Age-Layered Population Structure

Andrew Pozzuoli
Brock University
St. Catharines, ON
6017735
ap15yl@brocku.ca

***Abstract*—Premature convergence is one of the major problems that can come up during evolutionary computation. This occurs when the individuals of a population converge onto a sub-optimal solution and progress in finding a better solution ceases. The only way to get a population out of a basin of attraction pulling all the individuals into the local optimum is to introduce new, randomly generated individuals. The Age-Layered Population structure is one such method. In order to give these newly created individuals a fighting chance against more evolved, fitter individuals, the population is segregated into layers based on an age attribute that measures how long solutions have been evolving. Competition is restricted to remain within these layers so younger individuals are not immediately dominated by older ones. This model has been shown to successfully reduce premature convergence. This paper explores ALPS in more detail, how it has been used, and variants to ALPS that have been employed.**

I. Introduction

One major issue arising from the use of evolutionary computation is premature convergence. This occurs when an evolutionary algorithm (EA) stagnates into a sub-optimal solution and progress in finding a better solution halts. This occurs when new individuals cannot be generated that move the population into other areas of the search space where better solutions could be found [1].

One common way of dealing with the problem of premature convergence is to conduct multiple runs of an evolutionary algorithm where each run uses a different random seed to create its initial random population [2]. With this approach, additional challenges come in the form of selecting when to restart.

An alternative to restarts is to regularly introduce new, randomly generated individuals into the population so that the algorithm never converges and constantly explores more of the search space. This method was used by Gregory Hornby in his 2006 paper *ALPS: The Age-Layered Population Structure for Reducing the Problem of Premature Convergence* in the form of the ALPS paradigm [1]. In this method, each individual is attributed with an age which measures how many generations the genetic material has been evolving. The population is segregated into layers, each with a maximum age allowed for individuals within. Recombination and mutation is restricted to take place within these layers and not across different layers so that younger individuals are not dominated by fitter solutions with older genetic material. This effectively gives the new, randomly generated individuals that are regularly added to the population a fighting chance against the evolved solutions.

This paper will detail the ALPS paradigm introduced by Hornby, go over how it has been applied and how it fares against other methods, and explore variants to standard ALPS.

II. Premature Convergence

The problem of premature convergence is a major issue in evolutionary computation. After some amount of time, the population converges onto a sub-optimal solution which is not improved with further iterations [3]. If the randomly generated initial population started around one basin of attraction, that population will never be able to explore outside of that region to find better solutions [1]. The major problems with this are that resources are wasted on iterations when no better solutions are going to be found and the run will never have any chance of finding the global optimum in a search space once it has converged on a mediocre solution.

A typical way of tackling this issue is to perform multiple runs with different seeds in order to create new, randomly generated initial populations [2]. This approach comes with an added challenge of deciding how long to run the algorithm before restarting. Too short and the population may not have enough time to reach an optimum. Too long and the population may converge earlier than the total generations leading to wasted time. This approach also can have the problem of wasted resources since entire populations are repeatedly discarded whenever a restart occurs [4]. Good solutions within each run are also unable to share parts of their solution with populations of other runs.

An alternative way of preventing premature convergence without restarting the algorithm is to regularly inject new, randomly generated individuals into the population. Hu and Goodman used this in their model called Hierarchical Fair Competition (HFC) [5]. Randomly generated individuals stand little chance of selection for recombination against the individuals of the population who have already evolved into higher fitness solutions so to protect the new individuals from being dominated, the population is separated into layers by fitness. Once an individual reaches a certain fitness, it can move up a layer. Hornby points out that HFC has a problem where individuals that have converged in a fitness layer can prevent other individuals from passing through that layer [1].

III. Age-Layered Population Structure

In order to improve HFC to allow for better integration of newly generated individuals, the ALPS paradigm was proposed by Gregory Hornby in his paper *ALPS: The Age-Layered Population Structure for Reducing the Problem of Premature Convergence* (2006) [1].

Like with HFC, new, randomly generated individuals are inserted into the population at regular intervals but with ALPS individuals have an additional attribute called *age* which is used to restrict competition rather than using fitness as in HFC.

Age is a metric that measures how long genetic material has been evolving in the population. A newly created individual starts with an age of 1 since the genetic material has just been randomly generated. Every generation where this individual is selected to produce offspring, its age increases by 1. Even if the individual is selected to breed more than once in a generation, the age only increases by 1 since it is meant to reflect the number of generations. Individuals that are offspring created through recombination and mutation have an age of 1 plus the age of its oldest parent, reflecting the age of the genetic material.

The population is segregated into layers by age. Each layer has a maximum age for individuals to be allowed to reside within it with no lower bound. The last layer can have individuals of any age in it. There are various methods of setting age limits of the layers. Hornby's paper gives three examples of linear, exponential, or polynomially increasing limits. Furthermore, an age-gap parameter which each limit is multiplied by is used to keep a manageable number of layers and to keep age limits separate enough so there is not segregation between individuals few generations apart. For example, with an exponentially increasing age limit, an age-gap set to 10, and 6 layers the age brackets are 0-10, 0-20, 0-40, 0-80, 0-160, and 0-∞.

The normal evolutionary algorithm runs within each layer where individuals are allowed to breed only with individuals of the current layer or the previous layer. This allows a smooth passing of offspring across different layers.

In order to insert new individuals into the population, an interval equivalent to the age-gap parameter is used to determine the number of generations before all individuals in layer 0 are replaced with new randomly generated individuals.

As with the fitness-layers of HFC, this separation ensures that newly generated individuals are not completely dominated by individuals who have evolved for longer. This method also solves the issue of HFC where layers whose individuals have converged into a local optimum prevent other individuals from crossing the layer. With ALPS, individuals will always move up the layers as the generations pass and will never get stuck in a layer like with HFC. The only exception would be if an individual is the global optimum where it will remain in the last layer and not be out-competed by better solutions.

In Hornby's paper, ALPS was compared to HFC and canonical EA on antenna design optimization. It was found that ALPS outperformed canonical EA by a large margin. It was also seen that HFC did not perform differently from EA in a statistically significant way showing that just introducing new individuals into the population and only restricting breeding by fitness does not work as well as the age parameter. These results suggest that segregating by age gives new solutions enough time to evolve and develop while HFC does not adequately develop the newly introduced solutions into the population well enough to be effective over canonical EA.

IV. Circuit Design

ALPS was used to aid in novel circuit design in McConaghy et al's paper *Genetic Programming with Design Reuse for Industrially Scalable, Novel Circuit Design* (n.d.) [6]. They adapted ALPS to be multi-objective with each layer using a non-dominated sorting genetic algorithm (NSGA-II). The objectives being optimized were in trustworthiness and in novelty in order to create innovative circuits that work. ALPS allowed for more exploration in creative, novel circuits since it constantly injected new, random individuals into the population. ALPS was also used to prevent premature convergence.

V. ALPS Variants

A. Steady State

Hornby also developed a steady-state version of the ALPS model in his 2010 paper *A Steady-State Version of The Age-Layered Population Structure EA* [3]. In this version, a steady-state EA is run in each of the age-layers. With the original ALPS model, age could be measured in generations but a steady-state EA does not use generations so a different measure of age needed to be used. Instead of using generations, steady-state ALPS (ALPS-SS) takes the number of evaluations that the genetic material has been around for and divides by the population size. Newly generated individuals keep track of the number of evaluations performed so far while offspring take the number of evaluations of their oldest parent. All in all, the measure of age is calculated by the following equation:

$$age = 1 + (evals_{current} - evals_{created})/popsize \quad (1)$$

Where $evals_{current}$ is the number of evaluations performed thus far, $evals_{created}$ is the number of evaluations performed at the time of the genetic material's creation, $popsize$ is the total number of individuals across all layers, and a constant of 1 is added so that newly generated individuals start with an age of 1.

With this new age measure, ALPS-SS works similarly to traditional ALPS where age-layers are determined by the implementer. An initial, randomly generated population is created and fitness values are assigned to individuals as normal. Layers are iterated over, updated, and new individuals are created to be inserted into the population.

ALPS-SS was compared with a standard, steady-state EA in antenna design optimization and evolving 3D tables from cubes. It was found that ALPS-SS greatly outperformed the standard EA in both tasks.

B. FSALPS: Feature Selection Age Layered Population Structure

Awuley & Ross' 2016 paper *Feature Selection and Classification Using Age Layered Population Structure Genetic Programming* [7] modified Hornby's ALPS model for the task of classification in a genetic programming (GP) context instead of an EA context. They presented the algorithm called Feature Selection Age Layered Population Structure (FSALPS) which counts the frequencies of features in the terminal nodes of GP trees to generate probabilities for terminal selection. These probabilities are used to select terminal nodes for generation of new individuals in layer 0. The idea behind this being that features that evolve up in the layers would be more refined and the use of their terminals would lead to improved performance in the newly created trees. This modified model performed significantly better than both canonical GP and ALPS in terms of feature reduction. ALPS and FSALPS both produced smaller, more efficient GP trees over standard GP. None of the models significantly performed more accurate classifications than any other.

C. *Age-Fitness Pareto Optimization*

Schmidt & Lipson's paper *Age-Fitness Pareto Optimization* modified the ALPS model to be multi-objective, optimizing for both fitness and age. Rather than using layers to separate the population, individuals are injected into a two dimensional space of both fitness and age. Even though the newest individuals are introduced into the same population as the more evolved, fitter solutions, they do not get dominated in the age dimension so they can have a chance to evolve and develop. They compared this approach to ALPS and deterministic crowding in the task of symbolic regression and results showed that age-fitness pareto optimization was able to find the target solution more often than the other methods. They also found that problems that were notoriously difficult to solve with ALPS were solved using age-fitness pareto optimization with one third of the computing effort.

VI. Conclusion

Age-layered population structure (ALPS) was explored along with variants. ALPS was shown to be an effective model for reducing premature convergence since it constantly inserts new, randomly generated individuals into the population. To prevent these new individuals from being dominated by fitter, more evolved individuals, and age attribute is added to measure how long the genetic material has been evolving and the population is separated into layers with similarly aged individuals allowed to breed only with members of their layer or the one previous. This method is general and can be applied to any optimization that uses randomness as a way of fighting premature convergence.

In addition to this, the solutions resulting from running ALPS are shown to often perform better than canonical EAs since the inserting of random individuals allows the population to explore more of the search space rather than getting stuck in a local optimum.

This is also an improvement over simply restarting the EA with a different seed since good genetic material can be shared with the population and there is no need to decide how long to run the EA for before restarting.

This does come at the cost of adding more tuning that the implementer needs to do before running the ALPS model. Namely, they must choose an age-gap and a structure for determining maximum ages for the layers. While typically the oldest parent is selected for determining the age of offspring, youngest parent has also been used but not explored in depth. Future work could be done in deciding a method for tuning these parameters.

Future work could also be to look into using the age parameter in different ways such as what Schmidt & Lipson did with age-fitness pareto optimization [4] or with using evolved solutions in higher layers to help with creating new individuals as with Awuley & Ross with FSALPS [7].

References

[1] G. S. Hornby, "ALPS: the age-layered population structure for reducing the problem of premature convergence," in *Proceedings of the 8th annual conference on Genetic and evolutionary computation - GECCO '06*, (Seattle, Washington, USA), p. 815, ACM Press, 2006.

[2] T. Jansen, "On the Analysis of Dynamic Restart Strategies for Evolutionary Algorithms," in *Parallel Problem Solving from Nature — PPSN VII* (G. Goos, J. Hartmanis, J. van Leeuwen, J. J. M. Guervós, P. Adamidis, H.-G. Beyer, H.-P. Schwefel, and J.-L. Fernández-Villacañas, eds.), vol. 2439, pp. 33–43, Berlin, Heidelberg: Springer Berlin Heidelberg, 2002. Series Title: Lecture Notes in Computer Science.

[3] G. S. Hornby, "A Steady-State Version of the Age-Layered Population Structure EA," in *Genetic Programming Theory and Practice VII* (R. Riolo, U.-M. O'Reilly, and T. McConaghy, eds.), pp. 87–102, Boston, MA: Springer US, 2010. Series Title: Genetic and Evolutionary Computation.

[4] M. D. Schmidt and H. Lipson, "Age-fitness pareto optimization," p. 2.

[5] Jian Jun Hu and E. Goodman, "The hierarchical fair competition (HFC) model for parallel evolutionary algorithms," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, vol. 1, (Honolulu, HI, USA), pp. 49–54, IEEE, 2002.

[6] T. McConaghy, P. Palmers, G. Gielen, and M. Steyaert, "GENETIC PROGRAMMING WITH DESIGN REUSE FOR INDUSTRIALLY SCALABLE, NOVEL CIRCUIT DESIGN," p. 18.

[7] A. Awuley and B. J. Ross, "Feature selection and classification using age layered population structure genetic programming," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, (Vancouver, BC, Canada), pp. 2417–2426, IEEE, July 2016.