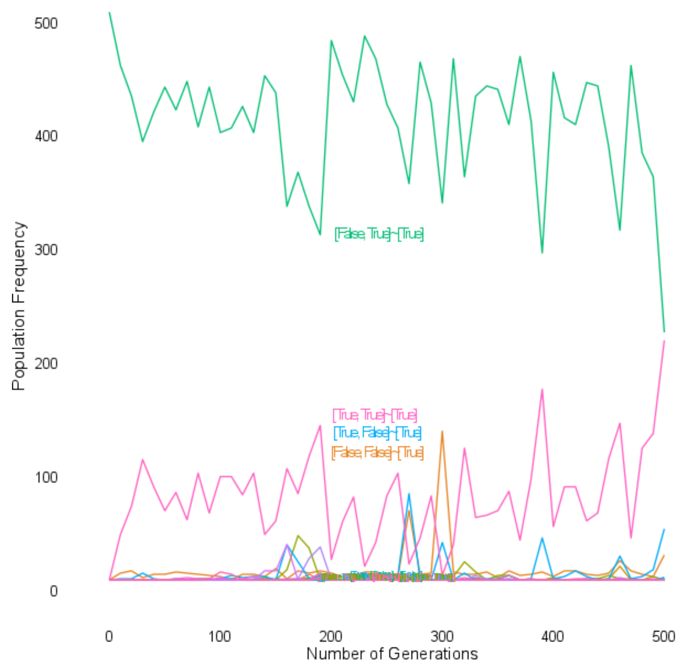


FIGURE 7. A GRAPH OF THE MOST COMMON STRATEGIES OBSERVED IN THE UNDER THE STATIC ENVIRONMENT EXPERIMENT USING .025 MUTATIONAL RATES USING DATA FROM THE 1ST RUN

Medium Mutation Rate - .05

In the .05 static environment experiment as shown in Table 17, as with previous mutation rates, one bit strategies are the most prominent strategies because they are easy to evolve in comparison to other more complex strategies.

TABLE 17. THE MOST COMMON STRATEGIES OBSERVED UNDER THE STATIC EXPERIMENT USING .05 MUTATIONAL RATES

Static Environment (Medium) - .05 Mutational Rate	# of Runs
[False, True]~[True]	**4**
[True, True]~[True]	**4**
[False, False]~[True]	**4**
[False, True, True, True]~[True, True]	**1**
False, False, True, True]~[True, True]	**1**
[True, False, False, True]~[True, True]	**1**
[True, True, False, True]~[True, True]	**1**
[False, True, False, True]~[True, True]	**1**
[False, False, False, True]~[True, True]	**1**

There is an increase in the variety of strategies under this experiment, especially with reference to two bit strategies. Six notable two bit strategies were adapted by the majority of the population during the first run as shown in figure 10. Three bit strategies and one bit strategies evolved with minimal frequency across the population. With the exception of the first run, all other runs had similar results; Tit for Tat, Always Cooperate and Always Defect where the most common strategies in these runs as shown in figure 11.

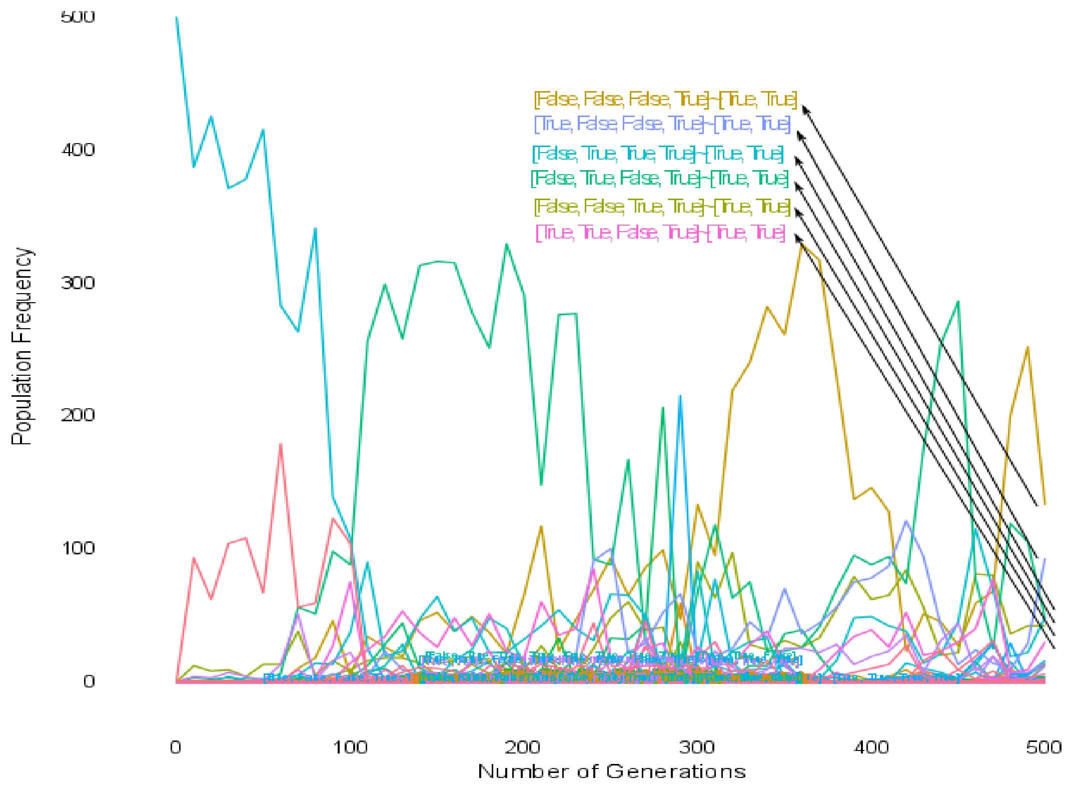


FIGURE 8. A GRAPH OF THE MOST COMMON STRATEGIES OBSERVED IN THE UNDER THE STATIC ENVIRONMENT EXPERIMENT USING .05 MUTATIONAL RATES USING DATA FROM THE 1ST RUN

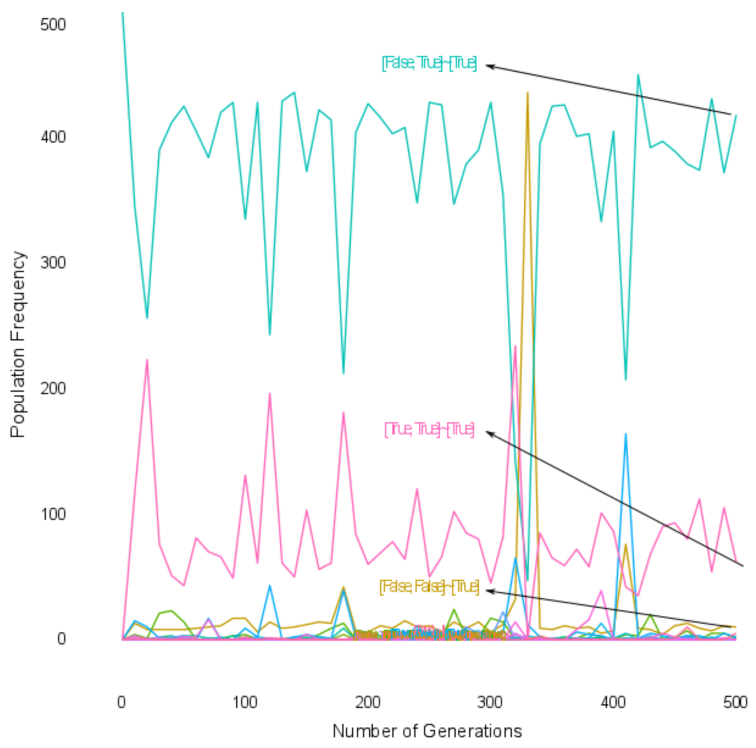


FIGURE 9. A GRAPH OF THE MOST COMMON STRATEGIES OBSERVED IN THE UNDER THE STATIC ENVIRONMENT EXPERIMENT USING .05 MUTATIONAL RATES USING DATA FROM THE 3RD RUN

High Mutation Rate - .075

As show in Table 18, Tit for Tat (one bit,[False, True]) is the most common strategy as it was frequent in four of the five experimental runs. The most common two bit strategies were Tit for Tat (two bit, [False, True, False, True]), Two Tits for Tat ([False, False, False, True]) and [True, True, False, True]. A Tit for Two Tats is a strategy that punishes defection by defecting twice in a row. The strategy [True, True, False, True], only defects if the oldest memory it remembers is defection and the opponents previous move is cooperation. In this environment, we see successful evolution of zero bit strategies in the 2nd (Always Defect-[False]) and 5th (Always Cooperate-[True]) runs. In the fifth run we see a population that mostly adapted four bit strategies.

TABLE 18. THE MOST COMMON STRATEGIES OBSERVED UNDER THE STATIC EXPERIMENT USING .075 MUTATIONAL RATES

Static Environment (High) - .075 Mutational Rate	# of Runs
[True]~[]	1
[False]~[]	1
[False, True]~[True]	4
[True, True]~[True]	2
[True, False]~[True]	2
[False, False]~[True]	1
[False, True, True, True]~[True, True]	1
[False, False, True, True]~[True, True]	1
[False, False, False, True]~[True, True]	2
[True, False, True, True]~[True, True]	1
[False, True, False, True]~[True, True]	2
[True, True, False, True]~[True, True]	2
[True, True, True, True]~[True, True]	1
[True, False, False, True]~[True, True]	1
[False, False, False, True, False, True, True, True, False, False, False, False, False, False, True, True]~[False, True, True, True]	1
[False, False, False, True, False, True, True, True, False, False, False, False, True, False, True, True]~[False, True, True, True]	1

Although there were a great number of different four bit strategies in this run, as shown in Figure 12 the most used strategy ([False, False, False, True, False, True, True, True, False, False, False, False, False, False, True, True]) has a population frequency slightly better than the second most common ([False, False, False, True, False, True, True, True, False, False, False, False, True, False, True, True]). Both strategies were slightly more successful in contrast to other strategies used in the same population. Competition in this run was substantial as there were a variety of strategies in this run each adapted in small amounts by the population.

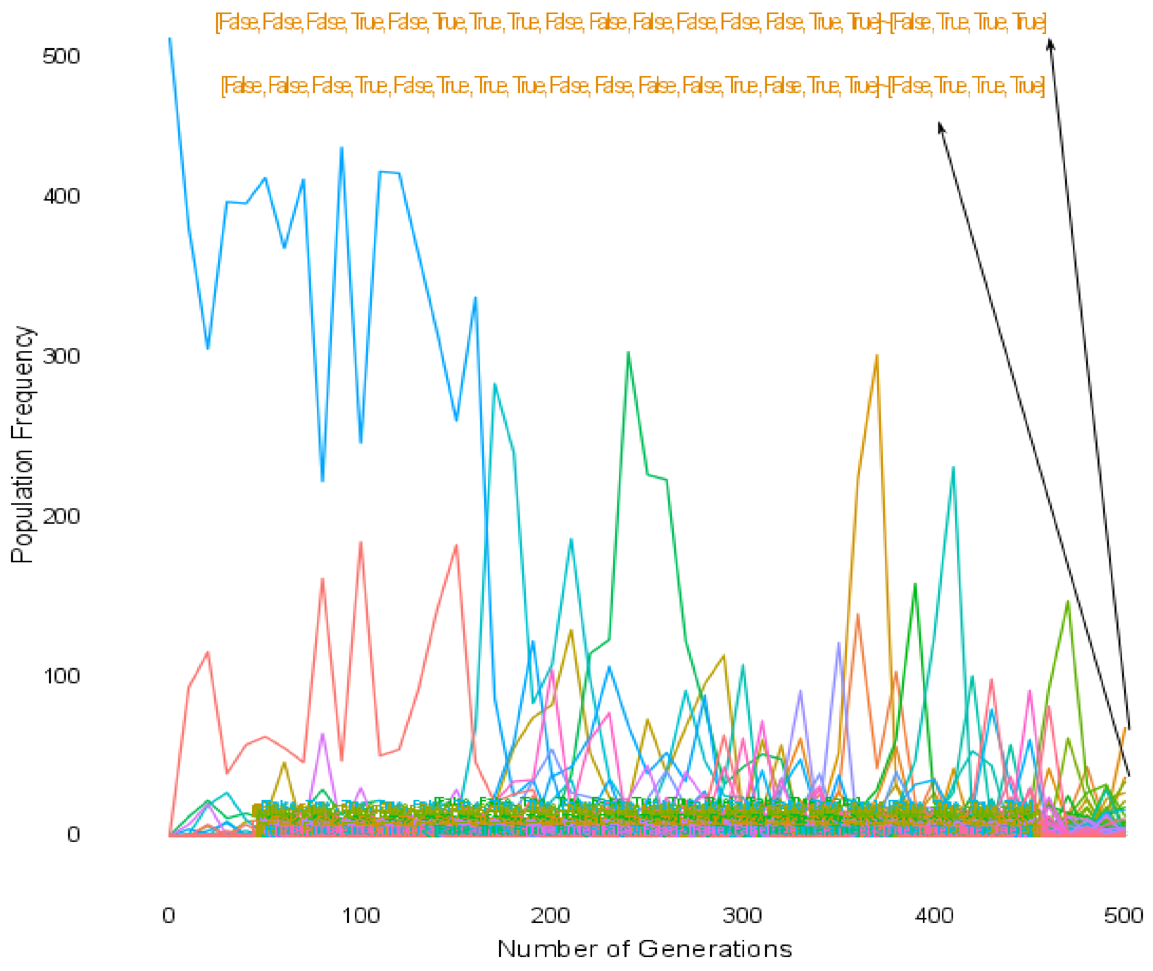


FIGURE 10. THE MOST COMMON STRATEGIES OBSERVED IN THE UNDER THE STATIC ENVIRONMENT EXPERIMENT USING .075 MUTATIONAL RATES USING DATA FROM THE 4TH RUN

High Mutation Rate - .1

As shown in table 19, results in this experiment are similar to the previous experiment. Although one bit strategies evolved in this experiment, they only did so in two of the runs. Each of these runs were mostly populated by one to two bit strategies, the most successful being Tit for Tat, Always Cooperate and Always Defect. The rest of the runs had populations using four bit strategies, which made for a highly competitive environment. Results in this experiment had the highest number of runs populated by the most complex strategies (4 bit strategies).

TABLE 19: THE MOST COMMON STRATEGIES OBSERVED IN THE STATIC ENVIRONMENT EXPERIMENT TESTING .1 MUTATIONAL RATES

Static Environment (High) - .1 Mutational Rate	# of Runs
[False, True]~[True]	2
[True, True]~[True]	2
[True, False]~[True]	1
[False, False]~[True]	2
[False, True, False, True]~[True]	1
[False, False, False, False, True, False, False, False, False, False, False, False, False, True, False, True]~[True, True, True, True]	1
[True, True, False, True, False, False, False, False, False, False, False, True, False, True, False, True]~[True, True, True, True]	1
[False, False, True, True, False, True, True, False, True, True, False, True, False, False, True, True]~[True, True, True, True]	1
[False, False, False, False, False, True, True, True, False, False, False, False, False, False, False, True]~[True, True, True, True]	1

Dynamic Environment Experiments

Low Bit Mutation Rate - .01

As shown in table 20, in this experiment, one bit strategies were the most common. Although there were a number of two bit strategies that did evolve, each of these strategies evolved all in the same run (4th run shown in figure 13). Low bit mutation rates decrease the chance of changes dealing with increasing or decreasing bits of memory. If an organism evolves with a high number of memory bits and successfully creates offspring, competition in the population increases due to the potential amount of different strategies generated by increasing or decreasing amounts of memory.

TABLE 20. THE MOST COMMON STRATEGIES OBSERVED UNDER THE DYNAMIC EXPERIMENT TESTING .01 BIT MUTATIONAL RATE

Dynamic – Bit Mutation Rate .01	# of Runs
False, True]~[True]	**4**
[True, True]~[True]	**4**
[False, False]~[True]	**3**
[True, True, True, True]~[True, True]	**1**
[False, True, True, True]~[True, True]	**1**
[True, True, False, True]~[True, True]	**1**
[False, True, False, True]~[True, True]	**1**
[False, False, False, True]~[True, True]	**1**

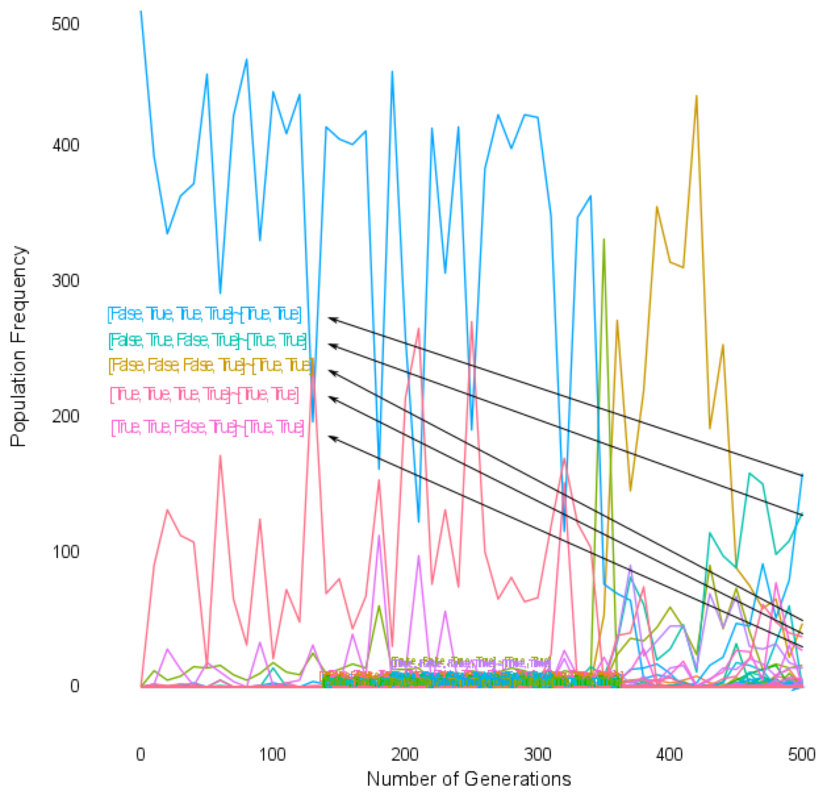


FIGURE 11. A GRAPH OF THE MOST COMMON STRATEGIES OBSERVED IN THE UNDER THE DYNAMIC ENVIRONMENT EXPERIMENT TESTING BIT MUTATION WITH .01 MUTATIONAL RATES USING DATA FROM THE 4TH RUN

High Bit Mutation Rate - .1

As shown in Table 21, organisms can still adapt to strategies that use low amounts of memory in an environment with high bit mutation rates. Although fewer strategies appear compared to runs in the prior bit mutation experiment, there is an increasing number of organisms in the experiment adapting more complex strategies such as Tit for Tat (two bit).High bit mutation rates allow for more complex strategies to evolve, thereby increasing competition. In order to survive, organisms adapt that can compete against a myriad of opponents, which could explain a decrease number of common strategies as the most adaptable strategies survive.

TABLE 21. THE MOST COMMON STRATEGIES OBSERVED UNDER THE DYNAMIC EXPERIMENT TESTING .1 BIT MUTATION RATE

Dynamic – Bit Mutation Rate .1	# of Runs
[False, True]~[True]	**4**
[True, True]~[True]	**4**
[True, False]~[True]	**1**
[False, False]~[True]	**2**
[False, True, False, True]~[True, True]	**3**
[False, False, False, True]~[True, True]	**1**
[False, False, True, True]~[True, True]	**1**

As show in Figure 14, under high bit mutation rates, a multitude of strategies can evolve. The amount of competition caused by high mutation rates causes a population to adopt a number of strategies. Organisms adapting new strategies have a chance to become extinct. There are times that new strategies are relevant enough to affect a population. This is presented by the various spikes in the figure 14 as the population over time.

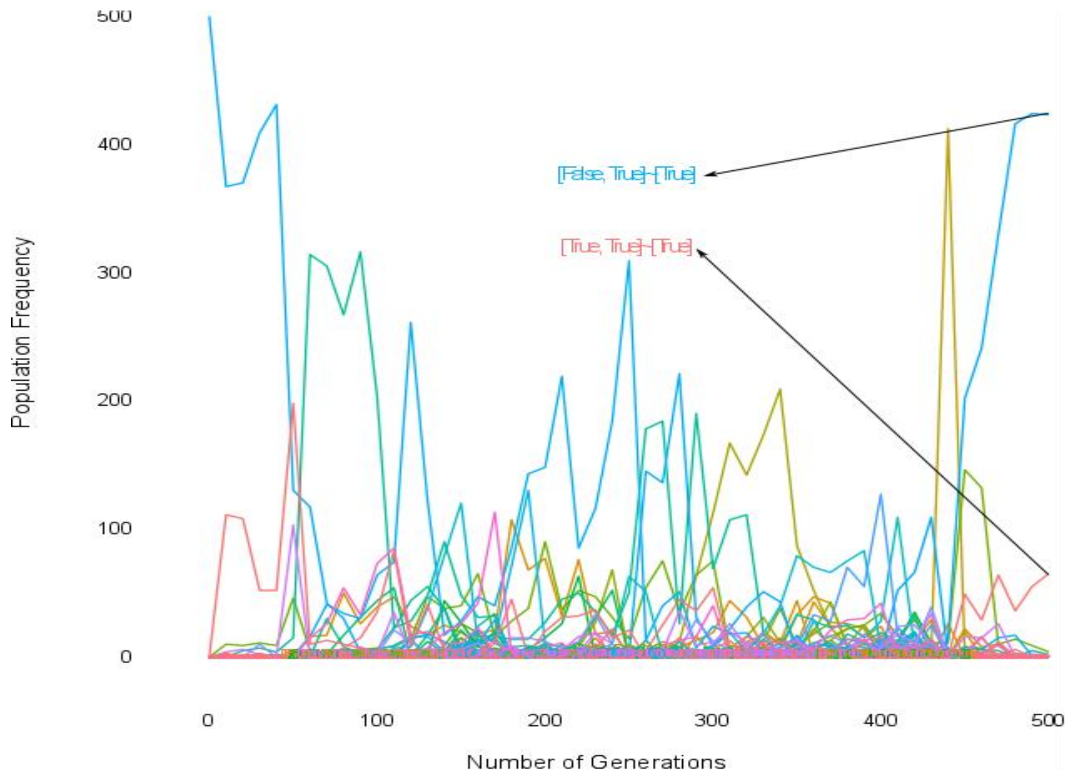


FIGURE 12. A GRAPH OF THE MOST COMMON STRATEGIES OBSERVED IN THE UNDER THE DYNAMIC ENVIRONMENT EXPERIMENT TESTING BIT MUTATION WITH .1 MUTATIONAL RATES USING DATA FROM THE 5TH RUN

Low (.01) and High (.1) Memory Mutation Rate

The results in both experiments shown in tables 22 and 23 , low (.01) memory mutation rates have a greater number of common strategies than high (.1) mutation rates. As these results, did not correlate with information gathered from previous experiments, both experiment were run a second time. Running both experiments again led to results similar to previous experiments, which contradicted initial results gathered from both experiments.

Changes in memories can greatly affect the score of individual organisms within a population. The opposite is also true as memory can differ from each individual. The memory an organism has changes with each encounter it has with another organism. As organism's makes its decision by mapping its memory to an appropriate decision in its decision list, the possibility exists that the memories an organism has may not affect its decisions. For example, an organism that adapts a strategy like Always Cooperate will cooperate regardless of its memories. Results in these two experiments suggest although changes in memory can affect the outcome of decisions made by organisms, the influence it has on the success of a strategy over time is potentially overshadowed by other factors (Bit mutation, General mutation).

TABLE 22. THE MOST COMMON STRATEGIES OBSERVED UNDER THE DYNAMIC EXPERIMENT TESTING .01 MEMORY MUTATION RATE

Dynamic – Memory Mutation Rate .01	# of Runs
[False, True]~[True]	**3**
True, True]~[True]	**3**
[False, False]~[True]	**3**
[False, False, True, True]~[True, True]	**1**
[False, True, False, True]~[True, True]	**1**
False, False, False, True]~[True, True]	**1**
[False, True, False, False, True, True, True, True]~[True, True, True]	**1**
[False, True, False, False, True, True, False, True]~[True, True, True]	**1**
[False, True, False, True, True, True, False, True]~[True, True, True]	**1**

TABLE 23. THE MOST COMMON STRATEGIES OBSERVED UNDER THE DYNAMIC EXPERIMENT TESTING .1 MEMORY MUTATION RATE

Dynamic - Memory Mutation Rate .1	# of Runs
[False, True]~[True]	**5**
[True, True]~[True]	**5**
[False, True, True, True]~[True, True]	**1**
[False, False, True, True]~[True, True]	**1**

Low General Mutation Rate - .01

As shown in Table 24, this experiment contained the smallest number of common strategies adapted throughout each population under Dynamic Environment. This experiment also evolved the least complex set of strategies for experiments outside of the control environment. Most of the common strategies that evolved were one bit strategies, with the exception of one run. This run evolved the zero-bit strategy, Always Defect. Tit for Tat and Always Cooperate were the most common strategies adapted in populations living in this environment and as a result, populations this experiment were stable.

TABLE 24. THE MOST COMMON STRATEGIES OBSERVED UNDER THE DYNAMIC EXPERIMENT TESTING .01 GENERAL MUTATION RATE

Dynamic - General Mutation Rate .01	# of Runs
[False]~[]	**1**
[False, True]~[True]	**4**
[True, True]~[True]	**4**

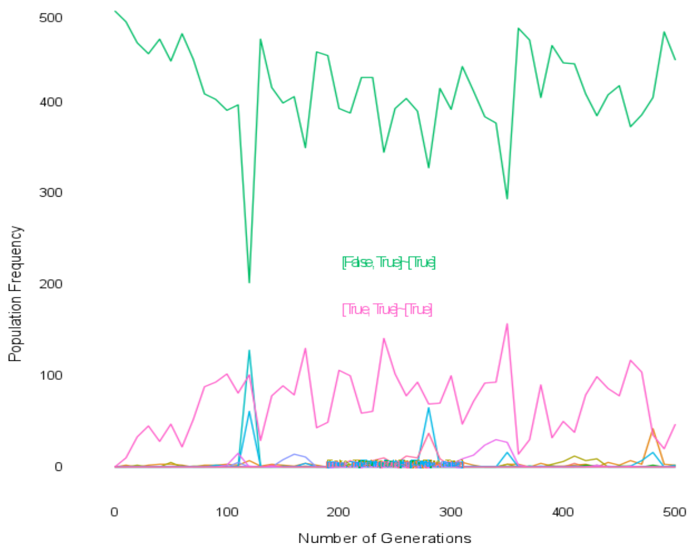


FIGURE 13. A GRAPH THAT SHOWS THE MOST COMMON STRATEGIES OBSERVED IN THE UNDER THE DYNAMIC ENVIRONMENT EXPERIMENT TESTING GENERAL MUTATION WITH .01 MUTATIONAL RATES USING DATA FROM THE 2ND RUN

High General Mutation Rate - .1

As shown in Table 25, this experiment contained the largest bit variation in common strategies adapted throughout population under Dynamic Environment. As shown in Figure 16, runs under this experiment were highly competitive as strategies used by each population were constantly changing per generation. With the exception of two bit strategies, each strategy, at most, only appeared only once over the course of the experiment.

Results from general mutation rate experiments .01 and .1 suggests that this mutation is the most impactful mutation out of all three types of mutation as the data in these experiments correlate similarly to both low and high Static Environment experiments which had the lowest and highest rates per mutation type.

TABLE 25. THE MOST COMMON STRATEGIES OBSERVED UNDER THE DYNAMIC EXPERIMENT TESTING .1 GENERAL MUTATION RATE

Dynamic - General Mutation Rate .1	# of Runs
[False]~[]	1
[False, True]~[True]	2
[True, True]~[True]	2
True, False]~[True]	2
[False, False]~[True]	2
[False, False, True, True, True, True, False, True]~[True, True, True]	1
[False, False, True, True, True, False, False, True]~[True, True, True]	1
[True, True, False, False, True, False, False, True]~[True, True, True]	1
False, False, False, False, False, False, False, True]~[True, True, True]	1
[False, False, True, True, True, True, False, True]~[True, True, True]	1
[False, True, False, True, False, False, False, False, False, False, True, False, False, False, False, True]~[True, True, True, True]	1

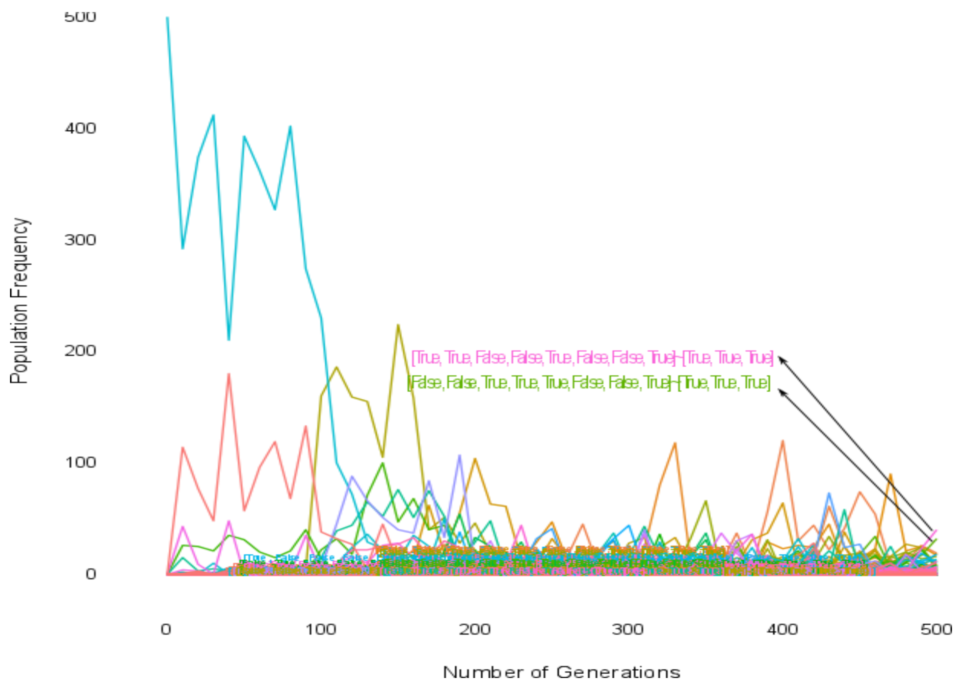


FIGURE 14. THE MOST COMMON STRATEGIES OBSERVED IN THE UNDER THE DYNAMIC ENVIRONMENT EXPERIMENT TESTING GENERAL MUTATION WITH .1 MUTATIONAL RATES USING DATA FROM THE 3RD RUN

CHAPTER IV

CONCLUSION AND FUTURE WORK

Discussion

This thesis focuses on investigating the effects different mutations and mutation rates have on populations using strategies while playing Iterative Prisoners Dilemma (IPD). The software for this study was the system used genetic algorithms in order to observe the effects of memory on populations using strategies while playing IPD [4]. Experiments in this study were sectioned into three different categories: Control, Static, and Dynamic Environments. The first set of experiments, the Control Environment Experiments, were used as benchmark for the system and as a baseline and reference for Static and Dynamic Environment Experiments. The second set of experiments, the Static Environment Experiments presented data regarding populations using strategies while affected by different levels of mutational rates. The purpose of this set of experiments is analyze the effects mutation rates ranging from low to high, as stated in the hypothesis. All three types of mutation under this environment were kept at the same rate. Different rates were tested in each experiment which had two categories low (.01, .025) and high (.075, .1) with .05 being the median rate. The third set of experiments, the Dynamic Environment Experiments presented data on populations affected by three different types of mutation, Bit, Memory and General. The purpose of this set of experiments analyze the effects of different types of mutation on populations using strategies while playing Iterative Prisoner's Dilemma. Two experiments (low: .01, high: .1) were run for each of the three mutational rates. Of each experiment, only one of the three types of mutation were changed, the rest were kept in the median rate (.05). As with the previous environment, all experimental populations were seeded with Tit for Tat as its initial strategy.

Control Environment

Results for the first two control experiments show that the initial strategies within a population greatly impact future generations under an environment where no mutation is present. The next three experiments presented the data on environments lacking one of the three mutational rates: Bit, Memory and General. Out of these three types of mutation, general mutation was the most influential as it acts both as the primary rate which the system checks as a when making changes within organisms in a population. This mutation also decides when decision list mutations can occur within organisms in a population.

Static Environment

Results show that as mutation rate increases, the possibility of more complex strategies evolving also rises. Populations with low mutation rates have difficulty mutating strategies with high amounts of memory given that low mutation rates slow down the process of mutation within a population. Populations under these conditions tend to adopt strategies that interact with Tit for Tat [Always Cooperate, Always Defect], because this strategy serves as the initial strategy for the population. This leads to stable populations where organisms adopt certain strategies. On the other hand, populations under high mutation rates tend to exhibit a broader range of strategies and strategies that require high amounts of memory. This leads to highly competitive environments filled with a variety of strategies. Results in this environment support with the hypothesis of this thesis.

Dynamic Environment

In the dynamic environment, the experiments show that each of the three different types of mutation have an effect on population's use of strategies while playing the Iterative Prisoner's Dilemma (IPD). Results show that Bit mutation rates limit the complexity of strategies that evolve within a population. Populations under low Bit mutation rates have strategies with a similar number of memory bits. High Bit mutation rates tend to increase the diversity of strategies within a population. Populations under high Bit mutation rates adapt strategies with high greater of memory in comparison to the initial strategy of a population. Low bit mutation keeps stability, and high Bit Mutation allows for increased variability in reference to the strategies within a population.

Memory mutation was the least impactful type of mutation as runs in both low and high rates fluctuate between stable and competitive populations. The results in the memory mutation experiments contradicted previous experiments that showed populations with organisms adapting strategies using low amounts of memory in populations under low mutational rates, and organisms adapting strategies with high amounts of memory in populations under high mutational rates. Another experiment was run with same conditions as the previous experiment. This experiment contradicted these results. The results from both experiments suggest that the effects of memory mutation are overshadowed by other mutation types.

General Mutation had the most impact with respect to change in organisms in a population. As dictated by the system, the general mutation rate acts as the primary mutation rate used by the system. It governs the rate of all mutations in organisms within the population. Results in populations with low general mutation rates act similar to low mutational experiments in the Static Environment while high general mutation rates act similarly to high mutational experiments in the Static Environment.

Final Thoughts

General mutation, through system design, had the most impact of any mutation type in the system. Bit mutation and memory mutation are only allowed to occur in an organism that can undergo general mutation. General mutation also governs decision list mutations, which can affect strategies in a population over time. With the ability to change decisions over time, general mutation has the greatest impact in comparison to other mutations because the number of strategies that can be created through slight permutations in an organism’s decision list is larger than the amount of strategies generated by other mutations types.

Bit mutation was the second most impactful mutation. Bit mutation rates have great impact on the complexity of strategies evolved organisms within a population. The higher bit mutation rates are, the higher the chance that organisms will evolve strategies that use high amounts of memory. As organisms within a population adapt strategies that use high amounts of memory, competition within the population increases. As specified by the system, the size of an organism’s decision list is 2^m were m is the amount of memory bits an organism has. As memory increases the amount of decisions an organism can make also increases. Although new strategies can be generated solely through increasing the memory of an organism, mutations in an organism’s decision list can have a greater impact on the amount of strategies within a population. Therefore bit mutation rates can have a great impact on the amount of strategies adapted by a population especially if general mutation rates are high.

Out of all three types of mutation within the system, the memory mutation was the least impactful mutation. The contents of an organism’s memory change rapidly through mutation and an organism’s encounters and the amount of influence this has is significantly less than the amount of influence other mutations have on a population as m strategies created by changes in memory cannot compete with 2^m strategies created by mutation s in an organism’s decision list.

Although memory mutation can affect the speed in which mutations in contents of memory occur, these changes can happen independently of this mutation. As an organism encounters other individuals within a population, the decision of each individual can potentially change the memories recorded within an organism’s memory bits. Changing Memory mutation rates is potentially synonymous to affecting the ability of an organism to remember a sequence of events correctly (forgetfulness or remembering wrongly).

As specified by the system, an organism makes a decision by turning its memory bits into a value, and mapping that value to its appropriate decision in its decision list. These changes can have great or minimal impact on an organism’s decision. Certain strategies are independent of memory. For example and organism that uses the strategy Always Defect, will defect regardless of the values in its memory bits. Strategies that depend on memory values are greatly influenced by these changes. Tit for Tat, a strategy that cooperates initially then mimics its opponent’s previous actions needs one bit of memory. Mutations or changes in this memory bit can greatly influence the decision of an organism using this strategy. The fact that strategies may or may not be influenced by the contents of an organism’s memory and that changes in the contents of these memory bits can happen independently of Memory mutation rates help explain the results found in the Dynamic Environment experiment testing Memory Mutation rates.

In reference to Axelrod’s Tournament [10], the most successful strategies, i.e., the most used strategies in each population had aspects of the four characteristics specified by Axelrod.

Many of the most common strategies in each experiment adapted strategies that were nice; strategies that cooperated first. An example of this is show in figure 10. The most popular strategy, A Tit for Two Tats ([False, False, False, True]~[True, True]) cooperates initially as bits of memory remembered by this organism maps towards cooperation.

Most organisms developed punishing strategies. An example of this is shown in table 25, as most of the strategies in this table have defection implemented in its decision list.

With the aspect of forgiving an example of this is shown in table 24, as most of the strategies in this table have cooperation implemented in its decision list. Most strategies in this table have ability to punish, but with the inclusion of cooperation in the decision list of these strategies, organisms adapting these strategies have the ability to forgive others.

The last aspect consistency is specified by the system. The size of an organism’s decision list is 2^m were m is the amount of memory bits an organism has. An organism makes a decision by turning its memory into a value and mapping it to a specific decision in its decision list. This mapping allows for consistency as each permutation in the values of memory within an organism’s memory bits map to a unique value in an organism’s decision list.

Conclusion

The focus of the experiments in this thesis is to study the effects of mutation on populations using strategies while playing Iterative Prisoner’s Dilemma (IPD). I observed how both different types of mutation and mutation rates can have various effects when dealing with strategies while playing IPD, as well as how Axelrod’s tournament [10] and the characteristics he defined, correlate to the success of organism’s using strategies while playing IPD.

Of the three types of mutation I studied under the system used by Leas [4], general mutation was the most impactful, followed by bit and memory mutations. On the topic of mutation rates, results suggest that as mutation rates increase, the competition between organisms within a population using strategies while playing IPD increases. A main cause of this increase in competition is the amount of strategies generated by mutations in the strategies of organisms. As mutation rates increase, the amount of different strategies within a population increases as well.

In reference of Axelrod tournament [10], many of common strategies organisms adapted in each experiment contained the four characteristics that were specified in Axelrod's results. As shown in the results in this thesis, these four characteristics hold great importance when considering successful strategies in the game IPD.

The results in this thesis can have great applications in fields of social sciences, politics and economics as studying behavior in this context can help explain many situations in real life.

Future Work

A Study on the Initial Population of Organisms Playing the IPD.

As observed in the Control Environment experimental results, initial strategies within a population has potential impact in the evolution of strategies in populations playing the IPD. Future work in this subject includes, studying the effects of different initial strategies in both Static and Dynamic Environments.

System Expansion

This system holds great potential in studying aspects of game theory through the use of evolutionary computational algorithms such as genetic algorithms. An expansion of this system would allowfor the study of game theory in greater detail and fields of research that are known to have used evolutionary algorithms could potentially benefit from this as well.

REFERENCES

[1] John Maynard Smith, *Evolution and the Theory of Games*. New York: Cambridge University Press, 1982.

[2] John Maynard Smith and George Robert Price, "The Logic of Animal Conflicts," *Nature*, pp. 15-18, 1973.

[3] Thomas Back, Ulrick Hammel, and H-P Schwefel, "Evolutionary computation: Comments on the history and current state," in *IEEE transactions on Evolutionary Computation 1.1*, 1997, pp. 3-17.

[4] Mikaela Leas et al., "The Prisoner's Dillemma, Memory and the Early Evolution of Intelligence," in *Proceedings of the Artificial Life Conference 2016*, Mayan Riviera, 2016, pp. 409-415.

[5] Rabin Matthew, "Incorporating fairness into game theory and economics," *The American economic review* , pp. 1281-1302, 1993.

[6] Andrew M Colman, "Cooperation, psychological game theory, and the limitations of rationality in social interaction," *Behavioral and brain sciences*, pp. 139-153, 2003.

[7] Peter Hammerstein and Reinhard Selten, "Game theory and evolutionary biology," in *Handbook of game theory with economic applications*.: North Holand, 1994, pp. 929-993.

[8] Kevin C Clements and David W Stephens, "Testing models of non-kin cooperation:mutualism and the Prisoner's Dilemma," *Animal Behavior*, vol. 50, no. 2, pp. 527-535, August 1995.

[9] Andreoni James and John H Miller, "Rational cooperation in the finitely repeated prisoner's dilemma: Experimental evidence," *The economic journal* , pp. 570-585, 1993.

[10] Robert Axelrod, "Effective choice in the prisoner's dilemma," *Journal of conflict resolution*, pp. 3-25, 1980.

[11] Martin A. Nowak, "Evolving Cooperation," *Journal of theoretical biology*, pp. 1-8, 2012.

[12] Charles Robert Darwin, *On the origin of species by means of natural selection, or, the preservation of favoured races in the struggle for life*. London: John Murray, 1859.

[13] Robert Axelrod, "The evolution of strategies in the iterated prisoner's dilemma," in *The Dynamics of Norms*. New York: Cambridge University Press, 1987, pp. 1-16.

APPENDIX

APPENDIX

Functions

pd_qsub_generator.py

- Convert_config_file_name_to_job_string() – this function takes a file name and output directory base. Edits done to this file affect file input pathing. The directory of the files needed for the experiment must be specified in this file for a job to be executed by the HPCC.
- create_config_files() -this function creates and names output data files(config files) created by the system after a successful job is finished by the HPCC. Edits done to this file will affect output file names created by the system.

Main File

- create_initial_population () – this sets the initial population of organisms in an experiment. By default, population generation is random. For the experiments in this thesis the population was set to begin with Tit for Tat as used by Leas.

The Process of Running Experiments using the System (figure 4)

Configure Experimental Settings – This stage involves editing the variables in the configuration file (optional: main file- create_initial_population()).

Set file names – This stageinvolvesediting the pd_qsub_generator.py file specifying file paths and input/output file names.

Execution – This stageinvolvesrunning the pd_qsub_generator.py with the specified configfile(pd_config.init) as input. Execution of this file sends a job request to the HPCC(High Performance Compute Cluster). Jobs on average take from 10 – 30+ mins and varies on the number jobs sent, and the availability of nodes in the HPCC. Output file directories are generated after execution is finished, which is used for the Data Calculation step.

Data Calculation – This stageinvolvesexecuting the pd_beaker_one.py using output file directories generated by the Execution step as input. The output of this file is used as input for pd_beaker_two_strategies_df_r.r .

Data Generation – This stageinvolvesrunning the pd_beaker_two_strategies_df_r.r on the file out of pd_beaker_one.py. This file generates a graphic representation of the data presented by the previous pd_beaker.

Definitions

Genetic Drift – the change of gene frequency within a population due to the random nature of reproduction.

Examples:

- The last regular crocodile dies, creating a population of albino crocodiles.
- A certain type of plant produces red and blue flowers. A volcano explodes and destroys most of the red flowers. Since blue is now the dominant trait, the plant has a great chance of producing blue flowers and over time only produces blue flowers.
- A certain type of plant produces red and blue flowers. A landslide occurs and destroys most of the blue flowers leaving mostly red flowers. Over time through a span of generations red flowering plants started to dwindle. Now thisplantmostly produces blue flowers.

BIOGRAPHICAL SKETCH

Ramses Romulus De Guzman Reyes was born during February 2 at Manila, Philippines. He moved to the Pharr, Texas at age 10. He attended Cesar Chavez elementary during 2003. After graduating from 5th grade he they attended, Liberty Middle School from 2004 to 2007. After graduating middle school he attended PSJA High School for one year(2007) before moving to Edinburg High School(2008-2011). He attended the University of Texas Pan American at 2011 and graduated with a Bachelor of Science in Computer Science at 2015. He immediately started pursuing his Master of Science in Computer Science at 2015. He obtained this degree in May 2017. He currently lives in 1938 Katherine Ave at Edinburg Texas.