Whale Optimization Algorithm with Exploratory Move for Wireless Sensor Networks Localization

Nebojsa Bacanin[1] [0000-0002-2062-924X], Eva Tuba[1] [0000-0003-4866-9048], Miodrag Zivkovic[1] [0000-0002-4351-068X], Ivana Strumberger[1] [0000-0002-1154-6696], and Milan Tuba[1] [0000-0003-3794-3056]

Singidunum University, 11000 Belgrade, Serbia
{nbacanin,mzivkovic,istrumberger}@singidunum.ac.rs,{etuba,tuba}@ieee.org

Abstract. In the modern era, with the development of new technologies, such as cloud computing and the internet of things, there is a greater focus on wireless distributed sensors, distributed data processing and remote operations. Low price and miniaturization of sensor nodes have led to a large number of applications, such as military, forest fire detection, remote surveillance, volcano monitoring, etc. The localization problem is among the greatest challenges in the area of wireless sensor networks, as routing and energy efficiency depend heavily on the positions of the nodes. By performing a survey of computer science literature, it can be observed that in the wireless sensor networks localization domain, swarm intelligence metaheuristics have generated compelling results. In the research proposed in this paper, a modified/improved whale optimization swarm intelligence algorithm, that incorporates exploratory move operator from Hooke-Jeeves local search method, applied to solve localization in wireless networks, is presented. Moreover, we have compared the proposed improved whale optimization algorithm with its original version, as well as with some other algorithms that were tested on the same model and data sets, in order to evaluate its performance. Simulation results demonstrate that our presented hybridized approach manages to accomplish more accurate and consistent unknown nodes locations in the wireless networks topology, than other algorithms included in comparative analysis.

Keywords: Node localization · Wireless sensor networks · Swarm intelligence · Hybridization · Whale optimization algorithm

1 Introduction

The wireless sensor networks (WSNs) are networks that consist of many small and cheap wireless devices, i.e. sensor nodes, used to detect different phenomena from the physical world. Due to very limited resources, every node can process just a small portion of the collected data. However, a large number of nodes working together can measure given physical variable very precisely. WSNs rely on the coordination of a large number of nodes in dense layout to perform its task [1].

Today, with the development of novel technologies and computing paradigms like cloud computing and the internet of things (IoT), the focus is on wireless distributed sensors, distributed data processing and remote operation [7]. Additionally, low price and miniaturization of sensor nodes led to a large number of applications, such as military, forest fire detection, remote surveillance, volcano monitoring, etc. The exact location of target phenomena is often unknown, therefore distributed sensor nodes allow better coverage and closer positioning (very important for hostile areas, such as war zones, radioactive areas, etc.). Additionally, the monitored area usually does not have any existing infrastructure such as telecommunications or power supply. In such an environment, sensors that are deployed randomly, must operate with limited resources and communicate in a wireless manner [23]. It can be safely assumed that their exact positions are not known. Therefore, among the main challenges from the WSNs domains is localization, which refers to finding positions of the deployed sensors. Using the global positioning system (GPS) is not feasible, because sensors have limitations in terms of computing power and attainable energy.

The WSNs localization problem is an NP-hard by nature and classical algorithms can not be implemented (for example deterministic algorithms), due to high complexity and often unacceptable computational time [3]. For tackling this problem, stochastic algorithms like swarm intelligence are capable of generating satisfying solutions within a relatively short interval.

Swarm intelligence metaheuristics fall into the category of bio-inspired optimization methods. According to the literature survey, these approaches have been successfully applied to solving many complex NP-hard real-life problems. Some examples of the swarm algorithms that have many practical applications include: artificial bee colony (ABC) [22], fireworks algorithm (FWA) [19, 20], bran storm optimization (BSO) [18], monarch butterfly optimization (MBO) [17], firefly algorithm [21] and tree growth algorithm (TGA) [16]. Moreover, many hybridized swarm algorithms exist [10, 12, 15], as well as their implementations for WSNs localization problem [11, 13, 14].

The research presented in this paper aims towards achieving further enhancements in tackling WSNs localization problem by applying swarm intelligence algorithms. We propose in this paper a hybrid whale optimization swarm intelligence algorithm, that adopts exploratory move operator from Hooke-Jeeves local search method, adapted for solving localization challenge.

The structure of the paper can be summarized as follows. Localization problem mathematical formulation, which was used in simulations, is given in Section 2. The Section 3 presents hybridized whale optimization algorithm tuned for node localization problem. In Section 4, simulation environment, as well as accomplished results and side-by-side comparisons are given. The Section 5 wraps up the proposed paper and also provides references for the upcoming research.

2 Background and proposed model

In the WSNs topology, there are typically two basic types of sensors, anchors and targets. Anchors usually utilize GPS for determining their location. Target nodes are randomly distributed in target area and their locations must be estimated by applying localization algorithms. The estimation is usually conducted in two phases [5]. In the first phase, that is known as the ranging phase, methods estimate neighboring anchors and unknown sensors nodes distance. On the contrarily, position of sensors is estimated by applying geometry principles in the second phase.

The objective of localization is estimation of coordinates of randomly distributed sensors (targets), with the goal to minimize the objective function. The position of target node is estimated by the range-based localization technique.

In the first phase, metric received signal strength indicator (RSSI) was used to assess the distance between the target and anchor nodes, and that signal was corrupted by Gaussian noise. In the second phase, positions of target nodes were estimated by using trilateration, together with the results from the ranging phase. In order the trilateration technique to work, the distances between at least three anchors and the node with unknown location should be known in advance. Since measurements have imprecision in both phases, swarm intelligence can be utilized to minimize the error of localization.

The M target and N anchors sensors are randomly deployed in a $2D$ environment, which the range of transmission denoted as R. The distance between each target node and anchors in its range is given by equation $\hat{d_i} = d_i + n_i$, where n_i is an additive Gaussian noise, and d_i is the real distance determined by using the following expression:

$$d_i = \sqrt{(x - x_i)^2 + (y - y_i)^2}, \tag{1}$$

where target and anchor nodes positions are represented as (x, y) and (x_i, y_i), respectively.

The variance of n_i, as the noise that affects the measured distance between anchors and target senors, is given as:

$$\sigma_d^2 = \beta^2 \cdot P_n \cdot d_i, \tag{2}$$

where P_n is the percentage noise in distance calculation $d_i \pm d_i(\frac{P_n}{100})$, and β is a parameter whose value is usually adjusted to 0.1 in practical implementations.

The target node is localized if there are at least three anchors with the known positions A (x_a, y_a), B (x_b, y_b), and C (x_c, y_c), within its transmission range R, and with distance d_i from the target node n.

The swarm intelligence metaheuristic was executed independently for each localizable target node to estimate its position. Artificial individuals are initialized within the anchor nodes centroid by using the following expression:

$$(x_c, y_c) = \left(\frac{1}{N} \sum_{i=1}^{N} x_i, \frac{1}{N} \sum_{i=1}^{N} y_i \right), \tag{3}$$

where N denotes the number of anchors within target node range.

The $f(x, y)$, that represents node localization problem objective function, is formulated as the mean square distance between the anchor and target node, given in Eq. (4), where $N \geq 3$ [5].

$$f(x, y) = \frac{1}{N}\left(\sum_{i=1}^{N} \sqrt{(x - x_i)^2 + (y - y_i)^2}\right)^2, \tag{4}$$

Localization error E_L is given by the Equation (5), as the mean of squared distance between the estimated (X_i, Y_i) and the real node coordinates (x_i, y_i):

$$E_L = \frac{1}{N_L} \sum_{i=1}^{N} \sqrt{(x_i - X_i)^2 + (y_i - Y_i)^2} \tag{5}$$

The efficiency of the algorithm is measured by the localization error average value E_L and the number of non-localized sensors N_{NL}, where $N_{NL} = M - N_L$.

3 Hybridized whale optimization algorithm

The original implementation of the WOA was introduced in 2016 by Mirjalili and Lewis [9] for tackling unconstrained and constrained continuous optimization problems [8], [6]. In this paper, hybridized WOA will be presented.

The search process of the WOA is performed by mathematically modeling the humpback whales hunting behavior. In the nature, humpback whales express cooperating behavior while hunting their prey by performing a distinctive hunting strategy, which is in the literature refereed as a bubble-net feeding strategy. These whales chaise small fishes by producing a spiral bubble path which surrounds their prey and swimming up to the surface of the ocean.

The WOA's search process is being conducted by simultaneously performing diversification and intensification phases. The process of exploitation models the encircling of prey and spiral bubble-net strategy, while the exploration emulates a search for a prey.

In the phase of exploitation, each candidate solution performs a search in its neighborhood and it is directed towards the location where is the current global best solution. When for each solution in the population a fitness is calculated, positions of all solutions are updated respect to the location of fittest solution [9]:

$$\overrightarrow{D} = |\overrightarrow{C} \cdot \overrightarrow{X}^*(t) - \overrightarrow{X}(t)| \tag{6}$$

$$\overrightarrow{X}(t+1) = \overrightarrow{X}^*(t) - \overrightarrow{A} \cdot \overrightarrow{D}, \tag{7}$$

where $\overrightarrow{X}(t)$ and $\overrightarrow{X}^*(t)$ denote candidate and current best solutions in iteration t, $\overrightarrow{A}$ and $\overrightarrow{C}$ represent coefficients given by the following expressions [9]:

$$\overrightarrow{A} = 2\overrightarrow{a} \cdot \overrightarrow{r} - \overrightarrow{a} \tag{8}$$

$$\overrightarrow{C} = 2 \cdot \overrightarrow{r} \tag{9}$$

Original WOA version emulates the bubble-net strategy by utilizing the expression [9]:

$$\overrightarrow{a} = 2 - t\frac{2}{maxIter}, \tag{10}$$

where t and $maxIter$ represent the current and maximal iteration numbers, respectively.

Second mechanism that guides the process of exploitation executes in two steps: first, the length of the space between the fittest solution ($\overrightarrow{X}^*(t)$) and current solution ($\overrightarrow{X}(t)$) in iteration t is calculated, and then, a new (updated) candidate solution ($\overrightarrow{X}(t+1)$) can be determined by using a spiral equation [9]:

$$\overrightarrow{X}(t+1) = \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X}^*(t), \tag{11}$$

where $\overrightarrow{D'}$ is defined as $\overrightarrow{D'} = |\overrightarrow{X}^*(t) - \overrightarrow{X}(t)|$, b represents a constant used to define a shape of logarithmic spiral, while l denotes pseudo-random number between -1 and 1.

The whales simultaneously move around the pray together with a spiral path and shrink the circle, which is simulated by choosing between shrinking and spiral-shaped path in each iteration with equal probability p:

$$\overrightarrow{X}(t+1) = \begin{cases} \overrightarrow{X}^*(t) - \overrightarrow{A} \cdot \overrightarrow{D} \text{ , if } p < 0.5 \\ \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X}^*(t) \text{ , if } p \geq 0.5 \end{cases} \tag{12}$$

The exploration phase is conducted by updating each candidate solution in the population with respect to the position of a randomly chosen solution rather than of the global fittest solutions, as it is the case in the process of exploitation. The following expression models WOA's exploration phase [9]:

$$\overrightarrow{X}(t+1) = \overrightarrow{X}_{rnd}(t) - \overrightarrow{A} \cdot \overrightarrow{D}, \tag{13}$$

where $\overrightarrow{D}$, distance between the i-th candidate and the random solution from the population rnd at iteration t, is given by $\overrightarrow{D} = |\overrightarrow{C} \cdot \overrightarrow{X}_{rnd}(t) - \overrightarrow{X}(t)|$.

By conducting empirical simulations, we concluded that the original WOA version exhibits the behavior of premature convergence, and as a consequence, algorithm usually traps in one of the suboptimal regions of the search domain. The exploration is conducted only in cases when conditions $p < 0.5$ and $|A \geq 1|$ are satisfied. The exploration process should be more intensive, especially in the early phases of algorithm's execution. The basic WOA implementation is explained in more details in [9].

In order to overcome observed deficiencies, we adapted exploratory move (EM) from the Hooke-Jeeves local search method, that proved to be an efficient optimization technique [4]. With an assumption that $\overrightarrow{X}^*$ represents the current best solution (the base point), f_{min} is the current minimum objective function value, $\overrightarrow{\delta} = (\delta_1, \delta_2, ..., \delta_n)$ denote step sizes in n directions and $\overrightarrow{x_t}$ is temporary vector,the main steps of EM can be summarized in Algorithm 1.

Algorithm 1 EM pseudo-code

```
Initialization: x_t = X*
for (i = 1 to n) do
    x_{t,i} = X*_i + δ_i
    if (f(x_t) < f_min) then
        continue
    else
        x_{t,i} = X*_i − δ_i
        if (f(x_t) < f_min) then
            continue
        else
            x_{t,i} = X*_i
        end if
    end if
end for
```

Moreover, in our implementation we used adaptive step size $\overrightarrow{\delta}$. First, all solutions in the population are ranked based on the value of the objective function, and after the first 10% best solutions are selected for the step size calculation by the following expression $\delta_j = 0.1 \cdot (\sum_{i=1}^{m}(\overrightarrow{X_{i,j}} - \overrightarrow{X}_j^*))/m$,where δ_j denotes the step size in j-th dimension, and m represents the number of 10% fittest solutions from the population.

With the assumption that in later iterations of the algorithm, a proper part of the search space is found, our proposed approach utilizes EM operator only in first 50% of iterations. In this phase of execution, the EM operator is executed instead of exploitation process by using Eq. (11).

By incorporating EM into original WOA approach, hybridized WOA-EM is devised, which pseudo-code is summarized in the Algorithm 2.

4 Simulation results and analysis

Due to the research purpose and for the sake of more precise comparative analysis, we utilized the same simulation setup as in [2] and [14]. A two-dimensional ($2D$) WSN deployment area with a size of 100 $U \times 100$ U was used. Static target sensors and anchors with coordinates (x, y) are randomly deployed on the WSN deployment area by using pseudo-random number generator.

In the first set of experiments, simulations with 40 target nodes (M) and 8 anchor nodes (N) were performed, while in the second experiment instance, we utilized varying number of anchors (from 10 to 35) and target (from 25 to 150) nodes. In both experiments we have taken into account the additive Gaussian noise signal, which is given by $\hat{d}_i = d_i + n_i$. For more information, please refer to Eq. (2).

Algorithm 2 Pseudo-code of the WOA-EM

```
Initialization. Generate random initial population X_i(i = 1, 2, 3, ..., N) and initialize values of
control parameters.
Fitness calculation. Calculate fitness of each generated solution and determine the fittest solu-
tion X*
while (t < maxIter) do
  for each candidate solution do
    Recalculate A, C, a, l and p
    if p < 0.5 then
      if |A| < 1 then
        Recalculate current candidate solution X by using Eq. (7)
      else
        Choose random solution rnd form the population
        Update current candidate solution X by using Eq. (13)
      end if
    else
      if t < maxIter * 0.5 then
        Update current candidate solution by applying EM operator
      else
        Update current candidate solution X by using Eq. (11)
      end if
    end if
  end for
  If any solution goes beyond feasible region of the search space, modify it
  Evaluate all solutions in the population by calculating fitness
  Update position of the global best solution X* if necessary
  t = t + 1
end while
return The fittest (X*) from the current population
```

The size of population (N) and the maximum iteration number ($maxIter$) were set to 30 and 200, respectively for both algorithms, WOA and WOA-EM. The same parameter adjustments were used in [2] and [14]. The basic WOA parameters were adjusted as in [9]. Also, in both experiments, as performance indicators, we took the following metrics: the mean number of non-localized nodes (N_{NL}) and the mean localization error (E_L). Values of performance indicators were averaged over 30 independent runs.

In the first round of experiments, the goal was to measure the influence of the noise percentage (P_n) in distance measurement on the localization accuracy. For this purpose we ran original and hybridized WOA metaheuristics with the value of P_n set to 2 and 5, respectively. With each particular value of P_n we executed all algorithms in 30 independent runs.

Comparative analysis was performed between WOA-EM and original WOA, buttery optimization algorithm (BOA), firefly algorithm (FA), particle swarm optimization (PSO), elephant herding optimization (EHO), hybridized EHO (HEHO), TGA and dynamic TGA (dynsTGA). For this research we implemented WOA and WOA-EM, while the results for other approaches were taken form [2] and [14].

Simulation results of the proposed algorithm along with the results of the algorithms used for comparison are given in Table 1, where better results from each category are marked bold. Visualization of results for one run of WOA and WOA-EM, when $P_n = 5$ is given in Figure 1.

Table 1: Comparative analysis and simulation results for $M = 40$, $N = 8$ with different values for P_n averaged in 30 runs

Algorithms	$P_n = 5$			$P_n = 2$		
	Mean N_{NL}	**Mean E_L**	**Computing Time (s)**	**Mean N_{NL}**	**Mean E_L**	**Computing Time (s)**
BOA	4.7	0.28	0.65	4.5	0.21	0.53
FA	6.6	0.72	2.15	6.2	0.69	1.94
PSO	5.9	0.81	0.54	5.6	0.78	0.49
EHO	6.8	0.79	1.1	6.2	0.71	0.9
HEHO	5.3	0.45	1.2	5.1	0.37	1.0
TGA	5.5	0.42	0.9	5.0	0.36	0.8
dynsTGA	4.5	0.19	1.2	4.3	0.16	1.1
WOA	5.9	0.75	1.1	5.6	0.73	1.1
WOA-EM	**4.4**	**0.17**	1.3	4.3	**0.15**	1.2

From the results presented in Table 1, it can be noticed that in average WOA-EM obtains the best results. Only in the case when P_n is set to 2, for N_{NL} indicator, WOA-EM performs the same like the dynsTGA. At the other hand, improvements of WOA-EM over the original WOA are significant in all test instances. Original WOA obtains similar performance like PSO algorithm.

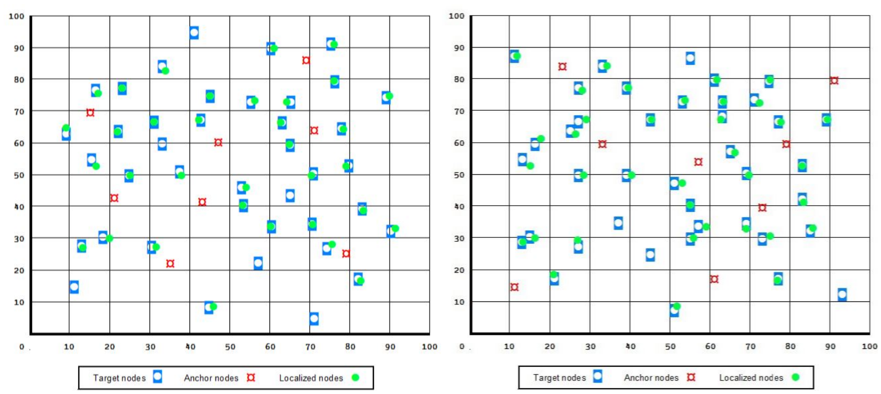

Fig. 1: Visualization of results when $Pn = 5$ for one run - WOA (left), WOA-EM (right)

Results from the second set of experiments, with the varying number of anchor and target nodes are given in Table 2.

Table 2: Comparative analysis between WOA-EM and WOA for varying number of target and anchor nodes averaged in 30 runs

Anchors	Targets	WOA		WOA-EM	
		Mean N_{NL}	**Mean** E_L	**Mean** N_{NL}	**Mean** E_L
25	10	5	0.73529	**1**	**0.19155**
50	50	3	0.55039	**2**	**0.22731**
75	20	3	0.69401	**2**	**0.18900**
100	25	0	0.64912	0	**0.17302**
125	30	3	0.61857	**1**	**0.28251**
150	35	1	0.71594	1	**0.49302**

Based on the results that are given in Table 2, it is obvious that the WOA-EM significantly improved performance of the basic WOA in terms of convergence, as well as of results' quality.

5 Conclusion and future work

The work that has been presented is aimed to improve solving localization problem in WSNs by utilizing WOA swarm approach. We have modified and improved the basic WOA and it was used for solving this problem.

The scientific contribution of this paper is twofold: improvements of the original WOA metaheuristics and advances in solving WSNs localization problem. Based on the comparison with other state-of-the-art approaches, that were implemented for the same WSNs localization problem, it can be said that it has proved the robusteness and effectiveness of our proposed WOA-EM approach.

As part of our future research activities, we will try to further improve WOA approach, and also to apply it to other WSNs localization problem modes that are current research topics.

Acknowledgment

The paper is supported by the Ministry of Education, Science and Technological Development of Republic of Serbia, Grant No. III-44006.

References

1. Ahmed, A., Ali, J., Raza, A., Abbas, G.: Wired vs wireless deployment support for wireless sensor networks. In: TENCON IEEE Region 10 Conference. pp. 1–3 (2006)
2. Arora, S., Singh, S.: Node localization in wireless sensor networks using butterfly optimization algorithm. Arabian Journal for Science and Engineering **42**(8), 3325–3335 (2017)
3. Goyal, S., Patterh, M.S.: Wireless sensor network localization based on cuckoo search algorithm. Wireless Personal Communications **79**, 223–234 (2014)
4. Hooke, R., Jeeves, T.A.: "Direct Search" solution of numerical and statistical problems. Journal of the ACM (JACM) **8**(2), 212–229 (1961)
5. Lavanya, D., Udgata, S.K.: Swarm intelligence based localization in wireless sensor networks. Springer **79**, 317–328 (2011)
6. Ling, Y., Zhou, Y., Luo, Q.: Lévy flight trajectory-based whale optimization algorithm for global optimization. IEEE access **5**, 6168–6186 (2017)
7. Liu, C., Liu, S., Zhang, W., Zhao, D.: The performance evaluation of hybrid localization algorithm in wireless sensor networks. Mob. Netw. Appl. **21**(6), 994–1001 (2016)
8. Mafarja, M.M., Mirjalili, S.: Hybrid whale optimization algorithm with simulated annealing for feature selection. Neurocomputing **260**, 302–312 (2017)
9. Mirjalili, S., Lewis, A.: The whale optimization algorithm. Advances in Engineering Software **95**, 51–67 (2016)

10. Strumberger, I., Tuba, E., Bacanin, N., Beko, M., Tuba, M.: Hybridized moth search algorithm for constrained optimization problems. In: International Young Engineers Forum (YEF-ECE). pp. 1–5 (2018)
11. Strumberger, I., Tuba, E., Bacanin, N., Beko, M., Tuba, M.: Monarch butterfly optimization algorithm for localization in wireless sensor networks. In: 28th IEEE International Conference Radioelektronika. pp. 1–6 (2018)
12. Strumberger, I., Bacanin, N., Tuba, M.: Hybridized elephant herding optimization algorithm for constrained optimization. In: Hybrid Intelligent Systems, Advances in Intelligent Systems and Computing. vol. 734, pp. 158–166. Springer, Cham (2018)
13. Strumberger, I., Beko, M., Tuba, M., Minovic, M., Bacanin, N.: Elephant herding optimization algorithm for wireless sensor network localization problem. In: Technological Innovation for Resilient Systems. pp. 175–184. Springer, Cham (2018)
14. Strumberger, I., Minovic, M., Tuba, M., Bacanin, N.: Performance of elephant herding optimization and tree growth algorithm adapted for node localization in wireless sensor networks. Sensors **19**(11), 2515 (2019)
15. Strumberger, I., Tuba, E., Bacanin, N., Beko, M., Tuba, M.: Modified and hybridized monarch butterfly algorithms for multi-objective optimization. In: Hybrid Intelligent Systems, Advances in Intelligent Systems and Computing. vol. 923, pp. 449–458. Springer, Cham (2020)
16. Strumberger, I., Tuba, E., Zivkovic, M., Bacanin, N., Beko, M., Tuba, M.: Dynamic search tree growth algorithm for global optimization. In: Technological Innovation for Industry and Service Systems, IFIP Advances in Information and Communication Technology book series. vol. 553, pp. 143–153. Springer, Cham (2019)
17. Strumberger, I., Tuba, M., Bacanin, N., Tuba, E.: Cloudlet scheduling by hybridized monarch butterfly optimization algorithm. Journal of Sensor and Actuator Networks **8**(3), 1–44 (2019)
18. Tuba, E., Strumberger, I., Zivkovic, D., Bacanin, N., Tuba, M.: Mobile robot path planning by improved brain storm optimization algorithm. In: IEEE Congress on Evolutionary Computation (CEC). pp. 1–8 (2018)
19. Tuba, E., Strumberger, I., Bacanin, N., Tuba, M.: Bare bones fireworks algorithm for capacitated p-median problem. In: Advances in Swarm Intelligence, LNCS. vol. 10941, pp. 283–291. Springer, Cham (2018)
20. Tuba, E., Tuba, M., Beko, M.: Support vector machine parameters optimization by enhanced fireworks algorithm. Advances in Swarm Intelligence, Lecture Notes in Computer Science **9712**, 526–534 (2016)
21. Tuba, M., Bacanin, N.: JPEG quantization tables selection by the firefly algorithm. In: International Conference on Multimedia Computing and Systems (ICMCS). pp. 153–158. IEEE (2014)
22. Tuba, M., Bacanin, N., Beko, M.: Multiobjective RFID network planning by artificial bee colony algorithm with genetic operators. In: Advances in Swarm and Computational Intelligence, LNCS. vol. 9140, pp. 247–254. Springer, Cham (2015)
23. Zivkovic, M., Branovic, B., Markovic, D., Popovic, R.: Energy efficient security architecture for wireless sensor networks. In: 20th Telecommunications Forum (TELFOR). pp. 1524–1527 (2012)