HYBRIDIZED PARTICLE SWARM OPTIMIZATION FOR CONSTRAINED PROBLEMS

Nebojša Bačanin Džakula*,
Ivana Štrumberger,
Eva Tuba,
Milan Tuba

Singidunum University,
Belgrade, Serbia

Abstract:
This paper presents hybridized implementation of the well-known particle swarm optimization algorithm that belongs to the family of swarm intelligence metaheuristics. The proposed approach was adapted for tackling constrained optimization problems. With the basic goals to enhance the converge of the algorithm and to improve the exploitation – exploration tradeoff, the mechanism that replaces exhausted solutions from the population with randomly generated solutions from the search domain was adopted from the artificial bee colony approach. Proposed metaheuristic was tested on standard constrained engineering benchmark, and comparative analysis with other state-of-the-art algorithms was conducted. Empirical results obtained from practical simulations proved that the hybridized particle swarm optimization for constrained problems is able to successfully tackle this type of NP hard challenges.

Keywords:
particle swarm optimization, NP hardness, constrained optimization, engineering problems, swarm intelligence.

1. INTRODUCTION

Since many real-world problems can be formulated as optimization tasks, optimization has been widely applied in the domains of computer science and mathematics. The hardness of particular problem depends on the types of mathematical relationships between decision variables, objective function and in same cases constraints.

Many real-life challenges can be categorized as NP hard problems. These problems can be be further divdied into two basic groups: discrete (combinatorial) and global optimization (conitnuous) problems. Global optimization tasks can be distinguished as unconstrained (bound constrained) and constrained.

In this paper, the nonlinear continious constrained optimization problems are particulary addressed, and they can be mathematically expressed as:

$$\min f(x), x = (x_1, x_2, x_3, \dots, x_n) \in R^n, \quad (1)$$

Correspondence:
Nebojša Bačanin Džakula

e-mail:
nbacanin@singidunum.ac.rs

DOI: 10.15308/Sinteza-2019-17-25

where $x \in F \subseteq S$. Symbol S denotes the search space, as n-dimensional hyper-rectangular space in R^n. The R^n is defined by lower and upper bounds of decision variables:

$$lb_i \leq x_i \leq ub_i,\ 1 \leq i \leq n \tag{2}$$

Finally, the feasible domain of the search space $F \subseteq S$ is determined with a set of m linear and nonlinear constrains that are formulated as following:

$$\begin{aligned} &g_j(x) \leq 0, for\ j = 1, \ldots q \\ &h_j(x) = 0, for\ j = q+1, \ldots, m, \end{aligned} \tag{3}$$

where q denotes the number of inequality constraints, while the number of equalities constrains is m-q.

The utilization of classic deterministic algorithms (algorithms that for the same set of input always generate the same output), or exhaustive search approaches for solving NP hard problems is not feasible since it would take too long time for algorithms to execute. In the case of such problems, it is better to employ some of the metaheuristic methods, that could not guarantee to obtain an optimal solution, but could provide satisfying solution within reasonable execution period.

In recent years, many metaheuristic algorithms have been devised for solving large variety of optimization problems from both areas, combinatorial and global. Usually, metaheuristics are population and iterative based approaches that work with a set of solutions that are being improved in each iteration of algorithm's execution by applying the search equation. When a new metaheuristic approach is developed, it is good practice to test it first on standard benchmark problems to evaluate its solutions' quality and robustness. Later, the metaheuristics can be adapted for solving various kinds of real-life optimization problems.

In this paper, we present our implementation of the well-known particle swarm optimization (PSO) metaheuristics that was adjusted for solving global constrained optimization tasks. The PSO approach belongs to the group of swarm intelligence metaheuristics. The solutions' quality and the performance of the proposed approach was validated against the well-known conastraned optimization benchmarks.

The rest of the paper is structured as follows. In Section I, we brifely present basic principles and literature review of swarm intelligence metaheuristics. The implementation of the hybridized PSO algorithm for constrained optimization is given in Section II. Experimental setup, empircal results and comparative analysis is given in Section III, while Section IV concludes this paper and provides guidelines for future work in this domain.

2. REVIEW OF SWARM INTELLIGENCE METAHEURISTICS

At the very general level, metaheuristics can be divided into two groups: those that are inspired by the nature, and those that are not inspired by the nature. Nature-inspired metaheuristics can further fall into one of two categories: evolutionary algorithms (EAs) and swarm intelligence. The EAs conduct the search by simulating the process of natural evolution by applying operators adopted from the nature, such are crossover, mutation and selection, The most well-known example of EA is genetic algorithm (GA) [1]. The GA was successfully applied to various kinds of optimization challenges [2], [3].

Swarm intelligence metaheuristics mimic collective behavior and social interactions between individuals in swarms, like groups of birds and fish, bees, fireflies, bats, ant, etc [4]. Every artificial agent (individual) in swarm intelligence algorithms is relatively unsophisticated and simple. However by established communication between such agents, a sofisticated system is developed that is directed towards achiving a particular goal.

Two essential mechanisms that guide the search process in swarm intelligent approaches are exploitation (intensification) and exploration (diversification). The intensification perform the local search process around the current solutions in the population, while the diversification conduct exploration of the still undiscovered domain of the search space. Since many benchmark and real-life optimization problems have many local and/or global optimums, exploration prevents the algorithm to be trapped in the local optimu,, while the exploitaiton enables fine search process around the current best solutions in the population.

Artificial bee colony (ABC) is one of the most prominent representative of swarm algorihtms. The ABC simulates behavior of honey bee swarms by utilizing three types of artificail agents: employees, scouts and onlookers. This metaheuristics has many successful implementations for many benchmark problems [5], as well as for practical challenges [6], [7]. Firefly algorithm (FA) is another widely applied representative of swarm algorithms, that was devised by Yang in 2008 [8], and later was improved [9]. Some of the FA's applications

include: constrained benchmark challenges [10], wireless sensor network localization [11] and portfolio optimization [12]. Many hybridized FA approaches can be also found in the literature survey [7], [13].

Fireworks algorithm (FWA), that emulates the process of fireworks' explosion, was firstly proposed by Tan and Zhu in 2010 for tackling global optimization challenges [14]. According to the literature survey, since its creation, this metaheuristic has eight versions that have been adopted and applied to different benchmark [15] and practical challenges, for example retinal image registration [16], constrained portfolio optimization [17], multilevel image thresholding [18], RFID network planning problem [19], capacitated p-median problem [20]. Another swarm intelligence metaheuristic, that models herds of elephants (elephant herding optimization – ECHO), emerged in 2015. EHO is known as good NP hard problems optimizer with many implementations in both domains, benchmark [21], and real-life challenges. Some of the EHO's implementations for the practical tasks include: support vector machine (SVM) [22], static drone placement problem [23] and localization of sensors with unknown location in wireless sensor networks (WSNs) topology [24].

Relatively new monarch butterfly optimization (MBO) has been recently proposed by Wang and Deb for global benchmark problems [25]. Despite of this fact, MBO already has adaptations for practical tasks [26] and multi-objective optimization problems [27]. Another representative of swarm intelligence algorithm, moth serach (MS), that was also proposed by Wang [28], qualifies as the state-of-the-art metaheuristic. Besides implementations for the benchmark problems [29], [30], many MS's adaptations for the real-world problems can be found in the literature survey [31], [32].

Besides all above mentioned, other swarm intelligence approaches that are worth of mentioning encompass: seeker optimization algorithm (SOA) [33], cuckoo search (CS) [34], [35], [36], bran storm optimization (BSO) [37], [38], bat algortithm (BA) [39], [40], and state-of-the-art approach for cobinatorial optimization, ant colony optimization (ACO) [41].

3. PARTICLE SWARM OPTIMIZATION FOR CONSTRAINED PROBLEMS

Particle swarm optimization (PSO) is well-known swarm intelligence approach devised in 1995 by Kennedy and Eberhart [42], [43]. The PSO proved to be robust and state-of-the-art metaheuristics with many implementations and adaptations [44]. In this Section of the paper we present our adaptations of this outstanding optimizer adapted for tackling constrained optimization problems.

In the basic PSO implementation, potential problem solution (individual in the population) within the boundaries of the search space domain is represented as particle. Each solution i in the D-dimensional search space is defined by its position and velocity, denoted as $x_i=(x_{i1},x_{i2},x_{i3},\ldots,x_{iD})$ and $v_i=(v_{i1},v_{i2},v_{i3},\ldots,v_{iD})$, respectively.

At the beginning of algorithm's execution, in the initialization phase, all solutions in the population are generated randomly within the lower and upper boundaries of the search space (see Eqs. (1)-(2)). After random initialization, in each iteration of algorithm's execution, the position and velocity of each solution are updated using the following expressions [43]:

$$v_i(t+1) = wv_i(t) + c_1 r_1\left(p_i - x_i(t)\right) + c_2 r_2(p_g - x_i(t)) \tag{4}$$

$$x_i(t+1) = x_i(t) + v_i(t+1), \tag{5}$$

where the ω denotes the inertia weight that is utilized for the purpose of controlling the influence of the old velocity to the new one, c_1 and c_2 control constants that are used for determining the weights of p_g and p_i. Previously best position of the i-th individual in the population is denoted as p_i, while the p_g represents the best previous position of solutions in the current iteration. Finally, r_1 and r_2 are pseudo-random numbers uniformly generated in the range [0,1], t is the current iteration, and $t+1$ is the next iteration.

By utilizing presented equations (Eq. (4) and Eq. (5)) the exploitation and exploration process of the PSO metaheuristics is being performed. By adjusting the values of c_1 and c_2 control parameters, the balance (trade-off) between the intensification and diversification is also being adjusted. If the values of c_2 and c_2 parameters are too high, this balance is shifted towards exploitation. Otherwise, if the values of c_2 and c_2 parameters are too low, the trade-off favours the process of exploration.

According to prevously conducted studies [12], [13], [33], in the first iterations of the algorithm's execution, the exploration should be more intensive, due to the assumption that the search process has not yet converged to the optimum domain of the search space. However, in the later iterations, with the basic assumption that the

promising part of the search domain is found, the power of intensification should be enhanced.

For the sake of better controlling the balance between exploitation and exploration, in our PSO implementation, we introduced the *limit* parameter from the ABC metaheuristic. The basic idea behind this approach can be summarized into one sentence: each solution in the population that can not be enahnced in a predefined number of iterations is being discarded from the population and replaced with the pseudo-random solution generated within the lower and upper boundaries of the search domain. Every time when a particular solution could not be improved, the value of its *limit* parameter is incremented. When a value of the *limit* parameter reaches the threshold value (*tvalue*), this solution is discarded from the population.

One of the greatest challenges in the domain of constrained optimization is how to handle constraints. Equality constraints could pose serious problem due to the fact that they could significantly shrink the feasible part of the search space that becomes very small compared to the entire search space. In our PSO implementation, we replaced the equality constraints by inequality by using small violation limit $\varepsilon > 0$ [45]:

$$|h(x)| - \varepsilon \leq 0 \tag{6}$$

Feasibility of the solutions depends on the value of violation limit ε. If this value is too low, the search process may not find feasible part of the search space. Otherwise, if the chosen value for the ε is too high, obtained results may be far from the feasible space. One of the best approaches is to adapt dynamic violation limit approach, by starting with the large value of violation limit in early iterations, and then to gradually decrease its value during the course of algorithm's execution.

In our PSO implementation for this purpose we used the following expression [13], [33]:

$$\varepsilon(t+1) = \frac{\varepsilon(t)}{dec}, \tag{7}$$

where t and $t+1$ denote the current iteration and next iterations, respectively, and $dec>1$ represents the decreasing coefficient in each iteration.

In our implementation we perform the selection process between old and new solution, after utilization of the search equations (Eq. (4) and Eq. (5)), by employing Deb's rules [46], [47]. In this way, we make sure that the feasible solutions are favoured over infeasible solutions, and also that if both solutions, old and new, are infeasible, we make sure that the infeasible solution that is closer to the feasible part of the search region is favoured over other infeasible solution, that is distant from the feasible space.

By introducing all described modifications in the basic PSO version, we devised hybridized constrained PSO (HCPSO) metaheuristic. All execution steps of the HCPSO are summarized in the pseudo-code given below.

1. Initialize pseudo-random population
2. Evaluate population
3. Set the violation limit to 1
4. *iter* = 1
5. **repeat**
6. For every solution in the population, generate new solution by using Eq. (4) and Eq. (5)
7. Apply selection between old and new solution based on the Deb's rules [46], [47]
8. For all solutions that can not be improved, increment the value of the *limit* parameter
9. Replace all solutions from the population whose *limit* parameter value reach the *tvalue*. with randomly generated solutions
10. Memorize the best solution obtained so far
11. If the condition is met, dynamically adjust the value by using Eq. (7)
12. *iter* = *iter* +1
13. **until** *iter* = *MITER*
14. Output the best solution in the population

In the presented pseudo-code, *MITER* denotes the maximum iteration number in one algorithm's execution (run).

4. PARAMETER SETUP, EMPIRICAL RESULTS AND COMPARATIVE ANALYSIS

In this section of the paper we first show one of the well-known constrained engineering benchmarks that was used for validation purposes of our proposed approach. Then we show HCPSO control parameters adjustments, and finally we present empirical results along with comparative analysis with other state-of-the-art metaheuristics tested on the same problem.

Speed reducer design problem

In order to prove robustness, convergence speed and solution's quality of the HCPSO metaheuristic, we utilized speed reducer design problem that was firstly introduced by Golinski [48]. The basic objective of this problem is to minimize the weights of the speed reducer. Visual representation of the speed reducer is given in Fig. 1.

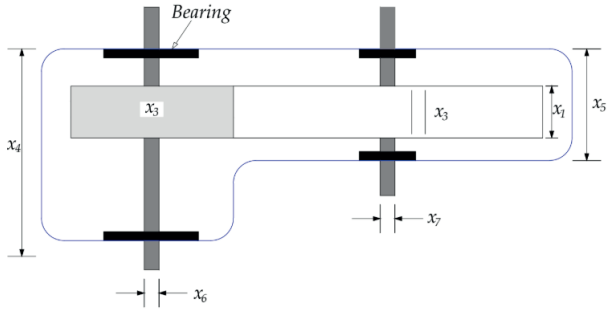

Fig. 1. Example of a figure caption. *(figure caption)*

The speed reducer problem includes seven design variables: face width (x_1), teeth module (x_2), pinion teeth number (x_3), shaft between the bearings (x_4), first shaft length (x_5), first shaft diameter (x_6) and the second shaft diameter (x_7). Moreover, the speed reducer problem incorporates eleven inequality constraints.

Mathematical formulation of the speed reducer problem is given in Eqs. (8) – (20).

$$min\, W(x) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2x_7^2) + 7.4777(x_6^3x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2) \quad (8)$$

Subject to:

$$g_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \le 0 \quad (9)$$

$$g_2(x) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0 \quad (10)$$

$$g_3(x) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \le 0 \quad (11)$$

$$g_4(x) = \frac{1.93x_4^3}{x_2x_3x_7^4} - 1 \le 0 \quad (12)$$

$$g_5(x) = \frac{1}{110x_6^3}\sqrt{\left(\frac{754x_4}{x_2x_3}\right)^2 + 16.9 \cdot 10^6} - 1 \le 0 \quad (13)$$

$$g_6(x) = \frac{1}{85x_7^3}\sqrt{\left(\frac{754x_5}{x_2x_3}\right)^2 + 157.5 \cdot 10^6} - 1 \le 0 \quad (14)$$

$$g_7(x) = \frac{x_2x_3}{40} - 1 \le 0 \quad (15)$$

$$g_8(x) = \frac{5x_2}{x_3} - 1 \le 0 \quad (16)$$

$$g_9(x) = \frac{x_1}{12x_2} - 1 \le 0 \quad (17)$$

$$g_{10}(x) = \frac{(1.5x_6+1.9)}{x_4} - 1 \le 0 \quad (18)$$

$$g_{11}(x) = \frac{(1.1x_7+1.9)}{x_5} - 1 \le 0 \quad (19)$$

with parameter's bounds:

$$2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28$$
$$7.3 \le x_4 \le 8.3, 7.8 \le x_5 \le 8.3, 2.9 \le x_3 \le 3.9 \quad (20)$$
$$5 \le x_7 \le 5.5$$

HCPSO Control parameters setup

We adjusted the global HCPSO control parameters as follows: number of solutions in the population (*N*) was set to 40, and *MITER* was set to the value of 750. These settings yield to the total number of 30,000 objective function evaluations in one algorithm's run (40 750 = 30,000).

The HCPSO local control parameters c_1 and c_2 were both set to the value of 2, while the inertia weight () was adjusted to 0.7. For setting the threshold value (*tvalue*) to 19, by using the following equation:

$$tvalue = round(\frac{MITER}{N}) \quad (21)$$

As already stated in the Section II, at the beginning of the algorithm's execution we set to 1, whose value was gradually decreasing during the algorithm's execution according to Eq. (7). The value of the *dec* was set to 1.002, while the threshold for the was adjusted to 0.0001.

Empirical results and comparative analysis

Comparative analysis was performed between our proposed approach HCPSO and society and civilization (SCA) metaheuristic, hybridized artificial immune system with clearing procedure (AIS-GA^{c}), ABC and BA metaheuristics. Results of other approaches for the purpose of comparative analysis were taken from [49].

Results of conducted comparative analysis are presented in the Table I. The notation "V" used for some results indicate that such results are not feasible. From the presented comparative analysis, it is clear that HCPSO obtains the best value for objective function (w). Results reported for the BA metaheuristic are slightly better, but violate g_5 and g_6 constrains, and such are infeasible.

It also should be noted that the SCA and AIS-GA^{S} were tested with 54,456 and 36,000 function evaluations [50], which is significantly more than 30,000 function evaluation that were utilized in tests conducted with HCPSO metaheuristic. The ABC approach was also tested with 30,000 objective function evaluations, while the BA metaheuristic utilized only 15,000 evaluations [50]. However, the results reported for the BA approach are not feasible and cannot be taken into consideration.

Table 1. Simulation Results And Comparative Analysis

	Algorithms				
	SCA	*AIS-GA^{C}*	*ABC*	*BA^{V}*	*HCPSO*
x_1	3.50000681	3.5	3.499999	3.5	3.50000001
x_2	0.70000001	0.7	0.7	0.7	0.7
x_3	17	17	17	17	17
x_4	7.32760205	7.3000035	7.3	7.30001	7.30000012
x_5	7.71532175	7.7153225	7.8	7.71532	7.71532019
x_6	3.35026702	3.3502147	3.350215	3.35021	3.3502145
x_7	5.28665450	5.2866545	5.287800	5.28665	5.28665437
g_1	−0.073917	−0.07391524	−0.073915	−0.074	−0.07391518
g_2	−0.198000	−0.19799852	−0.197999	−0.198	−0.19799853
g_3	−0.493501	−0.49917156	−0.499172	−0.499	−0.49917226
g_4	−0.904644	−0.90464383	−0.901555	−0.905	−0.9046439
g_5	−6.362E−07	−2.451E−08	−2.990E−07	4.195E−06^{v}	−3.013E−08
g_6	−1.954E−08	−1.938E−08	−6.335E−04	2.534E−05^{v}	−2.754E−09
g_7	−0.702500	−0.7025	−0.7025	−0.7025	−0.7025
g_8	−1.931E−06	0	2.857E−07	0	−9.431E−09
g_9	−0.583333	−0.5833333	−0.583333	−0.583	−0.58333333
g_{10}	−0.054889	−0.05132616	−0.051326	−0.051	−0.05132576
g_{11}	−2.333E−07	−3.305E−07	−0.010695	−6.481E−07	−4.315E−08
w	2,994.7442	2,994.4712	2,997.0584	2,994.4671	**2,994.470123**

5. CONCLUSION

In this paper we presented implementation of the hybridized particle swarm optimization for constrained problems. In order to enhance the convergence, and to improve the balance between intensification and diversification, we incorporated the *limit* control parameter from the well-known artificial bee colony metaheuristic. The proposed approach was named hybridized constrained particle swarm optimization (HCPSO).

Proposed metaheuristic was tested on standard constrained engineering optimization benchmark – speed reducer design problem. According to the results of comparative analysis with other state-of-the-art optimizers that were tested for the same problem instance, and under the equivalent experimental conditions it can be concluded that the proposed HCPSO represents promising approach for tackling these types of NP hard challenges.

Since many real-world optimization tasks belong to this group of problems, the HCPSO will be adapted for other problems as part of the future research.

ACKNOWLEDGMENT

This research is supported by Ministry of Education and Science of Republic of Serbia, Grant No. III-44006

REFERENCES

[1] D. E. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning". Addison-Wesley Longman Publishing, 1989.

[2] E. P. Ijjina and K. M. Chalavadi, "Human action recognition using genetic algorithms and convolutional neural networks," Pattern Recognition, vol. 59, pp. 199 – 212, 2016. Compositional Models and Structured Learning for Visual Recognition.

[3] A. Suriya and J. D. Porter, "Genetic algorithm based approach for RFID network planning," in TENCON 2014 - 2014 IEEE Region 10 Conference, pp. 1–5, Oct 2014.

[4] X.-S. Yang, "Swarm intelligence based algorithms: a critical analysis," Evolutionary Intelligence, vol. 7, pp. 17–28, April 2014.

[5] N. Bacanin and M. Tuba, "Artificial bee colony (ABC) algorithm for constrained optimization improved with genetic operators," Studies in Informatics and Control, vol. 21, pp. 137–146, June 2012.

[6] N. Bacanin, M. Tuba, and I. Strumberger, "RFID network planning by ABC algorithm hybridized with heuristic for initial number and locations of readers," in 2015 17th UKSim-AMSS International Conference on Modelling and Simulation (UKSim), pp. 39–44, March 2015.

[7] M. Tuba and N. Bacanin, "Artificial bee colony algorithm hybridized with firefly metaheuristic for cardinality constrained mean-variance portfolio problem," Applied Mathematics & Information Sciences, vol. 8, pp. 2831–2844, November 2014.

[8] X.-S. Yang, Nature-Inspired Metaheuristic Algorithms: Second Edition. Luniver Press, July 2010.

[9] X.-S. Yang, "Firefly algorithm, stochastic test functions and design optimisation," International Journal of Bio-Inspired Computation, vol. 2, no. 2, pp. 78–84, 2010.

[10] I. Strumberger, N. Bacanin, and M. Tuba, "Enhanced firefly algorithm for constrained numerical optimization, IEEE congress on evolutionary computation," in Proceedings of the IEEE International Congress on Evolutionary Computation (CEC 2017), pp. 2120–2127, June 2017.

[11] E. Tuba, M. Tuba, and M. Beko, "Two stage wireless sensor node localization using firefly algorithm," in Smart Trends in Systems, Security and Sustainability (X.-S. Yang, A. K. Nagar, and A. Joshi, eds.), (Singapore), pp. 113–120, Springer Singapore, 2018.

[12] N. Bacanin and M. Tuba, "Firefly algorithm for cardinality constrained mean-variance portfolio optimization problem with entropy diversity constraint," The Scientific World Journal, special issue Computational Intelligence and Metaheuristic Algorithms with Applications,, vol. 2014, no. Article ID 721521, p. 16, 2014.

[13] I. Strumberger, N. Bacanin, and M. Tuba, "Hybridized elephant herding optimization algorithm for constrained optimization," in Hybrid Intelligent Systems (A. Abraham, P. K. Muhuri, A. K. Muda, and N. Gandhi, eds.), (Cham), pp. 158–166, Springer International Publishing, 2018.

[14] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," Advances in Swarm Intelligence, LNCS, vol. 6145, pp. 355–364, June 2010.

[15] N. Bacanin, M. Tuba, and M. Beko, "Hybridized fireworks algorithm for global optimization," in Mathematical Methods and Systems in Science and Engineering, pp. 108–114, 2015.

[16] E. Tuba, M. Tuba, and E. Dolicanin, "Adjusted fireworks algorithm applied to retinal image registration," Studies in Informatics and Control, vol. 26, pp. 33–42, March 2017.

[17] N. Bacanin and M. Tuba, "Fireworks algorithm applied to constrained portfolio optimization problem," in Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC 2015), May 2015.

[18] M. Tuba, N. Bacanin, and A. Alihodzic, "Multilevel image thresholding by fireworks algorithm," in 2015 25th International Conference Radioelektronika (RADIOELEKTRONIKA), pp. 326–330, April 2015.

[19] I. Strumberger, E. Tuba, N. Bacanin, M. Beko, and M. Tuba, "Bare bones fireworks algorithm for the RFID network planning problem," in 2018 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8, July 2018.

[20] E. Tuba, I. Strumberger, N. Bacanin, and M. Tuba, "Bare bones fireworks algorithm for capacitated p-median problem," in Advances in Swarm Intelligence (Y. Tan, Y. Shi, and Q. Tang, eds.), (Cham), pp. 283–291, Springer International Publishing, 2018.

[21] I. Strumberger, N. Bacanin, and M. Tuba, "Hybridized elephant herding optimization algorithm for constrained optimization," in Hybrid Intelligent Systems (A. Abraham, P. K. Muhuri, A. K. Muda, and N. Gandhi, eds.), (Cham), pp. 158–166, Springer International Publishing, 2018.

[22] E. Tuba and Z. Stanimirovic, "Elephant herding optimization algorithm for support vector machine parameters tuning," in Proceedings of the 2017 International Conference on Electronics, Computers and Artificial Intelligence (ECAI), pp. 1–5, June 2017.

[23] I. Strumberger, N. Bacanin, M. Beko, S. Tomic, and M. Tuba, "Static drone placement by elephant herding optimization algorithm," in Proceedings of the 24th Telecommunications Forum (TELFOR), pp. 374-377, November 2017.

[24] I. Strumberger, M. Beko, M. Tuba, M. Minovic, N. Bacanin, "Elephant Herding Optimization Algorithm for Wireless Sensor Network Localization Problem", Chapter in Technological Innovation for Resilient Systems, IFIP Advances in Information and Communication Technology, Vol. 521, Springer, pp. 175 - 184, 2018.

[25] G.-G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," Neural Computing and Applications, pp. 1–20, May 2015.

[26] I. Strumberger, E. Tuba, N. Bacanin, M. Beko, and M. Tuba, "Monarch butterfly optimization algorithm for localization in wireless sensor networks," in 2018 28th International Conference Radioelektronika (RADIOELEKTRONIKA), pp. 1–6, April 2018.

[27] I. Strumberger, E. Tuba, N. Bacanin, M. Beko and M. Tuba, "Modified and Hybridized Monarch Butterfly Algorithms for Multi-Objective Optimization", Chapter in: Advances in Intelligent Systems and Computing: Hybrid Intelligent Systems, Vol. 923, pp. 449 - 458, Mar, 2019.

[28] G.-G. Wang, "Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems," Memetic Computing, Sep 2016.

[29] I. Strumberger and N. Bacanin, "Modified Moth Search Algorithm for Global Optimization Problems", International Journal of Computers, Vol. 3, pp. 44 - 48, Mar, 2018.

[30] I. Strumberger, E. Tuba and N. Bacanin, M. Beko, M. Tuba, "Hybridized Moth Search Algorithm for Constrained Optimization Problems", IEEE 2nd International Young Engineers Forum on Electrical and Computer Engineering (YEF-ECE), pp. 1 - 5, May, 2018.

[31] I. Strumberger, E. Tuba and N. Bacanin, M. Beko, M. Tuba, "Wireless Sensor Network Localization Problem by Hybridized Moth Search Algorithm", 14th IEEE International Wireless Communications and Mobile Computing Conference (IWCMC 2018), pp. 316 - 321, May, 2018.

[32] I. Strumberger, M. Sarac, D. Markovic and N. Bacanin, "Moth Search Algorithm for Drone Placement Problem", International Journal of Computers, Vol. 3, pp. 75 - 80, Apr, 2018

[33] M. Tuba and N. Bacanin, "Improved seeker optimization algorithm hybridized with firefly algorithm for constrained optimization problems", Neurocomputing, vol. 143, pp. 197–207, 2014.

[34] M. Tuba, A. Alihodzic, and N. Bacanin, "Cuckoo Search and Bat Algorithm Applied to Training Feed-Forward Neural Networks", pp. 139–162. Cham: Springer International Publishing, 2015.

[35] N. Bacanin, "Implementation and Performance of an Object-Oriented Software System for Cuckoo Search Algorithm", International Journal of Mathematics and Computers in Simulation, Vol. 6, No. 1, pp. 185 - 193, Mar, 2012.

[36] M. Tuba, A. Alihodzic, and N. Bacanin, "Cuckoo Search and Bat Algorithm Applied to Training Feed-Forward Neural Networks", pp. 139–162. Cham: Springer International Publishing, 2015.

[37] E. Tuba, I. Strumberger, D. Zivkovic, N. Bacanin, and M. Tuba, "Mobile robot path planning by improved brain storm optimization algorithm," in 2018 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8, July 2018.

[38] E. Tuba, I. Strumberger, N. Bacanin, D. Zivkovic, M. Tuba, "Cooperative Clustering Algorithm Based on Brain Storm Optimization and K-Means", IEEE 28th International Conference Radioelektronika, pp. 1 - 5, Apr, 2018.

[39] M. Tuba, N. Bacanin, "Hybridized Bat Algorithm for Multi-objective Radio Frequency Identification (RFID) Network Planning", IEEE Congress on Evolutionary Computation (CEC), pp. 499 - 506, 2015.

[40] I. Strumberger, N. Bacanin, M. Tuba, "Constrained Portfolio Optimization by Hybridized Bat Algorithm", 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS), pp. 83 - 88, Jan, 2016.

[41] R. Jovanovic, M. Tuba, S. Voss, "An Efficient Ant Colony Optimization Algorithm for the Blocks Relocation Problem", European Journal of Operational Research, Vol. 274, No. 1, pp. 78 - 90, Apr, 2019.

[42] J. Kennedy and R. Eberhart, "Particle swarm optimization," Vol. 4, pp. 1942–1948, 1995.

[43] R. Eberhart. and J. Kennedy, "A new optimizer using particle swarm theory," in Proceedings of the 6th international symposium on micromachine and human science, pp. 39–43, IEEE, 1995.

[44] H. Zhu, Y. Wang, K. Wang, and Y. Chen, "Modified particle swarm optimization for nonconvex economic dispatch problems," International Journal of Electrical Power & Energy Systems, vol. 69, pp. 304–312, July 2015.

[45] E. Mezura-Montes (editor), "Constraint-Handling in Evolutionary Optimization", vol. 198 of Studies in Computational Intelligence. Springer- Verlag, 2009.

[46] K. Deb, "An efficient constraint-handling method for genetic algorithms," Computer Methods in Applied Mechanics and Engineering, vol. 186, no. 2-4, pp. 311–338, 2000.

[47] K. Deb, "Optimization for engineering design, algorithms and examples," Computer Methods in Applied Mechanics and Engineering, p. 396, 2005.

[48] J. Golinski, "An adaptive optimization system applied to machine synthesis". Mech.Mach. Theory vol. 8, Issue 4, pp. 419–436, 1973.

[49] H.-P. Dai, D.-D. Chen, and Z.-S. Zheng, "Effects of random values for particle swarm optimization algorithm", Algorithms, Vol. 11, No. 2, 2018, doi: https://doi.org/10.3390/a11020023

[50] M. K. Dhadwal, S. N. Jung, and C. J. Kim, "Advanced particle swarm assisted genetic algorithm for constrained optimization problems," Computational Optimization and Applications, vol. 58, pp. 781–806, Jul 2014.