Effects of combining Particle Swarm Optimization with Genetic Algorithms to Minimize Deceptive Cases

Daniel N. Khalil
Computer Science
Artificial Intelligence Club
Daytona Beach, Florida
600 South Clyde Morris Blvd.

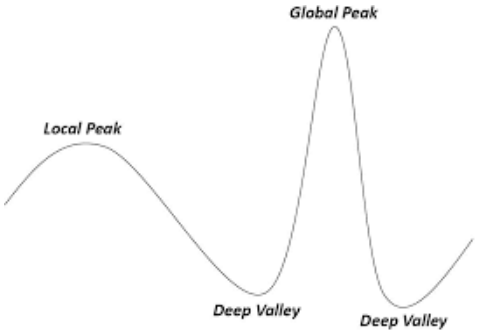
Luke R. Crump
Software Engineering
Artificial Intelligence Club
Daytona Beach, Florida
600 South Clyde Morris Blvd.

Abstract

This paper set out to find whether combing Particle Swarm Optimization and Genetic Algorithms together in a new hybridized algorithm would allow convergence to an optimal solution on a solution space with many local minimums to simulate solving a deceptive case. After experimentation, it was found that the algorithm proposed performed slightly better than Particle Swarm Optimization and Genetic Algorithms when executed on solution spaces with many local minimums.

Introduction

Heuristic optimization techniques have become a massive part of machine learning. Whether it be updating weights or topologies of Neural Networks or approximating solutions to NP-Complete problems, Heuristic evolutionary optimization algorithms, like genetic algorithms and particle swarm optimization, are handy. Although most heuristic optimization algorithm's drawback is that in searching for an optimal solution, an inaccurate solution can be converged upon because the algorithm ran on a deceptive solution space. This problem can cause unwanted behaviors in machine learning or undesirable solutions.



Our goal in writing this paper is to test whether hybridizing the heuristic evolutionary optimization algorithm's Genetic Algorithm and Particle Swarm Optimization together may have a positive effect on convergence speed, and accuracy in searching through deceptive solution spaces.

Definitions

Heuristic: a problem dependent algorithm that is supposedly faster and more efficient than traditional methods by sacrificing accuracy, or completeness.

Metaheuristic: a high-level problem independent algorithm that uses heuristic methods to allow application to a broad variety of problems.

Optimization: the process of finding the best solution or solutions out of a range of possible solutions

Solution Space: the set of all possible solutions to a specific problem

Search Space: the set of all possible solutions

Genetic Algorithm

Genetic Algorithms (GA) created in 1960 by John Holland are an evolutionary metaheuristic optimization algorithm based on Darwin's theory of evolution and survival of the fittest.

The Algorithm initially constructs a population of random solutions in a solution space, encodes them, then will run through the following steps:

1. evaluate solutions through defining some fitness function
2. select the solutions based off of those evaluations
3. breed the solutions together using one of an assortment of crossover methods
4. mutating each solution based on a defined mutation rate

Repeating these steps until an optimal solution has been found, Genetic Algorithms eventually converges the entire population to a solution inside the solution space.

The critical aspects of how GAs find a solution is in how they explore and converge. Exploration is affected by the selection and mutation methods of GA and is essential to help the population not get stuck in deceptive cases. Convergence is affected by the selection and crossover methods of GA and is essential to moving the entire population towards a specific solution. In some cases, if these deceptive cases contain too much noise, GAs would have a much lower chance of converging to the most optimal solution.

Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a metaheuristic optimization algorithm based off of the social behavior of birds. PSO instead of using the practice of selection and crossover, uses a mathematical vector equation that uses the position of the specific particle, the position of the best particle, and randomness to explore and converge on a solution

The algorithm initially constructs a population of particles that contain a random solution between the upper and lower bounds of the solution space, assigns each particle a random velocity vector $\mathbf{v_i}$ within the bounds of the solution space. Then runs through these steps.

ω – inertia parameter (normally between 0.6-0.9)
φ_p – social component parameter (normally 2)
φ_g – cognitive component parameter (normally = φ_p = 2)
r – random number between 0-1

1. Evaluates the particle's solution: $\mathbf{x}_i$
2. If the particle's solution is better, then it's personal best it updates that particles personal best: $\mathbf{p}_i$
3. If the particle's solution is the best solution out of the entire swarm then the global best solution is updated $\mathbf{g}_d$
4. Updates the particle's velocity using $\mathbf{v}_{i,d} \leftarrow \omega\, \mathbf{v}_{i,d} + \varphi_p\, r_p\, (\mathbf{p}_{i,d}-\mathbf{x}_{i,d}) + \varphi_g\, r_g\, (\mathbf{g}_d-\mathbf{x}_{i,d})$
5. Updates the particle's position $\mathbf{p}_i \leftarrow \mathbf{p}_i + \mathbf{v}_{i,d}$

These steps repeat for each particle for each epoch. To promote convergence the velocity vector equation uses the social component to move each of the particles towards the global best solution, to promote exploration of the solution space the random parameter **r** is multiplied to each component of the equation, as well as the inertia parameter ω.

Hybridization

To test whether the hybridization of PSO and GA is viable. GA, and PSO will be tested against 3 different hybrid algorithms. If these algorithms are awarded a higher fitness, in a shorter amount of iterations it will be accepted that these algorithms perform better than the nonhybrid versions of GA or PSO.

Hybrid: Tandem GAPSO

The Tandem GAPSO Algorithm will first construct a population of random solutions within the bounds of a solution space. Then encapsulate 1/2 of those individual solutions within a particle object that contains a pBest and gBest as well as a velocity vector.

1. pBest updates based on the current particles position.
2. gBest updates based off of the entire population's global best solution.
3. Particles update velocities of the first ½ of the population
4. Particles update position the first ½ of the population
5. GA will select parents from the entire population
6. GA performs crossover and mutation to fill up the other ½ of the population with children.
7.

These steps will repeat to converge to a solution.

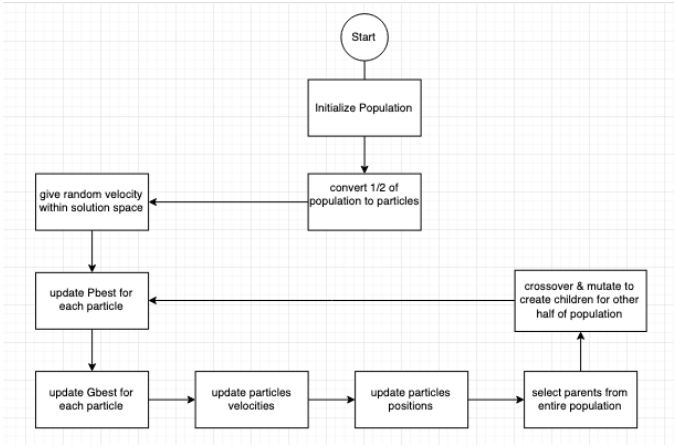


Figure 1

The idea behind this hybridization of Particle Swarm Optimization and Genetic Algorithm, is to increase the exploration of the population through the solution space. As the population will utilize the exploration components of Particle Swarm Optimization (Inertia, and random aspects of the velocity calculation), as well as the exploration aspects of Genetic Algorithm, (mutation rate, and selection).

Adding this extra exploration aspect will hopefully allow convergence to a more optimal solution within solution spaces that contain either extra noise, or more local minimums / local maximums.

Benchmark Functions

Name	Function
Ackley's function	$f(\mathbf{x}) = -a \exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right) + a + \exp(1)$
Eggholder function	$f(\mathbf{x}) = -(x_2+47)\sin\left(\sqrt{\left\lvert x_2 + \frac{x_1}{2} + 47\right\rvert}\right) - x_1 \sin\left(\sqrt{\lvert x_1 - (x_2+47)\rvert}\right)$
Holder Table function	$f(\mathbf{x}) = -\left\lvert \sin(x_1)\cos(x_2)\exp\left(\left\lvert 1 - \frac{\sqrt{x_1^2+x_2^2}}{\pi}\right\rvert\right)\right\rvert$
Levy function	$f(\mathbf{x}) = \sin^2(\pi w_1) + \sum_{i=1}^{d-1}(w_i-1)^2\left[1+10\sin^2(\pi w_i+1)\right] + (w_d-1)^2\left[1+\sin^2(2\pi w_d)\right]$, where $w_i = 1 + \frac{x_i - 1}{4}$, for all $i = 1, \ldots, d$
Rastrigin function	$f(\mathbf{x}) = 10d + \sum_{i=1}^{d}\left[x_i^2 - 10\cos(2\pi x_i)\right]$
Sphere function	$f(\mathbf{x}) = \sum_{i=1}^{d} x_i^2$

Experimental Design

Plotting the fitness over 20 iterations on each the 6 different optimization benchmark functions, all three algorithms are executed to understand efficiency. In this test the algorithms will be optimizing for the global minimum, so the favorable fitness of each algorithm will be the smallest value.

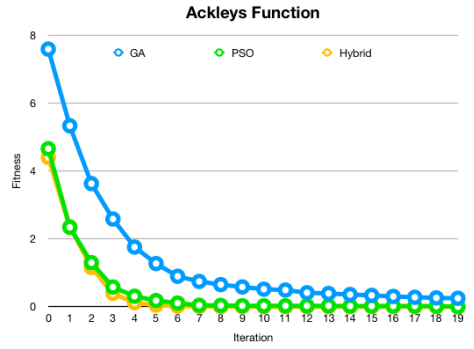
Algorithm	Parameters
GA	Mutation rate: 0.1 Population: 25
PSO	Cognitive Component: 2 Social Component: 2 Inertia: 0.9 Population
Hybrid Algorithm	Cognitive Component: 2 Social Component: 2 Inertia: 0.9 Mutation rate: 0.1 Population: 25

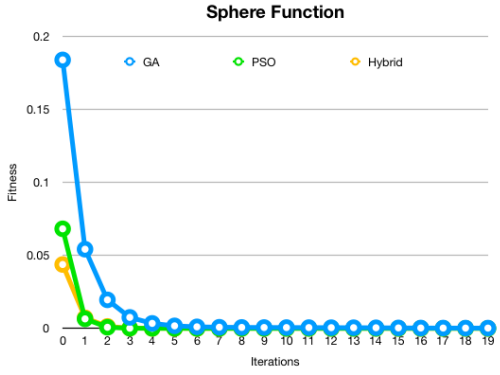
(Each Algorithm executed 100 times then averaged to find the data points that go inside the test cases) The test cases will go as follows:
For each algorithm

Iteration	GA	PSO	Hybrid
0			
1			
2			
…			
19			

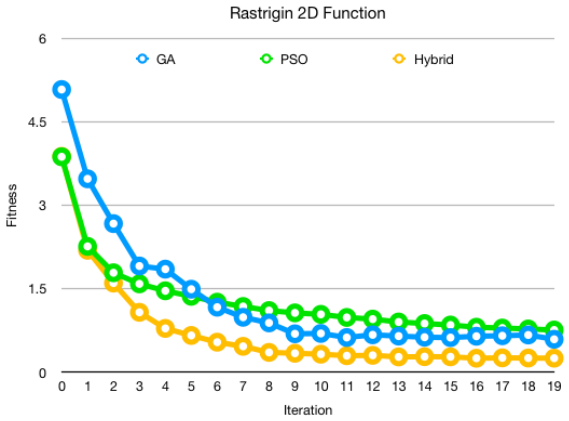
Results

The pattern found was that the hybrid algorithm did not outperform in benchmark functions with less prominent local minimums.

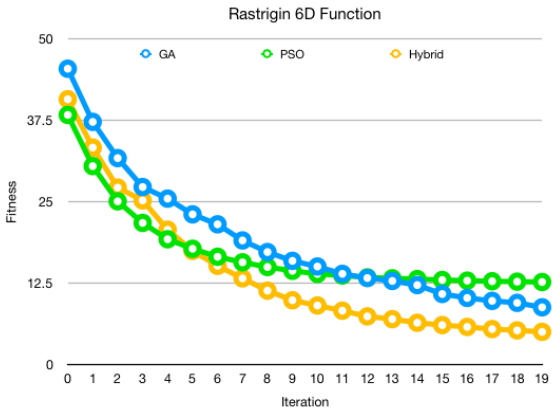


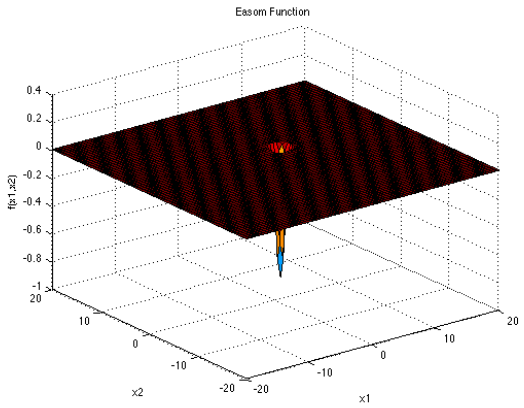


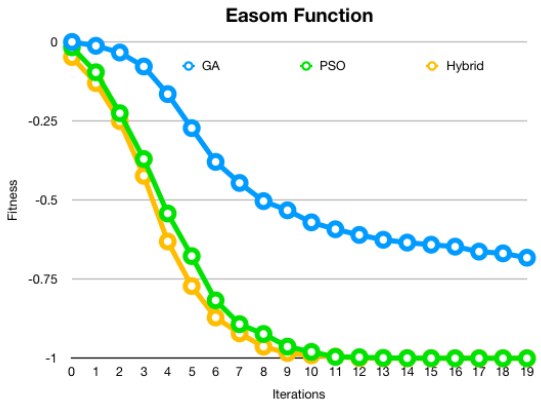
Although in functions that had many local minimums the hybrid algorithm performed slightly better than both Genetic Algorithm and Particle Swarm Optimization. For example, the Rastrigin function was a function that the hybrid algorithm would outperform GA and PSO for every dimension.



To check how the hybrid algorithm would perform on solution spaces that contained plateaus; the GA, PSO and the hybrid algorithm were executed using the Easom function which has a large area containing a plateau.







The Easom function's hybrid algorithm execution had a slightly quicker convergence speed then particle swarm optimization, although the difference was within 1-2 iterations.

Conclusion

The proposed Genetic Algorithm and Particle Swarm Optimization hybridization outperformed GA and PSO only slightly in situations that contained many local minimums / local maximums. Although, the act of changing the algorithm to expand on exploration affected the convergence speed of the algorithm on solution spaces with no/few local minimums or maximums.

Particle Swarm Optimization is an excellent algorithm when paired with real number problems. In the experiment, we executed with the parameters we utilized Particle Swarm Optimization had a faster convergence speed than the Genetic Algorithm; inversely, Genetic Algorithms performed better when it came to the exploration of the solution space. Thankfully when it came to optimizing solution spaces with many local minima and maxima, the hybrid GAPSO Algorithm used both of those strengths and converged on the optimal solution in fewer iterations because of it.

Particle Swarm Optimization and Genetic Algorithms, if executed with the right parameters, always finds the global minimum of the Sphere Function, and Ackley's function, although when these algorithms are put up against solution spaces with noise, or fluctuation, they stop becoming viable solutions to problems. The GAPSO hybrid created during this paper set out to push that boundary slightly further and succeeded. Although the improvement to the exploration to aid metaheuristic optimization algorithms to get over deceptive problems continues to hinder convergence speeds.

References

PSO-GSA (Particle Swarm Optimization, Gravitational Search Algorithm Hybrid
https://ieeexplore.ieee.org/abstract/document/6141614

Gene selection in cancer classification using PSO/SVM and GA/SVM hybrid algorithms
https://ieeexplore.ieee.org/abstract/document/4424483

GA and a novel PSO-GA-based hybrid algorithm
https://www.sciencedirect.com/science/article/pii/S0020019004003254

Variable-Length Particle Swarm Optimization for Feature Selection on High-Dimensional Classification
https://ieeexplore.ieee.org/document/8458226

How efficient are genetic algorithms to solve high epistasis deceptive problems?
https://ieeexplore.ieee.org/abstract/document/4630806

Development and validation of different hybridization strategies between GA and PSO
https://ieeexplore.ieee.org/document/4424823