Genetic Algorithms Report

Aimilia Palaska 10453 - aimiliapm@ece.auth.gr
Academic Supervisor: Anastasios Tefas

May 30, 2025

Introduction

This report is created as part of the Computational Intelligence & Deep Reinforcement Learning course in Electrical and Computer Engineering department of Aristotle University of Thessaloniki. It investigates the behavior and performance of a Genetic Algorithm (GA) [1] under various configurations and parameter settings, focusing on its convergence properties and schema propagation dynamics. The results provide insights into the stability and convergence behavior of the algorithm, highlighting the impact of selection pressure and genetic diversity introduced through crossover and mutation. Developed code in Python, along with detailed comments, can be found in the emily-palaska/GeneticFuzzyExercises GitHub Repository.

Exercise 9.1 - Linear Fitness

In this experiment, an instance of the abstraction class `genetic.abstraction.GeneticAlgorithm` was configured using binary chromosomes of length $l = 100$, represented as vectors. The fitness function was defined as

$$f_l(\mathbf{c}) = \sum_{i}^{l} c_i$$

effectively measuring the number of ones in a chromosome $\mathbf{c}$. Parent selection was performed using fitness-proportionate sampling (roulette wheel selection). The population size was set to $n = 100$, with one-point crossover applied at probability P_δ, and point mutation occurring at a fixed probability of $P_\mu = 0.001$.

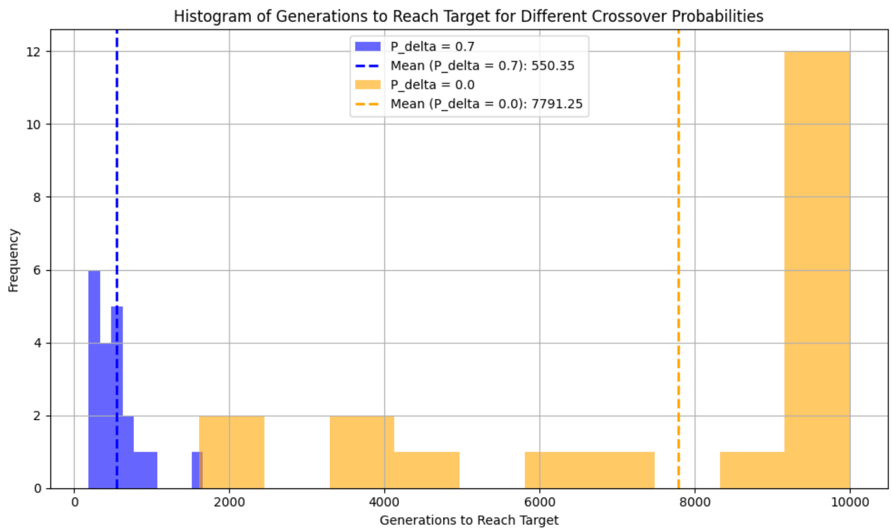

Figure 1: Convergence generation histogram of 20 runs for two $P\delta$ values for Exercise 9.2.

The objective was to evolve toward a target chromosome composed entirely of ones ($t_i = 1\ \forall i \in \{0, 1, ..., l-1\}$). . The algorithm iterates through selection, crossover, and mutation until the target appears in the population, with a termination cap at `max_generations`$=10^4$ to prevent infinite loops. The simulation was executed 20 times for $P_\delta = 0.7$ and $P_\delta = 0.0$, with the mean generation of first target appearance recorded.

Initial trials showed that, for both values of P_δ, the target chromosome was not found within 10^4 generations. To reduce computational cost while preserving the effect of crossover, chromosome length was reduced to $l = 50$. As shown in Figure 1, crossover significantly accelerated convergence, reducing the mean generation from 7791.25 to 550.35 and achieving a 93% improvement.

To analyze the impact of $P\delta$ more broadly, further experiments were conducted with values ranging from 0 to 1 in increments of 0.1, using chromosome length $l = 20$. Figure 2 illustrates that one-point crossover not only lowers the average convergence generation but also reduces variability across runs.

The results demonstrate that $P\delta$, the probability of crossover at a random point between two parents, is critical for accelerating fitness improvement. Without crossover ($P\delta$=0.0), evolution resembles resampling the current population, limiting novelty unless rare mutations occur. Consequently, several runs with this value reached the generation cap without producing the target chromosome.

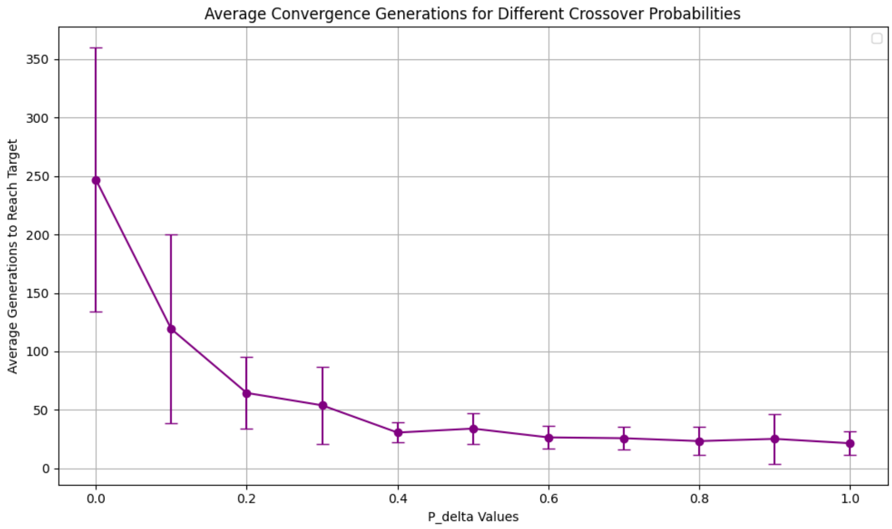

Figure 2: Convergence Generation average and deviation for different P_δ values for Exercise 9.1.

Exercise 9.2 - Binary Fitness

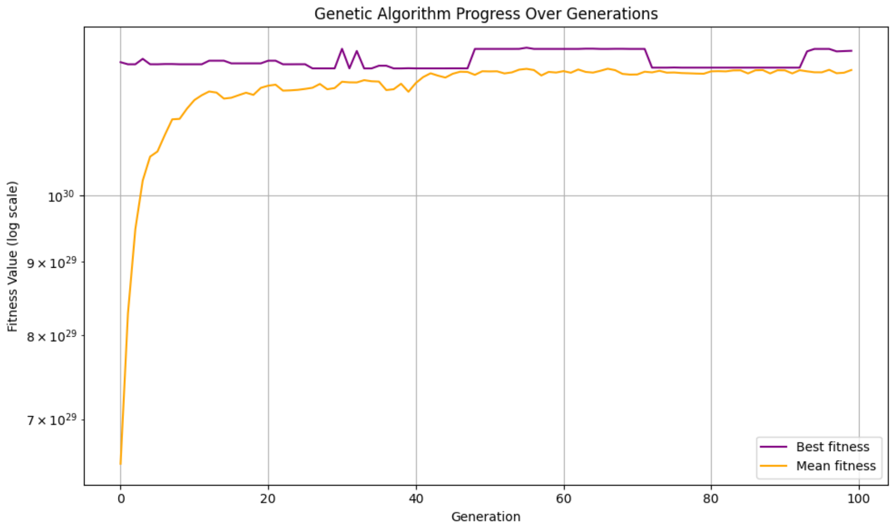

Figure 3: Progress of Best and Mean Fitness over 100 generations for Exercise 9.2.

In the second configuration, the same **GeneticAlgorithm** setup was employed, but with a different fitness function:

$$f_b(\mathbf{c}) = \sum_{i}^{l} 2^{c_{l-i}}$$

which interprets each chromosome $\mathbf{c}$ as its binary-to-decimal value. The population size remained at $n = 100$, with fitness-proportionate parent selection again employed. One-point crossover was applied with a fixed probability of $P_\delta = 0.7$, while the mutation probability remained at $P_\mu = 0.001$.

In the initial experiment, the genetic algorithm was executed for 100 generations, recording both best and mean fitness values per generation. The fitness progression is illustrated in Figure 3, where the best fitness remains relatively stable while the mean fitness converges to it around generation 60.

The effect of crossover probability P_δ was also investigated (Figure 4). Consistent with Exercise 9.1, crossover proves essential for rapid convergence. Notably, high crossover rates ($P_\delta > 0.8$) introduce instability, evidenced by oscillations in the mean fitness, while absence of crossover ($P_\delta = 0.0$) results in the slowest convergence.

Subsequent experiments examined the impact of varying the population size $n = 1, 10, 100, 1000$, with results shown in Figure 5. As anticipated, larger populations yield more stable behavior, with marginal improvements in robustness observed beyond $n = 100$.

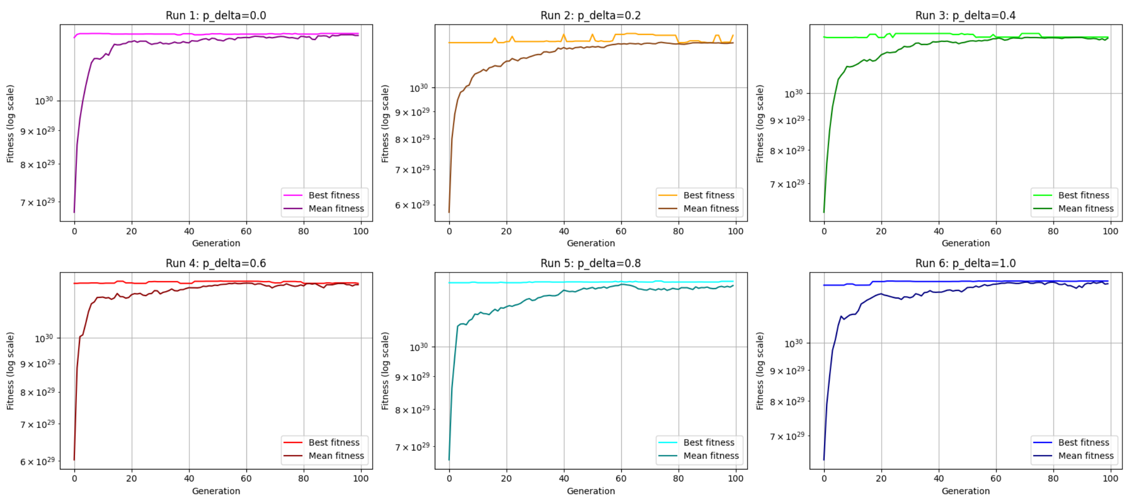

Figure 4: Fitness progression over 100 generations for varying crossover probabilities P_δ.

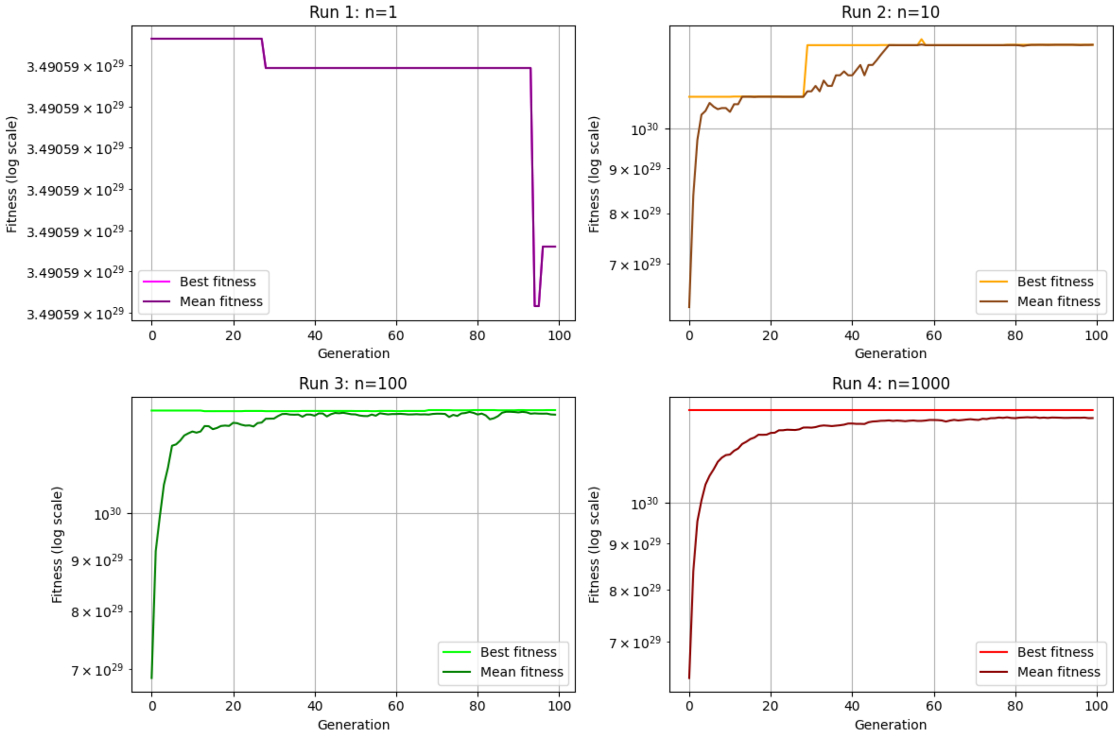

Figure 5: Fitness progression for varying population sizes over 100 generations for Exercise 9.2.

Finally, mutation probability P_μ was varied across six values, with results depicted in Figure 6. The data indicate that a small mutation rate enhances algorithmic stability compared to no mutation, though convergence speed is unaffected. Mutation rates exceeding $P_\mu = 0.1$ detrimentally affect performance, inducing oscillations in both mean and best fitness.

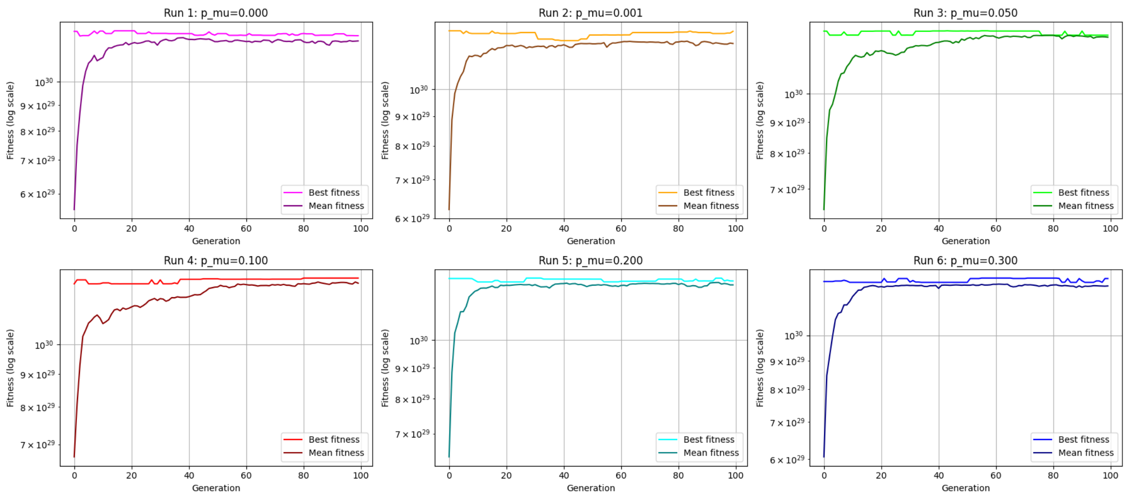

Figure 6: Fitness progression over 100 generations for varying mutation probabilities P_μ.

Exercise 9.3 - Schema Inspection

This experiment evaluates the Genetic Algorithm (GA) using the two defined fitness functions, analyzing the propagation of specific schemas over 100 generations. A schema denotes a subset of chromosomes characterized by fixed positions within binary strings, represented using symbols `0`, `1`, and `*` (wildcard). For instance, the schema `1**...` includes all chromosomes starting with a `1`. The frequency of each schema was tracked and is shown in Figure 7 for the linear and Figure 8 for the binary fitness function.

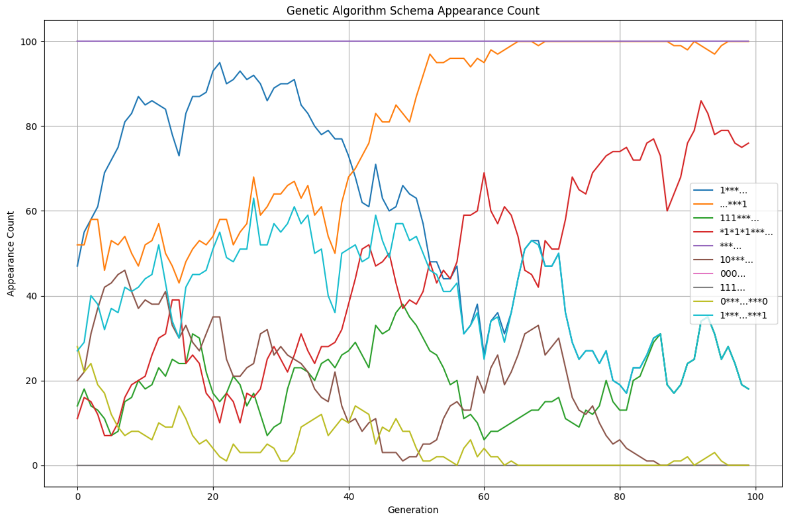

Figure 7: Schema frequencies over 100 generations using the linear fitness function.

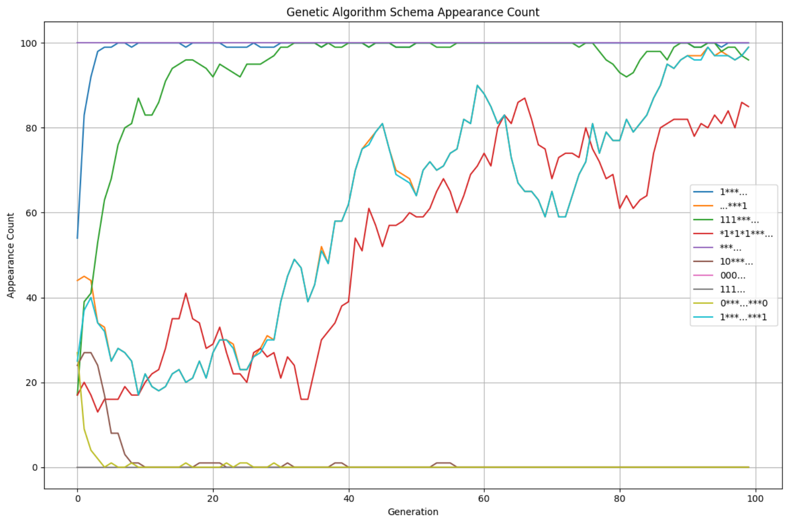

Figure 8: Schema frequencies over 100 generations using the binary fitness function.

The observed trends align with schema theory, accounting for the stochastic nature of GAs. The schema `***...` serves as a reference for the total population, while `111...` and `000...` correspond to the optimal and minimal fitness schemas, respectively. Schemas with a high proportion of `1`s generally increase in frequency, reflecting the convergence toward the optimal chromosome, whereas those with many `0`s decline. Moreover, more structurally complex schemas appear less frequently, as expected due to combinatorial rarity.

Comparison of fitness functions reveals notable differences. The linear function uniformly rewards each bit, resulting in similar schema trajectories for `1**...` and `...**1`. Conversely, the binary fitness heavily favors leading bits, with the first bit contributing $2^{l-1} = 2^{99} \approx 6.34 \times 10^{29}$, while the last contributes only $2^0 = 1$. This causes schemas like `1**...` to dominate early due to their overwhelming selection advantage. All in all, the binary fitness function exhibits greater stability, with schema frequencies stabilizing beyond the 20th generation and fewer oscillations.

Acknowledgments

Results presented in this work have been produced using the Aristotle University of Thessaloniki (AUTh) High Performance Computing Infrastructure and Resources. Additionally, the open-source language model ChatGPT [2] was utilized in parts of these experiments. It generated portions of the code which were then tested and analyzed, as well as enhanced the overall readability and clarity of this report.

References

[1] Wikipedia contributors, “Genetic Algorithm.” https://en.wikipedia.org/wiki/Genetic_algorithm, 2025. Accessed: 2025-05-23.

[2] OpenAI, “ChatGPT.” https://chat.openai.com/.