Study and Implementation of Age-Fitness Pareto Optimization in Genetic Algorithms

Vishal Uttamchand Sancheti, B000877378
Department of Computer Science
Dalhousie University
Halifax, NS, Canada B3H 4R2
vs488310@dal.ca

***Abstract*— Premature convergence is one of the common problems in genetics algorithms. Based on the previous research one of the effective ways to solve this problem is to introduce "Age" as an explicit optimization criterion. In this project, I try to study and implement one such approach called Age-Fitness Pareto Optimization [1]. Along with the implementation, I compare and study the results with standard genetic programming. During implementation, I selected an unbalanced dataset and performed various hyper tuning on model parameters to improve performance for the selected dataset.**

I. INTRODUCTION

A genetic algorithm is a method to identify the solution for a given problem. Initially, a given number of solutions known as individuals are initialized that collectively form population, each individual is then evaluated for fitness, based on fitness at each generation new individuals are introduced by crossover and mutation this repeats for several generations till termination condition is reached. The termination condition generally is when required fitness is reached or the maximum number of generations reached.

A common problem in many applications of genetic algorithms is when the progress of the algorithm stagnates and solutions stop improving [2]. Expending additional computational effort in the evolution often fails to make any substantial progress. This problem is known as premature convergence [3].

The previous study suggests various solutions to solve premature convergence. The most common solution is to run the algorithm for many generations or to randomize and restart the algorithm. Both the solutions are effective but, increase the cost. The method suggested in Age-Fitness Pareto Optimization [1] is a modern approach that uses two-dimensional Age-Fitness Pareto space. The main object of this method is to achieve high fitness for low age.

II. RELATED WORK

The key factor to avoid premature convergence is to maintain diversity in a population. Early methods use heavy mutation to maintain diversity but this resulted in more damage compared to benefits. Some modern work deal with this by suggesting alternate methods such as deterministic crowding [4] replacing individual by similar performing descendent, hierarchical fair competition [5] maintaining subpopulation for different fitness range, and Age-Layered Population Structure (ALPS) [6] maintaining subpopulation for different genotypic age range.

All the previously suggested methods deal with reducing premature convergence for the existing algorithm. The Age-Fitness Pareto method proposed by researchers is a different take inspired by ALPS and uses genotypic age, single population, and tournament selection to identify the best solution.

III. METHODS

In this project, I implemented the Age-Fitness Pareto optimization algorithm and compared results with a standard genetic algorithm for a poker hand classification dataset [7].

A. Dataset

The dataset is a collection of the hand of five cards drawn out of a standard deck of 52 playing cards. The dataset represents each card with suit and rank thus, consists of ten predictive attributes and one goal attribute. The goal attribute is a class of 10 possible values representing 10 possible poker hands. The training dataset class distribution is as shown in Table I.

TABLE I
DATASET ANALYSIS

Class	Poker Hand	Instances
0	Nothing in hand	501209
1	One pair	422498
2	Two pairs	47622
3	Three of a kind	21121
4	Straight	3885
5	Flush	1996
6	Full house	1424
7	Four of a kind	230
8	Straight flush	12
9	Royal flush	3

Based on the poker game the dataset statistics are as shown in Table II. We can observe that the probability of Nothing, One pair, Two pair, and Three of a kind is high compared to others thus, any algorithm will identify them better.

TABLE II
DATASET STATISTICS

Poker Hand	# of hands	Probability
Royal Flush	4	0.00000154
Straight Flush	36	0.00001385
Four of a kind	624	0.0002401
Full house	3744	0.00144058
Flush	5108	0.0019654
Straight	10200	0.00392464
Three of a kind	54912	0.02112845
Two pairs	123552	0.04753902
One pair	1098240	0.42256903
Nothing	1302540	0.50117739

B. Algorithm

The implementation is a modified version of the algorithm suggested in [1]. The various components for the algorithm are:

1) Individual: Individuals are represented using linear genetic programming and each individual is a set of instructions.

2) Fitness: The fitness is the calculated average of class-wise accuracy.

3) Age: The age represents the genotypic age of an individual. The genotypic age of the new individual is one, and the genotypic age of a derived individual is the youngest age of part of the genotype.

4) Execution: Sampling is done on training dataset for every five generations with uniform probability and without replacement. The sample is also oversampled to make a uniform distribution.

For given population size, individuals are initialized with different sizes of the random instruction set. These individuals are evaluated against a sample of training dataset and rank them against age-fitness Pareto to identify dominated individuals.

For each generation, one new random individual and one new mutated individual is added and the population is checked against max population size and if exceeds dominated individuals are removed.

The algorithm runs for max number of generations with termination condition where it terminates if desired fitness is reached.

C. Experiment

The main aim of the experiment is to compare standard genetic programming and Age-Fitness Pareto optimization. To establish common grounds for comparison we use a similar poker dataset to find a multi-class classifier to classify poker hands for a given set of suits and ranks.

Parameters used in standard genetic programming are as shown in Table III.

TABLE III
STANDARD GP PARAMETERS

Population Size	100
Sample Size	100
Instruction Set Size	48 to 256
Output Registers	10
Gap Percentage	30%
Crossover Probability	80%
Mutation Probability	30%

Parameters used in age-fitness pareto optimization are as shown in Table IV.

TABLE IV
AGE-FITNESS PARETO OPTIMIZATION PARAMETERS

Initial Population Size	95
Target Population Size	100
Sample Size	100
Instruction Set Size	48 to 256
Output Registers	10
Crossover Probability	80%
Mutation Probability	30%

IV. RESULTS

Experiments were performed several times to identify a common pattern for each algorithm and one such result for both is selected for the comparison.

A. Standard Genetic Programming

The method mostly failed to converge to global optima but, at times reached global optima in few generations. The selected result is from a case where it failed to reach global optima and converged to local optima.

The training statistics for the result are as shown in Table V and Fig 1. We can observe it reached fitness of 0.18 in 1000 generations and cost us 1min and 46s.

TABLE V
TRAINING STATISTICS

Generations	1000
Fitness	0.18
CPU time	1min 46s
Wall time	1min 47s

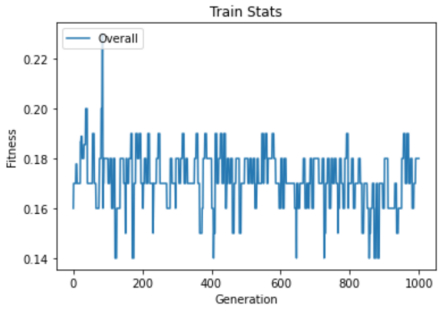

Fig. 1. Fitness-Generation Graph

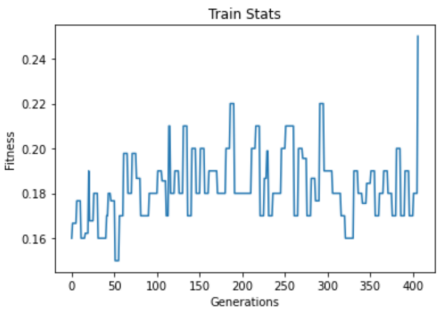

Fig. 2. Fitness-Generation Graph

The best classifier identified in training when applied on test dataset the result are as shown in Table VI. We can observe it classifies with an accuracy of 2.85% and identifies only the first two classes.

TABLE VI
TEST STATISTICS

Accuracy	2.85%
Detection Rate (DR)	10.11%
Class 1 DR	62.35%
Class 2 DR	38.78%
Class 3 DR	0.0%
Class 4 DR	0.0%
Class 5 DR	0.0%
Class 6 DR	0.0%
Class 7 DR	0.0%
Class 8 DR	0.0%
Class 9 DR	0.0%
Class 10 DR	0.0%

B. Age-Fitness Pareto Optimization

The method generally converges to global optima but, at times it needed more generations. The selected result is from a case where it reached global optima within 1000 generations.

The training statistics for the result are as shown in Table VII, Fig. 2, and Fig. 3. We can observe it reached fitness of 0.25 in 1000 generations and cost us 30.2s.

TABLE VII
TRAINING STATISTICS

Generations	409
Fitness	0.25
Age	293
CPU time	30.2 s
Wall time	30.4 s

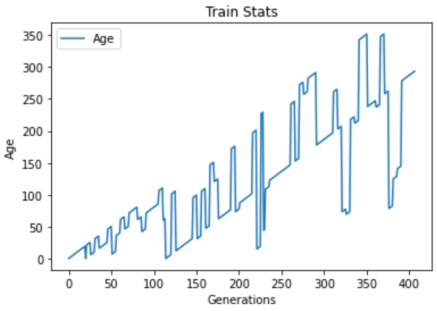

Fig. 3. Age-Generation Graph

The best classifier identified in training when applied on test dataset the result are as shown in Table VIII. We can observe it classifies with an accuracy of 0.81% and identifies only the first four classes.

TABLE VIII
TEST STATISTICS

Accuracy	0.81%
Detection Rate (DR)	8.9%
Class 1 DR	1.52%
Class 2 DR	44.99%
Class 3 DR	9.79%
Class 4 DR	32.87%
Class 5 DR	0.0%
Class 6 DR	0.0%
Class 7 DR	0.0%
Class 8 DR	0.0%
Class 9 DR	0.0%
Class 10 DR	0.0%

V. CONCLUSIONS

In conclusion, the Age-Fitness Pareto Optimization can solve premature convergence and be cost-effective but, at times the standard genetic programming performs well when it avoids premature convergence.

Overall, the concept of introducing age as an explicit optimization criterion can increase performance as young and well-performing individuals are not replaced with old and high-performing individuals.

VI. ACKNOWLEDGMENT

The project could not have been possible without the guidance of Dr. Malcolm Heywood.

A debt of gratitude is also owed to the researchers, Dr. Michael Schmidt and Dr. Hod Lipson, who drafted the implemented approach in their research paper.

Finally, I would like to thank my classmates at Dalhousie University for the Course CSCI6506, Summer'21, who always shared information and asked questions.

REFERENCES

[1] M. Schmidt and H. Lipson, "Age-fitness pareto optimization," in *GECCO*, vol. 8, 01 2010, pp. 543–544.

[2] G. Murphy and C. Ryan, *Manipulation of Convergence in Evolutionary Systems*. Springer US, 01 1970, pp. 33–52.

[3] C. Ryan, "Reducing premature convergence in evolutionary algorithms," 02 1970.

[4] S. Mahfoud, "Niching methods for genetic algorithms," vol. 51, 05 1995.

[5] J. Hu, E. Goodman, K. Seo, Z. Fan, and R. Rosenberg, "The hierarchical fair competition (hfc) framework for sustainable evolutionary algorithms," *Evolutionary computation*, vol. 13, pp. 241–77, 02 2005.

[6] G. Hornby, *A Steady-State Version of the Age-Layered Population Structure EA*. Springer US, 11 2009, pp. 87–102.

[7] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml