An Event-based Architecture for Multi-population Optimization Algorithms*

Erick Vargas Minguela$^{1[0000-1111-2222-3333]}$, Mario Garcia Valdez$^{2,3[1111-2222-3333-4444]}$, and Third Author$^{3[2222--3333-4444-5555]}$

National Technological Institute of Mexico erick.vargas.minguela@gmail.com
http://www.springer.com/gp/computer-science/lncs
{abc,lncs}

Abstract. Having the knowledge that both of them are population-based algorithms it can be defined that a migration between 2 or more populations are possible, and this kind hybrid can be helpful to increase the possibility to find the optimal result (the best of the best), there is where fits the concept of Multi-population. For this kind of work we used asynchronous functions, serverless functions, multithread and a distributed architecture taking advantage for functional programming and serverless architecture. Even nature works like that... parallel, asynchronous and distributed.

Keywords: Multi-population · Asynchronous · Sub-population · Serverless · Distributed.

1 Introduction

A universe of solutions can exist for a single problem and sometimes is too big and complex to solve them in a traditional way. That is why heuristic and population based algorithms are required. This kind of algorithms are very useful to solve combinatories problems, however, usually one is better than others to solve one thing and another is great to solve another problem, and there are several cases stucked in optimal local values.

Here we propose an architecture composed by serverless functions to create a multi-populations that will process sub-populations in distributed function architecture and they are going to be parallel and asynchronous making each sub-population distributed and independent. And comunicating using migrations to help each other preventing fall into optimal local values.

The distributed architectures are having extensive use in the software industry because of their high performance, many systems are being created and migrating step by step to microservices and... in a nearly future... the new architectures called serverless, which proposes the use of "Function as a Service" (FaaS).

* Supported by organization x.

1.1 Serverless

Recently, the cloud providers as Amazon Web Services (AWS), Google Cloud, etc. Offers a new alternative to programming throught interfaces called Serverless Computing, this kind of platform consist in a very simple mecanism where the developer upload the code into the platform and execute it as mamy times it is required scalling and allowing do this in a parallel way. This way the developers do not worry about servers, connections and other configurations. In serverless it pays only for what is used. Even there are some platform that allows to install them into your own server to do your own local architecture serverless.

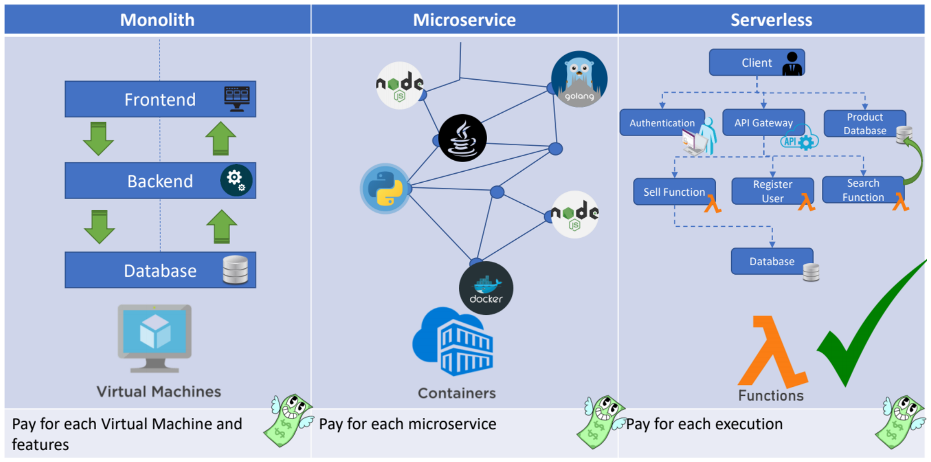

Fig. 1. Software architecture generations.

Serverless Function In math a function is a relation between a set of inputs and an allowed set of outputs with the idea that each input goes to a single output. But in computer science is small bits of code that do only one thing and are easily to understand and support. In serverless this functions could be triggered by an event that would be menssages, http request,etc. Also is known that each function scales independently and is stateless with a short duration. Displayed equations are centered and set on a separate line.

$$x + y = z \tag{1}$$

Please try to avoid rasterized images for line-art diagrams and schemas. Whenever possible, use vector graphics instead (see Fig. 15).

Theorem 1. *This is a sample theorem. The run-in heading is set in bold, while the following text appears in italics. Definitions, lemmas, propositions, and corollaries are styled the same way.*

Table 1. Table captions should be placed above the tables.

Heading level	Example	Font size and style
Title (centered)	**Lecture Notes**	14 point, bold
1st-level heading	**1 Introduction**	12 point, bold
2nd-level heading	**2.1 Printing Area**	10 point, bold
3rd-level heading	**Run-in Heading in Bold.** Text follows	10 point, bold
4th-level heading	*Lowest Level Heading.* Text follows	10 point, italic

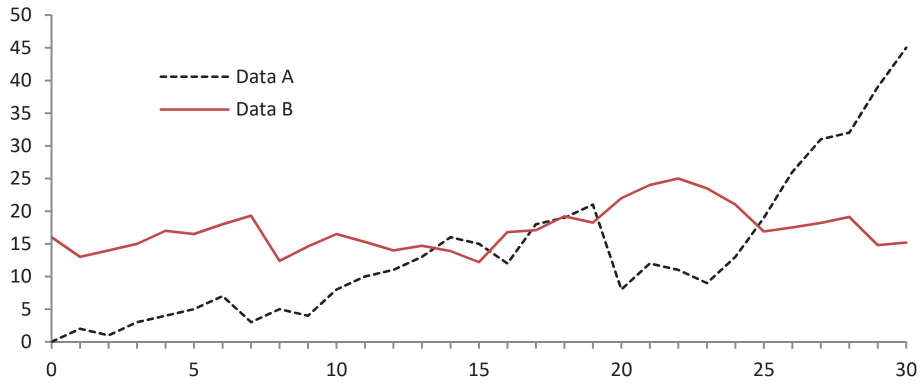

Fig. 2. A figure caption is always placed below the illustration. Please note that short captions are centered, while long ones are justified by the macro package automatically.

Proof. Proofs, examples, and remarks have the initial word in italics, while the following text appears in normal font.

For citations of references, we prefer the use of square brackets and consecutive numbers. Citations using labels or the author/year convention are also acceptable. The following bibliography provides a sample reference list with entries for journal

2 Experiments and Results

2.1 Experiments

Now that an interaction between sub-populations with different algorithms it is working and hybridation have been a success, using until now the added algorithms (GA and PSO) algorithms, all thanks to the developed architecture, lets procede to the experiments. This section is going to be the execution of several experiments from 2 to 40 dimensions, with a stop criterial of an error below 0.5E-8, without a parameter optimization method, waiting that the architecture by its self would be enough to increase the possibility to find a better optimal result than the traditional methods. All this hoping that the results will probe the needness of this kind of architecture on increasing dimensions. To test if the architecture was useful, several experiments were made to solve benchmark functions, for this case the functions are Sphere, Rastrigin and Rosenbrock. Using 10 sub-population for each experiment and maximum 4 migrations per sub-population with different algorithms and parameters for each sub-population.

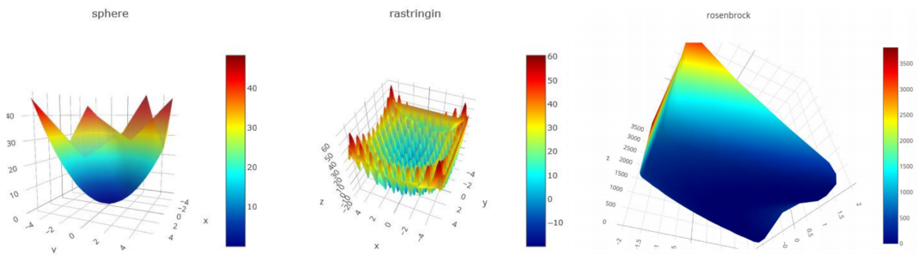

Fig. 3. Benchmark functions for experimentation.

2.2 Parameters Configuration

This architecture modifies the traditional way to work with population based algorithms, then the experiments could not be parameterized as usually are.

Then the experiments are scaled by their number of evaluations and the parameters must be configured to be adjusted to the next criterial, using the next expression:

$$Evaluations = 10^5 Dimensions \tag{2}$$

For example, if the experiment has 2 dimensions, the maximum number of evaluations will be 200,0000, for 10 dimensions will be 1,000,000 of evaluacions and the same with the others dimensions.

References

1. J. M. Hellerstein, J. Faleiro, J. E. Gonzalez, J. Schleier-Smith, V. Sreekanti, A. Tumanov, and C. Wu, "Serverless Computing: One Step Forward, Two Steps Back," vol. 3, 2018.
2. O. Kramer, "Genetic Algorithm Essentials," *Springer International Publishing AG*, vol. 679, pp. 11–20, 2017.
3. C. Guerrero, I. Lera, and C. Juiz, "Genetic Algorithm for Multi-Objective Optimization of Container Allocation in Cloud Architecture," *Journal of Grid Computing*, pp. 1–23, 2017.
4. S. Lalwani, H. Sharma, S. Chandra, S. Kusum, D. Jagdish, and C. Bansal, "REVIEW - COMPUTER ENGINEERING AND COMPUTER SCIENCE A Survey on Parallel Particle Swarm Optimization Algorithms," *Arabian Journal for Science and Engineering*, 2019.

Table 2. 2 dimension parameters

Parameter	Value
GA Optimization	Minimize
GA Generations	50
GA Dimensions	2
GA Population size	100
GA Mutation	Random(Tournament2,Tournament3,Random ,RandomLinearRank,Sequential,Fittest)
GA Crossover	Tournament3
GA Crossover percentage	Random[10%, 80%]
GA Mutation percentage	Random[10%,50%]
GA Crossover function	Uniforme de punto medio
GA Mutation Function	gaussian
PSO Optimization	Minimiza
PSO Iterations	50
PSO Dimensions	2
PSO Vector size	100
PSO Social factor	Random[0.5,4.0]
PSO Individual factor	Random[0.5,4.0]
PSO Inercia factor	Random[0.5,4.0]

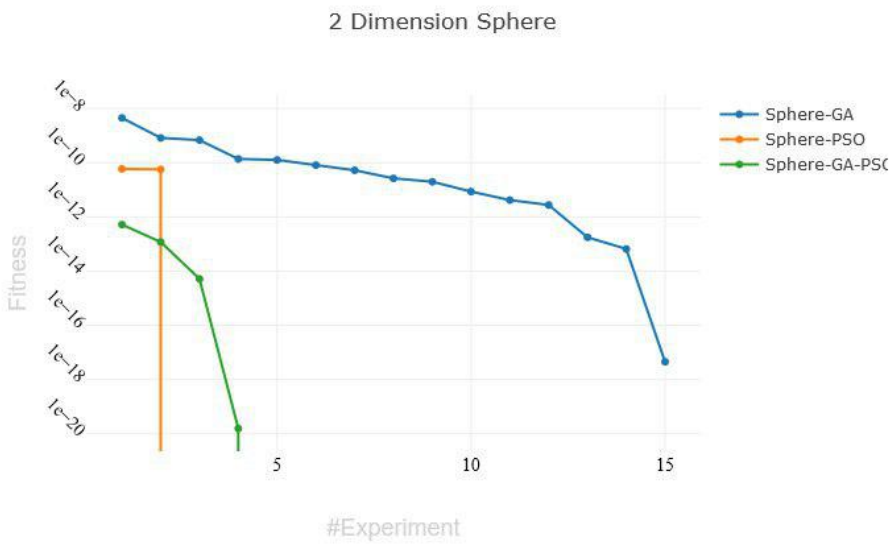

Fig. 4. 2 dimension experiments Sphere.

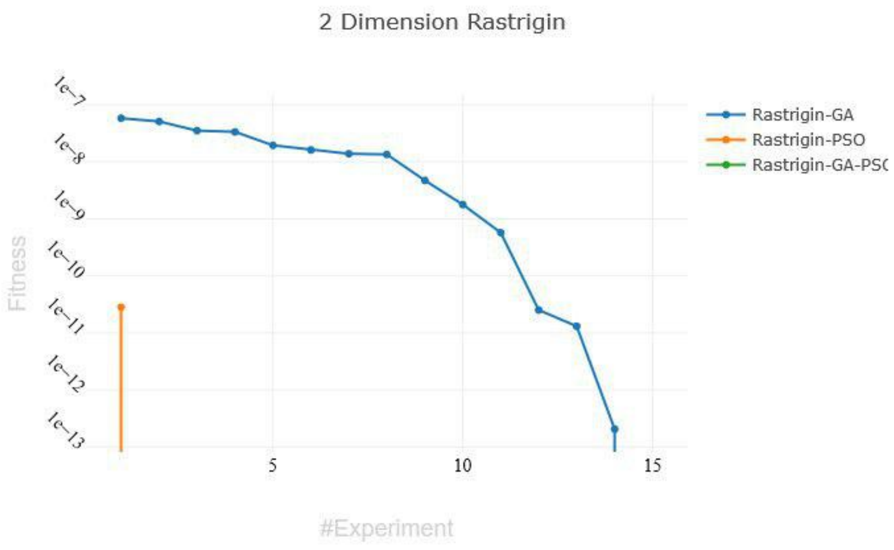

Fig. 5. 2 dimension experiments Rastrigin.

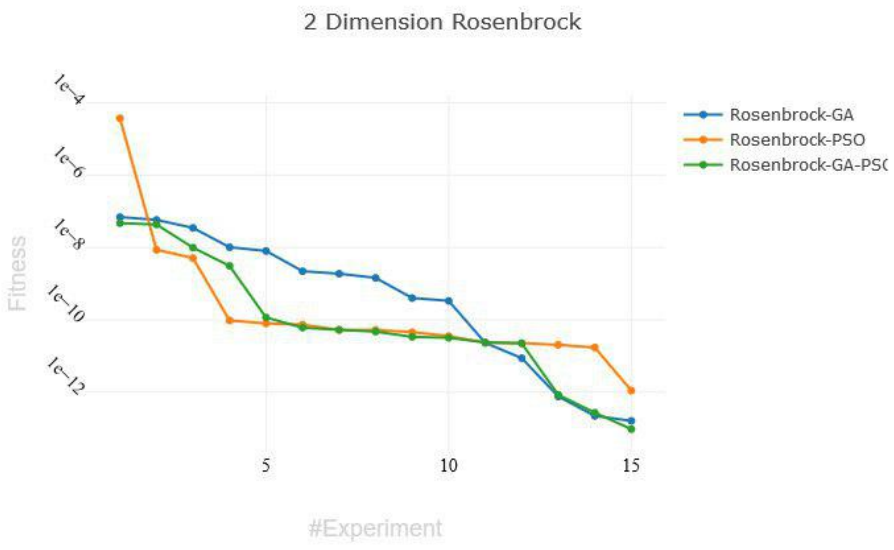

Fig. 6. 2 dimension experiments Rosenbrock.

Table 3. 2 dimensional experiment results

Fn	Best	AVG	Experiment Number
Rastrigin GA	0	1.65377E-08	15
Rastrigin PSO	0	1.8872E-12	15
Rastrigin GA-PSO	0	0	15
Sphere GA	4.53222E-18	4.36977E-10	15
Sphere PSO	0	7.8012E-12	15
Sphere GA-PSO	0	4.33161E-14	15
Rosenbrock GA	1.62335E-13	1.24176E-08	15
Rosenbrock PSO	1.11674E-12	2.47795E-06	15
Rosenbrock GA-PSO	9.5809E-14	6.90695E-09	15

Table 4. 10 dimensions parameters

Parameter	Value
GA Optimization	Minimiza
GA Generations	70
GA Dimensions	10
GA Population size	200
GA Mutation	Random(Tournament2,Tournament3,Random ,RandomLinearRank,Sequential,Fittest)
GA Crossover	
GA Crossover percentage	Random[10%, 80%]
GA Mutation percentage	Random[10%,50%]
GA Crossover function	Uniforme de punto medio
GA Mutation Function	gaussian
PSO Optimization	Minimiza
PSO Iterations	70
PSO Dimensions	10
PSO Vector size	200
PSO Social factor	Random[0.5,4.0]
PSO Individual factor	Random[0.5,4.0]
PSO Inercia factor	Random[0.5,4.0]

Table 5. 10 dimensional experiment results

Fn	Mejor	Promedio	No. Experimento
Rastrigin GA	3.21768E-09	2.38015E-06	15
Rastrigin PSO	7.8586E-11	2.715716161	15
Rastrigin GA-PSO	8.01492E-12	5.08668E-09	15
Sphere GA	1.84051E-09	2.5389E-08	15
Sphere PSO	4.50351E-11	4.72855E-09	15
Sphere GA-PSO	3.33851E-11	1.30062E-09	15
Rosenbrock GA	9.58323E-07	1.24176E-08	15
Rosenbrock PSO	4.16711E-07	4.431565444	15
Rosenbrock GA-PSO	3.62472E-07	0.000240251	15

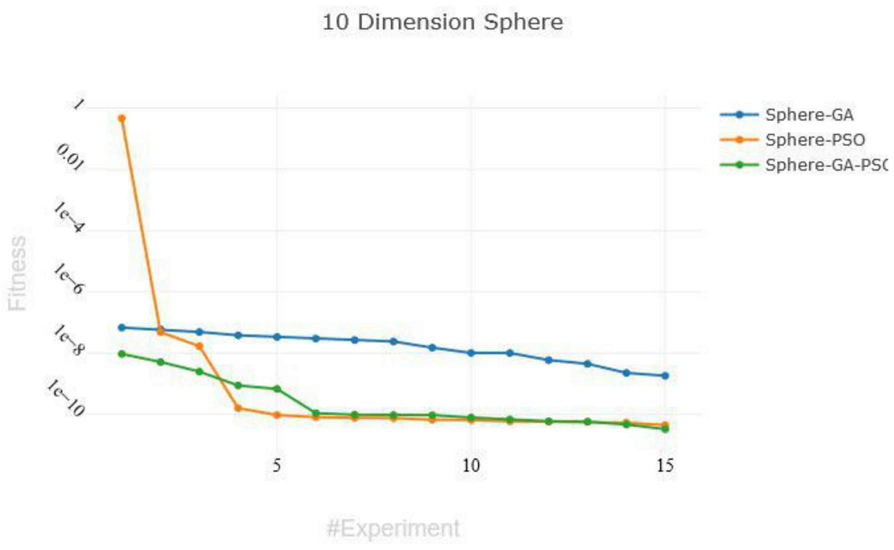

Fig. 7. 10 dimensions experiments Sphere.

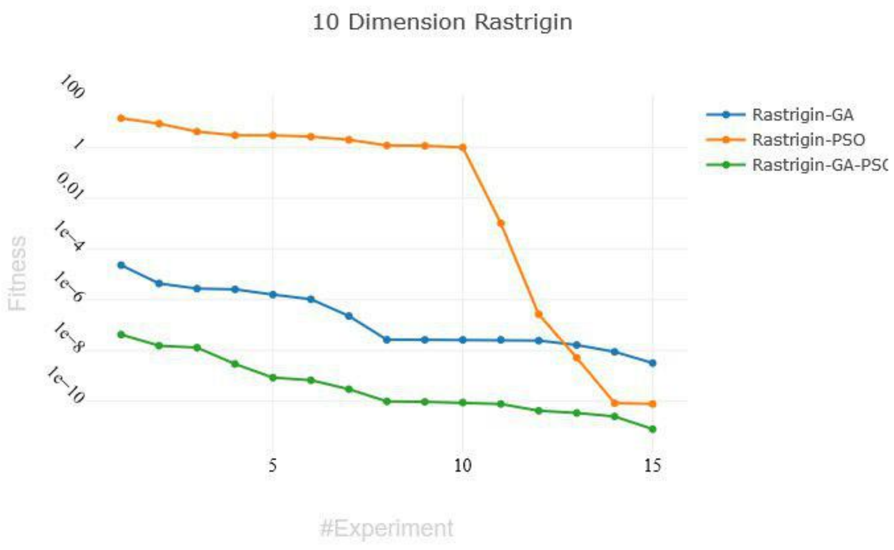

Fig. 8. 10 dimensions experiments Rastrigin.

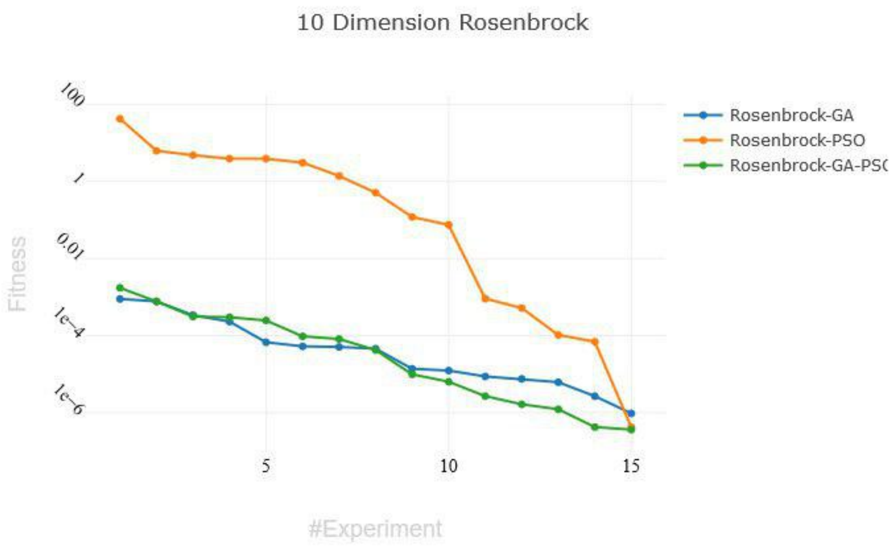

Fig. 9. 10 dimensions experiments Rosenbrock.

Table 6. Parametros experimentos 20 dimensiones

Parameter	Value
GA Optimization	Minimiza
GA Generations	70
GA Dimensions	20
GA Population size	200
GA Mutation	Random(Tournament2,Tournament3,Random ,RandomLinearRank,Sequential,Fittest)
GA Crossover	Tournament3
GA Crossover percentage	Random[10%, 80%]
GA Mutation percentage	Random[10%,50%]
GA Crossover function	Uniforme de punto medio
GA Mutation Function	gaussian
PSO Optimization	Minimiza
PSO Iterations	70
PSO Dimensions	20
PSO Vector size	200
PSO Social factor	Random[0.5,4.0]
PSO Individual factor	Random[0.5,4.0]
PSO Inercia factor	Random[0.5,4.0]

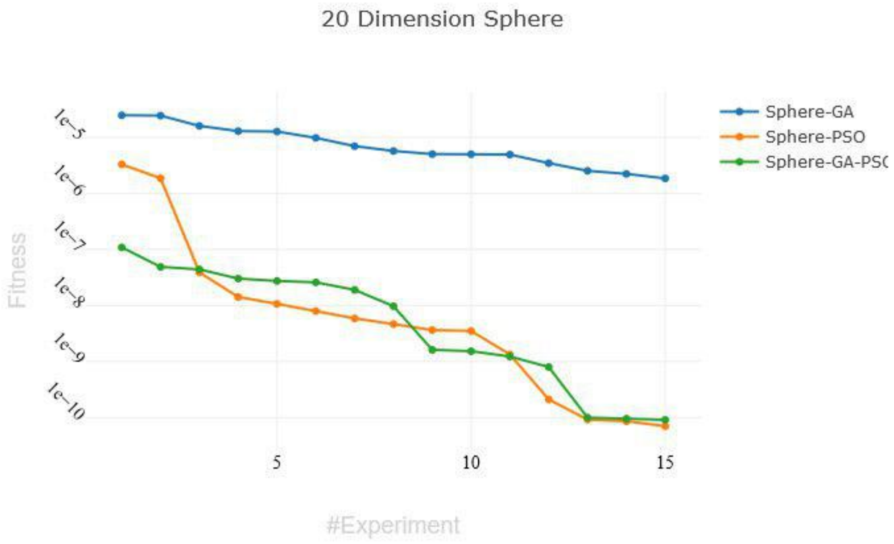

Fig. 10. 20 dimensions experiments Sphere.

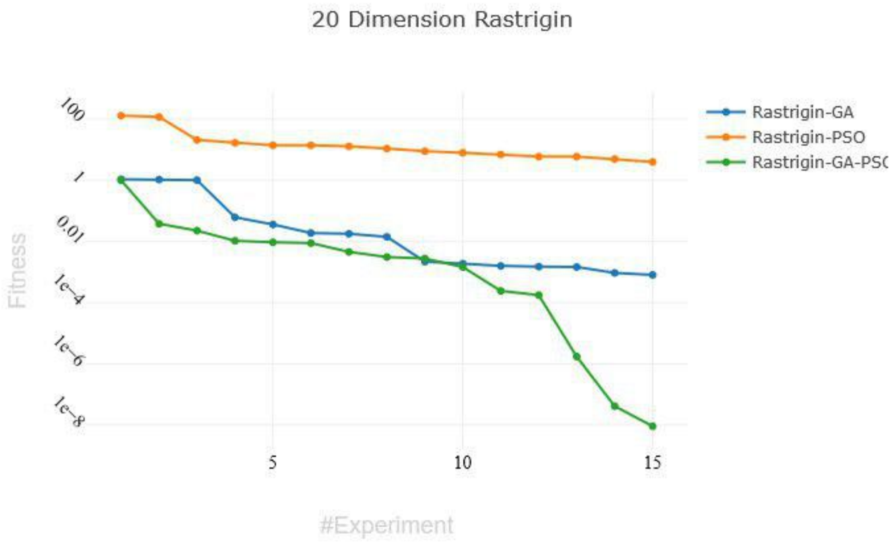

Fig. 11. 20 dimensions experiments Rastrigin.

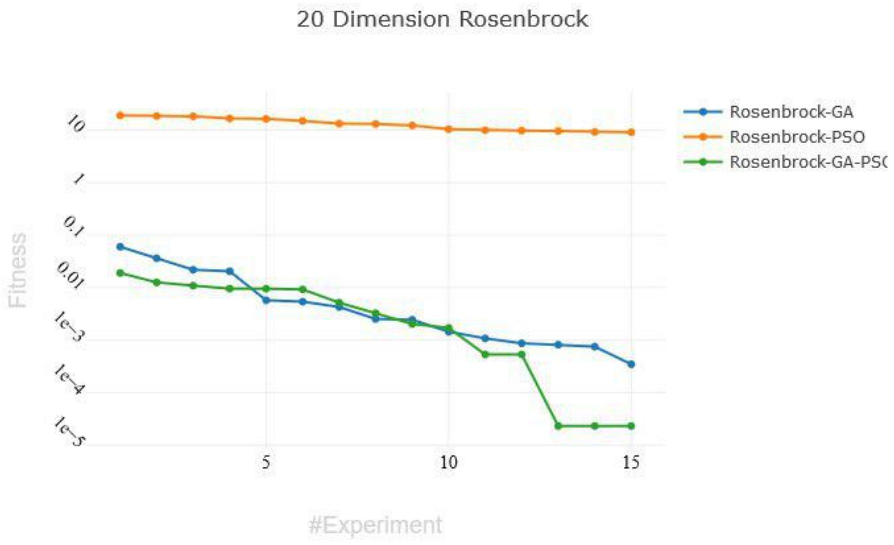

Fig. 12. 20 dimensions experiments Rosenbrock.

Table 7. Resultados 20 dimensiones

Fn	Mejor	Promedio	No. Experimento
Rastrigin GA	0.000808633	0.220596203	15
Rastrigin PSO	3.988070734	25.51777514	15
Rastrigin GA-PSO	9.13E-09	7.38E-02	15
Sphere GA	1.84051E-09	9.22715E-06	15
Sphere PSO	7.04E-11	3.50E-07	15
Sphere GA-PSO	9.11E-11	2.13E-08	15
Rosenbrock GA	0.000348015	0.010958941	15
Rosenbrock PSO	9.119539342	13.37613983	15
Rosenbrock GA-PSO	2.31663E-05	0.005608855	15

Table 8. Parametros experimentos 40 dimensiones

Parameter	Value
GA Optimization	Minimiza
GA Generations	70
GA Dimensions	40
GA Population size	200
GA Mutation	Random(Tournament2,Tournament3,Random ,RandomLinearRank,Sequential,Fittest)
GA Crossover	
GA Crossover percentage	Random[10%, 80%]
GA Mutation percentage	Random[10%,50%]
GA Crossover function	Uniforme de punto medio
GA Mutation Function	gaussian
PSO Optimization	Minimiza
PSO Iterations	70
PSO Dimensions	40
PSO Vector size	200
PSO Social factor	Random[0.5,4.0]
PSO Individual factor	Random[0.5,4.0]
PSO Inercia factor	Random[0.5,4.0]

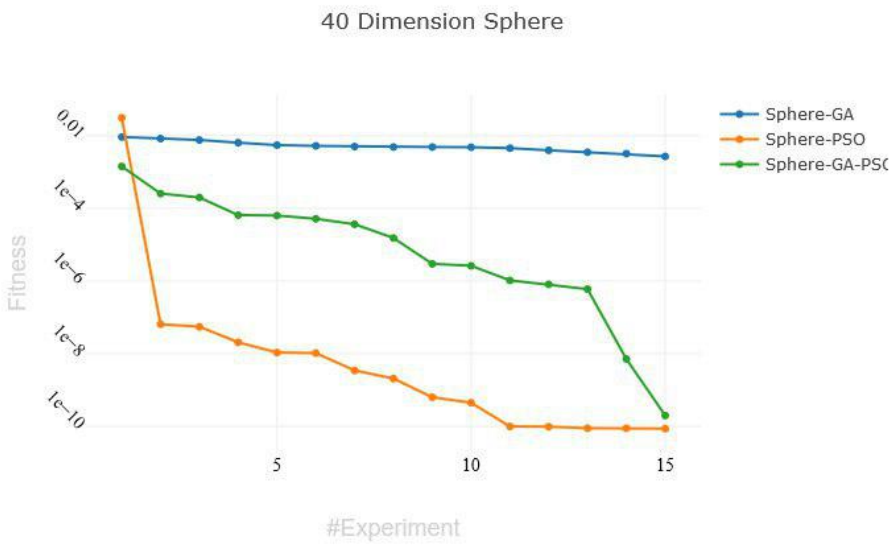

Fig. 13. 40 dimensions experiments Sphere.

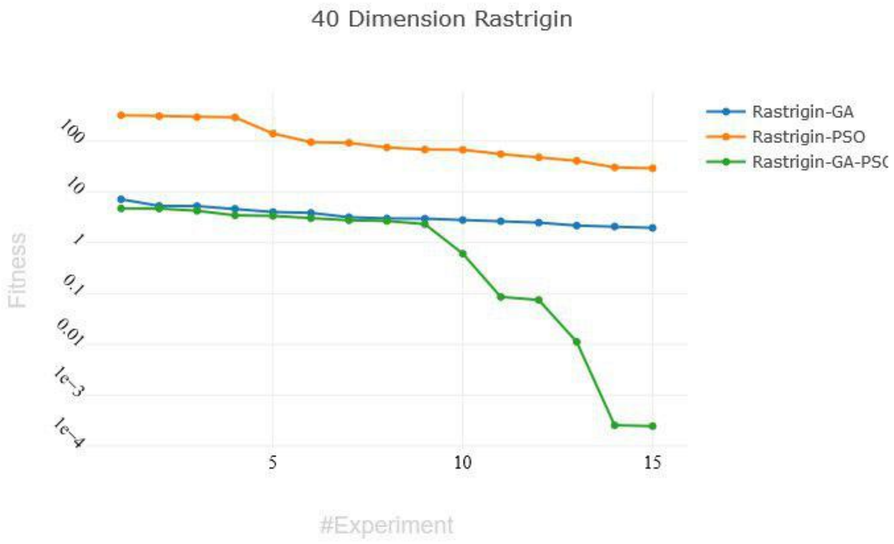

Fig. 14. 40 dimensions experiments Rastrigin.

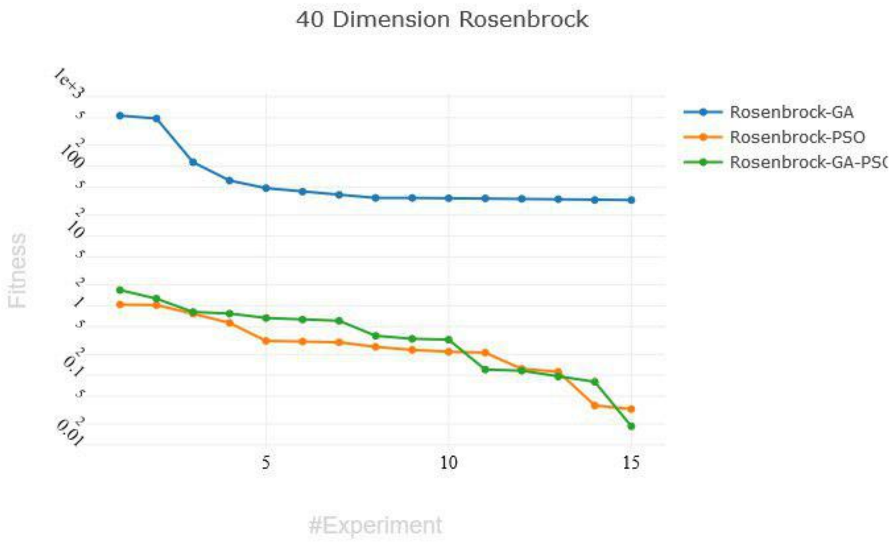

Fig. 15. 40 dimensions experiments Rosenbrock.

Table 9. Resultados 40 dimensiones

Fn	Mejor	Promedio	No. Experimento
Rastrigin GA	1.95478879	3.560837088	15
Rastrigin PSO	29.06596132	130.2865863	15
Rastrigin GA-PSO	2.46E-04	2.13E+00	15
Sphere GA	0.002686956	0.005302951	15
Sphere PSO	8.68E-11	2.07E-03	15
Sphere GA-PSO	2.00E-10	1.41E-04	15
Rosenbrock GA	0.000348015	106.9287542	15
Rosenbrock PSO	0.032708559	0.368395353	15
Rosenbrock GA-PSO	0.018538924	0.525086565	15

5. S. Blum, R. Puisa, and M. Wintermantel, "Adaptive Mutation Strategies for Evolutionary Algorithms," *2nd Weimar Optimization and Stochastic Days*, pp. 1–13, 2005.
6. S. Everywhere, "The Fn Project,"
7. H. Ma, S. Shen, M. Yu, Z. Yang, M. Fei, and H. Zhou, "Multi-population techniques in nature inspired optimization algorithms : A comprehensive survey," *Swarm and Evolutionary Computation*, vol. 44, no. July 2017, pp. 365–387, 2019.
8. S. Santander-jim and M. A. Vega-rodr, "Comparative Analysis of Intra-Algorithm Parallel Multiobjective Evolutionary Algorithms : Taxonomy Implications on Bioinformatics Scenarios," vol. 9219, no. c, pp. 1–15, 2018.
9. D. Sherry, K. Veeramachaneni, J. McDermott, and U. M. O'Reilly, "Flex-GP: Genetic programming on the cloud," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7248 LNCS, pp. 477–486, 2012.
10. R. Goebel, *29th Australasian Joint Conference on Artificial Intelligence, AI 2016*, vol. 9992 LNAI. 2016.
11. J. J. M. Guerv and J. M. Garc, "Introducing an Event-Based Architecture for Concurrent and Distributed Evolutionary Algorithms," vol. 1, pp. 399–410, 2018.
12. L. Moroney, *The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform.* 2017.
13. T. Ambler and N. Cloud, *JavaScript frameworks for modern web dev.* 2015.
14. A. D. Barwell, C. Brown, and K. Hammond, "USING PROGRAM SHAPING AND ALGORITHMIC SKELETONS TO PARALLELISE AN EVOLUTIONARY MULTI-AGENT SYSTEM IN ERLANG Wojciech Turek , Aleksander Byrski," vol. 35, pp. 792–818, 2016.
15. C. Paper and E. Alba, "Distributed Genetic Algorithms on Portable Devices for Smart Cities," no. May, 2017.
16. J. L. Technische, "C6.3 Island (migration) models: evolutionary algorithms based on punctuated equilibria," no. January 2000, 2016.
17. J. Kunasaikaran and A. Iqbal, "A Brief Overview of Functional Programming Languages," *electronic Journal of Computer Science and Information Technology (eJCSIT)*, vol. 6, no. 1, pp. 32–36, 2016.
18. D. Hows, P. Membrey, and E. Plugge, "MongoDB Basics," *MongoDB Basics*, 2014.
19. J. Cook, *Docker for Data Science.* 2017.
20. M. Løvbjerg and T. K. Rasmussen, "Hybrid Particle Swarm Optimiser with Breeding and Subpopulations," *Proc. 3rd Genetic Evolutionary Computation Conf.*, pp. 469 –476, 2001.

21. H. M. A. Jimeno, M. J. L. Sánchez, and R. H. Rico, “Multipopulation - based multi - level parallel enhanced Jaya algorithms,” *The Journal of Supercomputing*, no. 0123456789, 2019.
22. M. Roberts, “Serverless Architectures,” 2016.
23. Y. Kaya, M. Uyar, and R. Tek\D{j}n, “A Novel Crossover Operator for Genetic Algorithms: Ring Crossover,” no. May 2014, 2011.