Toby Hilliard
Nicholas Porter
Zhi Zheng

Warhammer 40k Squad Optimization

A Genetic Algorithm Project

1. Introduction

Project Overview:

In the beautiful and dark world of Warhammer 40k TRPG (Table Role Playing Game), an eternal war rages on in the 41st millennium rages across galaxies. For players across the world, setting up the loadout has always been the most time-consuming part of the game. In this project, we will focus on using genetic algorithms to optimize a Space Marine Squad weapon loadout against different opponents. This demonstration can be thought of as analogous to other significant AI (Artificial Intelligence) research in game playing, because games are among the best testing grounds for AI research. With controlled environments, fuzzy situations, many fantasize about the human-like interaction, a safe testing ground for actual gameplay, and many more compelling reasons all support using games as a means for developing AI techniques, before applying to real life scenarios.

The project is more than just a research project on mere game preparation, it is an adventure into the sophisticated balance between strategy and probability. The complex and rich universe of Warhammer 40K is a great subject for integrating computational power with tabletop role playing games. We seek to create a bridge between genetic algorithms with practical tabletop warfare.

Problem Definition:

The challenge of optimizing a Space Marine Squad loadout is the prerequisite of understanding the complex rules and lore of Warhammer 40K. It is like navigating through a complex labyrinth that can only be solved with deep understanding and thorough analysis. The initial task at hand is to identify the optimal combination of gear that can adapt to and triumph over numerous possible battlefield scenarios.

This optimization analysis is not about just picking the most dangerous weapon, it is an analysis requiring taking account of balancing firepower, versatility, and adaptability. The diversity of enemies, each with unique traits, vulnerabilities, and equipment. The game demands a loadout strategy that can navigate through the consideration for armor penetration, weapon range, and tactical flexibility. We will be facing hordes of Orks, forces of Chaos, hives of Tyranids, technologically advanced Tau, and many more malicious threats of 40K universe.

The project approaches the problem domain using genetic algorithms tailored to evolve and refine loadout configurations through successive iterations. Our goal is to converge at the global optimal loadout and not a local optimal loadout. This methodology aligns with our analytical needs for enriching our understanding of the strategic puzzle that is Warhammer 40K. We plan to contribute to the broader community for optimization of gaming, offering insight into methodologies that can be applied to various strategic games dealing with strategy and probability.

In summary, this project is a tribute to the strategic depth of Warhammer 40K and an academic exploration of optimization techniques. It stands as an intersection of gaming culture, strategic analysis, and computational intelligence.

2. Background

Players take control of a faction within the setting and assemble armies of miniatures to wage epic battles and tell stories of woe and conquest. Among these factions are the noble Space Marines, genetically engineered super soldiers who serve the Imperium of Mankind. Space Marines are typically found in squads and can be equipped with a set of special weaponry that can enhance their firepower.

Assembling a squad can be difficult in 40k, as there are several rules that restrict army size, composition, and equipment. How a player composes their army can decide a battle before it has even begun. Thus, several efforts have been made to optimize army loadouts and find the most effective spread of weapons and unit within the game. A popular example of this is UnitCrunch, a web application that creates statistical data to display the average output of a squad.

For our project, we are applying genetic algorithms to determine what set of special weapons a Space Marine squad can equip to be the most effective. This is determined by how many enemy models are destroyed in a skirmish.

Breakdown of Attack Sequence

Whenever a weapon attacks in 40k, a series of steps are taken to determine whether the weapon hits the target, whether its "wounds" the target, and whether the weapon ignores the target's armor.

The first stage, called "rolling to hit" is determined by the unit's Ballistics Skill. However, since we're using the same unit every time, we can ignore this value and simply calculate the average hit rate of a space marine, a one in two.

The second stage, called "rolling to wound", determines whether a weapon theoretically could wound a target. To calculate this, we compare the **strength** value of the weapon and the **toughness** of the target. From here, we follow simple if-then logic.

WOUND ROLL	
ATTACK'S STRENGTH VS TARGET'S TOUGHNESS	D6 ROLL REQUIRED
Is the Strength TWICE (or more) than the Toughness?	2+
Is the Strength GREATER than the Toughness?	3+
Is the Strength EQUAL to the Toughness?	4+
Is the Strength LOWER than the Toughness?	5+
Is the Strength HALF (or less) than the Toughness?	6+

The table gives us the dice roll that will result in a success, which we simply convert to probability (e.g. a 2+ has a 5/6 probability of success).

Finally, we must calculate whether or not **armor** prevents the damage from occurring, and calculate the total damage done. We'll focus on whether or not the armor saves fail. This is given by the function (Save - 1 + AP) / 6, where Save is the **armor save** for a given model and the **armor penetration** characteristic of the weapon. Finally, we can multiply that result by the damage of the weapon.

3. Methodology

Genetic Algorithm Introduction:

Genetic algorithms (GAs) are a subset of evolutionary algorithms inspired by natural selection and genetics. The Genetic algorithms simulate natural evolution, employing mechanisms like selection, crossover, and mutation to evolve solutions to optimization and search problems. GAs is particularly adept at navigating large and complex search spaces, making them ideal for challenges where traditional optimization methods are inadequate. The critical formula that represents the GA process is:

$$New\ Generation = Mutate(Crossover(Select(Current\ Generation)))$$

Representation of Chromosomes:

Each chromosome in our genetic algorithm represents a potential load-out for a Space Marine Squad. This load-out is encoded as a sequence of genes, where each gene corresponds to a specific weapon. A chromosome is represented as $C = [w_1, w_2, w_3, w_4]$.

Algorithm Parameters:

We assign the following parameters for the genetic algorithm:

- **Population Size (*N*)**: Set at 200, 500, and 1000 as parameters influences the diversity of load-out combinations.
- **Crossover Probability (P_c)**: At 0.6, 0.7, and 0.9 these rates determine how frequently chromosomes combine their genetic material, thus affecting population diversity.
- **Mutation Probability (P_m)**: At 0.01, 0.05, and 0.1 these rates controls the frequency of random changes in the genes, allowing for the introduction of new weapons at a controlled rate.

Fitness Function:

Our fitness function for the genetic algorithm is based on a single attack sequence with a given loadout. Using a tournament style of selection, we will test which weapon loadout in each generation will be most fit to accomplish this task. Fitness is determined by the number of enemies destroyed thus this fitness function iterates through the chromosome and finds the roll to wounds dealt based upon the parameters mentioned in the roll phases above:

$$f(w,\ t) = \ ceil(w_n \cdot 0.5) \cdot wounds(w_s,\ t_t) \cdot \left(\frac{t_s - 1 + w_{AP}}{6}\right) \cdot w_d$$

Where w is the weapon, t is a test case, and the subscripts are the attributes of the weapons and test cases. This is followed up by the total enemies destroyed, which is the number of enemies in the test cases destroyed, which will be the determinant of how well our weapon loadout performed, and thus how fit our chromosome is:

$$t_d = \frac{f(w,t)}{t_t}$$

Where t_d is the enemy targets destroyed. The scenario where the toughness of a target is greater than six is also considered for excellent performance.

Data:

To determine the results, we created two datasets. The first dataset contains all the potential weapons a space marine squad can be equipped with, including melee weapons. The second dataset is an array of units from different armies in the 40k setting to serve as test cases. These range from light infantry troops, such as the Imperial Guardsmen, to heavy vehicles, like the Eldar Wraithlord. Our genetic algorithm draws from this test case pool to determine whether a given squad loadout is fit to fight a diverse range of targets.

4. Implementation

Initial Population Creation:

The initial population is a set of chromosomes, each representing a potential solution (load-out). Mathematically, this is represented as:

$$Population_{initial} = \{\boldsymbol{C}_1, \boldsymbol{C}_2, \dots, \boldsymbol{C}_N\}$$

Where $\boldsymbol{C}_i = [g_1, g_2, \dots, g_m]$ is a chromosome consisting of m genes, and each gene g represents a weapon type. These genes are randomly selected from a predefined set of weapons, ensuring diversity within the initial population. The population size N is a predetermined parameter that influences the genetic diversity available for evolution.

Crossover and Mutation Mechanisms:

- **Crossover**: The crossover operation combines genetic material from two parent chromosomes to produce new offspring. For two parents $\boldsymbol{C}_{parent1}$ and $\boldsymbol{C}_{parent2}$, the crossover operation can be defined as:

$$\boldsymbol{C}_{\boldsymbol{child1}} = \boldsymbol{C}_{parent1}[:x] + \boldsymbol{C}_{parent2}[x:]$$
$$\boldsymbol{C}_{child2} = \boldsymbol{C}_{parent2}[:x] + \boldsymbol{C}_{parent1}[x:]$$

Here, x is a randomly chosen crossover point within the chromosome's length. This operation is applied with a probability P_c crossover probability), which governs the frequency of crossover events.

- **Mutation**: Mutation introduces new genetic variations into the population, enhancing the algorithm's ability to explore the solution space. The mutation operation for a chromosome **C** can be mathematically described as:

$$\boldsymbol{C}_{mutated}[i] = f(x) = \begin{cases} random\ weapon & if\ rand(0,1) < P_m \\ \boldsymbol{C}[i] & otherwise \end{cases}$$

Here, rand(0,1) generates a random number between 0 and 1 and P_m the mutation probability. Each gene in the chromosome is independently subjected to this mutation test.

Termination Criteria:

The termination criteria for the genetic algorithm can be mathematically formalized as follows:

$$Term\ if\ (G = MG)\ or (\Delta f \leq \epsilon\ for\ T\ generations)$$

where:

- Generation is the current generation number.
- MaxGenerations is the maximum number of generations allowed.
- Δf is the change in the maximum fitness value across generations.

- ϵ is a small threshold value indicating stagnation in fitness.
- T is the number of generations for which fitness changes are below the threshold.

These mathematical formulations provide a clear and precise framework for the implementation of the genetic algorithm, enhancing the understanding of its operational mechanics. As the project progresses and evolves, these formulations can be adjusted and refined to align with any changes or improvements in the algorithm's design and functionality.

5. Results

Interim Findings:

The genetic algorithm's initial runs have shown a progressive improvement in fitness scores, indicating an evolution toward more effective weapon combinations. Notably, the algorithm has consistently favored loadouts comprised entirely of 'Thunder Hammers.' By generation three, the best fitness score achieved was 452.0, with the optimal load-out identified as four 'Thunder Hammers,' each with an attack rate of 10.0, strength of -2.0, an AP of 3.0, and damage of 3.0. Additionally, the genetic algorithm converges on around third generations, and the highest generation to converge on is the 23rd generation.

Performance Metrics:

The evaluation of loadouts relies on a fitness function that assesses their combat effectiveness against a range of enemy units based on the following:

- **Expected Damage Output: This metric is crucial, as it measures each weapon's potential impact in various combat scenarios, considering its strength, armor penetration, damage, and rate of fire.**
- **Versatility**: The algorithm evaluates the load-out's ability to handle different enemy types, ensuring that it can adapt to diverse battlefield situations.
- **Combat Efficiency**: This aspect considers the potential of a load-out to maximize damage while minimizing vulnerabilities based on the strategic combination of weapons.

Ongoing Analysis:

While the current results are promising in terms of fitness score improvement, they also highlight the need for further analysis and potential adjustments:

- **Diversity in Weaponry**: The dominance of 'Thunder Hammers' in the optimal load-out suggests a need to re-examine the balance of weapon attributes and their representation in the fitness function.
- **Strategic Balance**: Future iterations will explore how different combinations of weapons can offer a more balanced approach to combat, considering factors like range, rate of fire, and adaptability.
- **Algorithm Tuning**: Adjustments to parameters such as mutation rate and selection

process will be considered to encourage a broader exploration of the solution space.

- **Fitness Function Refinement**: The fitness function will be reassessed to ensure it accurately evaluates the strategic and tactical value of diverse loadouts.

The results will be continually updated as the project progresses, with the aim to develop a load-out strategy that is not only statistically optimal but also aligns with the strategic complexities and varied combat scenarios of Warhammer 40K gameplay.

6. Discussion

Initial Insights:

The current results from the genetic algorithm reveal intriguing insights into the optimization of loadouts for Warhammer 40K Space Marine Squads favors the equipment 'Thunder Hammer'. The indication being that the 'Thunder Hammer' is the most effective weapon in the provided scenarios is the assumption of the algorithm. This is understandable, because the 'Thunder Hammers' provide significant advantages in high strength, armor penetration, and damage attributes. However, it is not the absolute best optimal loadout, because this is just a simplified version of genetic algorithms for the Warhammer 40K Space Marine loadout. The complexity of Warhammer 40K can take years to completely to setup the scenarios for training to find the true optimal loadout.

Adaptability and Versatility:

The current findings, while highlighting the effectiveness of a particular weapon type, also raise questions about the adaptability and versatility of the loadouts:

- **Lack of Tactical Diversity**: The dominance of a single weapon type of 'Thunder Hammer' limits the Space Marine Squad capability to adapt to diverse battlefield conditions. In actual gameplay, a mix of weapon types is crucial for handling different enemy units and combat ranges, which is important for tactical warfare.
- **Strategic Considerations**: The current loadouts need to account for the strategic complexity of Warhammer 40K, where factors like weapon range, mobility, specific enemy weaknesses, and the battlefield condition play a significant role in determining the outcome of battles.

Areas for Improvement:

Based on these observations, multiple areas for improvement in the algorithm and its implementation can be identified:

- **Fitness Function Enhancement**: The fitness function may need to be refined to better balance the attributes of different weapons and to more accurately reflect the varied conditions of Warhammer 40K combat scenarios.
- **Parameter Tuning**: Adjusting the algorithm's parameters, such as increasing the

mutation rate or modifying the selection process, could help in exploring a wider variety of loadouts.
- **Introducing Constraints**: Incorporating constraints or additional criteria into the algorithm might encourage a more diverse and strategically sound selection of weapons.
- **Algorithmic Robustness:** Enhancing the algorithm's robustness to ensure it only favors certain weapon types if they are unequivocally the best choice in most scenarios.

As the project continues, these areas will be addressed, and the algorithm will be continuously refined. The goal is to develop a more comprehensive and strategically genetic algorithm to help find optimal loadout in Warhammer 40K, ensuring that the results are not only statistically sound but also relevant to the actual game accounting for the diverse tactical landscape and wide-variety enemy units.

7. Conclusion

<u>*Summary:*</u>

Based on the results of the genetic algorithm, the project made significant progress exploring the puzzling domain between computational power and tabletop warfare. The genetic algorithm is fruitful in generating generation with ability to evolve and improve the fitness of the optimizing loadouts for Warhammer 40K Space Marine Squad. The most noticeable result of the genetic algorithms is the algorithm's heavy emphasis on picking "Thunder Hammers," as the optimal loadout, a weapon that offers substantial combat effectiveness.

This outcome, while insightful, also underlines the necessity of ensuring a more balanced and tactically versatile approach to load-out optimization. The initial results have provided a foundation upon which further refinements and adjustments can be made, aiming to achieve a more comprehensive solution that aligns with the strategic complexities of Warhammer 40K gameplay.

<u>*Future Directions:*</u>

Based on the results from the project, these are the areas that can be focus on in the future:

- **Refinement of the Fitness Function**: Adjusting the fitness function to more accurately reflect the diverse tactical scenarios encountered in Warhammer 40K, ensuring that it values versatility and adaptability alongside raw combat effectiveness. Increasing the dimensionality of can be a way to improve the fitness function to capture the more aspects of the game's complexity.
- **Algorithm Parameter Tuning**: Experimenting with different settings for mutation rates, crossover probabilities, and selection methods to encourage exploration of a broader range of loadouts.
- **Incorporation of Tactical Elements**: Introducing additional factors into the evaluation process, such as weapon range, unit mobility, and specific enemy counters, to develop loadouts that are not only powerful but also strategically sound.
- **Continuous Evaluation and Testing**: Running the algorithm with varied data sets and

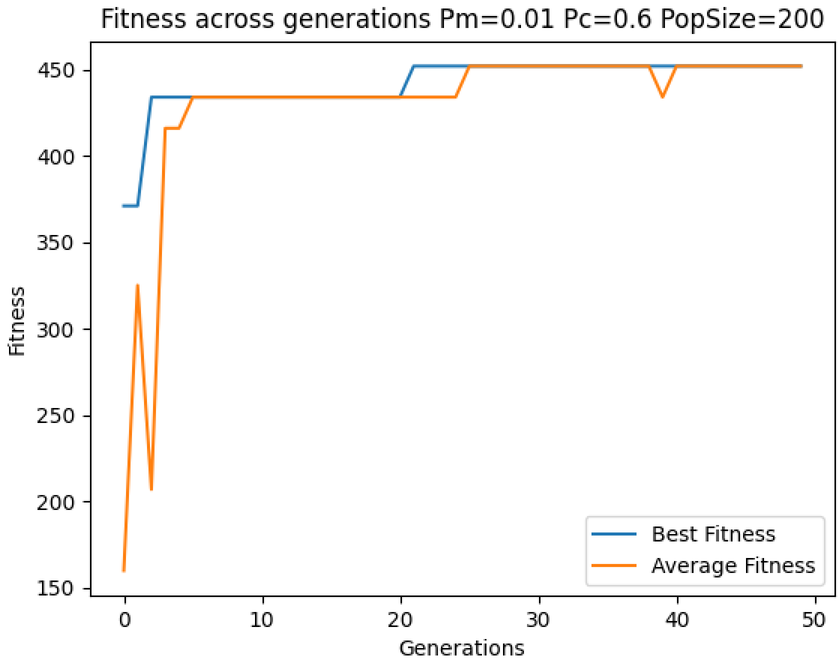
scenarios to validate its effectiveness and fine-tune its performance. Like trying to find optimal build for other factions.

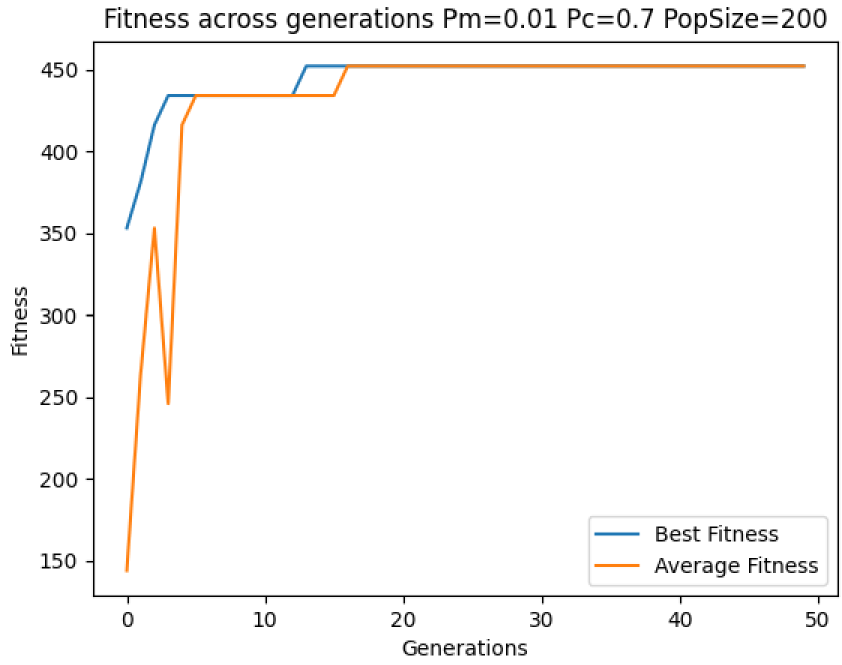
8. References

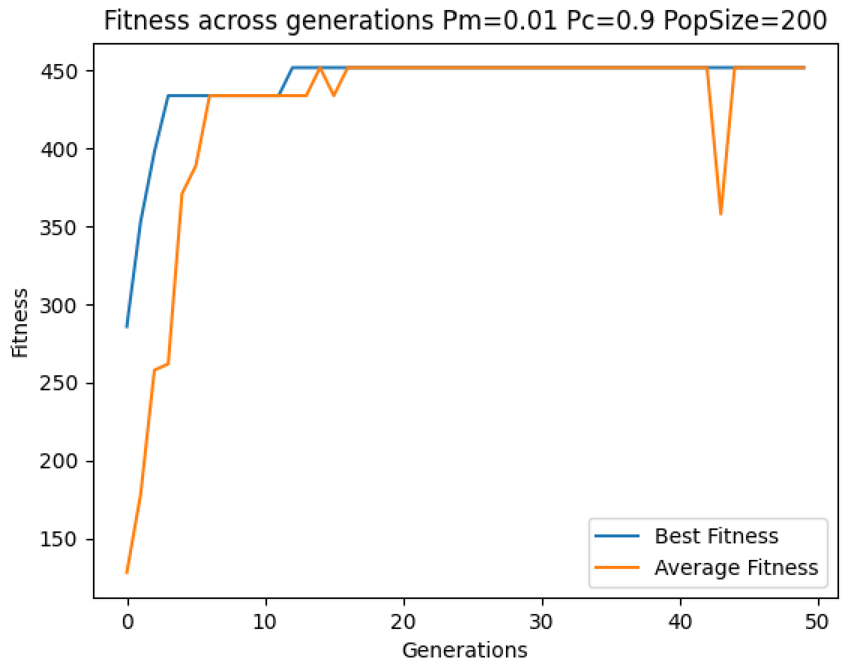
https://wahapedia.ru/wh40k9ed/the-rules/playing-this-game/

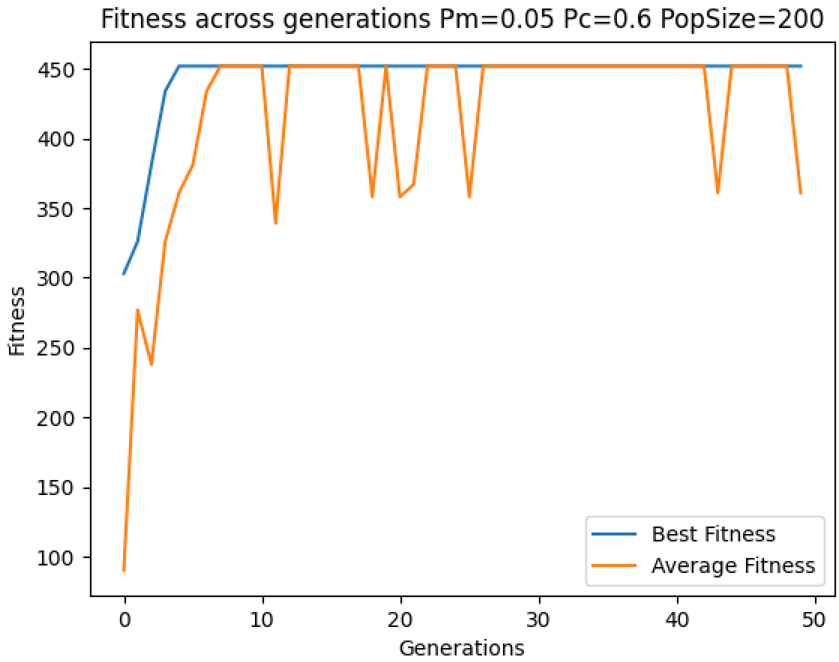
9. Appendix

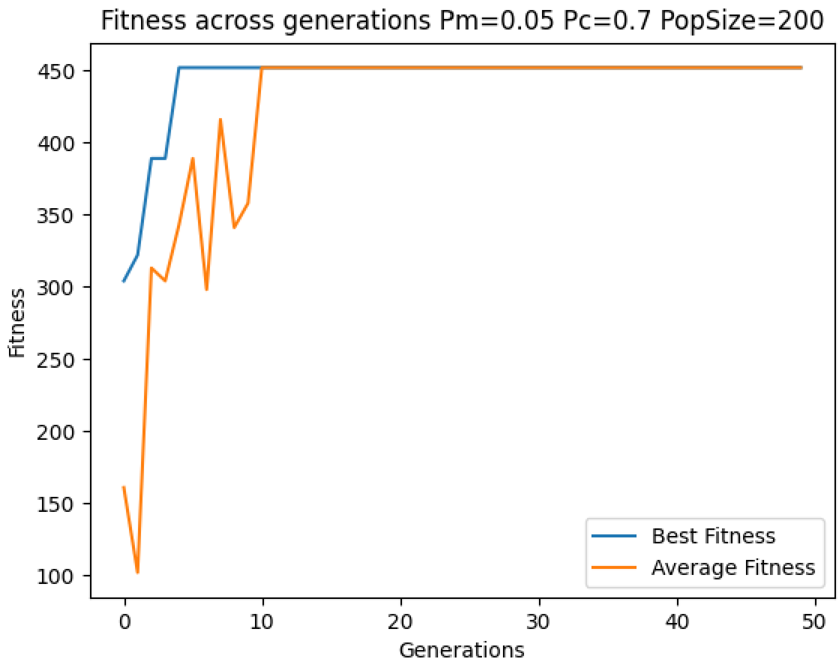
Graphs for Generational Performance across all parameters:

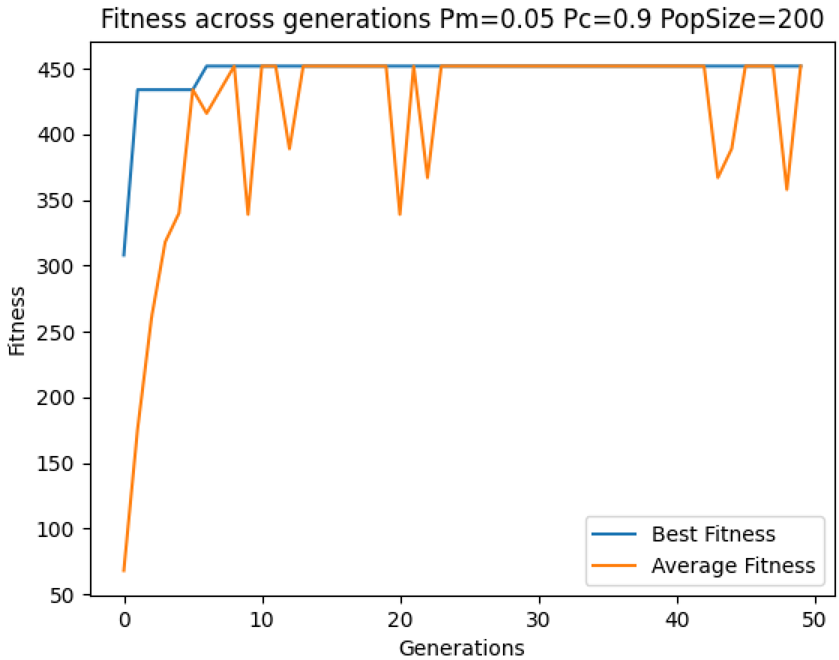


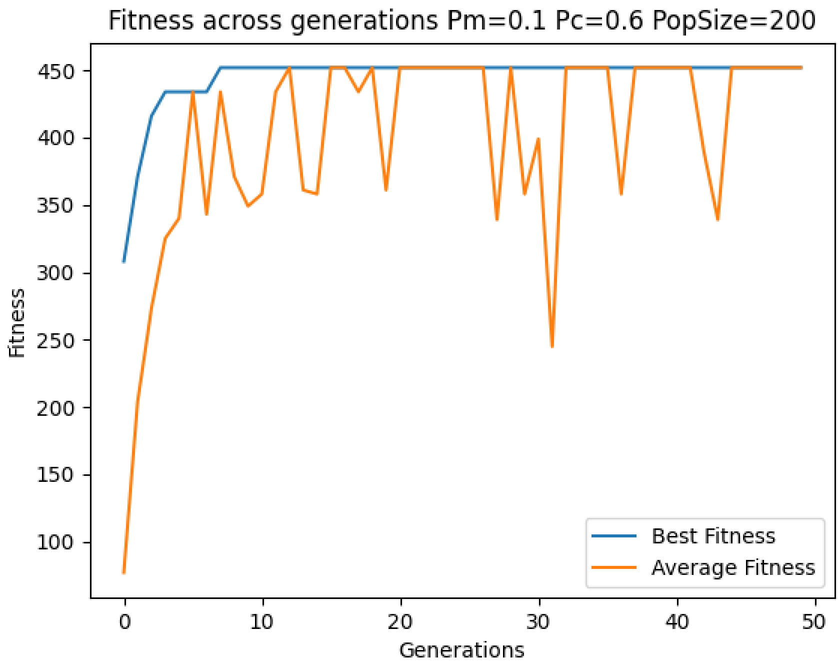

Fitness across generations Pm=0.01 Pc=0.7 PopSize=200
450
400
350
300
250
200
150
Fitness
0
10
20
30
40
50
Generations
Best Fitness
Average Fitness

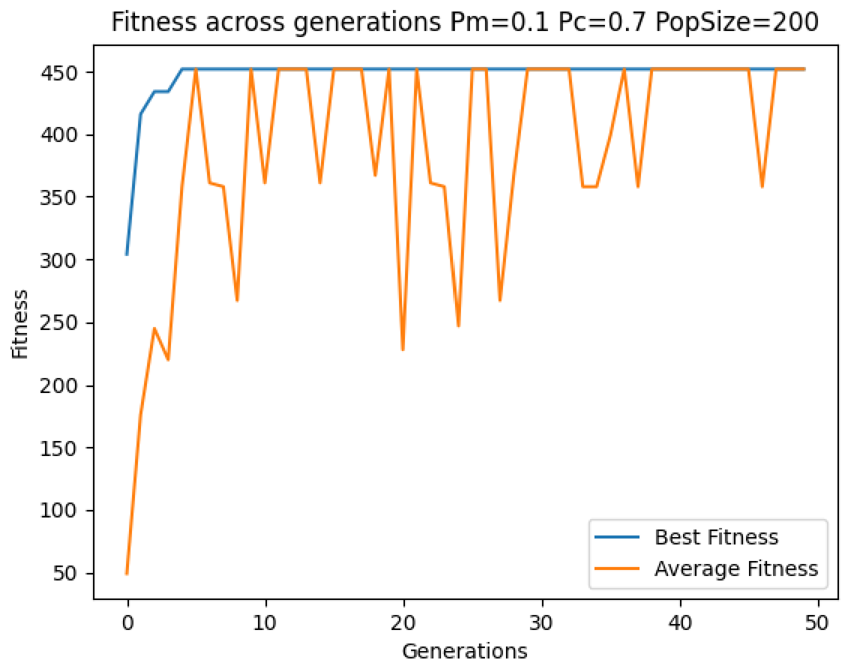

Fitness across generations Pm=0.01 Pc=0.9 PopSize=200
450
400
350
300
250
200
150
Fitness
0
10
20
30
40
50
Generations
Best Fitness
Average Fitness

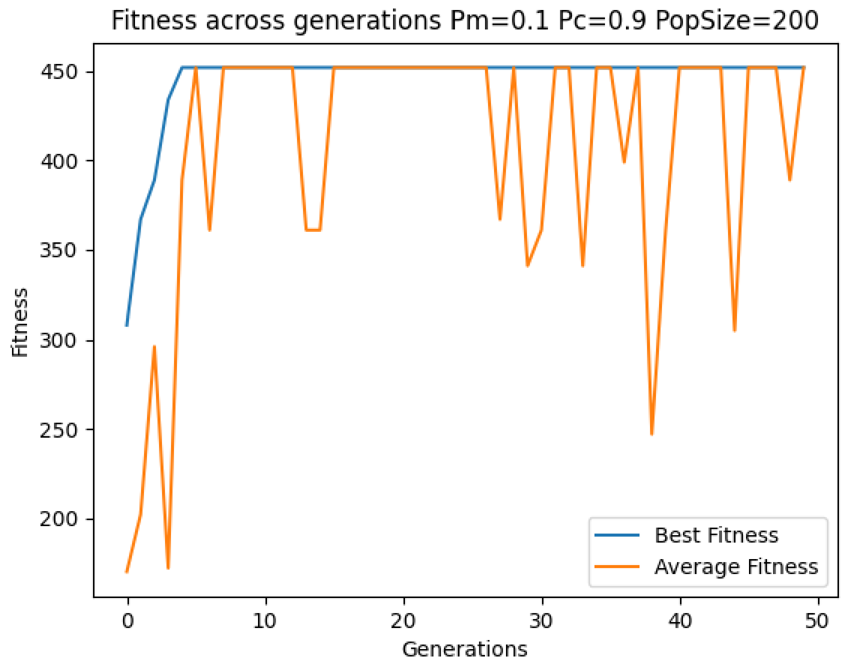

Fitness across generations Pm=0.05 Pc=0.6 PopSize=200
Fitness
Generations
Best Fitness
Average Fitness
450
400
350
300
250
200
150
100
0
10
20
30
40
50

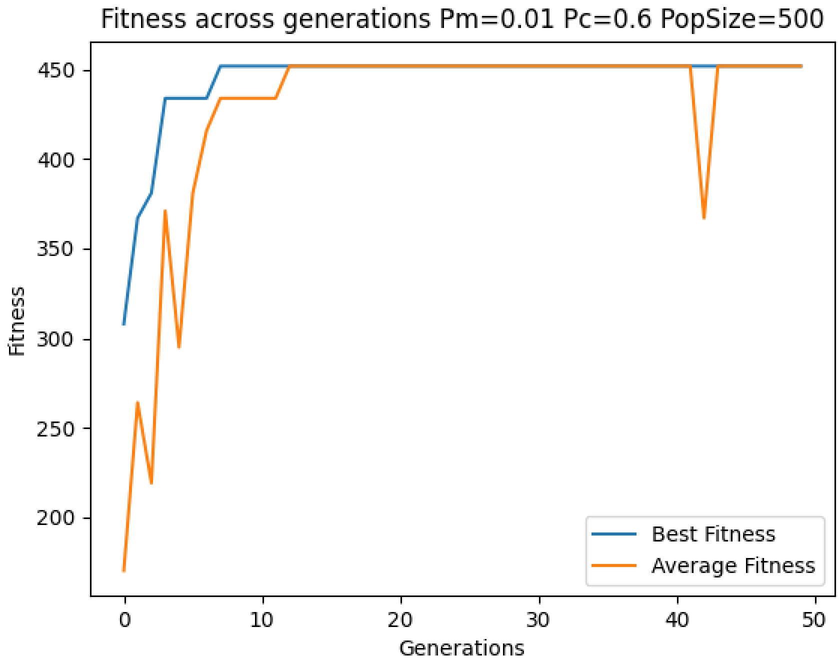

Fitness across generations Pm=0.05 Pc=0.7 PopSize=200
Fitness
Generations
Best Fitness
Average Fitness
450
400
350
300
250
200
150
100
0
10
20
30
40
50

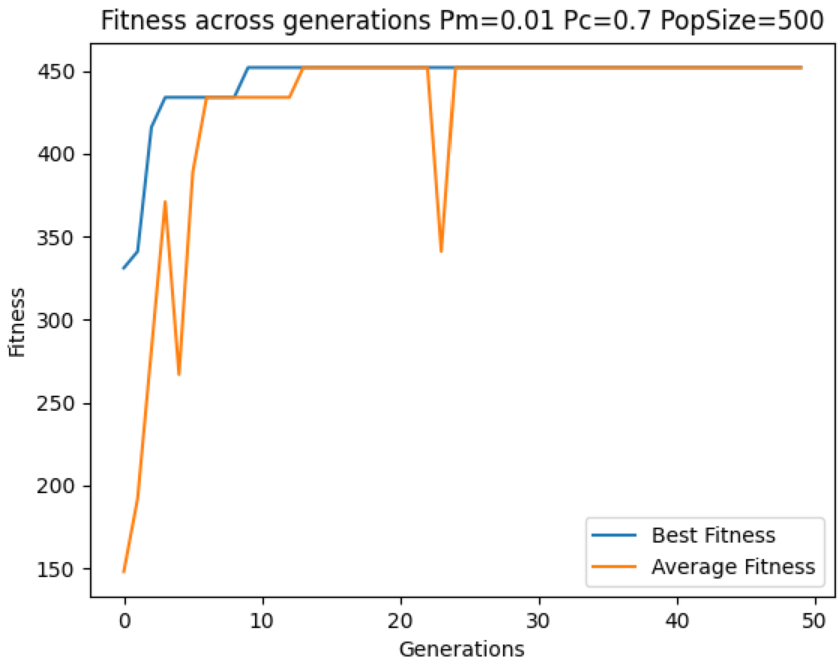

Fitness across generations Pm=0.05 Pc=0.9 PopSize=200
Fitness
Generations
Best Fitness
Average Fitness
450
400
350
300
250
200
150
100
50
0
10
20
30
40
50

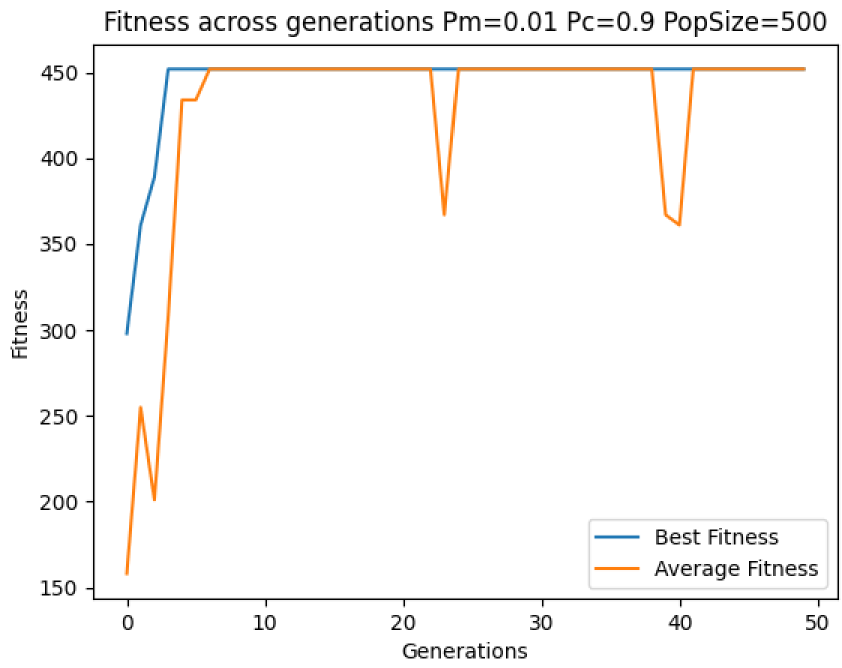

Fitness across generations Pm=0.1 Pc=0.6 PopSize=200
Fitness
Generations
Best Fitness
Average Fitness
450
400
350
300
250
200
150
100
0
10
20
30
40
50

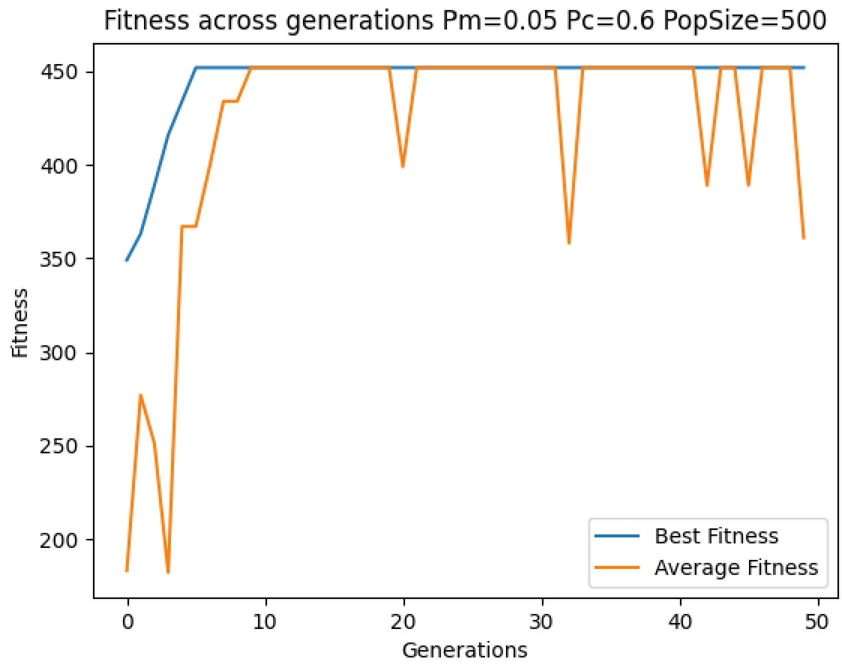

Fitness across generations Pm=0.1 Pc=0.7 PopSize=200
Fitness
Generations
Best Fitness
Average Fitness

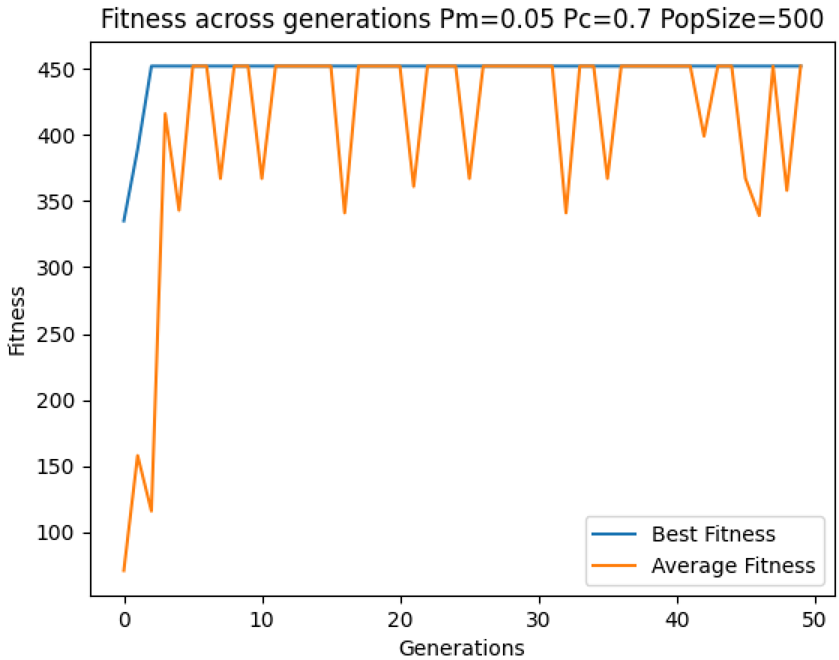

Fitness across generations Pm=0.1 Pc=0.9 PopSize=200
Fitness
Generations
Best Fitness
Average Fitness

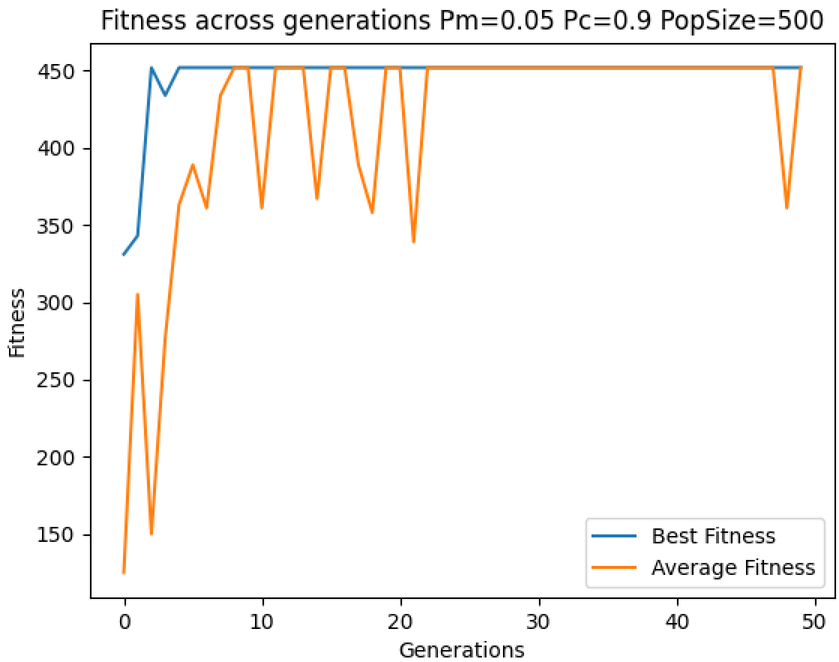

Fitness across generations Pm=0.01 Pc=0.6 PopSize=500
Fitness
Generations
Best Fitness
Average Fitness
450
400
350
300
250
200
0
10
20
30
40
50

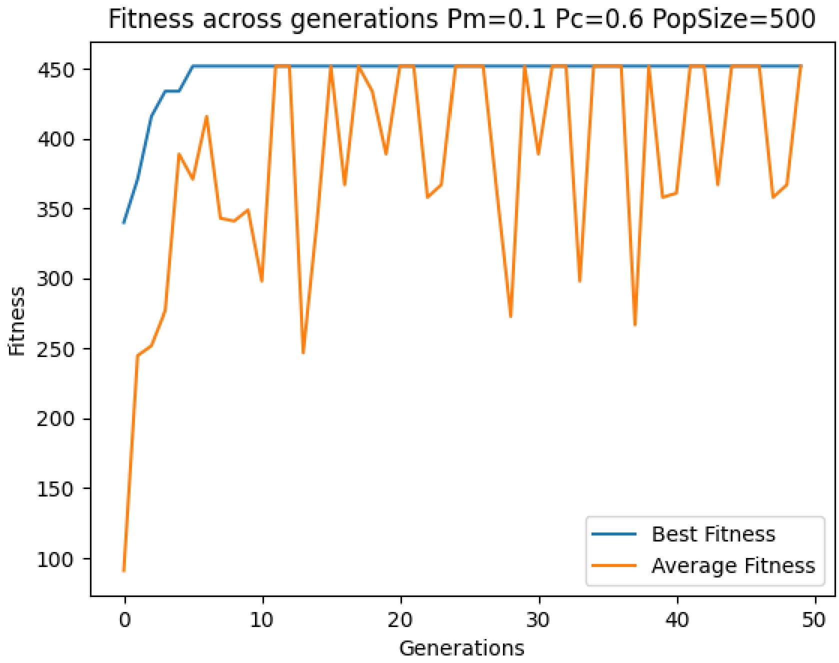

Fitness across generations Pm=0.01 Pc=0.7 PopSize=500
Fitness
Generations
Best Fitness
Average Fitness
450
400
350
300
250
200
150
0
10
20
30
40
50

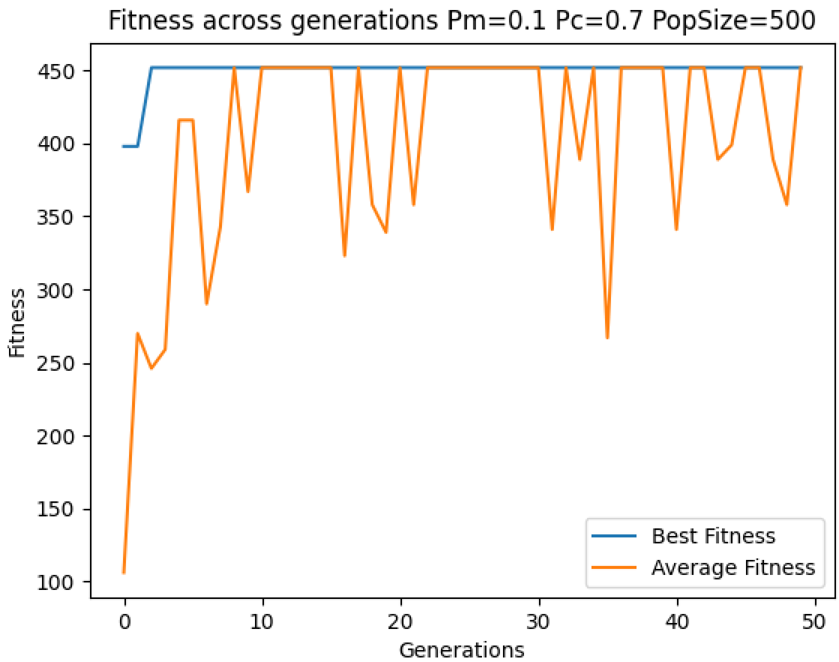

Fitness across generations Pm=0.01 Pc=0.9 PopSize=500
450
400
350
300
250
200
150
Fitness
0
10
20
30
40
50
Generations
Best Fitness
Average Fitness

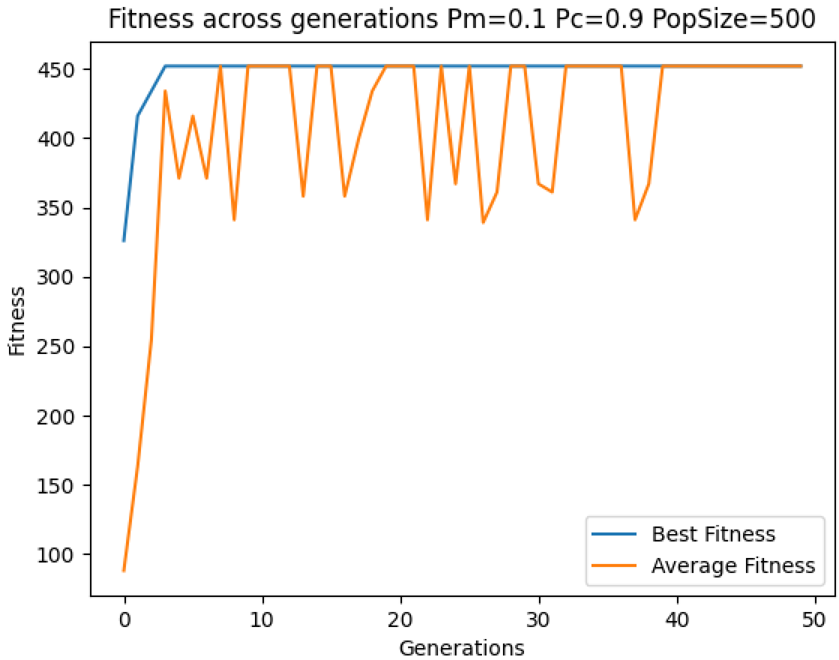

Fitness across generations Pm=0.05 Pc=0.6 PopSize=500
450
400
350
300
250
200
Fitness
0
10
20
30
40
50
Generations
Best Fitness
Average Fitness

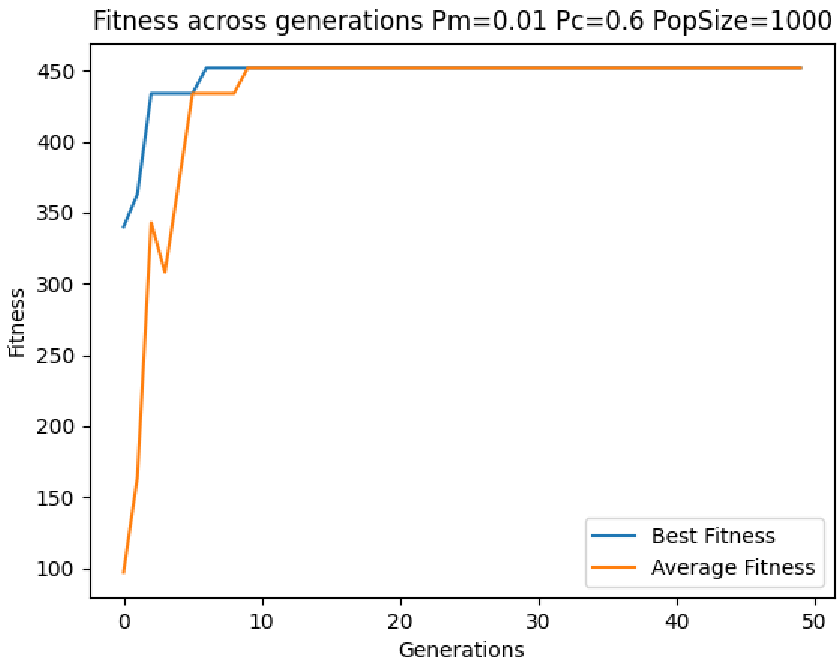

Fitness across generations Pm=0.05 Pc=0.7 PopSize=500
Fitness
Generations
Best Fitness
Average Fitness

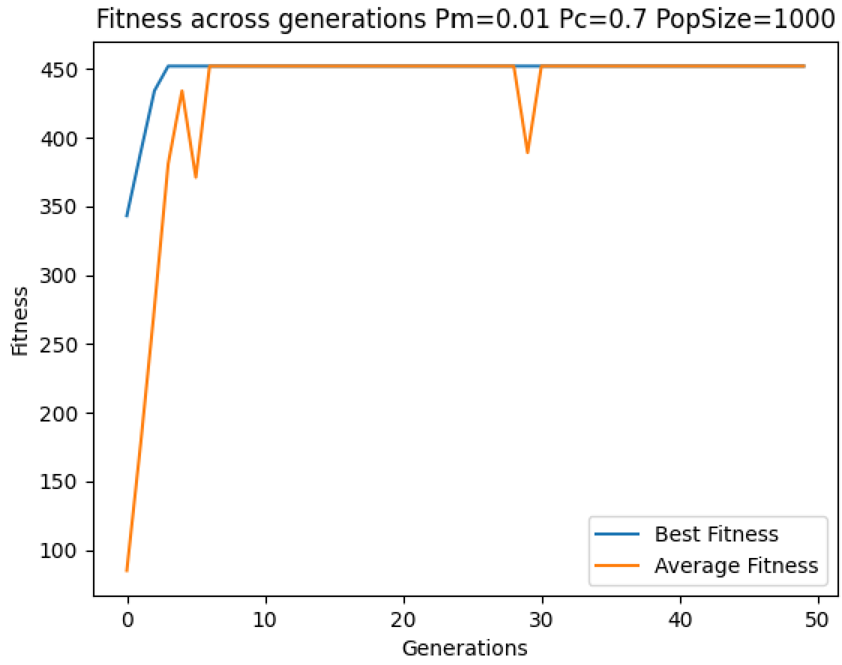

Fitness across generations Pm=0.05 Pc=0.9 PopSize=500
Fitness
Generations
Best Fitness
Average Fitness

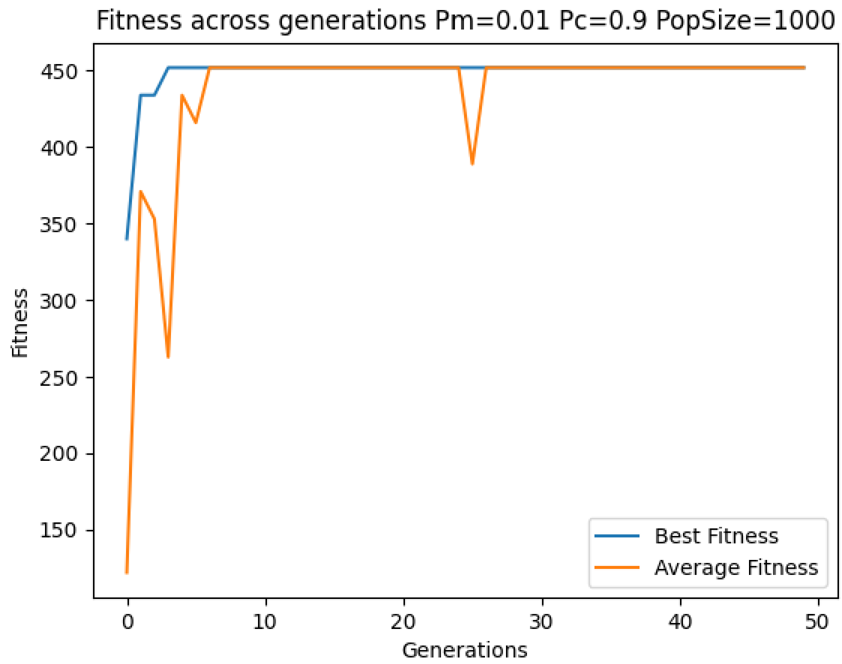

Fitness across generations Pm=0.1 Pc=0.6 PopSize=500
Fitness
Generations
Best Fitness
Average Fitness
450
400
350
300
250
200
150
100
0
10
20
30
40
50

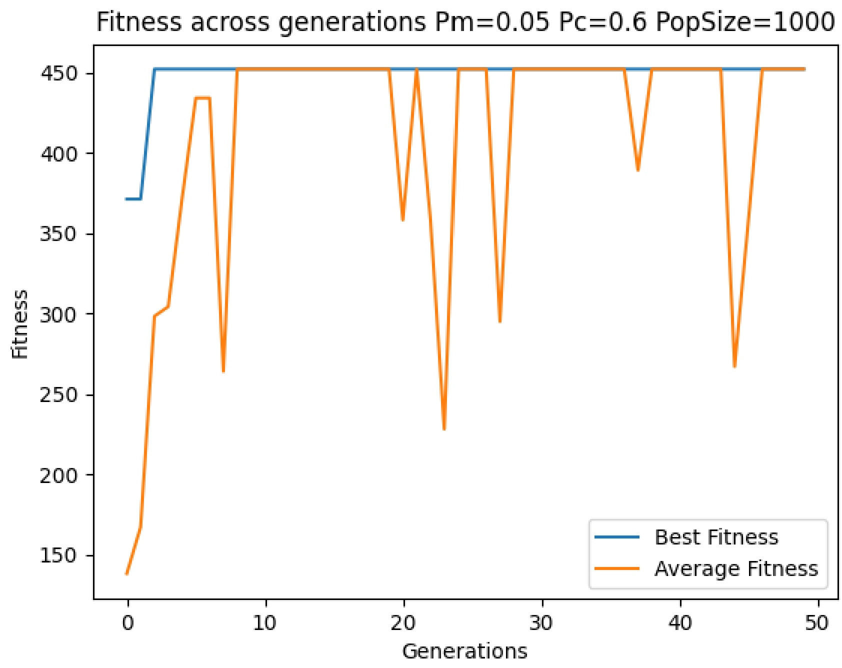

Fitness across generations Pm=0.1 Pc=0.7 PopSize=500
Fitness
Generations
Best Fitness
Average Fitness
450
400
350
300
250
200
150
100
0
10
20
30
40
50

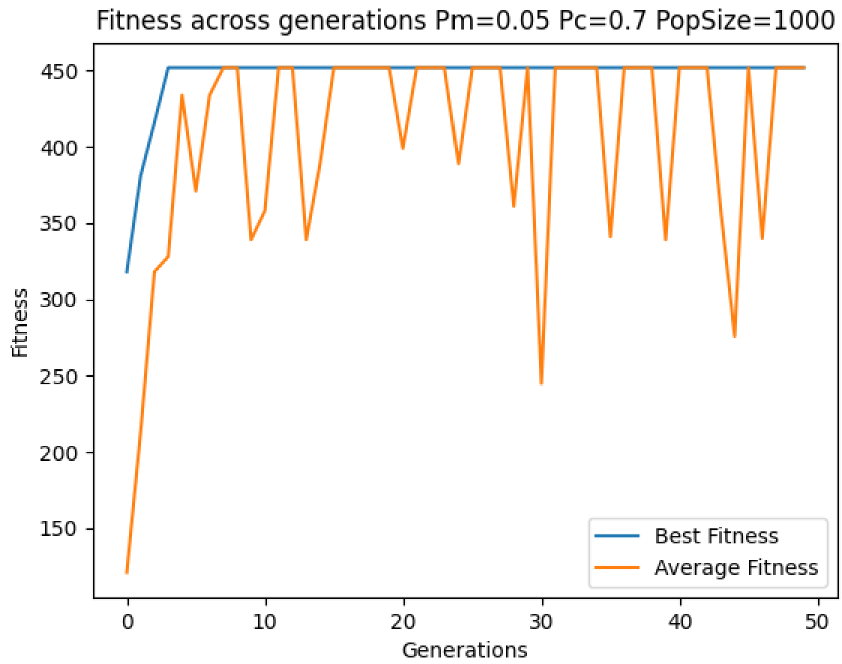

Fitness across generations Pm=0.1 Pc=0.9 PopSize=500
Fitness
Generations
Best Fitness
Average Fitness

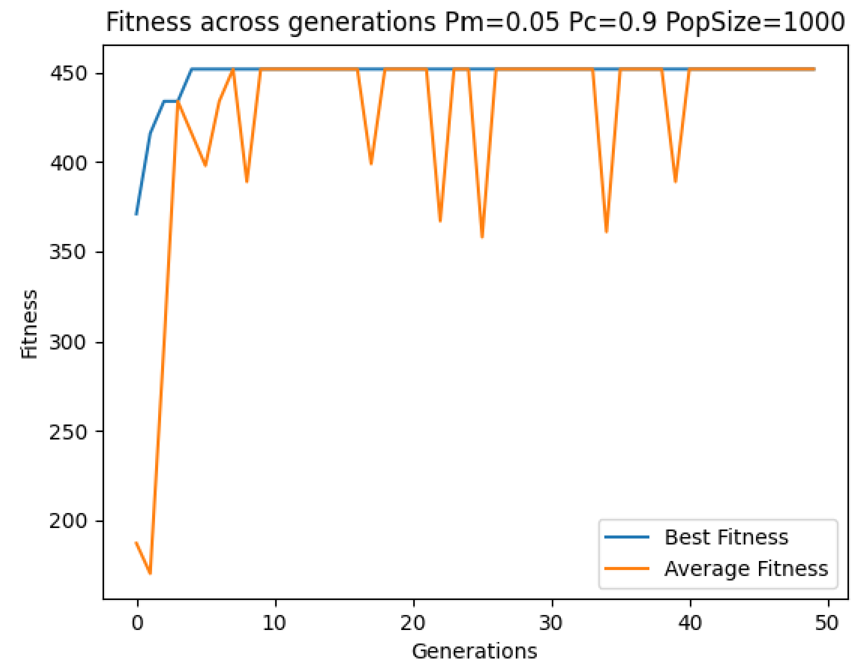

Fitness across generations Pm=0.01 Pc=0.6 PopSize=1000
Fitness
Generations
Best Fitness
Average Fitness

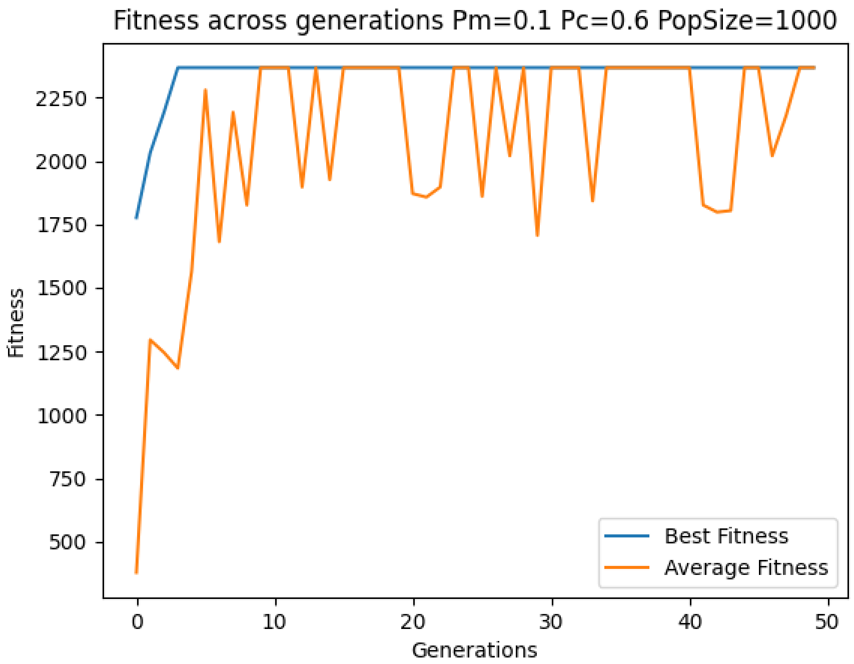

Fitness across generations Pm=0.01 Pc=0.7 PopSize=1000
Fitness
Generations
Best Fitness
Average Fitness

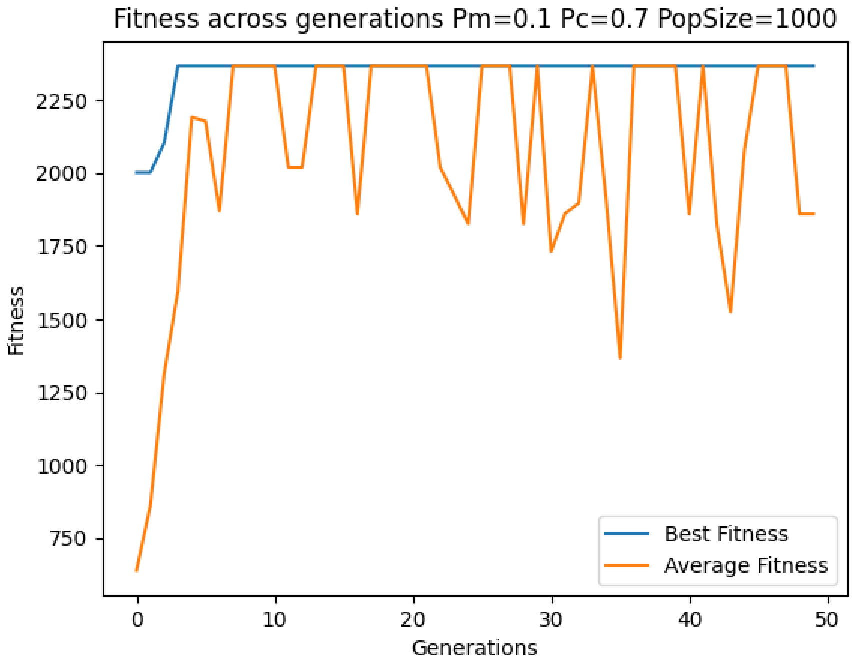

Fitness across generations Pm=0.01 Pc=0.9 PopSize=1000
Fitness
Generations
Best Fitness
Average Fitness

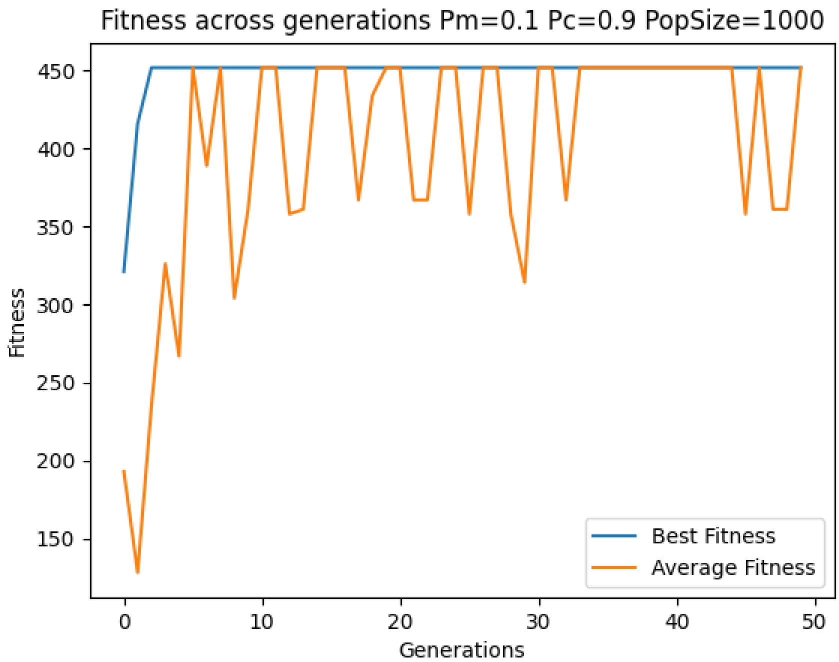

Fitness across generations Pm=0.05 Pc=0.6 PopSize=1000
450
400
350
300
250
200
150
Fitness
0
10
20
30
40
50
Generations
Best Fitness
Average Fitness


Fitness across generations Pm=0.05 Pc=0.7 PopSize=1000
450
400
350
300
250
200
150
Fitness
0
10
20
30
40
50
Generations
Best Fitness
Average Fitness


Fitness across generations Pm=0.05 Pc=0.9 PopSize=1000
Fitness
Generations
Best Fitness
Average Fitness
450
400
350
300
250
200
0
10
20
30
40
50


Fitness across generations Pm=0.1 Pc=0.6 PopSize=1000
Fitness
Generations
Best Fitness
Average Fitness
2250
2000
1750
1500
1250
1000
750
500
0
10
20
30
40
50


Fitness across generations Pm=0.1 Pc=0.7 PopSize=1000
Fitness
2250
2000
1750
1500
1250
1000
750
0
10
20
30
40
50
Generations
Best Fitness
Average Fitness


Fitness across generations Pm=0.1 Pc=0.9 PopSize=1000
Fitness
450
400
350
300
250
200
150
0
10
20
30
40
50
Generations
Best Fitness
Average Fitness