International Journal of Advances in Electronics and Computer Science, ISSN (p): 2394-2835 Volume-10, Issue-9, Sept.-2023
http://iraj.in

HYBRID SINGLE OPERATOR HHO, SMA AND HGS BASED METAHEURISTIC ALGORITHMS

[1]PABLO PIÑOL, [2]MARÍA.JOSÉ MOSCARDÓ, [3]HÉCTOR MIGALLÓN, [4]MIGUEL MARTÍNEZ-RACH, [5]OTONIEL LÓPEZ-GRANADO

[1,2,3,4,5] Department of Computer Engineering, Miguel Hernández University, E-03202, Elche, Alicante, Spain
E-mail: [1]pablop@umh.es, [2]maria.moscardo@goumh.umh.es,[3]hmigallon@umh.es,[4]mmrach@umh.es,[5]otoniel@umh.es

Abstract - There are many population-based meta-heuristic optimization algorithms, but none can outperform all existing algorithms on all existing optimization problems, or solve all optimization problems. This leads some algorithms to use one or more control parameters to adjust the properties of the algorithm depending on the problem to be solved. The optimization behavior of these algorithms can be improved by, among other aspects, incorporating hybridization techniques. Hybrid algorithms are suitable for a wide range of applications, but they are usually intended for specific engineering problems, and they increase the difficulty of correctly adjusting the control parameters. This paper presents hybrid algorithms based on three operators from prevalent configuration parameter-free optimization algorithms. Each hybrid approach uses a different strategy to change the algorithm responsible for generating each new individual. These algorithms are HHO, SMA and HGS. Experimental results show that the proposed algorithms perform better than the original algorithms, which implies that the optimal use of these basic algorithms depends on the problem to be solved. Another advantage of the hybrid algorithms is that there is no need for a prior process of adjusting the control parameters.

Keywords - Hybrid optimization algorithms, Meta-heuristics, Swarm-based algorithms, HHO algorithm, HGS algorithm, SMA algorithm.

I. INTRODUCTION

Meta-heuristic optimization is widely used to solve problems in various fields of science and engineering. Many of these methods are based on populations, also known as swarms, which iteratively generate new populations with the aim of increasing the diversity of the current generation and thus increasing the probability of reaching the optimum for the problem under consideration. These algorithms are proposed to replace analytical algorithms when the latter are unable to produce an acceptable solution. The failure to produce a quasi-optimal solution may be due to the characteristics of the objective function or to the large search space, which makes exhaustive search useless. In addition, classical optimization methods, such as greedy algorithms, must take into account several assumptions that make it difficult to solve the problem under consideration.

When using meta-heuristic methods, the objective function has no restrictions, i.e., it does not necessarily have to be differentiable. Each optimization method proposes its own rules for evolving the population or swarm towards the optimum. These algorithms usually have different capabilities for global exploration and local exploitation.

Some classical algorithms have been proven to be robust including: genetic algorithms (GA) [1] which reflect the process of natural selection; differential evolution (DE) [2] which attempts to iteratively improve a candidate solution for a given measure of quality; the evolutionary strategy (ES) algorithm [3] which is based on the processes of mutation and selection seen in evolution; genetic programming (GP) [4] and evolutionary programming (EP) [5] which are based on the choice of individuals for reproduction (crossover) and mutation; the ant colony optimization (ACO) algorithm [6] which imitates the foraging behavior of ant colonies; the particle swarm optimization (PSO) algorithm [7] based on the social behavior of fish schooling or bird flocking; the artificial bee colony (ABC) algorithm [8] inspired by the foraging behavior of honey bees; the firefly (FF) algorithm [9] inspired by the flashing behavior of fireflies; the cuckoo search (CS) algorithm [10] based on the obligate brood parasitic behavior of some cuckoo species in combination with the Levy flight behavior of some birds and fruit flies; the biogeography-based optimization (BBO) algorithm [11] which improves solutions stochastically and iteratively; the flower pollination algorithm (FPA) [12] inspired by the pollination process of flowers; and the teaching-learning-based optimization (TLBO) algorithm [13] based on the teaching and learning models. Recently, many other outstanding meta-heuristic optimization algorithms have been proposed. Among them, the following are the most efficient: the whale optimization algorithm (WOA) [14] which mimics the social behavior of humpback whales; the enhanced sine cosine algorithm (ESCA) [15], which is based on the sine and cosine vacillation outwards or towards the best solution; the grey wolf optimizer (GWO) [16] inspired by grey wolves by mimicking the leadership hierarchy and hunting mechanism of grey wolves; the arithmetic optimization algorithm (AOA) [17] that utilizes the distribution behavior of the leading arithmetic operators in mathematics; the Harris hawks optimization (HHO) [18] based on the cooperative

behavior and chasing style of Harris' hawks in nature; the gradient-based optimizer (GBO) [19] inspired by the gradient-based Newton's method; the hunger games search (HGS) [20] designed according to the hunger-driven activities and behavioral choice of animals; and the slime mould algorithm (SMA) [21] based on the oscillation mode of slime mould in nature.

One of the techniques used to improve the performance of optimization algorithms is hybridization. Since a metaheuristic optimization algorithm cannot outperform all algorithms insolving a given problem, hybridization can be a solution that combines the capabilities of differentalgorithms into one system, see for example [22–39]. If these algorithms use control parameters,combining several of these algorithms into a single hybrid algorithm increases the complexity ofoptimizing these control parameters. In addition, some hybridization techniques can be complicated ifthe strategies for managing and replacing individuals in the populations are not similar.

II. HYBRID ALGORITHMS

2.1. Operators

The proposed hybrid algorithms are designed using three operators obtained from the HHO [18], SMA [20] and HGS [21] algorithms. These operators have been selected considering their performance in solving constrained and unconstrained functions, but also, but to a lesser extent, because they share a similar structure that allows the implementation of different hybridization strategies.

The general algorithm follows the guidelines of any algorithm based on populations or swarms, where a population is first obtained randomly and then evolved. Depending on the operator for this evaluation, the best individual and/or individuals randomly selected from the population itself and/or randomly obtained individuals are used.

2.2. Hybrid algorithm based on populations

As mentioned above, algorithms without control parameters were considered and three hybrid algorithms were designed. The first proposed hybrid algorithm, shown in Algorithm 1, processes the entire population in each iteration using the same operator. This is the simplest hybridization technique, allowing operators with very different structures to be hybridized, but it is also the hybridization that improves the basic operators the least.In these algorithms the population or swarm has a selectable size equal to popSize, each new population is identified by newPop. The number of iterations (numIterations) is the chosenstopping criterion for these algorithms. Logically, the number of function evaluations is equal topopSize * numIterations.

Algorithm 1 Population based hybrid algorithm

```
1: NumOperators= 3
2: IndexOpSelected= 0
3: for i= 1 to numIterationsdo
4:          switch (IndexOpSelected) 5:      case0:
6:            OpSelected=SMA
7:          case1:
8:            OpSelected=HHO
9:          case2:
10:           OpSelected=HGS
11:         endswitch
12:         for m = 0 to popSizedo
13:           Compute newPopm usingOpSelected
14:         endfor
15:         IndexOpSelected=IndexOpSelected+1 16:        ifIndexOpSelected≥NumOperatorsthen
17:           IndexOpSelected=0
18:         endif
19: end for
20: Search for the current solution
```

2.3. Hybrid algorithm based on subpopulations

The second hybrid algorithm is described in Algorithm 2. The population is divided into as many sub-populations as there are operators to be used, in our case three operators. In the optimization process, each sub-population is processed by one of the

operators used in the hybridization. If we do not include lines 15-18, each sub-population is always processed by the same operator, but if we include them, each sub-population is processed by a different operator in each iteration.

Algorithm 2 Sub-population based hybrid algorithm

```
1: NumOperators= 3
2: Split popSizeintoNumOperatorssub-populations
3: for i= 1 to numIterationsdo
4:         for m = 0 to popSizedo
5:             subPopID=sub-popindexofindividual m
6:             switch(subPopID)
7:             case0:
8:                OpSelected=SMA
9:             case1:
10:               OpSelected=HHO
11:            case2:
12:               OpSelected=HGS
13:            endswitch
14:            Compute newPopm usingOpSelected
15:            subPopID= subPopID+1
16:            if subPopID≥ NumOperatorsthen
17:               subPopID= 0
18:            endif
19:        endfor
20: end for
21: Search for the current BestPop
```

2.4. Hybrid algorithm based on individuals

In the last hybrid proposal shown in Algorithm 3, a different operator treats each individual in the population in each iteration. Furthermore, the choice of operator is randomized so that, statistically, each operator treats a very similar number of individuals.

Algorithm 3 Individual based hybrid algorithm

```
1: NumOperators=3
2: for i= 1 to numIterationsdo
3: for m = 0 to popSizedo
4:             OpSelected= rand[0, NumOperators - 1]
5:             switch(OpSelected)
6:             case0:
7:                OpSelected=SMA
8:             case1:
9:                OpSelected=HHO
10:            case2:
11:               OpSelected=HGS
12:            endswitch
13:            Compute newPopm usingOpSelected
14:        endfor
15: end for
16: Search for the current BestPop
```

It should be noted that the objective of the proposed hybrid algorithms is not to improve the convergence rate of the algorithms used separately, nor to achieve optimal performance for a particular problem. The aim is to show how to improve the average behavior for different cost functions, and analyze the different hybridization techniques.

III. EXPERIMENTAL RESULTS

In this section, the performance of the proposed hybrid algorithms is analyzed by solving 4 unconstrained functions listed in Table 1, whose definitions can be found, for example, in [40].

Id.	Function	Num. variables	Domain	Optimum
F1	Sphere	30,50,100	-100,100	0
F2	Trid_10	10	-10^2,10^2	-210
F3	Rosenbrock	30,50,100	-30,30	0
F4	FoxHoles	2	-65.536,65.536	0.998

Table1: Benchmark functions

The population size is 24 individuals and the number of iterations is 625 to perform 15000 function evaluations. The number of variables for functions F1 and F3 is selectable, while the number of variables for functions F2 and F4 is fixed. First, the statistical analysis obtained from 30 independent runs of each experiment is shown. This analysis, shown inTable 2, 3 and 4, is performed in terms of best and worst solution, mean, and standard deviation, when F1 and F3 are optimized with 30 variables in Table 2, 50 variables in Table 3 and 100 variables in Table 4.

		Best	Worst	Average	St. Dev.
SMA	F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	F2	-2.10E+02	-2.09E+02	-2.10E+02	2.49E-01
	F3	2.74E+01	2.85E+01	2.80E+01	3.32E-01
	F4	9.98E-01	9.98E-01	9.98E-01	5.56E-13
HHO	F1	1.85E-132	8.14E-115	5.46E-116	2.10E-115
	F2	-2.10E+02	-2.09E+02	-2.10E+02	2.16E-01
	F3	7.89E-05	1.36E-01	2.06E-02	3.41E-02
	F4	9.98E-01	9.98E-01	9.98E-01	2.64E-10
HGS	F1	3.47E-36	1.27E-32	1.74E-33	3.59E-33
	F2	-2.10E+02	-2.07E+02	-2.10E+02	7.53E-01
	F3	2.35E+01	1.70E+02	4.12E+01	4.13E+01
	F4	9.98E-01	1.08E+01	1.98E+00	2.53E+00

Table2: Operator's analysis (F1 and F3 with 30 variables)

		Best	Worst	Average	St. Dev.
SMA	F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	F3	4.77E+01	4.85E+01	4.82E+01	2.14E-01
HHO	F1	1.09E-150	2.46E-108	1.64E-109	6.35E-109
	F3	2.07E-04	5.47E-02	1.49E-02	1.61E-02
HGS	F1	5.18E-29	6.41E-24	6.08E-25	1.67E-24
	F3	4.42E+01	1.63E+02	5.88E+01	3.39E+01

Table 3: Operator's analysis (50 variables)

		Best	Worst	Average	St. Dev.
SMA	F1	0.00E+00	1.21E-275	8.05E-277	3.12E-276
	F3	9.81E+01	9.86E+01	9.84E+01	1.34E-01
HHO	F1	6.76E-145	1.81E-112	1.20E-113	4.66E-113
	F3	3.16E-04	8.15E-01	9.89E-02	2.19E-01
HGS	F1	1.60E-20	3.97E-17	7.94E-18	1.41E-17
	F3	9.56E+01	1.77E+02	1.02E+02	2.06E+01

Table 4: Operator's analysis (100 variables)

As can be seen in Table 2, 3 and 4, one operator is not always the best for all functions, in this case the SMA operator gives better results for all functions except F3. The result for this function is quite poor, our aim is always to obtain acceptable near-optimal results.

We will now analyze whether the proposed hybrid algorithms achieve our objective and which of these algorithms performs best in terms of optimization performance.

Tables 5, 6 and 7 show the results for the first hybrid approach, shown in Algorithm 1, which processes the entire population with a single operator in each iteration.

	Best	Worst	Average	St. Dev.
F1	6.30E-283	5.62E-253	3.95E-254	1.45E-253
F2	2.10E+02	2.10E+02	2.10E+02	2.69E-02
F3	8.98E-02	2.84E+01	2.58E+01	7.14E+00
F4	9.98E-01	1.99E+00	1.13E+00	3.50E-01

Table 5: Algorithm 1 analysis (F1 and F3 with 30 variables)

	Best	Worst	Average	St. Dev.
F1	8.60E-269	4.40E-232	2.93E-233	1.14E-232
F3	1.82E+00	4.85E+01	4.52E+01	1.20E+01

Table 6: Algorithm 1 analysis with 50 variables.

	Best	Worst	Average	St. Dev.
F1	1.83E-266	5.20E-234	3.49E-235	1.34E-234
F3	9.77E+01	9.81E+01	9.79E+01	1.23E-01

Table 7: Algorithm 1 analysis with 100 variables.

It can be seen from these tables that the objective is achieved with 30 variables, but performance deteriorates as the number of variables increases.

This loss of performance is due to some underperforming iterations. Therefore, we analyze Algorithm 2 based on subpopulations, in which we have used two versions: 1) fixed assignment (FA), in which each subpopulation is always processed by the same operator, but the data related to the whole population (mean of the variables, hungry …) are shared; 2) alternating assignment (AA), in which each subpopulation is processed by a different operator in each iteration.

		Best	Worst	Average	St. Dev.
FA	F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	F2	-2.10E+02	-2.10E+02	-2.10E+02	1.28E-03
	F3	6.39E-02	2.49E+01	1.82E+01	1.12E+01
	F4	9.98E-01	9.98E-01	9.98E-01	1.57E-16
AA	F1	0.00E+00	2.29E-304	1.53E-305	5.91E-305
	F2	-2.10E+02	-2.10E+02	-2.10E+02	1.10E-03
	F3	2.49E+01	2.67E+01	2.57E+01	5.04E-01
	F4	9.98E-01	9.98E-01	9.98E-01	5.06E-14

Table 8: Algorithm 2 analysis (F1 and F3 with 30 variables)

		Best	Worst	Average	St. Dev.
FA	F1	0.00E+00	8.59E-303	5.73E-304	2.22E-303
	F3	1.42E-01	4.49E+01	1.28E+01	1.99E+01
AA	F1	0.00E+00	6.48E-306	4.32E-307	1.67E-306
	F3	4.55E+01	4.75E+01	4.63E+01	6.18E-01

Table 9: Algorithm 2 analysis with 50 variables

		Best	Worst	Average	St. Dev.
FA	F1	0.00E+00	3.91E-301	2.61E-302	1.01E-301
	F3	9.99E-01	9.80E+01	2.42E+01	3.77E+01
AA	F1	0.00E+00	7.06E-292	4.71E-293	1.82E-292
	F3	9.58E+01	9.80E+01	9.70E+01	6.21E-01

Table 10: Algorithm 2 analysis with 100 variables

Tables 8, 9 and 10 show the results for Algorithm 2 and we can see how the performance of the optimization improves, in particular the results in general remain the same, but the result for the function with the most performance problems (F3) improve noticeably.
Tables 11, 12 and 13 show the results for Algorithm 3 and we can see how the performance of the optimization is also improved over Algorithm 2, although it does not use a fixed allocation but a random one.

	Best	Worst	Average	St. Dev.
F1	0.00E+00	2.77E-229	1.85E-230	7.15E-230
F2	-2.10E+02	-2.10E+02	-2.10E+02	3.13E-02
F3	4.15E-01	2.84E+01	2.39E+01	9.54E+00
F4	9.98E-01	9.98E-01	9.98E-01	4.25E-14

Table 11: Algorithm 3 analysis (F1 and F3 with 30 variables)

	Best	Worst	Average	St. Dev.
F1	0.00E+00	2.13E-255	1.42E-256	5.51E-256
F3	4.29E+00	4.85E+01	4.48E+01	1.12E+01

Table 12: Algorithm 3 analysis with 50 variables

	Best	Worst	Average	St. Dev.
F1	3.79E-302	1.27E-239	8.46E-241	3.28E-240
F3	5.97E+00	9.80E+01	8.14E+01	3.43E+01

Table 13: Algorithm 3 analysis with 100 variables

V. CONCLUSION

As shown in this article, hybridization techniques improve the overall behavior of the operators used in hybridization. Logically, the better the operators, the better the hybrid algorithm.

It has also been shown that the basic operators have different capabilities depending on the cost function. Of the hybridization strategies analyzed, both the hybridization based on subpopulations of individuals with fixed assignment and the randomized strategy applied to individuals performed best.

Based on the above analysis, future work will focus on improving the operator with the best performance in each function for that particular case. In addition, techniques to reduce the size of the population as the algorithm progresses will be investigated.The latter technique, in the case of fixed allocation, can achieve both of the above objectives, as algorithms that do not see their subpopulation reduced are implicitly strengthened.

On the other hand, other operators need to be studied and analyzed continuously, and some of these operators also need to be developed. It should be borne in mind that operators should be selected not only on the basis of their general characteristics, but also on the basis of their complementarity with the other selected operators.

ACKNOWLEDGMENTS

This research was supported by the Spanish Ministry of Science and Innovation and the Research State Agency under Grant PID2021-123627OB-C55 co-financed by FEDER funds (MCIU/AEI/FEDER, UE)

REFERENCE

[1] Holland, J.H. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence; MIT Press, 1992.
[2] Price, K.V. New Ideas in Optimization; McGraw-Hill Ltd., UK: Maidenhead, UK, England, 1999; chapterAn Introduction to Differential Evolution, pp. 79–108.
[3] Schwefel, H.P. Evolutionsstrategie und numerischeOptimierung. Dr.-Ing. Thesis, Technical University ofBerlin, Department of Process Engineering, 1975.
[4] Koza, J.R. Genetic Programming: A Paradigm for Genetically Breeding Populations of Computer Programsto Solve Problems. Technical report, Stanford, CA, USA, 1990.
[5] Bäck, T.; Rudolph, G.; Schwefel, H.P. Evolutionary Programming and Evolution Strategies: Similaritiesand Differences. In Proceedings of the Second Annual Conference on Evolutionary Programming, 1997,pp. 11–22.
[6] Dorigo, M.; Di Caro, G. New Ideas in Optimization; McGraw-Hill Ltd., UK: Maidenhead, UK, England,1999; chapter The Ant Colony Optimization Meta-heuristic, pp. 11–32.
[7] Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. Swarm Intelligence 2007, 1, 33–57.doi:10.1007/s11721-007-0002-0.
[8] Karaboga, D.; Basturk, B. On the Performance of Artificial Bee Colony (ABC) Algorithm. Appl. Soft Comput.2008, 8, 687–697. doi:10.1016/j.asoc.2007.05.007.
[9] Xin-She, Y. Firefly Algorithm, Lévy Flights and Global Optimization. Research and Development in IntelligentSystems XXVI 2009, á, 209–218. doi:10.1007/978-1-84882-983-1_15.
[10] Yang, X.S.; Deb, S. Cuckoo Search via Lévy flights. 2009 World Congress on Nature Biologically InspiredComputing (NaBIC), 2009, pp. 210–214. doi:10.1109/NABIC.2009.5393690.

[11] Ma, H.; Simon, D.; Siarry, P.; Yang, Z.; Fei, M. Biogeography-Based Optimization: A 10-YearReview. IEEE Transactions on Emerging Topics in Computational Intelligence 2017, 1, 391–407.doi:10.1109/TETCI.2017.2739124.

[12] Yang, X.S. Flower Pollination Algorithm for Global Optimization. Unconventional Computation andNatural Computation; Durand-Lose, J.; Jonoska, N., Eds.; Springer Berlin Heidelberg: Berlin, Heidelberg,2012; pp. 240–249.

[13] Rao, R.V.; Savsani, V.; Vakharia, D. Teaching-learning-based optimization: A novel method forconstrained mechanical design optimization problems. Computer-Aided Design 2011, 43, 303–315.doi:10.1016/j.cad.2010.12.015.

[14] Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. Advances in Engineering Software 2016, 95, 51–67.doi:10.1016/j.advengsoft.2016.01.008.

[15] Belazi, A.; Migallón, H.; Gónzalez-Sánchez, D.; Gónzalez-García, J.; Jimeno-Morenilla, A.; Sánchez-Romero,J.L. Enhanced Parallel Sine Cosine Algorithm for Constrained and Unconstrained Optimization.Mathematics 2022, 10. doi:10.3390/math10071166.

[16] Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. Advances in Engineering Software 2014, 69, 46–61.doi:10.1016/j.advengsoft.2013.12.007.

[17] Abualigah, L.; Diabat, A.; Mirjalili, S.; AbdElaziz, M.; Gandomi, A.H. The ArithmeticOptimization Algorithm. Computer Methods in Applied Mechanics and Engineering 2021, 376, 113609.doi:10.1016/j.cma.2020.113609.

[18] Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawksoptimization: Algorithm and applications. Future Generation Computer Systems 2019, 97, 849 – 872.doi:10.1016/j.future.2019.02.028.

[19] Ahmadianfar, I.; Bozorg-Haddad, O.; Chu, X. Gradient-based optimizer: A new metaheuristic optimizationalgorithm. Information Sciences 2020, 540, 131–159. doi:10.1016/j.ins.2020.06.037.

[20] Yang, Y.; Chen, H.; Heidari, A.A.; Gandomi, A.H. Hunger games search: Visions, conception,implementation, deep analysis, perspectives, and towards performance shifts. Expert Systems withApplications 2021, 177, 114864. doi:10.1016/j.eswa.2021.114864.

[21] Li, S.; Chen, H.; Wang, M.; Heidari, A.A.; Mirjalili, S. Slime mould algorithm: A new method for stochasticoptimization. Future Generation Computer Systems 2020, 111, 300–323. doi:10.1016/j.future.2020.03.055.

[22] Moayedi, H.; Gör, M.; Khari, M.; Foong, L.K.; Bahiraei, M.; Bui, D.T. Hybridizing four wiseneural-metaheuristic paradigms in predicting soil shear strength. Measurement 2020, 156, 107576.doi:10.1016/j.measurement.2020.107576.

[23] Nguyen, B.M.; Tran, T.; Nguyen, T.; Nguyen, G. Hybridization of Galactic Swarm andEvolution Whale Optimization for Global Search Problem. IEEE Access 2020, 8, 74991–75010.doi:10.1109/ACCESS.2020.2988717.

[24] Kayabekir, A.E.; Toklu, Y.C.; Bekda¸s, G.; Nigdeli, S.M.; Yücel, M.; Geem, Z.W. A Novel Hybrid HarmonySearch Approach for the Analysis of Plane Stress Systems via Total Potential Optimization. Applied Sciences2020, 10. doi:10.3390/app10072301.

[25] Punurai,W.; Azad, M.S.; Pholdee, N.; Bureerat, S.; Sinsabvarodom, C. A novel hybridized metaheuristictechnique in enhancing the diagnosis of cross-sectional dent damaged offshore platform members.Computational Intelligence 2020, 36, 132–150. doi:10.1111/coin.12247.

[26] Pellegrini, R.; Serani, A.; Liuzzi, G.; Rinaldi, F.; Lucidi, S.; Diez, M. Hybridization ofMulti-Objective Deterministic Particle Swarm with Derivative-Free Local Searches. Mathematics 2020, 8.doi:10.3390/math8040546.

[27] Yue, Z.; Zhang, S.; Xiao, W. A Novel Hybrid Algorithm Based on Grey Wolf Optimizer and FireworksAlgorithm. Sensors 2020, 20. doi:10.3390/s20072147.

[28] Seifi, A.; Ehteram, M.; Singh, V.P.; Mosavi, A. Modeling and Uncertainty Analysis of Groundwater Level Using Six Evolutionary Optimization Algorithms Hybridized with ANFIS, SVM, and ANN. Sustainability2020, 12. doi:10.3390/su12104023.

[29] Chen, X.; Yu, K. Hybridizing cuckoo search algorithm with biogeography-based optimization for estimatingphotovoltaic model parameters. Solar Energy 2019, 180, 192 – 206. doi:10.1016/j.solener.2019.01.025.

[30] Zhang, X.; Shen, X.; Yu, Z. A Novel Hybrid Ant Colony Optimization for a Multicast Routing Problem.Algorithms 2019, 12. doi:10.3390/a12010018.

[31] Aljohani, T.M.; Ebrahim, A.F.; Mohammed, O. Single and Multiobjective Optimal Reactive PowerDispatch Based on Hybrid Artificial Physics–Particle Swarm Optimization. Energies 2019, 12.doi:10.3390/en12122333.

[32] Ahmadian, A.; Elkamel, A.; Mazouz, A. An Improved Hybrid Particle Swarm Optimization and TabuSearch Algorithm for Expansion Planning of Large Dimension Electric Distribution Network. Energies2019, 12. doi:10.3390/en12163052.

[33] Li, G.; Liu, P.; Le, C.; Zhou, B. A Novel Hybrid Meta-Heuristic Algorithm Based on the Cross-EntropyMethod and Firefly Algorithm for Global Optimization. Entropy 2019, 21. doi:10.3390/e21050494.

[34] Cherki, I.; Chaker, A.; Djidar, Z.; Khalfallah, N.; Benzergua, F. A Sequential Hybridization of GeneticAlgorithm and Particle Swarm Optimization for the Optimal Reactive Power Flow. Sustainability 2019, 11.doi:10.3390/su11143862.

[35] Hanem, W.A.H.M.; Jantan, A. Hybridizing artificial bee colony with monarch butterfly optimizationfor numerical optimization problems. Neural Computing and Applications 2018, 30, 163–181.doi:10.1007/s00521-016-2665-1.

[36] Das, P.; Behera, H.; Panigrahi, B. A hybridization of an improved particle swarm optimization andgravitational search algorithm for multi-robot path planning. Swarm and Evolutionary Computation 2016,28, 14 – 28. doi:10.1016/j.swevo.2015.10.011.

[37] Wang, G.G.; Gandomi, A.H.; Zhao, X.; Chu, H.C.E. Hybridizing harmony search algorithm with cuckoosearch for global numerical optimization. Soft Computing 2016, 20, 273–285. doi:10.1007/s00500-014-1502-7.

[38] Zhu, A.; Xu, C.; Li, Z.;Wu, J.; Liu, Z. Hybridizing grey wolf optimization with differential evolution forglobal optimization and test scheduling for 3D stacked SoC. Journal of Systems Engineering and Electronics2015, 26, 317–328. doi:10.1109/JSEE.2015.00037.

[39] Javaid, N.; Ahmed, A.; Iqbal, S.; Ashraf, M. Day Ahead Real Time Pricing and Critical Peak Pricing BasedPower Scheduling for Smart Homes with Different Duty Cycles. Energies 2018, 11. doi:10.3390/en11061464.

[40] Migallón, H.; Jimeno-Morenilla, A.; Sánchez-Romero, J.; Belazi, A. Efficient parallel and fast convergence chaotic Jaya algorithms. Swarm and Evolutionary Computation 2020, p.100698. doi:10.1016/j.swevo.2020.100698.

★★★