Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 4038 – 4054
© TÜBİTAK
doi:10.3906/elk-1902-189

On the performance of quick artificial bee colony algorithm for dynamic deployment of wireless sensor networks

Beyza GÖRKEMLİ[*], **Zahraa AL-DULAIMI**
Department of Computer Engineering, Faculty of Engineering, Erciyes University, Kayseri, Turkey

Received: 28.02.2019 • **Accepted/Published Online:** 21.08.2019 • **Final Version:** 26.11.2019

Abstract: In recent years, the use of wireless sensor networks (WSNs) has increased and there have been significant improvements in this field. Especially with smarter, cheaper, and smaller sensor nodes, various kinds of information can be detected and collected in different environments and under different conditions. WSNs have thus been used in many applications such as military, surveillance, target tracking, home, medical, and environmental applications. As the popularity of WSNs increases, problems related to these networks are being realized. The dynamic deployment problem is one of the main challenges that have a direct effect on the performance of WSNs. In this study, a novel optimization technique named the quick artificial bee colony (qABC) algorithm was applied to the dynamic deployment problem of WSNs. qABC is a new version of the artificial bee colony algorithm (ABC) and it redefines the onlooker bee phase of ABC in a more detailed way. In order to see the performance of qABC on this problem, WSNs that include only mobile sensors or both stationary and mobile sensors were considered with binary and probabilistic detection models. Some experimental studies were conducted for tuning the colony size (CS) and neighborhood radius (r) parameters of the qABC algorithm, and the performance of the proposed method was compared with the standard ABC algorithm and some other recently introduced approaches including a parallel ABC, a cooperative parallel ABC, a version of ABC powered by a transition control mechanism (tlABC), and a parallel version of tlABC. Additionally, some CPU time analyses were provided for qABC and ABC considering different dimensions of the problem. Simulation results show that the qABC algorithm is an effective method that can be used for the dynamic deployment problem of WSNs, and it generally improves the convergence performance of the standard ABC on this problem when $r \geq 1$.

Key words: Quick artificial bee colony algorithm, wireless sensor networks, dynamic deployment problem, probabilistic detection model, binary detection model

1. Introduction

Wireless sensor networks (WSNs) are important topics in network science and one of the main focuses of researchers is to improve their performance in order to make them more comfortable for different applications. This kind of networking has many advantages like easy installation and low cost. In recent years, WSNs have become more interesting worldwide. Especially with the growth of micro-electro-mechanical systems (MEMS) technology, it is easy to develop WSNs that contain autonomous smart sensor nodes that are inexpensive and small and have limited processing and computing resources [1, 2]. In addition, these sensor nodes have the ability of sensing, measuring, and collecting information from the environment; transmitting the sensed data to the user; and communicating with other sensors within a certain area [2].

[*]Correspondence: bgorkemli@erciyes.edu.tr

Sensor networks can be used to discover or monitor a variety of physical parameters or conditions, such as pressure, composition of the soil, humidity, voice, light, vehicular movement, temperature, and quality of the air or water [3]. Applications of WSNs can be classified as environmental, military, home, health, and other commercial fields. Due to the increase in the use and development of WSNs, many research fields emerged. One of the most important issues that directly affect the behavior and performance of WSNs is the deployment problem of the sensors. The major subject of the deployment of a sensor network is finding the sensor positions that are directly related to the coverage of the area of the interest [4]. Some sensor networks consist of many sensor nodes deployed very close to the phenomenon or inside it, and the positions of these sensor nodes do not need to be engineered or predetermined [5]. In inaccessible terrain or disaster relief operations, random deployment can be used with the sensor network protocols and algorithms that have self-organizing capabilities [5]. Dynamic deployment problems occur in such cases. Generally, every sensor knows its position by using some technologies like Global Positioning System (GPS), and by communicating with their neighbor sensors, they can also learn other sensors' positions. Since there is no previous information about the interest area, sensors are randomly scattered at the beginning in dynamic deployment problem. Then, if the type of the sensors is stationary, they cannot change their initial positions. However, if the network includes mobile sensors, which have the ability to change their positions, an optimization process can be carried out to achieve the maximum total coverage.

Optimization of the total covered area is one of the main issues that directly affect the performance of WSNs in dynamic deployment, and several evolutionary computation-based algorithms have been used to solve this problem in the literature [4, 6–21]. The artificial bee colony (ABC) algorithm is also one of these algorithms. In some studies, a binary detection model is used for the sensors in the network while in some of them a probabilistic detection model is applied. Also, some studies consider only mobile sensors in the network, while some of them use mobile and stationary sensors together. Ozturk et al. applied the ABC algorithm to the problem of dynamic deployment of WSNs consist of only mobile sensors considering a binary detection model [4]. They also used ABC with a probabilistic detection model for a WSN that includes both mobile and stationary sensors [6]. Aslan et al. examined the performance of a parallelized implementation of ABC on solving the dynamic deployment problem considering a probabilistic detection model and a network that includes only mobile sensors [22]. Yadav et al. proposed a modified version of ABC by addressing two drawbacks of the standard ABC algorithm using a crossover operator and a hybrid local search, and they tested its performance on a dynamic deployment problem with mobile sensors and considered a probabilistic detection model [23]. Aslan applied a parallel ABC and a cooperative parallel ABC to this problem by using a binary detection model and a WSN consisting of only mobile sensors [24]. He also proposed a new modified ABC by using a transition control mechanism and used it as a serial version and a parallel version for solving the dynamic deployment problem of WSNs that include mobile sensors with a binary detection model [25]. Yu et al. proposed a sensor deployment algorithm that is based on a modified ABC in which the updating equation of the onlooker bees and scout bees of the original ABC is changed, and 100 mobile sensors are used in the WSN for an ideal room [26]. They also carried out some experiments for a room with obstacles. He and Jiang used an improved version of the ABC algorithm for WSNs' dynamic deployment problem [10]. In their study, the standard ABC is improved by introducing a distance factor, changing the scout bee's working mode and the limit for the scout. This literature review shows that many ABC-based algorithms are used for solving the dynamic deployment problem. However, metaheuristics like ABC do not guarantee the optimum solution; rather, they provide a near optimum solution in an acceptable time. Thus, researchers try to improve the performances of these methods by making some changes to them. In particular, the flexible and simple algorithmic structure of ABC allows them to develop better versions.

Although the standard ABC has been successfully applied to many problems including dynamic deployment, it has some drawbacks. One of them is about the performance of ABC in terms of local search ability [27]. Karaboga and Gorkemli introduced a new version of ABC by redefining the onlooker bee phase of the algorithm in a more detailed way in 2012 [28]. Since this new definition enhances the convergence performance of the algorithm for the local minimum, they called it the quick ABC (qABC) algorithm [28, 29]. It improves the local search ability of the standard ABC on numerical optimization problems [28, 29]. To the best of our knowledge, this method has not been used for solving WSNs' dynamic deployment problem. In this study, considering the good results of the ABC algorithm for this problem and the advantages of qABC over the standard ABC, the qABC algorithm was applied to the problem of dynamic deployment of WSNs to achieve better performance. Also, this study was carried out to see the performance of qABC, which is a novel optimization algorithm, on some challenging problems in a different field. Using various scenarios with different detection models and network structures (according to sensor types), performance of the algorithm was examined in a detailed way. For each scenario, some tuning studies on colony size and neighborhood radius parameters of the qABC were carried out and the simulation results were compared with the results of the standard ABC algorithm. In order to evaluate the speed of the qABC and ABC algorithms on this problem, some CPU time analyses were done. Additionally, a comparison study was performed between the proposed method and some other approaches that have been recently applied to the dynamic deployment problem.

The remaining parts of the paper are organized as follows. The dynamic deployment problem of WSNs and the detection models are defined in Section 2. In Section 3, the qABC algorithm and the proposed approach are explained. Experimental studies and simulation results are provided in Section 4, and finally the study is concluded in Section 5.

2. Dynamic deployment problem

The performance of a sensor network depends on the sensors' positions. Sensors must be deployed by maximizing the quantity and quality of the information that they have the ability to get in the area of interest. After the first positioning of the sensors, there will not be any additional mobility in the network in the static version of the deployment problem. Optimal locations can be found as an optimization problem of the facility location by an offline scheme. Conversely, some sensors have the ability to move in the mission space in the dynamic version of the problem.

In the dynamic deployment problem, any prior information about the area of sensing is not found at the beginning. Therefore, initial positions of the sensors are chosen randomly. The maximum coverage ratio is sought by changing the positions of mobile sensors in the network. The coverage ratio (CR) can be formulated as in (1):

$$CR = \frac{\bigcup_{i \in S} cov_i}{A}, \tag{1}$$

where cov_i represents the coverage of i and A represents the total size of the area of the interest. S is the set of all sensors in the network.

In 2D environments, the detection range of each sensor can be represented with a circular structure such that the center of this circle is also the center of the related sensor, and the radius of the circle represents the sensor's detection range.

The binary detection model supposes that readings of a sensor have no associated unreliability. If a target is outside (inside) of a sensors' detection range, it is not detected (is detected) with complete certainty by this sensor [30]. The sensor field is considered as a 2D grid, every sensor has the same detection radius r_d, and sensor s is located at point $P_s(x_s, y_s)$. For any point $P(x, y)$, the Euclidean distance between $P(x, y)$ and s is denoted as $d(s, P(x, y))$. The basic principle of the binary detection model is given by (2):

$$c_{xy}(s) = \begin{cases} 1 & \text{if } d(s, P(x,y)) < r_d \\ 0 & otherwise, \end{cases} \tag{2}$$

where $c_{xy}(s)$ denotes the probability of being covered by sensor s for point $P(x, y)$. In the case of the Euclidean distance $d(s, P(x, y))$ is less than the detection radius and this indicates that point $P(x, y)$ is within the sensing range and covered by s. However, in the case of the distance being greater than the detection radius, it is not covered by sensor s.

While the binary model is concerned with only the range of detection, the probabilistic model also considers the detection uncertainty. For the probabilistic detection model, a new parameter r_e is introduced such that $r_e < r_d$, where r_d represents the detection range similar to the binary detection model and r_e represents the detection uncertainty range. According to this model, the probability of being covered by sensor s is given by (3) for point $P(x, y)$ [31]:

$$c_{xy}(s) = \begin{cases} 0 & \text{if } r_d + r_e \le d(s, P(x,y)) \\ e^{\left(\frac{-\lambda_1 \alpha_1^{\beta_1}}{\alpha_2^{\beta_2}} + \lambda_2\right)} & \text{if } r_d - r_e < d(s, P(x,y)) < r_d + r_e \\ 1 & \text{if } d(s, P(x,y)) \le r_d - r_e, \end{cases} \tag{3}$$

where β_1, β_2, and λ_1 are parameters to measure the detection probability; $\alpha_1 = r_e - r_d + d(s, P(x, y))$ and $\alpha_2 = r_e + r_d - d(s, P(x, y))$; and λ_2 denotes the disturbing effect.

In the probabilistic detection model, all points in the area of interest are covered with different probabilities by each sensor, and overlapping of the covered regions can be very important for recompensing the potential low detection probability of the points far from a sensor node. By using S as the set of all sensors in the network, (4) provides the probability of being covered for point $P(x, y)$ considering the effect of all sensors in S and their overlapping issues [32]:

$$c_{xy}(S) = 1 - \prod_{i \in S} (1 - c_{xy}(i)). \tag{4}$$

In order to calculate the CR, a desired coverage threshold c_{th} is used to decide the effectiveness of the coverage as shown in (5):

$$\begin{cases} \text{Accept the point } P(x,y) \text{ as covered} & \text{if } c_{xy}(S) \ge c_{th} \\ \text{Accept the point } P(x,y) \text{ as not covered} & \text{else.} \end{cases} \tag{5}$$

3. qABC algorithm and proposed approach

qABC is a novel version of the ABC algorithm [28, 29]. It describes onlooker bees' behavior more precisely and improves the standard ABC's local search ability [28, 29]. It was introduced in a lecture in 2012 [28]. In this algorithm, artificial bees are categorized into three different groups considering the colony's foraging behavior. The first group involves employed bees. These bees have a position of a food source in their mind when they leave the hive. After they return to the hive, they perform dances according to their food sources in the area of dancing. Some of the bees choose the food source regions by watching the employed bees' dances. This bee group is named onlookers. After selecting the region in the hive, they determine the food source to exploit when they achieve that region. The last bee group includes scout bees. A scout finds a new food source randomly and begins to consume it. She continues her mission as an employed bee. At the beginning, all employed bees work as scouts, and they start with the food sources found randomly. Through the optimization process, an employed bee can turn into a scout when her food source is abandoned. In the qABC algorithm, the position of a food source is related to a possible solution and the nectar amount or the quality of a food source is related to this solution's fitness value as in the ABC algorithm. The number of the employed bees is equal to the number of the food sources (SN), and the amount of the onlooker bees is equal to the amount of the employed bees. Food sources are randomly initialized by using (6):

$$x_{m,i} = l_i + \text{rand}(0,1) \times (u_i - l_i), \tag{6}$$

where $x_{m,i}$ represents the ith dimension value of the mth solution. l_i and u_i are the lower and upper bounds for the ith dimension, respectively. In the algorithm, employed bees determine a candidate food source using (7):

$$\upsilon_{m,i} = x_{m,i} + \phi_{m,i}(x_{m,i} - x_{k,i}), \tag{7}$$

where x_k represents a food source selected randomly from the population ($k \neq m$), i represents a dimension chosen randomly, and $\phi_{m,i}$ is a number that is also chosen randomly within the range $[-1,1]$. The solution's fitness $fit(x_m)$ is determined from its objective function value $f(x_m)$ by using (8):

$$fit(x_m) = \begin{cases} 1/(1+f(x_m)) & \text{if } f(x_m) \geq 0 \\ 1+abs(f(x_m)) & \text{if } f(x_m) < 0. \end{cases} \tag{8}$$

As mentioned before, the onlooker bees select their food sources differently from the employed bees in the qABC algorithm. An onlooker bee first watches the dances of the employed bees and she selects a food source region. When she reaches this region, she examines the food sources and selects the best (fittest) one to exploit. Onlookers choose the food source, which will be the center of their food source region, with a probability p_m and it is calculated by (9):

$$p_m = \frac{fit(x_m)}{\sum_{m=1}^{SN} fit(x_m)}. \tag{9}$$

Onlooker bees determine the candidate solution by (10) after selecting their central food source x_m for the region and finding the best one among all food sources in that region:

$$\upsilon_{N_m,i}^{best} = x_{N_m,i}^{best} + \phi_{m,i}(x_{N_m,i}^{best} - x_{k,i}), \tag{10}$$

where $x_{N_m}^{best}$ is the best solution among the neighbors of x_m and itself (N_m). In order to obtain the neighborhood of x_m, a mean Euclidean distance md_m is calculated between x_m and the rest of the solutions. $d(m,j)$

represents the Euclidean distance between x_m and x_j, and the mean Euclidean distance md_m of x_m is calculated by (11):

$$md_m = \frac{\sum_{j=1}^{SN} d(m,j)}{SN-1}. \tag{11}$$

For determining the neighbors of x_m, a definition similar to the one given in (12) [28, 29] is used:

$$\begin{cases} x_j \text{ is a neighbor of } x_m & \text{if } d(m,j) < r \times md_m \\ x_j \text{ is not a neighbor of } x_m & \text{else.} \end{cases} \tag{12}$$

In this definition, r is the neighborhood radius and it has to be used as $r \geq 0$. The neighborhood of x_m shrinks or enlarges as the r value decreases or increases, respectively. Thus, when $r = 0$, the qABC algorithm works as the standard ABC algorithm. The flowchart of the qABC algorithm is given in Figure 1.

In this study, the qABC was employed for the dynamic deployment problem of WSNs. This optimization algorithm was used to maximize the network's coverage rate that is given with (1). In this study, the objective function of a solution x_m for minimizing the uncovered area of the network is provided by (13):

$$f(x_m) = 1 - CR, \tag{13}$$

where CR is the coverage rate calculated by (1). Coverage of a WSN depends on the sensor detection model. The following assumptions are made, similarly to [6]:

- All sensors work with the same detection radius r_d,
- All sensors can communicate with other sensors,
- A WSN consists of only mobile sensors or both mobile and stationary sensors together,
- The mobile sensors are able to change their positions,
- The sensor field is represented by a two-dimensional (2D) grid,
- Each sensor realizes its location.

Figure 2 shows the basic steps for solving the dynamic deployment problem of WSNs simply.

Since the optimization process is carried out by changing the position of mobile sensors, a solution (food source) of the qABC is structured by using the solution string with $2 \times numberofmobilesensors$ items. Algorithm 1 is given to explain the qABC algorithm for the dynamic deployment of WSNs.

4. Results and discussion

In order to see the performance of the qABC on the dynamic deployment problem of WSNs, first a series of experiments were carried out considering different scenarios. With these experiments, after the tuning studies of the parameter colony size, the performance of the standard ABC algorithm and different r valued qABCs were compared. Then CPU time analyses were provided for ABC and qABC on this problem. Lastly, the results of the proposed method were compared with the results of some recent approaches given in the literature.

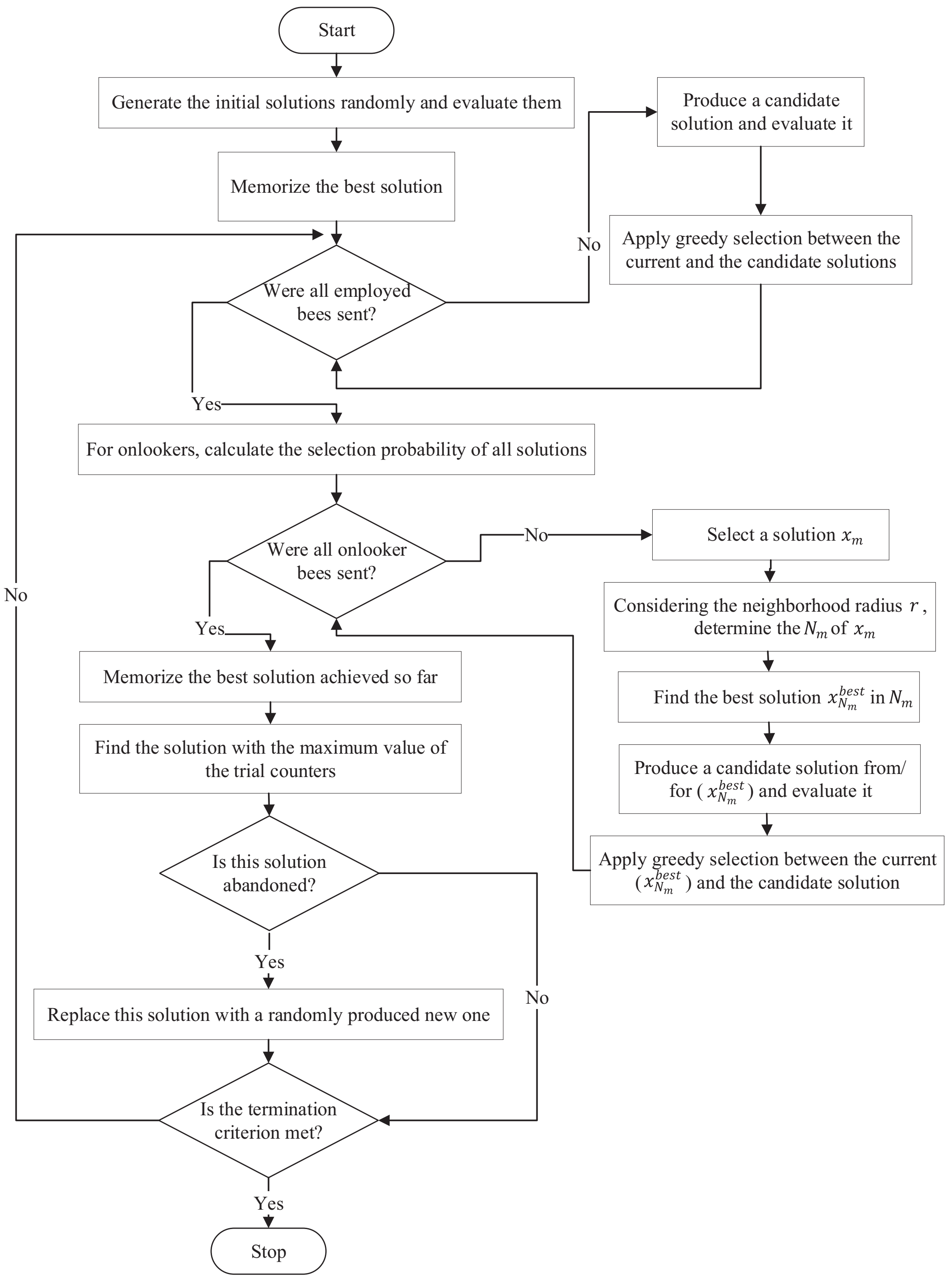
Start
Generate the initial solutions randomly and evaluate them
Memorize the best solution
Were all employed bees sent?
No
Produce a candidate solution and evaluate it
Apply greedy selection between the current and the candidate solutions
Yes
For onlookers, calculate the selection probability of all solutions
Were all onlooker bees sent?
No
Select a solution x_m
Considering the neighborhood radius r, determine the N_m of x_m
Find the best solution $x_{N_m}^{best}$ in N_m
Produce a candidate solution from/for ($x_{N_m}^{best}$) and evaluate it
Apply greedy selection between the current ($x_{N_m}^{best}$) and the candidate solution
Yes
Memorize the best solution achieved so far
Find the solution with the maximum value of the trial counters
Is this solution abandoned?
Yes
No
Replace this solution with a randomly produced new one
Is the termination criterion met?
No
Yes
Stop

Figure 1. Flowchart of the qABC algorithm.

4.1. Experimental settings

For the simulation studies, the same problem settings were used as in [6]. Accordingly, all of the sensors have the same features and the detection range r_d is 7 m for each sensor. The total area of interest A is 10000

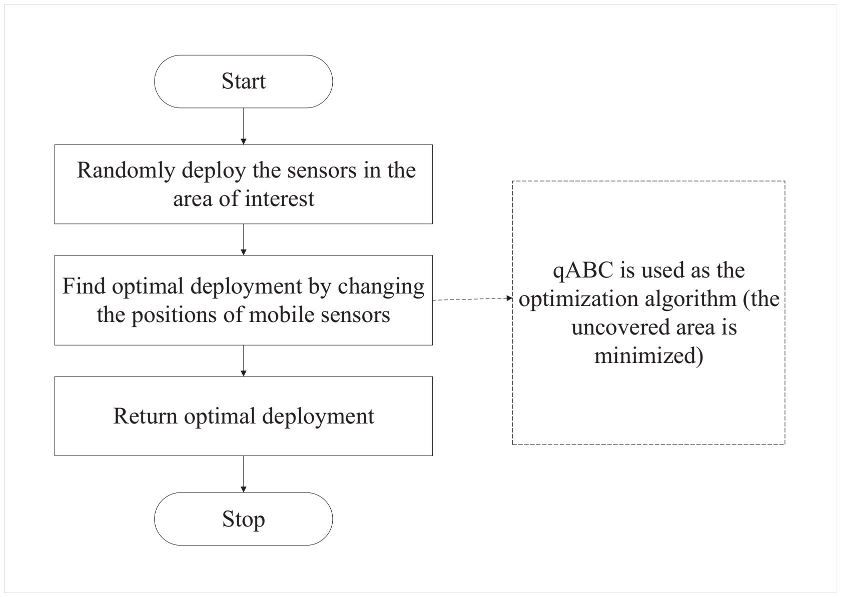
Start
Randomly deploy the sensors in the area of interest
Find optimal deployment by changing the positions of mobile sensors
Return optimal deployment
Stop
qABC is used as the optimization algorithm (the uncovered area is minimized)

Figure 2. The basic steps for solving the dynamic deployment problem of WSNs.

m^2 with 100 m length and 100 m width. When determining the coverage of a point $P(x,y)$, these points were picked with 1-m fixed intervals starting from the point at $(0,0)$ in the region of 100 m $\times$ 100 m. Thus, there are 10000 (100×100) points to cover in the area of interest. In this study, both binary and probabilistic detection models were used. Parameters of the probabilistic detection model were set as $\lambda_1 = 1$, $\lambda_2 = 0$, $\beta_1 = 1$, $\beta_2 = 0.5$, $c_{th} = 0.1$, and $r_e = 3.5\ m$. For each detection model two different cases were considered. While the dynamic deployment of a WSN only consisting of mobile sensors was optimized in the first case (100 mobile sensors), in the second case, the network consisted of both mobile and stationary sensors together (20 mobile and 80 stationary sensors). The performance of the qABC algorithm was thus analyzed for four different scenarios. It should be demonstrated that optimization of this problem has different kinds of difficulties for each scenario. When a WSN contains only mobile sensors, since the total number of sensors is 100 for all scenarios and the solution string contains the x and y coordinates of all sensor nodes, the dimension (D) of the problem is 200, while in the case of stationary and mobile sensors it is 40 with 20 mobile sensors. In the scenarios that have stationary and mobile sensors together, since the stationary ones cannot change their positions and are randomly deployed in the area of the interest, stationary sensors' deployment also becomes a part of the problem, and the optimization algorithm tries to achieve maximum coverage by only changing the positions of mobile sensors. Therefore, in this case, the area in which the algorithm carries out the optimization process has a more confusing structure than the one in the other case. The minimum coverage rate obtained by using only stationary sensors was 0.65 for the WSNs that include mobile and stationary sensors together.

For a fair comparison, the maximum number of fitness evaluations was used as the termination criterion. For all scenarios, first some experimental studies were carried out for determining the colony size ($r = 1$ [29]). After determining the colony size, some experiments were conducted for the neighborhood radius (r) parameter of qABC algorithm. By trying different neighborhood radii, performances of different r valued qABC algorithms were compared. When $r = 0$, the qABC works as the standard ABC. These experimental studies allow to compare the performance of the qABC with the standard ABC algorithm that has been successfully used to solve the dynamic deployment problem of WSNs in previous studies. Table 1 shows the parameter settings.

Algorithm 1 The qABC algorithm for the dynamic deployment of WSNs.

```
1: Initialization phase:
2: Initialize the value of the parameters:
3: Problem settings: detection radius $r_d$, size of the interest area $A$, number of mobile sensors $ms$, number
   of stationary sensors $ss$.
4: Control parameters of qABC: colony size $CS$, limit $l$ for the scout, neighborhood radius $r$, maximum
   number of cycles $MaxNum$.
5: Determine the initial positions of $ms$ mobile sensors randomly for each solution (food source) $x_m$ using
   (6). Set the value of the trial counter as $0$ for each solution.
6: Evaluate the solutions using (13) and (8).
7: Memorize the best solution.
8: $c = 0$.
9: repeat
10:   Employed bee phase:
11:   for all employed bees do
12:     Produce a candidate solution $\upsilon_m$ from $x_m$ using (7) and evaluate it using (13) and (8).
13:     Apply a greedy selection process between $x_m$ and $\upsilon_m$.
14:     If $\upsilon_m$ is selected, then set the trial counter of this solution as 0; otherwise, increase the value of $x_m$'s
        trial counter.
15:   end for
16:   Calculate the selection probability values $p_m$ of the solutions for the onlooker bees using (9).
17:   Onlooker bee phase:
18:   for all onlooker bees do
19:     Select a solution $x_m$ depending on $p_m$ values.
20:     Determine the $N_m$ of $x_m$.
21:     Find the best solution $x_{N_m,i}^{best}$ in $N_m$.
22:     Produce a candidate solution $\upsilon_{N_m,i}^{best}$ from $x_{N_m,i}^{best}$ using (10) and evaluate it using (13) and (8).
23:     Apply a greedy selection process between $x_{N_m,i}^{best}$ and $\upsilon_{N_m,i}^{best}$.
24:     If $\upsilon_{N_m,i}^{best}$ is selected, then set the trial counter of this solution as $0$; otherwise, increase the value of
        $x_{N_m,i}^{best}$'s trial counter.
25:   end for
26:   Memorize the best solution achieved so far.
27:   Scout bee phase:
28:   Find the solution that has the maximum trial counter value in the population. If this value is higher
      than $l$, replace it with a new solution produced by (6) and set the trial counter of this new solution as
      $0$. Then evaluate it using (13) and (8).
29:   $c = c + 1$.
30: until ($c = MaxNum$)
```

Table 1. The parameter settings.

Parameter	Value
Number of evaluations	200000
CS	$10, 20, 40, 80$
Number of runs	30
r	$0, 0.5, 1, 2, 3, \infty$

The value of the limit parameter (l) for the scout bee is calculated by using (14) [29, 33]:

$$l = \frac{CS \times D}{2}, \tag{14}$$

where CS is the colony size and D is the dimension of the problem. Since the aim is maximizing the total covered area in this problem, the objective function, which gives the total uncovered area, was minimized through the optimization process. This objective function is given by (13).

In this study, the algorithms were implemented using C Sharp programming language and the .net framework was used.

4.2. Simulation results and discussion

For each scenario, simulation results were collected and analyzed in tables that show the mean, standard deviation (Std Dev), best, and worst of the objective function values over 30 independent runs. Tables 2–4 show the results of CS while Tables 5–7 provide the results of r experiments for the first, second, and fourth scenario, respectively. In the third scenario, which considers the probabilistic detection model with the WSN that contains only mobile sensors, the qABC algorithm found the optimum solution for every run with all considered values of the CS and r parameters at the end of 200000 function evaluations. Therefore, convergence graphs are used to analyze the effect of these parameters. Figure 4 illustrates the mean of the best objective function values found in each run through the evaluations for different CS values, and similarly Figure **??** gives the convergence graphs for different r values for the third scenario. Since the main advantage of the qABC model is demonstrated as improving the local search ability and the convergence speed of the standard ABC in the literature [28, 29], convergence graphs of the different r valued qABC algorithms are also presented for the first, second, and fourth scenarios in Figures 5–7, respectively.

Table 2. Effect of CS on the performance of the qABC for the WSN that contains only mobile sensors with the binary detection model.

CS	Mean	Std Dev	Best	Worst
10	0.00158	0.00051	0.00060	0.00290
20	0.00109	0.00035	0.00030	0.00170
40	0.00090	0.00038	0.00030	0.00180
80	0.00105	0.00045	0.00040	0.00230

Table 3. Effect of CS on the performance of the qABC for the WSN that contains stationary and mobile sensors with the binary detection model.

CS	Mean	Std Dev	Best	Worst
10	0.09589	0.01071	0.07540	0.11770
20	0.08518	0.00081	0.08410	0.08740
40	0.08464	0.00087	0.08260	0.08670
80	0.08401	0.00071	0.08260	0.08590

Table 4. Effect of CS on the performance of the qABC for the WSN that contains stationary and mobile sensors with the probabilistic detection model.

CS	Mean	Std Dev	Best	Worst
10	0.02786	0.00782	0.01370	0.04250
20	0.04294	0.00099	0.04130	0.04530
40	0.04238	0.00121	0.03950	0.04470
80	0.04175	0.00119	0.03830	0.04490

For the first scenario, all CS values except 10 give similar results and the better objective function values are achieved with 40. Also, when $r \geq 1$, better results are obtained and the best one was achieved with the value of 3 for mean, best, and worst fields. The highest mean value was found when $r = 0$. By comparison with the performance of the standard ABC algorithm ($r = 0$), it can be said that qABC gives better results for this scenario in the meaning of both the final best result and general convergence performance, especially when $r \geq 1$.

Table 5. Effect of r on the performance of the qABC for the WSN that contains only mobile sensors with the binary detection model.

r	Mean	Std Dev	Best	Worst
0	0.00163	0.00048	0.00060	0.00240
0.5	0.00161	0.00052	0.00070	0.00280
1	0.00090	0.00038	0.00030	0.00180
2	0.00094	0.00035	0.00030	0.00170
3	0.00079	0.00037	0	0.00160
∞	0.00083	0.00031	0.00040	0.00160

Table 6. Effect of r on the performance of the qABC for the WSN that contains stationary and mobile sensors with the binary detection model.

r	Mean	Std Dev	Best	Worst
0	0.08474	0.00077	0.08360	0.08640
0.5	0.08454	0.00069	0.08330	0.08610
1	0.08401	0.00071	0.08260	0.08590
2	0.08408	0.00081	0.08270	0.08600
3	0.08375	0.00065	0.08230	0.08500
∞	0.08391	0.00081	0.08250	0.08580

Table 7. Effect of r on the performance of the qABC for the WSN that contains stationary and mobile sensors with the probabilistic detection model.

r	Mean	Std Dev	Best	Worst
0	0.04457	0.00119	0.04210	0.04640
0.5	0.04390	0.00138	0.04010	0.04680
1	0.02786	0.00782	0.01370	0.04250
2	0.04465	0.00092	0.04310	0.04680
3	0.04425	0.00130	0.04050	0.04720
∞	0.04464	0.00105	0.04270	0.04700

While the value of CS increases, the qABC generally finds better solutions for the second scenario, and the best mean, standard deviation, and worst results were achieved when $CS = 80$. In this scenario, there are small differences between the results, and the algorithm's performance is slightly better when $r \geq 1$ in terms of final objective function values (after 200000 evaluations). However, it is clear that, when $r = 3$, it gives the best results. When the general convergence performances are examined, this time a remarkable difference can be seen between the performances of the qABC with $r \geq 1$ and $r < 1$. Especially in early evaluations, the algorithm provides much faster convergence with $r \geq 1$. The results also point out that, by tuning the value of the neighborhood radius to be greater than or equal to 1, the qABC algorithm achieved a performance superior to the performance of the standard ABC for the dynamic deployment of the WSN that contains both stationary and mobile sensor nodes according to the binary detection model.

In the third scenario, the qABC algorithm found the optimum solution for every run with all considered values of CS and r. However, the general convergence performance seems better when $CS = 10$, and the worst convergence performance was shown when $CS = 80$. Also, the convergence graphs show that the qABC outperforms the standard ABC in this scenario when $r \geq 1$.

In the fourth scenario, similar performances are shown when CS equals 20, 40, and 80, and by considering only these values it can be said that, as the colony size increases, the mean and best values are improved slightly. However, when $CS = 10$, the algorithm gives the best results for the mean, best, and worst values. Values of the mean and best fields decrease while the value of the r parameter increases until 2. When $r \geq 2$, the algorithm gives worse results than when $r = 1$. Actually, in terms of the mean, best, and worst values, the best results are obtained when $r = 1$ and the other values of r cause worse performances. However, the qABC produces

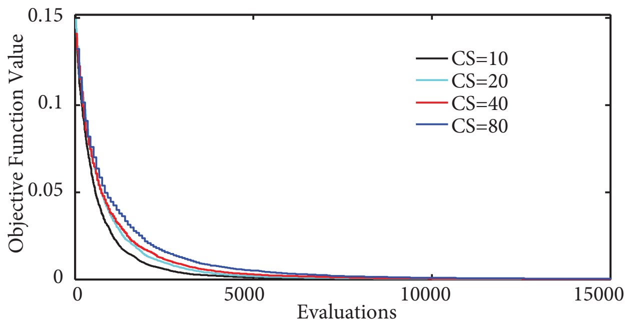

Figure 3. qABC algorithm's convergence performance for different CS values for the WSN that contains only mobile sensors with the probabilistic detection model.

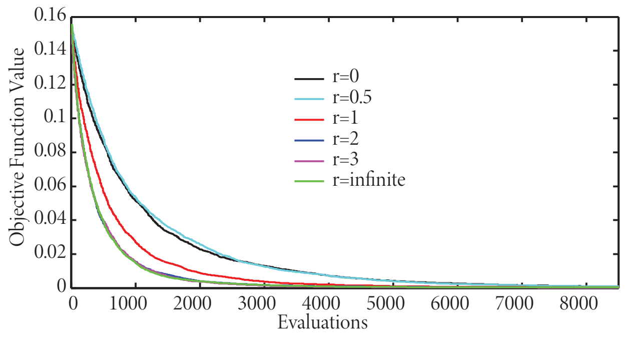

Figure 4. qABC algorithm's convergence performance for different r values for the WSN that contains only mobile sensors with the probabilistic detection model.

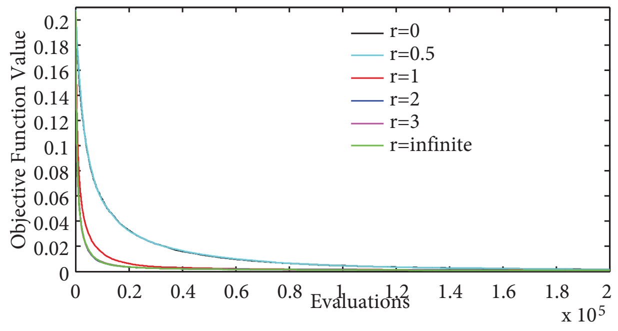

Figure 5. qABC algorithm's convergence performance for different r values for the WSN that contains only mobile sensors with the binary detection model.

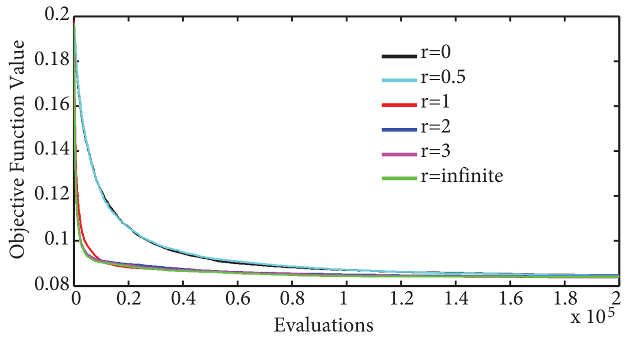

Figure 6. qABC algorithm's convergence performance for different r values for the WSN that contains stationary and mobile sensors with the binary detection model.

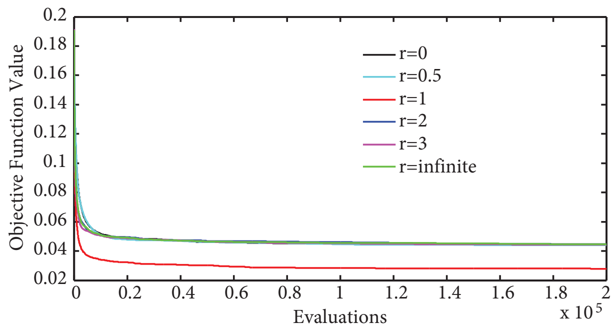

Figure 7. qABC algorithm's convergence performance for different r values for the WSN that contains stationary and mobile sensors with the probabilistic detection model.

similar results for all r values except 1. In this scenario, the qABC gives lower mean objective function values than the standard ABC algorithm if parameter r is equal to 1, 0.5, or 3. On the other hand, the standard ABC shows slightly better performance than the qABC when the value of parameter r equals 2 or ∞.

4.3. CPU times of the ABC and qABC algorithms on the dynamic deployment problem

In this section, some experiments were conducted to compare the CPU times of the ABC and qABC algorithms on the dynamic deployment problem of WSNs. In this comparison, the binary detection model is used, and all sensor nodes are mobile. CS is 40, the total number of function evaluations is 2000, and for the qABC, r is 1. In order to assess the CPU times of the algorithms considering different dimensions, they were executed with different numbers of mobile sensors (25, 50, and 100 mobile sensors). Other parameters were set as in the previous experiments. The means and standard deviations of 30 independent runs are presented in Table 8. For these experiments, the algorithms were implemented by using C Sharp programming language and .net framework 4.6.1 was used, and the runs were performed on Windows 10 Education on an Intel Core M-5Y51 1.20 GHz processor with 8 GB RAM. The qABC algorithm needs more time than the ABC since it has additional operations in the onlooker bee phase. The results in Table 8 show that, although the CPU times are increased when the mobile sensor number is increased for both algorithms, the increment rates for the qABC are lower than the ones for the ABC.

Table 8. CPU times of the ABC and qABC for the dynamic deployment problem with different numbers of mobile sensors.

ms	ABC		qABC	
	Mean	Std Dev	Mean	Std Dev
25	10.71558	1.10840	13.78004	0.20462
50	11.86260	1.43462	14.92413	0.22802
100	13.42507	1.62656	15.79890	0.20315

4.4. Additional study

Performance of the qABC was compared with some other optimization methods including the ABC, a parallel ABC (pABC) [24], a cooperative parallel ABC (coop-pABC) [24], a version of the ABC powered by a transition control mechanism (tlABC) [25], and a parallel version of the tlABC (p-tlABC) [25] on the dynamic deployment problem of WSNs. Details of these algorithms can be found in [24, 25]. Since the results of the ABC, pABC, and coop-pABC algorithms were taken from [24] and the results of the tlABC and p-tlABC algorithms were taken from [25], similar parameter settings and scenarios were used in this comparison study. The binary detection model is considered with 100 mobile sensors, r_d is 7 m, and the area of interest is 10000 m^2 with 100 m length and 100 m width. $CS = 20$ and the total number of fitness evaluations is 1000, 2000, and 10000. For the qABC, the value of the parameter limit was calculated by (14) as in the previous experiments and $r = 1$. In addition, the other algorithms used in the comparison were applied to this problem by considering the position information of a sensor (both x and y coordinates) as the value in one dimension while the qABC handle the coordinate information in different dimensions. As in [24, 25], 20 independent runs were executed, and the means and standard deviations of them are given in Table 9.

According to the table, the performance of the qABC is better than the performance of all the other algorithms for all considered numbers of evaluations. Since the qABC strengthens the convergence performance of the ABC algorithm, the difference between the mean values of the qABC and the algorithm that performs best after the qABC is the highest for 1000 evaluations and the smallest for 10000 evaluations.

5. Conclusion

In this study, a novel artificial intelligence optimization technique, the qABC algorithm, was applied to the dynamic deployment problem of WSNs. First, the proposed approach was explained in a detailed way and then its performance was examined by using four different scenarios. In the first scenario, the deployment

Table 9. Performance comparison.

Algorithm	1000 evaluations		2000 evaluations		10000 evaluations	
	Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
ABC [24]	0.88257	0.00410	0.91207	0.00638	0.96755	0.00226
pABC [24]	0.87507	0.00594	0.90887	0.00483	0.96904	0.00372
coop-pABC [24]	0.87970	0.00457	0.91530	0.00362	0.97063	0.00553
tlABC [25]	-	-	0.92330	0.00574	0.96932	0.00280
p-tlABC [25]	-	-	0.91632	0.00553	0.95702	0.00694
qABC	0.91154	0.01538	0.94556	0.01078	0.98792	0.00309

of a WSN that contains only mobile sensors was optimized considering the binary detection model. Also, in the second scenario the binary detection model was used. However, this time the WSN consisted of both stationary and mobile sensor nodes. Based on studies about the dynamic deployment problem that calculate the effectively covered area with the probabilistic detection model to get more realistic results, we used a probabilistic detection model, too. The qABC was tested on a WSN that included only mobile sensors in the third scenario and both stationary and mobile sensors in the fourth scenario with the probabilistic detection model. For each scenario, the algorithm was tested using different values of CS to see the effect of this control parameter on the performance of the qABC. The most appropriate CS value was determined and was used for the experiments of r. By trying different neighborhood radii, the performances of different r valued qABC and standard ABC algorithms were compared. Also, some CPU time analyses were provided for both qABC and ABC algorithms considering different dimensions of the problem. In addition, a comparison study was carried out between the proposed approach and some other recent ones given in the literature. According to the simulation results, some concluding remarks can be demonstrated as follows:

- In the first scenario, better results are obtained when $r \geq 1$ and the best mean value was achieved when $r = 3$ by the qABC. The worst performance was shown by the ABC algorithm. Compared with the ABC, the qABC gives better results in terms of not only the final best result but also the general convergence performance, especially when $r \geq 1$.
- In the second scenario, qABC's performance is slightly better when $r \geq 1$ in terms of the final objective function values. It finds the best results with $r = 3$. When the performances are evaluated considering the general convergence to the optimum through the evaluations, a remarkable difference can be seen between the qABC with $r \geq 1$ and $r < 1$. The qABC (with $r \geq 1$) provides much faster convergence than the ABC, especially in early evaluations.
- In the third scenario, all of the algorithms achieved optimum results in every independent run. When the convergence graphs are examined, it can be seen that qABC outperforms the standard ABC when $r \geq 1$.
- In the fourth scenario, the best results were found by the qABC algorithm when $r = 1$ and the other considered values of r gave similar results. The performance of the qABC is better than the ABC's performance if the value of parameter r is equal to 1, 0.5, or 3.
- Since the qABC algorithm has some additional operations in the onlooker bee phase, it needs more CPU time compared to the ABC. For both algorithms, the time spent is increased when the dimension of the problem is increased. However, the increment rates for the qABC are lower than those for the ABC.

- The comparison between the proposed approach and the other optimization algorithms including the ABC, pABC, coop-pABC, tlABC, and p-tlABC shows that, for every considered number of evaluations, the qABC performs best. Also, the results demonstrate the powerful effect of the qABC on the convergence performance of the ABC since all the other algorithms used in the comparison are based on ABC, too.

Consequently, it can be said that the qABC algorithm can be used to solve the dynamic deployment problem of WSNs effectively. With this study, it is also observed that the new definition of the onlooker bee phase that is provided in the qABC generally improves the performance of the standard ABC on not only the literature benchmarks but also the dynamic deployment problem of WSNs when $r \geq 1$.

As future work, first, it is suggested to compare the performance of the qABC algorithm on the dynamic deployment problem with a wider set of well-known optimization techniques. Secondly, the algorithm can be applied to different deployment scenarios with different assumptions (there could be some obstacles in the area of interest or different detection radius values can be used for each sensor). Additionally, an experimental design study can be carried out for the control parameters of the qABC algorithm such as the limit, maximum number of evaluations, colony size, and neighborhood radius.

References

[1] Patra RR, Patra PK. Analysis of k-coverage in wireless sensor networks. International Journal of Advanced Computer Science and Applications 2011; 2 (9): 91-96.

[2] Yick J, Mukherjee B, Ghosal D. Wireless sensor network survey. Computer Network 2008; 52 (12): 2292-2330.

[3] Sohraby K, Minoli D, Znati T. Wireless Sensor Networks: Technology, Protocols, and Applications. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2007.

[4] Öztürk C, Karaboğa D, Görkemli B. Artificial bee colony algorithm for dynamic deployment of wireless sensor networks. Turkish Journal of Electrical Engineering and Computer Science 2012; 20 (2): 255-262.

[5] Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a survey. Computer Network 2002; 38: 393-422.

[6] Ozturk C, Karaboga D, Gorkemli B. Probabilistic dynamic deployment of wireless sensor networks by artificial bee colony algorithm. Sensors 2011; 11 (6): 6056-6065.

[7] Wang X, Wang S, Bi D. Virtual force-directed particle swarm optimization for dynamic deployment in wireless sensor networks. In: Huang DS, Heutte L, Loog M (editors). Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues. ICIC 2007. Lecture Notes in Computer Science, Vol. 4681. Berlin, Germany: Springer, 2007, pp. 292-303.

[8] Wang X, Wang S, Ma J. An improved co-evolutionary particle swarm optimization for wireless sensor networks with dynamic deployment. Sensors 2007; 7 (3): 354-370.

[9] Wang G, Guo L, Duan H, Liu L, Wang H. Dynamic deployment of wireless sensor networks by biogeography based optimization algorithm. Journal of Sensor Actuator Networks 2012; 1 (2): 86-96.

[10] He P, Jiang M. Dynamic deployment of wireless sensor networks by an improved artificial bee colony algorithm. Application of Mechanical Materials 2014; 511-512: 862-866.

[11] Uppal RS, Kumar S. Big bang-big crunch algorithm for dynamic deployment of wireless sensor network. International Journal of Electrical and Computer Engineering 2016; 6 (2): 596-601.

[12] Kukunuru N, Thella BR, Davuluri RL. Sensor deployment using particle swarm optimization. International Journal of Engineering Science and Technology 2010; 2 (10): 5395-5401.

[13] Tuba E, Tuba M, Simian D. Wireless sensor network coverage problem using modified fireworks algorithm. In: 2016 International Wireless Communications and Mobile Computing Conference (IWCMC); Paphos, Cyprus; 2016. pp. 696-701.

[14] Liu XL, Zhang XS, Zhu QX. Enhanced fireworks algorithm for dynamic deployment of wireless sensor networks. In: 2017 2nd International Conference on Frontiers of Sensors Technologies (ICFST); Shenzhen, China; 2017. pp. 161-165.

[15] Tuba E, Tuba M, Beko M. Mobile wireless sensor networks coverage maximization by firefly algorithm. In: 2017 International Conference on Radioelektronika; Brno, Czech Republic; 2017. pp. 182-186.

[16] Wang L, Wu WH, Qi JY, Jia ZP. Wireless sensor network coverage optimization based on whale group algorithm. Computer Science and Information Systems 2018; 15 (3): 569-583.

[17] Alia OM, Al-Ajouri A. Maximizing wireless sensor network coverage with minimum cost using harmony search algorithm. IEEE Sensors Journal 2017; 17 (3): 882-896.

[18] Özdağ R, Karcı A. Probabilistic dynamic distribution of wireless sensor networks with improved distribution method based on electromagnetism-like algorithm. Measurement 2016; 79: 66-76.

[19] Shan W, Chen X. Improved invasive weed optimization algorithm in sensor deployment for wireless sensor networks. Boletín Técnico 2017; 55 (9): 310-316.

[20] Binh HTT, Hanh NT, Quan LV, Dey N. Improved cuckoo search and chaotic flower pollination optimization algorithm for maximizing area coverage in wireless sensor networks. Neural Computing & Applications 2018; 30 (7): 2305-2317.

[21] Farsi M, Elhosseini MA, Badawy M, Ali HA, Eldin HZ. Deployment techniques in wireless sensor networks, coverage and connectivity: a survey. IEEE Access 2019; 7: 28940-28954.

[22] Aslan S, Aksoy A, Gunay M. Performance of parallel artificial bee colony algorithm on solving probabilistic sensor deployment problem. In: 2018 International Conference on Artificial Intelligence and Data Processing; Malatya, Turkey; 2018. pp. 1-21.

[23] Yadav RK, Gupta D, Lobiyal DK. Dynamic positioning of mobile sensors using modified artificial bee colony algorithm in a wireless sensor networks. International Journal of Control Theory and Applications 2017; 10 (18): 167-176.

[24] Aslan S. Deployment in wireless sensor networks by parallel and cooperative parallel artificial bee colony algorithms. International Journal of Optimization and Control: Theories & Applications 2019; 9 (1): 1-10.

[25] Aslan S. A transition control mechanism for artificial bee colony (ABC) algorithm. Computational Intelligence and Neuroscience 2019; 2019: 1-24.

[26] Yu X, Zhang J, Fan J, Zhang T. A faster convergence artificial bee colony algorithm in sensor deployment for wireless sensor networks. International Journal of Distributed Sensor Networks 2013; 2013: 1-9.

[27] Karaboga D, Gorkemli B, Ozturk C, Karaboga N. A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artificial Intelligence Review 2014; 42 (1): 21-57.

[28] Karaboga D, Gorkemli B. A quick artificial bee colony -qABC- algorithm for optimization problems. In: 2012 International Symposium on Innovations in Intelligent Systems and Applications; Trabzon, Turkey; 2012. pp. 1-20.

[29] Karaboga D, Gorkemli B. A quick artificial bee colony (qABC) algorithm and its performance on optimization problems. Applied Soft Computing 2014; 23: 227-238.

[30] Mohamed SM, Hamza HS, Saroit IA. Coverage in mobile wireless sensor networks (M-WSN): a survey. Computer Communications 2017; 110: 133-150.

[31] Li SJ, Xu CF, Pan WK, Pan YH. Sensor deployment optimization for detecting maneuvering targets. In: 7th International Conference on Information Fusion; Philadelphia, PA, USA; 2005. pp. 1629-1635.

[32] Zou Y, Chakrabarty K. Sensor deployment and target localization based on virtual forces. In: 22th Annual Joint Conference of the IEEE Computer and Communications Societies; San Francisco, CA, USA; 2003. pp. 1293-1303.

[33] Karaboga D, Akay B. A comparative study of artificial bee colony algorithm. Applied Mathematics and Computation 2009; 214 (1): 108-132.