Exclusion Radius Particle Swarm Optimizer

To cite this article: Alvin Soo Chun Kit et al 2019 IOP Conf. Ser.: Mater. Sci. Eng. 551 012069

View the article online for updates and enhancements.
Exclusion Radius Particle Swarm Optimizer

Alvin Soo Chun Kit1, Haza Nuzly Abdull Hamed1, Mohd Adham Isa2

1Applied Industrial Analytics Research Group, Universiti Teknologi Malaysia, 81310
UTM Johor Bahru, Malaysia
2Software Engineering Research Group, Universiti Teknologi Malaysia, 81310 UTM
Johor Bahru, Malaysia

Email: alvinsoock@gmail.com, haza@utm.my

Abstract. Function optimization is a problem that has existed since modern engineering
began. With multiple variables, brute-forcing or simple regression models become unfeasible.
This paper proposes a shrinking population control method for a dynamic population particle
swarm optimizer. The proposed control mechanism creates an exclusion zone, where particles
are periodically purged from the simulation. The selection of particles purged relies on their
particle best i.e. the position where it found its personal best fitness. Particles with their particle
best outside of the exclusion zone with a predetermined radius from the current global best are
removed from the simulation. The purging is done periodically in stages of equal iteration
counts, and the radius is shrunk by a factor after every stage. By testing 5 benchmark
mathematical functions, the results show that the proposed population control mechanism
achieves better final optima than only a basic particle swarm optimizer and a simple shrinking
dynamic population particle swarm optimizer where the worst performing particle is removed
every so often.

1. Introduction
Particle Swarm Optimization (PSO) has been one of the earliest optimization methods. It was first
introduced by Kennedy and Eberhart [1] to solve a mathematical function which has now become
ways to evaluate the performance of PSO. Many adjustments to the original algorithm has been made
since then, such as the addition of \( v \) as an inertial weight [2] and clamping of \( v_{max} \) [3]. Due to its
simplicity in implementation, it is relatively fast to execute, and as such is rather popular as a way to
solve optimization problems. It does have its drawbacks, however; as it cannot guarantee that a
globally optimal solution is found. PSO is heavily dependent on the initial conditions, such as the
search space and the control variables used to encourage exploration and exploitation respectively.
Furthermore, PSO faces issues when used to solve higher dimensional problems [4]. As such, this
work proposes a variant of Dynamic Population Particle Swarm Optimization, where a shrinking
population control mechanism is used.

2. Experiments
The proposed approach still uses the basic Particle Swarm Optimizer algorithm as its base. As such,
the following subsections discuss the parameters inherent to a basic PSO, and the modifications made
in the proposed approach.
2.1. Base PSO Parameters
The base parameters for the PSO are as follows: $c_1$ and $c_2$ are set to 2.0, $\omega$ is set to scale linearly as the iteration count increases, from 0.9 at $t = 0$ to 0.7 at $t = t_{max}$ based on work by Shi and Eberhart [2]. The initial particle count is 20, while the maximum iteration count is set to 2000.

Velocity and position clamping is used in the proposed approach. The equations used are as detailed by Shi and Eberhart [2]. The position clamping is only used on problems with constrained search spaces. If the search space is deemed infinite, then $x_{\text{min}}$ and $x_{\text{max}}$ will be set to $-\text{IntegerMAX}$ and $\text{IntegerMAX}$ respectively.

2.2. Exclusion Radius PSO
This subsection details a shrinking population control mechanism, hereby referred to as Exclusion Radius Particle Swarm Optimizer (ERPSO). Exclusion Radius is used here to fence off an area that is safe from deletion. Particles outside of the exclusion radius at the time of purging (stage boundary).

The simulation is first separated into stages. For each stage, the following variables are set and initialized.

- Maximum deleted particles
  \[ p_{\text{Del}}_{\text{max}} = \frac{p_{\text{Initial}}}{\text{stageCount}} \]  

- Exclusion radius (per dimension)
  \[ \text{excludeRadius} = 0.5 \ast \text{shrinkFactor}^{(\text{phaseCount} - 1)} \ast \frac{(\text{searchRange}_{\text{max}} - \text{searchRange}_{\text{min}})}{2} \]  

- Shrinking factor, shrinkFactor

The exact values for used in the calculation of exclusion radius and shrinking factor may differ according to the problem. A grid search was performed during testing, and in general values lesser than 0.5 for the exclusion radius multiplier in Equation (2), and values greater than 0.5 for the shrinking factor itself were optimal.

A function to calculate the normalized distance between two particles is also defined:

\[ \text{distance} = \sum_{d=1}^{\text{dim}} \left( \frac{p_1 \text{position}[d] - p_2 \text{position}[d]}{\text{searchRange}_{\text{max}}[d] - \text{searchRange}_{\text{min}}[d]} \right)^2 \]  

---

Figure 1. Pseudocode for the proposed ERPSO method.
In each stage, the simulation is allowed to run as is typical. At the end of each stage, the exclusion safety zone (shown as the shaded circle in Figure 2) is determined. This exclusion safety zone shrinks after each stage in accordance to the exclusion radius formula in Equation (2).

Figure 1 shows the general flow of the algorithm. At the stage boundary, the entire population is evaluated by calculating the distances of their $p_{best}$ from the current global best (using Equation (3)), and removing those that are outside of the exclusion safety zone. If there are more particles outside of the exclusion safety zone than $pDel_{max}$, whichever particles the simulation finds first will be deleted. This is effectively a random removal method of the targeted particles as the particles are not ordered in any particular pattern.

![Figure 2. 2-dimensional representation of ERPSO in action.](image)

3. Benchmark Evaluation
The proposed framework is evaluated based on the average (over 30 runs) best fitness value found. This is to mitigate the effect of “lucky” runs and is recommended in the work done by Uriarte, Melin [5]. A lower fitness value (closer to 0) indicates a better performing algorithm.

All of the evaluation functions used are benchmark optimization functions as seen in various other works concerning function optimization [4-8] and have multi-dimensional forms. This paper uses the Sphere, Rastrigin, Schwefel, Ackley and Griewank functions as benchmarks. These particular functions have been selected due to the ease of implementation and encompass a variety of shapes in the solution space, allowing for more varied testing conditions. The 30-dimension variants of these functions are used in this work, and are otherwise unchanged.
4. Results and Discussion
Table 1 shows the comparison between the 3 tested approaches. Base PSO refers to the basic PSO algorithm with linearly decreasing inertial weight as first proposed by Shi and Eberhart [2]. Remove Worst One refers to a shrinking population control mechanism, where the worst performing particle (as determined by current position from global best) is removed after each stage. Proposed - ERPSO is the proposed approach of this paper. Table 1 lists the comparison between a basic PSO approach, a shrinking (remove worst one) approach, and the proposed (ERPSO) approach.

Table 1. Results Comparison.

| Problem   | Metric  | Base PSO | Remove Worst One | Proposed - ERPSO |
|-----------|---------|----------|------------------|------------------|
| Sphere    | Min     | 0.00E+00 | 0.00E+00         | 0.00E+00         |
|           | S.D.    | 5.71E+01 | 7.78E+01         | 0.00E+00         |
| Sphere    | Mean    | 1.50E+01 | 3.00E+01         | 0.00E+00         |
| Schwefel  | Min     | 2.69E+03 | 2.79E+03         | 2.37E+03         |
| Schwefel  | S.D.    | 4.60E+03 | 4.84E+03         | 3.99E+03         |
| Rastigrin | Min     | 0.00E+00 | 0.00E+00         | 0.00E+00         |
| Rastigrin | Mean    | 4.80E+01 | 8.80E+01         | 5.15E+01         |
| Rastigrin | S.D.    | 3.41E+01 | 4.86E+01         | 4.29E+01         |
| Ackley    | Min     | 0.00E+00 | 0.00E+00         | 0.00E+00         |
| Ackley    | Mean    | 1.59E+00 | 2.31E+00         | 9.59E-02         |
| Ackley    | S.D.    | 4.60E+00 | 5.22E+00         | 3.71E-01         |
| Griewank  | Min     | 0.00E+00 | 0.00E+00         | 0.00E+00         |
| Griewank  | Mean    | 9.08E+00 | 9.10E+00         | 0.00E+00         |
| Griewank  | S.D.    | 2.77E+01 | 2.76E+01         | 0.00E+00         |

Table 1. Results Comparison.

As seen in Table 1, the proposed approach outperforms the basic PSO approach, and the Remove Worst One shrinking approach. This can be attributed to the culling of poorly performing particles, and preventing the simulation as a whole from being trapped in local minima that is far away from the global best. The shrinking of the exclusion radius effectively funnels the remaining particles into fully exploring the local search space around the global best. However, this approach may have a weakness in that it might lock-on too early to a global best and lead the simulation towards other local optima. This can be combated by tweaking the stage boundaries, to allow more exploration of the larger search space in the beginning.

5. Conclusion
From the research, we can conclude that the proposed approach performs better in terms of final fitness found compared to the Base PSO approach and a simple Remove Worst Performer approach. This is due to the effectiveness of culling particles outside of a predetermined radius around the current global best, rather than simply removing the worst performer.

Acknowledgements
This research was funded by the Ministry of Education Malaysia (MOE) through the Fundamental Research Grant Scheme FRGS/1/2018/ICT02/UTM/02/1, Cost Center No.: RJ130000.7851.5F069.
References
[1] Kennedy J et al. 1995 International Conference on Neural Networks 1942-1948.
[2] Shi Y et al. 1998 IEEE Int'l Conf. on Evolutionary Computation Proceedings. 69-73.
[3] Eberhart R et al. 1995 MHS'95. Proc. of the Sixth Int'l Sym. on Micro Machine and Human Science 39-43.
[4] Arasomwan M A et al. 2013 IEEE Congress on Evolutionary Computation 2352-2359.
[5] Uriarte A et al. 2016 IEEE 8th International Conference on Intelligent Systems 128-132.
[6] Jana N D et al. 2014 IEEE Congress on Evolutionary Computation 573-578.
[7] Li J et al. 2015 IEEE Transactions on Cybernetics 45 2350-2363.
[8] Yu H et al. 2013 Ninth International Conference on Natural Computation 471-475.