Evolutionary Computing

Assignment II: Generalist Agent

Diversity Management via Adaptive Blend Crossover with Delayed Exploitation in Evolution of EvoMan

Arina Fayzulina
(Student Number: 2803544)

Mateusz Kędzia
(Student Number: 2666752)

Joaquín Cardona Ruiz
(Student Number: 2819713)

Frank Le
(Student Number: 2669711)

October 20, 2023

1 ABSTRACT

Balancing exploration and exploitation of the solution space is a crucial element in achieving effective Evolutionary Algorithms (EA). To achieve this, we propose "Adaptive Blend Crossover with Delayed Exploitation," a method that actively adjusts this balance based on population diversity during the process of evolution, to extend the exploration phase and stave off premature convergence. Applying this approach in the evolution of generalist EvoMan agents, we found no significant improvement in the performance of generalist EvoMan agents compared to a non-adaptive baseline algorithm.

2 INTRODUCTION

In this study, we focus on the role of population diversity in Evolutionary Algorithms (EA) and propose a method of active parameter control that modulates it to achieve a prolonged period of exploration of the solution space, before a period of heightened exploitation. This biphasic adaptive mechanism, "Adaptive Blend Crossover with Delayed Exploitation", adjusts the alpha (α) value of the Blend Crossover operator according to the population diversity to maintain high levels of diversity during the beginning of the evolution process, to allow for diverse solutions to be explored for longer and avoid premature convergence[1]. **Our central research question**, then, poses: *"Does the incorporation of Adaptive Blend Crossover with Delayed Exploitation influence the performance of generalist EvoMan agents evolved through evolutionary algorithms?"* We seek to examine this question because we expect a problem like the evolution of generalist agents for EvoMan to be prone to local optima, where highly-fit strategies that are easy to converge to can be far from the optimal solution. Our previous study [3] showed that maintaining diversity didn't always result in more fit individuals. Hence, we revisit this issue with an adaptive approach in a generalist environment, where we expect diversity to play an even more crucial role. We conduct experiments with two EAs, one using Adaptive Blend Crossover and Delayed Exploitation. We analyze the highest total gain values achieved by the evolved agents and conduct two-way ANOVA tests to assess the impact of our approach.

3 ALGORITHMS DESCRIPTION

3.1 Baseline Algorithm

We utilize the DEAP Framework to simplify the development of the EAs, making use of its built-in functions for population management, genetic operators, and selection strategies [4]. Our individuals' genotype consists of an array of weights that get mapped into the neural network provided by the default player controller. Individuals are evaluated according to the fitness function given, expressed by $f = 0.9(100 - e) + 0.1p - \log t$, where p represents the player's current energy level, e denotes the enemy's current energy level, and t signifies the total number of timesteps required to conclude the fight[5]. We have implemented a generational "comma strategy" algorithm where each generation completely replaces the previous one. The offspring is produced through Blend Crossover. This method was chosen due to the control it provides over the expected distance between the offspring and their parents, which is used for managing diversity. The population first undergoes a parent selection stage to determine which individuals get to crossover. Those selected then produce a number of offspring larger than the population size, which undergo mutation using the gaussian mutation method, chosen for its frequent small changes to genes, and less frequent (but still possible) large, radical changes that help with exploration of new solutions. A survivor selection stage is then performed, which establishes the population for the next generation, reducing the number of offspring to the appropriate population size. Both selection stages are performed using non-replacing tournament selection. Tournament selection contributes to a low selection pressure environment, allowing for low-fitness individuals to maintain an attainable chance of getting selected, which, along with implementing non-replacement to avoid identical individuals, encourages a high diversity of solutions in each population.

3.2 Diversity Management through Delayed Exploitation and Adaptive Blend Crossover

The Blend Crossover operator selects a uniformly random value for each gene that falls between the corresponding values of its two parents, plus or minus a proportional margin given by a parameter α, which expands the range proportionally to the distance (d) between the parents' gene values. In our approach, we propose dynamically adjusting the value of α in each generation based on population diversity, as measured by the Shannon Entropy of the population genotype. The function used to update is:

$$\alpha' = \frac{3}{e^{C \cdot H(P)}}$$

where $H(P)$ is the Shannon entropy of the population genotype, and C is a configurable hyperparameter (Diversity Bias), which indirectly determines the level of diversity which will be stably maintained while the adaptive alpha is enabled. The algorithm begins in its "Exploration phase". An Initial α value is used until the diversity of the population falls below a certain Diversity Threshold. Then, α begins to be adjusted according to the function defined above. After a certain number of generations (Exploitation Delay), the algorithm moves to an "Exploitation phase". α stops being adjusted and is fixed to a lower Base α value. This approach ultimately results in the algorithm being able to maintain a high level of diversity during a period of time, before switching to emphasizing exploitation by lowering α to allow for population convergence.

4 EXPERIMENTAL SETUP

In this section, we describe our experimental setup, including model parameters and evolutionary algorithm characteristics.

4.1 Parameter tuning

We fine-tuned the model parameters with the **dual annealing global optimization** method, configured for minimizing the negative of the fitness value.

The number of hidden neurons, the population size, number of parents selected, and the number of children per parent, were set manually in accordance with computational resource constraints.

Table 1: Parameters utilized for the EAs

Parameter	Value
Hidden neurons	10
Population size	100
Parents Selected	50
Children per parent	4
Mutation probability	0.05
Mutation parameters (μ, σ, p)	(0,4,0.3)
Parent selection Tournament size	12
Survivor selection Tournament size	12
Type of Algorithm	Generational
Generations	75
Blend crossover α (model 1)	0.25
Base α (model 2)	0.1
Initial α (model 2)	0.25
Diversity bias (model 2)	0.9
Diversity Threshold (model 2)	2.3
Exploitation Delay (model 2)	25

The mutation probability and mutation parameters, as well as both tournament sizes, were found using the parameter tuning method described above. Reasonable constraints were applied to the parameter tuning algorithm to keep values within acceptable bounds. To minimize resource expenditure on parameter tuning, the procedure was run without adaptive blend crossover enabled. The values specific to model 2 (base α, initial α, diversity bias and threshold, and exploitation delay) were determined by subsequent experimentation, keeping the other parameters fixed. Finally, the number of generations was set such that it allows both models against both sets of enemies to converge on a solution.

4.2 Enemy Selection

We selected 2 enemy groups based on insights from Da Silva Miras De Araújo and De França (2016) [5] and our gameplay experiences. The first group includes simple enemies, Airman and Metalman (enemies 2 and 5), defeated with timely jumps. The second group, consisting of Airman, Heatman, Crashman, and Bubbleman (enemies 2, 4, 6, 7), offers a diverse set of challenges. Choosing 2 distinct groups allows us to analyze the models' behaviors in different circumstances. For parameter tuning, we focused on Enemies 2 and 5, because of their straightforward behavior and computational efficiency.

4.3 Experimental Design and Data Collection

The experiment includes two Evolutionary Algorithms: one using adaptive blend crossover with delayed exploitation and the other without it. Both models will undergo 10 rounds of training and testing on the two separate enemy groups, resulting in 40 observations. This data will be analyzed to address our research question. For each of the runs, we tracked several essential metrics. "Mean Fitness" recorded the average population fitness, while "Max Fitness" captured the highest individual fitness value achieved each generation. We also stored the weights for the individual with the highest fitness (Best Individual) in each run, so that they can be re-evaluated. The Mean Fitness and Max Fitness data were aggregated by averaging across each generation in all 10 runs, giving us a view of the algorithm's performance and progression. Additionally, the weights of the 40 Best Individuals were re-evaluated 5 times each, to quantify the resulting gain of the individuals against all 8 enemies, measured as the accumulated difference between the player's life and the enemies' life across all enemies.

5 RESULTS AND DISCUSSION

5.1 Results Visualization and Analysis

In this section, we visualize the aggregated data described previously. We also employ statistical tests to assess the significance of our findings and analyze the acquired results.

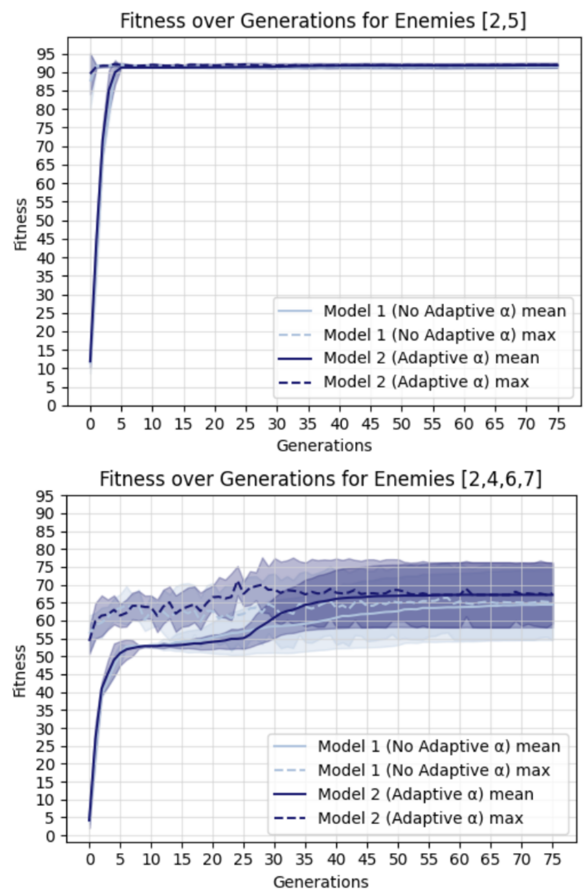

Figure 1: Mean and maximum fitness averages with standard deviations across generations for models with and without adaptive blend crossover for 2 groups of enemies: (2, 5) and (2, 4, 6, 7).

Fig. 1 compares the mean and maximum fitness values across generations for both models within enemy groups (2, 5) and (2, 4, 6, 7). These measures represent averages across all 10 experiment runs, with shaded regions indicating the standard deviation. These visualizations show that both models behave extremely alike, especially towards latter generations, where they converge at very

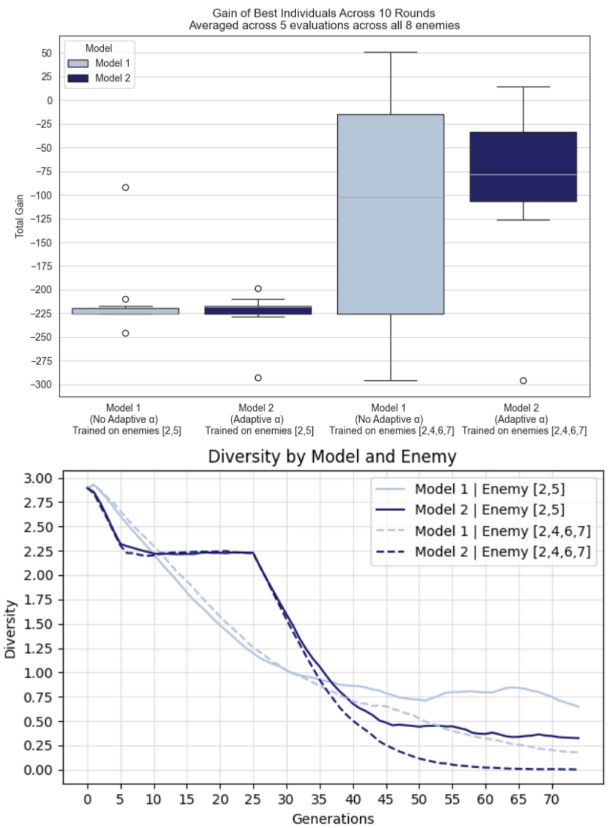

Figure 2: Box plots depicting gain distribution,Diversity evolution over generations

Table 2: Average health points (of 5 repetitions) of player and enemy for the best solution (as measured by total gain).

Enemy	1	2	3	4	5	6	7	8
Player's energy	0	76	0	0	61.6	0	55	28.6
Enemy's energy	90	0	20	40	0	20	0	0

similar fitness values. It is also of note that the maximum fitness increases very little across the generations, starting out with a value very similar to its final one. This is especially notable on the first enemy group (2, 5). This indicates that the EA presented here is not effective at evolving novel individuals with higher fitness values than the ones that get achieved during the random initialization process. It is to be expected that this effect is particularly visible on the first enemy group, as its comparative simplicity means that it's more likely that a close-to-optimal individual is generated during the random initialization process.

Fig.2.1 shows the gains from the top individuals in 10 runs of each model against each enemy. These individuals were re-evaluated 5 times, and their gains were averaged. Each box in the plot contains 10 data points, one for each best individual associated with the model and enemy. Statistical analysis will be provided below to determine whether the differences are statistically significant.

Fig 2.2 displays the population diversity, measured as the mean Shannon Entropy, averaged across all runs in each generation. It clearly illustrates the effect that Adaptive Blend Crossover for Delayed Exploitation has on the population diversity across generations, achieving a stable value that is held until generation 25, as set by the exploitation delay parameter, and then rapidly converging as the algorithm is allowed to focus on exploitation of the solutions found by that point. Shannon entropy is considered a superior measure of diversity according to existing research [6], as compared to the more common Mean Hamming Distance. It is notable, however, that we have found the latter to behave nearly identical to the former in our case.

Table 2 displays the average energy points for both the player and the enemies (across all enemy types) for the best solution found across the 40 runs performed. This solution was achieved using Model 1 (without adaptive alpha) and the set 2 of enemies (2, 4, 6, 7). On average, this solution successfully defeated 4 out of the 8 enemies, resulting in a general gain of 51.2, calculated as the difference between the sum of players' life and the sum of enemies' life across all 8 enemies.

5.2 ANOVA test

All tests used a significance level of $p < 0.05$. An ANOVA model examined differences between experimental groups with "gain" as the dependent variable and "model" (1,2) and "enemy-set" (1,2) as independent variables. There was no significant interaction between the independent variables ($p = 0.376$).Analyzing average data, we found no strong evidence to reject the null hypothesis ($p = 0.717 > 0.05$) Analogously, two one-way ANOVA tests for the impact of algorithm choice (with and without adaptive alpha) on average gain for each enemy set showed that there is no statistically significant differences between the two algorithms for both sets (p-value = 0.4055 for set (2, 5) and p-value = 0.515 for Set (2, 4, 6, 7).

5.3 Comparison to Baseline Paper

In our baseline paper by Sofiane Achiche et al. [7], it was found that prioritizing early-stage exploration (using high α for blend crossover), mid-stage relaxation (α=0.5), and late-stage exploitation (low α) outperformed the reverse strategy. Although our experimental setup differed from the reference study, our results contrasted with their findings. Our model, in which population diversity reached a stable value and then converged by the end of the training, did not improve total gain for the EvoMan generalist agent.

6 CONCLUSION

To conclude, the study found that the "Adaptive Blend Crossover with Delayed Exploitation" mechanism, while effective in maintaining diversity and balancing exploration, did not significantly improve the performance of generalist EvoMan agents compared to a non-adaptive baseline algorithm. This outcome, as well as the difference between our findings and those of the baseline paper, can be explained by the unique characteristics of the EvoMan problem domain, suboptimal parameter tuning, or population characteristics that may influence the impact of the adaptive mechanism.

7 REFERENCES

[1] T. Gabor, T. Phan, and C. Linnhoff-Popien, "Productive fitness in diversity-aware evolutionary algorithms," Nat. Comput., vol. 20, no. 3, pp. 363–376, Sep. 2021, doi: 10.1007/s11047-021-09853-3.

[2] C. Segura, A. Hernandez-Aguirre, S. Valdez Peña, and S. Botello, "The Importance of Proper Diversity Management in Evolutionary Algorithms for Combinatorial Optimization," vol. 663, pp. 121–148, Aug. 2017, doi: 10.1007/978-3-319-44003-3.6.

[3] A. Fayzulina, F. Le, J. Cardona, and M. Kędzia, "Diversity Preservation via Fitness Sharing in Evolution of EvoMan Agents," p. 3, Oct. 2023.

[4] "DEAP documentation — DEAP 1.4.1 documentation." Accessed: Oct. 19, 2023. [Online]. Available: https://deap.readthedocs.io /en/master/

[5] K. da Silva Miras de Araujo and F. O. de Franca, "Evolving a generalized strategy for an action-platformer video game framework," in 2016 IEEE Congress on Evolutionary Computation (CEC), Jul. 2016, pp. 1303–1310. doi: 10.1109/CEC.2016.7743938.

[6] R. A. Fisher, "Statistical Methods for Research Workers," in Breakthroughs in Statistics, S. Kotz and N. L. Johnson, Eds., in Springer Series in Statistics. , New York, NY: Springer New York, 1992, pp. 66–70. doi: 10.1007/978-1-4612-4380-9.6.

[7] S. Achiche, L. Baron, and M. Balazinski, "Scheduling exploration/exploitation levels in genetically-generated fuzzy knowledge bases," in IEEE Annual Meeting of the Fuzzy Information, 2004. Processing NAFIPS '04., Banff, Alta., Canada: IEEE, 2004, pp. 401-406 Vol.1. doi: 10.1109/NAFIPS.2004.1336316.