Event-driven multi-population optimization: mixing Swarm and Evolutionary strategies

ANONYMOUS[*]
Institute for Clarity in Documentation
Dublin, Ohio
ANONYMOUS@ANONYMOUS.com

ANONYMOUS ANONYMOUS
ANONYMOUS Research Laboratories
ANONYMOUS ANONYMOUS, ANONYMOUS
ANONYMOUS@ANONYMOUS.com

ABSTRACT

Recently, in the field of nature-inspired optimization, researchers have proposed multi-population asynchronous algorithms that distribute the evolutionary process among heterogeneous seach paradigms.These algorithms execute the seach strategy by taking streams of populations from message queues, and replacing them with evolved populations. Moreover, current studies suggest that having a high number of populations interacting in parallel, the effect of the individual parameters of each population is compensated by those selected in other populations improving the performance of the algorithm. In this work, we propose a simple reactive migration method for the asynchronous execution of multi-population, multi-strategy algorithms that improves over homogeneous configurations. We evaluate this method by comparing between homogeneous and an ensemble of multi-populations, using Genetic Algorithms (GAs) and Particle Swarm Optimization (PSO) in the noisless BBOB testbed for the optimization of continuous functions. Results show, that this method offers better performance, even when compared with other asynchronous population based algorithms.

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability;

KEYWORDS

ACM proceedings, LaTeX, text tagging

ACM Reference Format:
ANONYMOUS and ANONYMOUS ANONYMOUS. 2020. Event-driven multi-population optimization: mixing Swarm and Evolutionary strategies. In *Proceedings of the Genetic and Evolutionary Computation Conference 2020 (GECCO '20).* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Nature-inspired optimization algorithms have been used successfully in the last decades to tackle complex problems [19]. Algorithms taking inspiration in nature include evolutionary algorithms (EAs) [2] and swarm intelligence (SI) [11]. Genetic Algorithms (GAs) [5, 9], and Differential Evolution (DE) [10] are two popular EAs, while examples of (SI) [11] are particle swarm optimization (PSO) [3], and ant colony algorithms (ACO) [4]. A common characteristic of these kind of methods is the use of an initial set of random candidate solutions that are manipulated by the algorithm to generate a new set of candidates, and because of this, we commonly referred to them as population-based algorithms.

A drawback of population-based algorithms is that they can be computationally expensive, and that is why researchers have been proposing some form of parallelization [15] to increase their scalability. One of the first methods of parallelization was the island model, which lead to an increased performance [6, 7]. The concept was to divide the population into smaller populations that communicated with each other. Other population-based algorithms have adopted the technique, and since then, researchers have found additional advantages besides the execution speed, these include avoiding a premature convergence and maintaining the diversity of the global population [12], we are going to call these methods multi-population algorithms [14]. The relative isolation in which populations carry out the algorithm, together with the synchronous or asynchronous communication, helps to increase the overall diversity since each population will search in a particular area, at least between communications. Multi-population algorithms use a form of communication to recombine or migrate candidate solutions between populations to avoid premature convergence, since smaller populations are known to perform better for a given problem than bigger populations [13, 18]. Even in some cases, a multi-population based algorithm scales better due to these interactions, and the parallelism of the operation [1].

Having many populations offers researchers many configuration options and additional challenges when designing efficient multi-population algorithms. Options include the number and size of populations, the interaction between them, the search area of each population, and the search strategy and parametrization of each population. In this paper, we are interested in the latter, that is, having multiple populations running with distinct parameters and optimization algorithms. We can find in the literature, several heterogeneous multi-population algorithms integrating variations of optimization algorithms, and these often perform better than single-population or homogeneous multi-population optimization algorithms[16, 18]. Heterogeneous algorithms add to the problem of finding the correct parameter settings for each population; because some parameters affect the accuracy of the solution and the convergence speed of the individual algorithms as they tip the balance between exploration and exploitation of the search space. On the other hand, current studies show that by having a high number of

[*]Dr. ANONYMOUS insisted his name be first.

Table 1: DEAP GA EvoWorker Parameters

Selection	Tournament size=12
Mutation	Gaussian $\mu = 0.0$, $\sigma = 0.5$, indbp=0.05
Mutation Probability	[0.1,0.3]
Crossover	Two Point
Crossover Probability	[0.2,0.6]

Table 2: EvoloPy PSO Parameters

V_{max}	6
W_{max}	0.9
W_{min}	0.2
C_1	2
C_2	2

populations communicating in parallel, the effect of the individual parameters of each population is compensated by those selected in other populations [13, 17]. Some level of heterogeneity can be implemented by just changing the configuration parameters of each population, but in this case, we are interested in heterogeneous search strategies.

Therefore, in this paper, we implemented an asynchronous multi-population algorithm version, using a message queue for inter-process communication [8], and a reactive migration procedure, in which we compare three heterogeneous configurations using a randomized parameter technique. We experimented with all populations using a GA or PSO search strategies, versus an ensemble multi-population with both GA and PSO algorithms, using as a benchmark, the first five functions of the BBOB testbed. We compare the options by measuring the average running time (aRT) as the number of functions (#FEs), as the objective is to prove that the advantage of heterogeneous configurations resides not only in the increased scalability but also in the search performance.

The organization of the paper is as follows: First, Section 2 presents state of the art relevant to our work. In Section 3, we present the proposed method. Section 4 describes the design of the empirical evaluation we designed to assess the effectiveness of the method, and in Section 5, we report and discuss the results. Finally, we offer the conclusions of this paper and suggestions on future work in Section 6.

2 STATE OF THE ART

3 PROPOSED METHOD

4 SETUP

5 RESULTS

6 CONCLUSIONS

ACKNOWLEDGMENTS

The authors would also like to thank the anonymous referees for their valuable comments and helpful suggestions. The work is supported by the

Table 3: Parameters used in the experiments with ten populations

Dimension	2	3	5	10	20	40
Generations	40	25	28	50	66	80
Population Size	50	60	60	70	100	125
Populations	10	10	10	10	10	10
Iterations	10	20	30	30	30	40

REFERENCES

[1] Enrique Alba. 2002. Parallel evolutionary algorithms can achieve super-linear performance. *Inform. Process. Lett.* 82, 1 (2002), 7 – 13. https://doi.org/10.1016/S0020-0190(01)00281-2 Evolutionary Computation.

[2] Thomas Back. 1996. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms.* Oxford university press.

[3] Maurice Clerc. 2010. *Particle swarm optimization.* Vol. 93. John Wiley & Sons.

[4] Marco Dorigo and Gianni Di Caro. 1999. Ant colony optimization: a new meta-heuristic. In *Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*, Vol. 2. IEEE, 1470–1477.

[5] Agoston E Eiben and James E Smith. 2003. Genetic algorithms. In *Introduction to evolutionary computing.* Springer, 37–69.

[6] Martina Gorges-Schleuter. 1990. Explicit parallelism of genetic algorithms through population structures. In *International Conference on Parallel Problem Solving from Nature.* Springer, 150–159.

[7] P Grosso. 1985. Computer simulations of genetic adaptation: Parallel subcomponent interaction in multilocus model. *Ph. D. Dissertation, University of Michigan* (1985).

[8] Juan J Merelo Guervós and J Mario García-Valdez. 2018. Introducing an Event-Based Architecture for Concurrent and Distributed Evolutionary Algorithms. In *International Conference on Parallel Problem Solving from Nature.* Springer, 399–410.

[9] John Henry Holland et al. 1992. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* MIT press.

[10] Derviş Karaboğa and Selçuk Ökdem. 2004. A simple and global optimization algorithm for engineering problems: differential evolution algorithm. *Turkish Journal of Electrical Engineering & Computer Sciences* 12, 1 (2004), 53–60.

[11] James Kennedy. 2006. Swarm intelligence. In *Handbook of nature-inspired and innovative computing.* Springer, 187–219.

[12] Changhe Li, Trung Thanh Nguyen, Ming Yang, Shengxiang Yang, and Sanyou Zeng. 2015. Multi-population methods in unconstrained continuous dynamic environments: The challenges. *Information Sciences* 296 (2015), 95–118.

[13] Xiangtao Li, Shijing Ma, and Yunhe Wang. 2016. Multi-population based ensemble mutation method for single objective bilevel optimization problem. *IEEE Access* 4 (2016), 7262–7274.

[14] Haiping Ma, Shigen Shen, Mei Yu, Zhile Yang, Minrui Fei, and Huiyu Zhou. 2019. Multi-population techniques in nature inspired optimization algorithms : A comprehensive survey. *Swarm and Evolutionary Computation* 44, July 2017 (2019), 365–387. https://doi.org/10.1016/j.swevo.2018.04.011

[15] Heinz Mühlenbein, Martina Gorges-Schleuter, and Ottmar Krämer. 1988. Evolution algorithms in combinatorial optimization. *Parallel computing* 7, 1 (1988), 65–85.

[16] Shams K Nseef, Salwani Abdullah, Ayad Turky, and Graham Kendall. 2016. An adaptive multi-population artificial bee colony algorithm for dynamic optimisation problems. *Knowledge-based systems* 104 (2016), 14–23.

[17] Ryoji Tanabe and Alex Fukunaga. 2013. Evaluation of a randomized parameter setting strategy for island-model evolutionary algorithms. In *Evolutionary Computation (CEC), 2013 IEEE Congress on.* IEEE, 1263–1270.

[18] Guohua Wu, Rammohan Mallipeddi, Ponnuthurai Nagaratnam Suganthan, Rui Wang, and Huangke Chen. 2016. Differential evolution with multi-population based ensemble of mutation strategies. *Information Sciences* 329 (2016), 329–345.

[19] Xin-She Yang. 2014. *Nature-inspired optimization algorithms.* Elsevier.

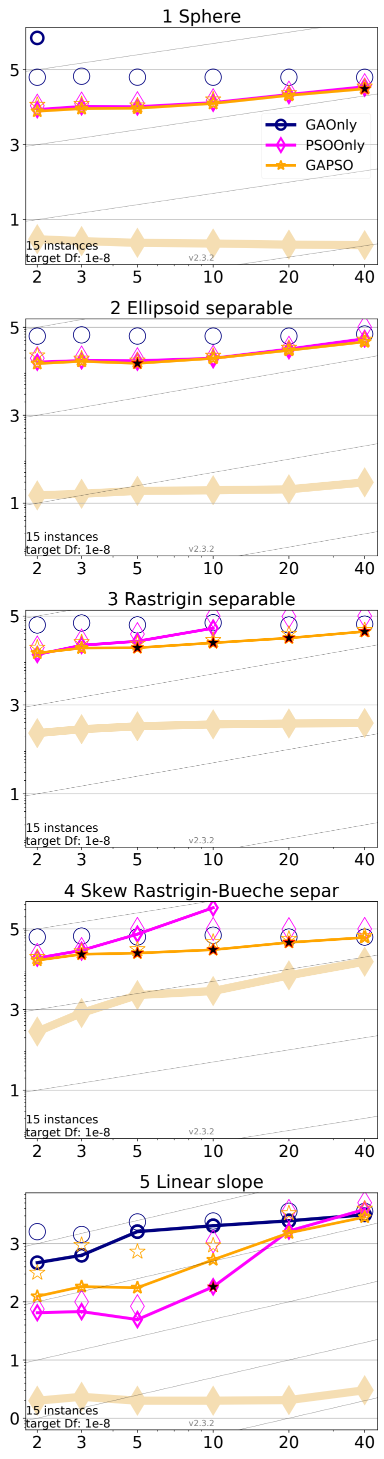

Figure 1: Average running time (in #FEs as log_{10} value), divided by dimension for target function value 10^{-8} vs dimension. Algorithms legends are given in f_1. Light symbols give the maximum number of function evaluations from the longest trial divided by dimension. Black stars indicate a statistically better result compared to all other algorithms ($p < 0.01$) and Bonferroni correction number of dimensions (six).