Journal of Information Hiding and Multimedia Signal Processing ©2015 ISSN 2073-4212
Ubiquitous International Volume **6**, Number **2**, March **2015**

A Compact Articial Bee Colony Optimization for Topology Control Scheme in Wireless Sensor Networks

Thi-Kien Dao, Tien-Szu Pan* and Trong-The Nguyen

Department of Electronic Engineering
National Kaohsiung University of Applied Sciences
415 Chien-Kung Road, Kaohsiung, 807, Taiwan
*Corresponding author
jvnkien@gmail.com; tpan@cc.kuas.edu.tw; vnthe@hpu.edu.vn

Shu-Chuan Chu

School of Computer Science, Engineering and Mathematics
Flinders University
Australia
jan.chu@infoeng.flinders.edu.au

Received July, 2014; revised November, 2014

Abstract. *In this paper, a compact Articial Bee Colony optimization method (cABC) for applying to the topology optimization of wireless sensor networks (WSNs) is presented. The purpose of compact algorithms is to address to the computational requirements in the limited resources of hardware devices such as memory size or low price. A probabilistic representation random of the collection behavior of social bee colony is inspired to employ for this proposed algorithm. The real population is replaced with the probability vector updated based on single competition. These lead to a modest memory usage when the entire algorithm is applied. Four selected test functions are used to evaluate the accuracy, computational time and memory saving of the proposed method. The experimental results show that the proposed cABC method is not only as accurate as the existing original Articial Bee Colony optimization but also requires less calculative time than the original method and uses a modest memory with only six agents needed for storing space. In addition, compared with the genetic algorithm (GA) method and the particle swarm optimization (PSO) method, the proposed cABC method can provide the highest robust structure and lowest contention topology schemes.*
Keywords: Bee colony algorithm, Compact artificial bee colony algorithm, Optimizations, Swarm intelligence, Topology control, Wireless sensor networks.

1. **Introduction.** Computational intelligence algorithms have been used to solve optimization problems in engineering, financial, and management fields. For example, genetic algorithms (GA) have been used successfully in engineering, financial, and security [1-3]. Particle swarm optimization (PSO) and enhanced PSO techniques [4-6] have been employed to forecast the exchange rates, segment images, optimize multiple interference cancellations [7-9], construct the portfolios of stock, and segment color images based on human perception [3, 10, 11]. Ant colony optimization (ACO) techniques have been utilized to solve the routing problem of networks and secure watermarking [12, 13]. Artificial bee colony (ABC) and Interactive Artificial Bee Colony have been used to solve the numerical problems and support the passive continuous authentication systems [14, 15]. Cat

swarm optimization (CSO) techniques have been used to solve the numerical problems, the aircraft schedule, and the lot-streaming flow shop scheduling problem [16-18] and discover proper positions for information hiding [19], respectively. In addition, bat algorithm (BA) is used for engineering design [20] and classifications[21]. Some applications require the solution of a complex optimization problem in limited hardware conditions caused by the cost and space constraints of computational devices. For example, wireless sensor networks (WSNs) are the networks of small, battery-powered, and memory-constraint devices (i.e., sensor nodes). Due to memory and power constraints, they need to be well arranged to build a fully functional network to have the capability of wireless communication in a restricted area [22]. For telecommunications [23] and energy production [13], a fast solution of the optimization problem is required. In addition, space shuttle control [24] and underwater communication [25] require high fault-tolerance and the avoidance of device rebooting. However, computational devices do not have enough memory to store a population composed of numerous candidate solutions of those computational intelligence algorithms for the aforementioned applications.

Compact algorithms are a promising answer to hardware limitations because they use an efficient compromise to present some advantages of population-based algorithms without storing an actual population of solutions. Compact algorithms simulate the behavior of population-based algorithms by employing their probabilistic representation instead of a population of solutions. Consequently, compact algorithms require less memory to store the number of parameters compared to their corresponding population-based structures.

The rst implementation of compact algorithms is the compact Genetic Algorithm (cGA) [26]. The cGA simulates the behavior of a standard binary encoded Genetic Algorithm (GA). The performance of cGA is almost as good as that of GA and requires less memory. The compact Differential Evolution (cDE) algorithm has been introduced in [11]. The success implementation of cDE is based on the combination of two factors. First, a cDE scheme benets from the introduction of a certain degree of randomization due to the probabilistic model. Second, the one-to-one spawning survivor selection typical of cDE (the offspring replaces the parent) can be naturally encoded into a compact logic.

The compact Particle Swarm Optimization (cPSO) has been dened in [27]. The implementation of cPSO algorithm benets from the same natural encoding of the selection scheme employed by cDE and another ingredient of compact optimization, i.e., a special treatment for the best solution ever detected and reinterpreted as an evolutionary algorithms in order to propose a compact encoding of PSO.

In this paper, the behavior and the characteristic of the bees are reviewed to improve the Artificial Bee Colony algorithms [16, 28] and to present the compact Artificial Bee Colony Algorithm (cABC) based on the framework of the original Artificial Bee Colony (oABC). According to the experimental results, our proposed cABC presents the same result in finding solutions as the original Artificial Bee Colony algorithm [29].

Moreover, WSNs, an emerging and promising technology, have been widely used in a variety of long-term and critical applications [30]. However, sensor nodes are limited in the computation capability and storage capacity of a computing unit, the communication range and radio quality of a communication unit, the sensing coverage and accuracy of a sensing unit, and the available energy of a power unit [31]. Topology control is one of the most fundamental problems in WSNs. It is an effective factor to ensure the quality of connectivity and coverage because it determines how to maintain network connectivity and transmit the power of each node while consuming as minimum power as possible. The new proposed cABC method would be applied to find out the solution for the topology control scheme in WSNs. The topology control scheme could be transformed into the problem of multi-objective degree-constrained minimum spanning tree. The multi-objective strategy

with a fitness function based on a niche and phenotype sharing function is also applied in cABC to obtain an approximation of the true Pareto front.

The rest of this paper is organized as follows: a brief review of ABC is given in Section 2; the statement of topology control in WSNs is reviewed in Section 3; the analysis and design for the cABC is presented in Section 4; the experimental results and the comparison between oABC and cABC are discussed in Section 5; the application of cABC for topology control is presented in Section 6; finally, the conclusion is made in Section 7.

2. **The Artificial Bee Colony algorithm.** The Artificial Bee Colony algorithm was proposed by Karaboga in 2005 [23], and the performance of ABC was analyzed in 2008 [24] by inspecting the behaviors of real bees on finding nectar and sharing the information of food sources to the bees in the nest. There are three kinds of bees defined in ABC as being the artificial agents known as the employed bee, the onlooker, and the scout. Every kind of these bees plays a different and important role in the optimization process. For example, the employed bee stays on a food source, which represents a spot in the solution space, and provides the coordinate for the onlookers in the hive for reference. The onlooker bee receives the locations of the food sources and selects one of the food sources to gather the nectar. The scout bee moves in the solution space to discover new food sources.

The process of ABC optimization is listed as follows:

Step 1. *Initialization*: Spray n_e percentage of the populations into the solution space randomly, and then calculate their fitness values, namely nectar amounts, where n_e represents the ratio of employed bees to the total population. Once these populations are positioned into the solution space, they are called the employed bees. The fitness value of the employed bees is evaluated to take account in their amount of nectar.

$$P_i = \frac{F(\theta_i)}{\sum_{k=1}^{S} F(\theta_k)} \tag{1}$$

Step 2. *Move the Onlookers*: Calculate the probability of selecting a food source by equation (1), where θ_i denotes the position of the i^{th} employed bee, $F(\theta_i)$ denotes the fitness function, S represents the number of employed bees, and P_i is the probability of selecting the i^{th} employed bee. The roulette wheel selection method is used to select a food source to move for onlooker bees and then determine their nectar amounts. The onlookers are moved by equation (2), where x_i denotes the position of the i^{th} onlooker bee, t denotes the iteration number, is the randomly chosen employed bee, j represents the dimension of the solution, and $\Phi(.)$ produces a series of random variable in the range from -1 to 1.

$$x_{ij}(t+1) = \theta_{ij}(t) + \Phi(\theta_{ij}(t) - \theta_{kj}(t)) \tag{2}$$

Step 3. *Update the Best Food Source Found So Far*: Memorize the best fitness value and the position, which are found by the bees.

Step 4. *Move the Scouts*: If the fitness values of the employed bees are not improved by a continuous predetermined number of iterations, namely *Limit*, those food sources are abandoned, and these employed bees become the scouts. The scouts are moved by equation (3), where r is a random number and $r \in [0, 1]$.

$$\theta_{ij} = \theta_{jmin} + r \times (\theta_{jmax} - \theta_{jmin}) \tag{3}$$

Step 5. *Termination Checking*: Check if the amount of the iterations satisfies the termination condition. If the termination condition is satisfied, terminate the program and output the results, otherwise go back to *Step*2.

The main steps of the algorithm are as below:

1. Initialize Population
2. repeat
3. Place the employed bees on their food sources
4. Place the onlooker bees on the food sources depending on their nectar amounts
5. Send the scouts to the search area to discover new food sources
6. Memorize the best food source found so far
7. until requirements are met

3. **Topology control scheme for wireless sensor networks.** A wireless sensor network is modeled as a directed, connected graph $G = (V, E)$, where V is a nite set of vertices (sensor nodes), $V = \{v_1, v_2, ..v_n\}$ and E is the set of edges (network links), representing connection of these vertices, $E = \{e_{1,2}, e_{1,3}, ..e_{i,j}, ..e_{n-1,n}\}$. Let $n = |V|$ be the number of network nodes and $l = |E|$ be the number of network links. The $linke = (v_i,)$ from node $v_i \in V$ to node $v_j \in V$ implies the existence of a link $e' = (, v_i)$ from node v_j to node v_i. A *link* can be defined as follows:

$$e_{i,j} = \begin{cases} 1, \ if \ v_i, v_j \ have \ edge \\ 0, \ otherwise \end{cases} (i = 1, 2, .., n-1; j = i+1, 2, .., \ n) \tag{4}$$

If the edge of $e_{i,j}$ exists, this edge has l associated positive real numbers. There are attributes in WSNs that weights could be defined on [28] [29]. Representing the weight could be denoted $w_{i,j}^k = \left\{w_{ij}^1, w_{ij}^2, ..w_{ij}^l\right\}$, where $k = 1, 2, , l$. There l positive real value functions are associated with each $linke(e \in E)$ such as: cos $C(e) : E \to R^+$, coverage $B(e) : E \to R^+$, delay $D(e) : E \to R^+$, data fusion $F(e) : E \to R^+$, loss rate $L(e) : E \to R^+$, power consumption $P(e) : E \to R^+$, etc. The link cost function, $C(e)$, may be either monetary cost or any measures of resource utilization that must be optimized. The link coverage, $B(e)$, is the reachable sensing radius of the sensors. The link delay, $D(e)$, is considered to be the sum of switching modes, queuing, transmission, and propagation delays. The link data fusion,$F(e)$, is the aggregation and integration data functions. The link loss rate, $L(e)$, is the packet loss rate on the receiving end on link e. The link power consumption, $P(e)$, is the energy for transiting, receiving and processing signals. These attributes $B(e), D(e), F(e), L(e), P(e)$ dene the criteria that must be constrained (bounded) because the sensor nodes are limited resources. Let $P_T(s, d)$ be path in the tree T from the source node s to a destination node $d \in M$. Let $m = |M|$ be the number of multi-criteria destination nodes, where M is the destination group and $s \cup M$ is the multi-criteria group. Let α be the coverage constraint, β be the delay constraint, δ be the data aggregation constraint, ζ be the loss rate constraint, and η be the dissipated energy constraint. The multi-constrained least-cost multi-criteria problem is defined as follows:

$$\begin{aligned} &MinimizeC(T(s, M)) \text{ subjects to:} \\ &B_{x_i,y_i}(X) \le \alpha \ \forall \ d \ \in M; D(P_T(s, d)) \le \beta \ \forall \ d \ \in M; F(P_T(s, d)) \le \delta \ \forall \ d \ \in M; \\ &L(P_T(s, d)) \le \zeta \ \forall \ d \ \in M; P(P_T(s, d)) \le \eta \ \forall \ d \ \in M \end{aligned} \tag{5}$$

A spanning tree of graph G can be expressed by the vector x.
$Letx = \ (x_{1,2}, x_{1,3}, \ \ldots, \ x_{i,j}, \ \ldots x_{n-1,n})$

$$x_{i,j} = \begin{cases} 1, \ if \ e_{i,j} = 1 \ and \ selected \\ 0, \ otherwise \end{cases} (i = 1, 2, \ldots, n-1; j = i+1, 2, \ldots, n) \tag{6}$$

A multi-criteria tree $T(s, M)$ is a subgraph of G spanning the source node $s \in V$ and the set of destination nodes $M \subseteq V - \{s\}$. Let X be the set of all such vectors corresponding

to spanning trees in graph G. The multi-criteria degree constrained minimum spanning tree problem can be formulated as follows:

$$\begin{gathered}\min f_1(x) = \sum w_{i,j}^1 x_{i,j} \\ \min f_2(x) = \sum w_{i,j}^2 x_{i,j} \\ \dots \\ \min f_m(x) = \sum w_{i,j}^l x_{i,j} \\ (i = 1, 2, .., n-1; j = i+1, \dots n) \\ 1 \leq \sum w_{i,j}^l x_{i,j} \leq d \\ (x \in X;\ i = 1, 2, .., n; j = 1, 2, \dots n)\end{gathered} \tag{7}$$

where $f_i(x)$ is the i^{th} objective to be minimized for the problem and d denotes the degree constraint. Wireless sensor network may suffer from poor network utilization, high end-to-end delays, and short network lifetime if its topology control scheme is not proper in a right place.

4. **The proposed Compact Artificial Bee Colony (cABC) method.** As mentioned above, compact algorithms process an actual population of solution as a virtual population. This virtual population is encoded within a data structure, namely Perturbation Vector (PV) as probabilistic model of a population of solutions. The distribution of individuals in the hypothetical swarms must be described by a probability density function (PDF) [30] defined on the normalized interval from -1 to +1. The distribution of each bee in the swarms could be assumed as Gaussian PDF with mean μ and standard deviation δ [20]. A minimization problem is considered in an m-dimensional hyper-rectangle in normalization of two truncated Gaussian curves (m is the number of parameters). Without loss of generality, the parameters are assumed to be normalized so that each search interval ranges from -1 to +1. Therefore, PV is a vector of $m2$ matrix specifying the two parameters of the PDF of each design variable. PV is defined as:

$$PV^t = \left[\mu^t, \delta^t\right] \tag{8}$$

where μ and δ are mean and standard deviation values of a Gaussian (PDF) truncated within the interval range from -1 to +1, respectively. The amplitude of the PDF is normalized in order to keep its area equal to 1. The apex t is time step. The initialization of the virtual population is generated for each design variable i, $\mu_i^1 = 0$ and $\delta^1 = k$, where k is set as a large positive constant (e.g., $k = 10$). The PDF height normalization is obtained sufficiently in the uniform distribution with a wide shape. The generating for a candidate solution x_i is produced from $PV(\mu_i, \delta)$. The value of mean μ and standard deviation δ in PV are associated to the equation of a truncated Gaussian PDF as follows:

$$PDF\left(trucNormal\left(x\right)\right) = \frac{e^{-\frac{(x-\mu_i)^2}{2\delta_i^2}}\sqrt{\frac{2}{\pi}}}{\delta_i(\operatorname{erf}\left(\frac{u_i+1}{\sqrt{2}\delta_i}\right) - \operatorname{erf}\left(\frac{u_i-1}{\sqrt{2}\delta_i}\right))} \tag{9}$$

The PDF in equation (9) is then used to compute the corresponding Cumulative Distribution Function (CDF). The CDF is constructed by means of Chebyshev polynomials by following the procedure described in [31]. The codomain of CDF ranges from 0 to 1. CDF is defined as a real-valued random variable X with a given probability distribution at a value less than or equal to x_i, as shown in equation (10). CDFs are also used to specify the distribution of multivariate random variables.

$$CDF = \int_0^1 \frac{e^{-\frac{(x-\mu_i)^2}{2\delta_i^2}} \sqrt{\frac{2}{\pi}}}{\delta_i(\operatorname{erf}\left(\frac{u_i+1}{\sqrt{2}\delta_i}\right) - \operatorname{erf}\left(\frac{u_i-1}{\sqrt{2}\delta_i}\right))} dx \tag{10}$$

The sampling of the design variable x_i from PV is performed by generating a random number rand [0, 1] from a uniform distribution and then computing the inverse function of CDF in rand [0, 1]. The newly calculated value is x_i by the sampling mechanism as equation (11):

$$x_i = inverse(CDF) \tag{11}$$

When the comparison between two design variables for individuals of the swarm (or better two individuals sampled from PV) is performed the winner solution biases the PV. The vector that scores a better tness value is regarded as the winner; the individual losing the (tness based) comparison is defined as the loser. Regarding the mean values l, the update rule for each of its elements is $\mu_i^t, \delta_i^t => \mu_i^{t+1}, \delta_i^{t+1}$

$$\mu_i^{t+1} = \mu_i^t + \frac{1}{N_p}(winner_i - loser_i) \tag{12}$$

where Np is virtual population size. Regarding δ values, the update rule of each element is given by:

$$\delta_i^{t+1} = \sqrt{(\delta_i^t)^2 + (\mu_i^t)^2 - \left(\mu_i^{t+1}\right)^2 + \frac{1}{N_p}(winner_i^2 - loser_i^2)} \tag{13}$$

$$[winner, loser] = complete\left(x_{best}, x^{t+1}\right) \tag{14}$$

The construction of equations (13) and (14) are persistent and non-persistent structures with tested results given in [32]. Similar to the binary cGA case, it was impossible to assess whether one or another elitist strategy was preferable. In elitist compact schemes, each moment of the optimum performance is retained in a separate memory slot. If a new candidate solution is computed, the tness-based comparison between it and the elite is carried out. If the elite is a winner solution, it biases the PV as shown in formulas (13) and (14).

Figure 1 shows the pseudo code of algorithm working principles of cABC. The tness value of the position x^t is calculated and compared with x_{best} to determine a *winner* and a *loser*. Equations (13) and (14) are then applied to update the probability vector PV. If rand is smaller than *Prob* (probability equation (1) is calculated from employment bee phrase), x^t will be calculated by equation (2). Update local and update global are implemented in Onlooker bee phrase. If $f(sol) < f_{best}$, the value of function is memorized, and the value of the global best is then updated.

5. **Experimental results.** This section presents the simulation results in running benchmark function tests and compares the cABC with the oABC, both in terms of solution quality and the number of memory variables evaluations taken. Four test standard functions are chosen for the experiments to evaluate the accuracy and computational speed of the proposed cABC. All experiments are averaged over different random seeds with 10 runs. The test standard functions used include Rosenbrock, Griewank, Rastrigin, and Sphere, which are listed in equations (15) to (18).

$$f_1(x) = \sum_{i=1}^{n-1} (100\left(x_{i-1} - x_i^2\right)^2 + (1 - x_i)^2 \tag{15}$$

$$f_2(x) = 1 + \sum_{i=1}^{N} \frac{x_i^2}{4000} + \prod_{i=1}^{N} cos\frac{x_i}{\sqrt{i}} \tag{16}$$

$$f_3(x) = \sum_{i=1}^{N} [10 + x_i^2 - 10cos2\pi x_i] \tag{17}$$

$$f_4(x) = \sum_{i=1}^{N} x_i^2 \tag{18}$$

```
1) Initialization probability vector (PV(μ, δ))
   for i=1:n do  μ_i^t =0; δ_i^t=k= 10;
2) Initialization parameters:  trial=0; limit=10; sol=x_best=up_bound;
   while termination is not satisfied do
3) Employed Bee Phase
   generate x^t from PV; Calculate f(x^t);
   if (f(x^t)<f(sol) then   sol=x^t;  f(sol)=f(x^t);
   else                     trial=trial+1;  end if
   //Update PV
   [winner,loser] =compete(x^t,x_best)
   Equation (12) and (13)
4) Onlooker bee phase
   x^t=sol; Prob = equation(1)
   if (rand<Prob)
       for i=1:n do
       x^t (i)= x^t (i)+rand*(x^t(i)-x(k)); with random k=1,..,n
    end if
    Calculate f(x^t);
   //Update local
   if (f(x^t)<f(sol)) then   sol=x^t;   f(sol)=f(x^t);
   else  trial=trial+1;  end if
   //Update global
   if (f(sol)<fbest) then  x_best=sol;  fbest=f(sol);
5) Scout Bee Phase
   if (trial==limit) then
      initial(sol);
   end if
  end while
```

FIGURE 1. The pseudo code of compact Artificial Bee Colony algorithm

The initial range and total iteration for all test functions are listed in Table 1.

The optimization goal for all of these test functions is to minimize the outcome. The parameters setting for both cABC and oABC are the initial $limit = 10$ of food source, the total population size $n = 20$, and the dimension of the solution space $dim = 10$. Each function contains the full iterations of 1000 is repeated by different random seeds with 10

TABLE 1. The initial range and the total iteration of test standard functions

Functions		**Initial range** $[x_{max}, x_{min}]$	**Total iterations**
Rosenbrock	$f_1(x)$	[-30,30]	1000
Griewangk	$f_2(x)$	[-100,100]	1000
Rastrigin	$f_3(x)$	[-5.12,5.12]	1000
Spherical	$f_4(x)$	[-100,100]	1000

runs. The final result is obtained by taking the average of the outcomes from all runs. The results are compared with the original ABC.

5.1. **Comparing optimizing performance algorithms.** Table 2 shows the comparison of the quality of performance and time running for numerical problem optimization between cABC and oABC. It is obvious that the average cases of the testing functions in compact Artificial Bee algorithm converge faster than the original cases. The mean of four test functions on the evaluation of minimum function for 10 runs is 6.78E+07 with average time consuming 1.174 s for oABC and 3.29E+07 with average time consuming 0.341 s for cABC.

TABLE 2. The comparison between oABC and cABC in terms of performance quality and speed

Functions	**Performance as mean of evaluation**		**Time running evaluation** *(seconds)*	
	oABC	***cABC***	***oABC***	***cABC***
$f_1(x)$	1.79E+08	1.75E+08	0.7503	0.2118
$f_2(x)$	0.2987	0.3615	1.0895	0.2457
$f_3(x)$	137.8214	139.2473	0.6979	0.2075
$f_4(x)$	2.2582	2.6112	0.6241	0.1891
Average value	**1.79E+08**	**1.76E+08**	**3.1618**	**0.8541**

Figures 2 to 3 show the average of function minimum of four test functions in 10 seed of output with the same iteration of 1000.

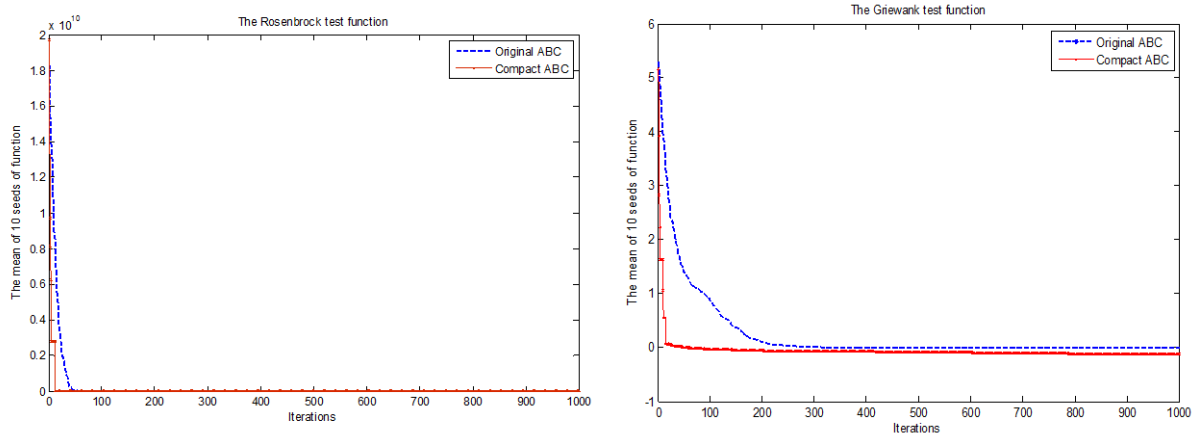

FIGURE 2. The mean of 10 runs of fitness function minimum curves in comparing cABC and oABC algorithms for the Rosenbrock and Griewank

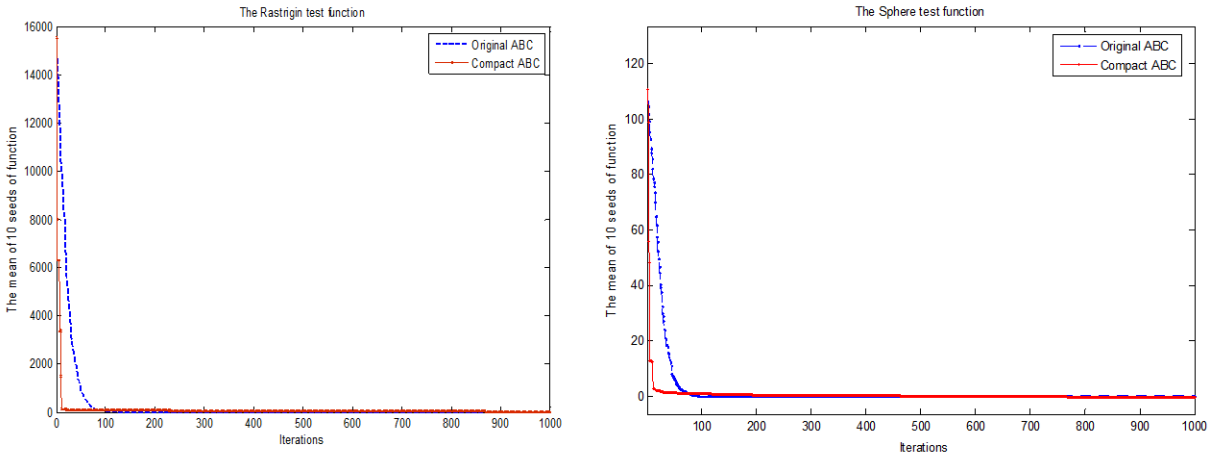

FIGURE 3. The mean of 10 runs of fitness function minimum curves in comparing cABC and oABC algorithms for the Rastrigin and Sphere

5.2. **Comparing saving-memory and time-complexity algorithms.** Table 3 shows the comparison of the saving-memory computations of two algorithms, cABC and oABC.

TABLE 3. The saving-memory comparison between compact ABC and original ABC

Algorith -ms	Popula -tion size	Dimen -sion	#Memory variables	# Equations	Computing complexity
Original ABC	N	D	$3 \times N \times D$	(1),(2),(3)	$3 \times T \times N \times D \times iteration$
Compact ABC	1	D	$6 \times D$	(1),(2),(3), (12),(13),(14)	$6 \times T \times D \times iteration$

It is evident that the number memory variables of cABC are smaller than those of oABC in the same computation condition, such as iterations. The real numbers of population or population size and dimension space of food source of oABC are N and D, but that size for cABC is only one with dimension D. Six equations (i.e., equations (1), (2), (3), (12), (13) and (14)) are used for optimizing computation in cABC. In contrast, three equations (i.e., equations (1), (2), and (3)) are used in oABC. The computing complexity of cABC and oABC is $6 \times T \times D \times iteration$ and $3 \times T \times N \times D \times iteration$, respectively. Thus, the rate of saving-memory equals the computing complexity of cABC per the computing complexity of oABC as given: rate $= 2/N$.

The computational times for both the algorithms cABC and oABC have been calculated by means of a PC Intel Core 2 Duo 2.4 GHz with 4 GB RAM employing in Windows7-OS, with Matlab (R2011b), version 7.13.0.564 32bits. Figure 4 illustrates the comparison of executing time between cABC and oABC in 10 trails with iteration 1000 for four test functions. It is clear that the execution time of test functions in the proposed cABC (red colored bars) is less than that in oABC (blue colored bars).

6. **Application of the proposed cABC method.** In this section, an application of the proposed cABC method is presented to solve the problem of topology control in WSNs. The objective function for optimal topology control schemes in WSNs is constructed based on the residual energy node and contention. The experimental results of the proposed

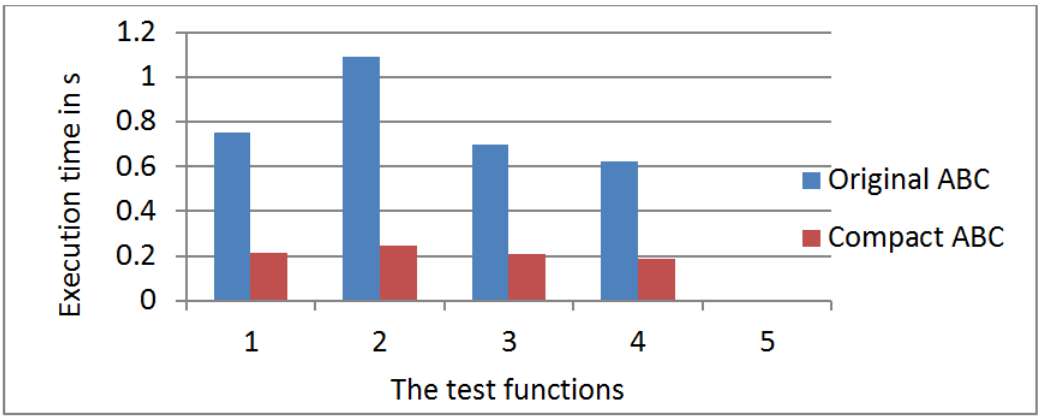

FIGURE 4. Comparison of two algorithms in term of time running for test functions

method is compared with the PSO-Optimized Minimum Spanning Tree-Based Topology Control Scheme method in [33] and the Genetic Algorithm (GA) for Multi-criteria Minimum Spanning Tree Problem in [34].

6.1. **Network Model Description.** As mentioned in section 4, a network model could be described as following: a wireless sensor network with n nodes is randomly distributed in desired areas. Each node can communicate with others by using transmission range r. That is, node i can receive the signal of node j if node i is in the transmission range r of node j. The topology of the network is abstracted by nodes with their own maximum transmission range as a directed graph, denoted by $G =< V, E >$ in which $V = \{V_1, V_2, .., V_n\}$ is the set of node numbers, and $E = \{E_1, E_2, .., E_n\}$ is the set of connectable communication between any of two nodes.

Let $D(i, j)$ be the Euclidean distance of nodes i and j in the network and CNi be the set of connectable neighbors which communicate with node i by using its maximum wireless transmission range r_{max}. For each node i that belongs to V, let $G_i =< V_i, E_i >$ be an induced sub-graph of G in which $V_i = CN_i, E_i$ is the subset of E. Consequently, G_i is the connectable node graph of node i.

Let $C(i, r)$ be the coverage of an edge, that is, the disk centered at node i with its certain radius r. Node i could affect at least all nodes located in the area of radius $r = D(i, j)$, centered at node i. The coverage of an edge between node i and node j is the measure of the number of nodes covered by the disk established by i and j [35].

$$L_{i,j} = [\{V_i \in V | D(V_i, i) \leq R_1\}] \cup [V_j \in V | D(V_j, j) \leq R_2)] \tag{19}$$

where R_1 and R_2 are the radius of $C(i, r)$ and $C(j, r)$, respectively. The strength of an edge between node i and node j is defined as:

$$S_{i,j} = \frac{e_i \times e_j}{\sqrt{e_i^2 + e_j^2}} \tag{20}$$

where e_i and e_j are the remaining energy of node i and j, respectively.

Local minimum spanning tree of node i is implemented based on the set of connectable neighbors G_i. It is assumed that $E = \{E_{1,2}, E_{1,3}, ..., E_{i,j}, ..., E_{n-1,n}\}$ is the set of edges constructed by $G_n =< V_n, E_n >$. If the connection can be formed between nodes V_i and V_j, then $E_{i,j} = E_{,ji} = 1$. Otherwise, $E_{i,j} = E_{,ji} = 0$, where $\forall V_i, V_j \in G_n; i = 1, 2, 3, ..., n; j = i + 1, ..., n$. The energy consumption between nodes V_i and V_j is defined as$P_{i,j} = kd^{\beta}$, where k is the system constant, d is the communication distance, and the

value of β is often predefined constant as the path loss exponent with value setting from 2 to 4. Each edge has a positive real number, W_i, defined as the weight of a communication link and calculated as the result of a topology with the aspects of low energy consumption, high stable structure, and low communication interference:

$$W_{i,j} = \alpha_1 \times \frac{P_{i,j}}{S_{i,j}} + \alpha_2 \times L_{i,j} \tag{21}$$

where α_1 and α_2 are the predefined parameters and $\alpha_1 + \alpha_2 = 1$. A spanning tree of graph G_n can be expressed by vector X as shown in equation (6), which can be transformed into the optimization problem for the minimal fitness function as shown in equation (7), which in turn can be formulated as the following equation:

$$Fitness_i = \sum W_{i,j} \times x_{i,j} \tag{22}$$

In reality, because the network topology in WSNs may vary with time as a result of many uncertain factors, the network topology is adjusted every Δ_t time. At the same time, uni-directional edges can be removed or some extra edges can be added in order to get a final topology consisting of only bi-directional edges. The coordinates of vertices in G and the weights of each edge are generated randomly. Each edge in the graph is supposed to have only two weights with real numbers

6.2. **Experimental results and comparison.** The environmental setting is as follows: the range of deployment of network is $100X100m$, the number of objectives is 2, and the number of vertices is 20. The setting parameters for both the proposed cABC-WSNs topology and oABC-WSNs topology are as follows: the initial *limit* of food source is 10, the total population size n is 20, and the dimension of the solution space *dim* is 10. Each function contains 1000 full iterations for each of the 10 random seeds. The experimental parameters for PSO-WNS topology are $c_1 = c_2 = 2.0$, inertia weight w is 0.9, population size is 20, and the maximum iteration times is 1000 in each run [36]. The parameters for the GA-WSNs topology are set as follows: population size is 20, crossover probability pc is 0.2, mutation probability pm is 0.05, and maximum generation max gen is 1000 in each run [1, 34]. The final result is obtained by taking the average of the outcomes from 10 runs. The experimental results of cABC-WSNs are compared with the results of ABC (ABC-WSN topology), GA (GA-WSN topology), and PSO (PSO-WSN topology).

Table 4 shows that the average convergent time of the fitness functions in the cABC method (9.288 minutes) is 30% faster than that of the oABC method (12.086 minutes). While cABC-WSNs topology evidently outperforms oABC-WSNs topology in time consumption, the mean of fitness functions evaluation of minimum function for 10 runs obtained using the cABC method is almost as good as that obtained using the oABC method. In addition, it is much more accurate than the means obtained using the GA-WSNs topology and PSO-WSNs topology by 16% and 8%, respectively.

Figure 5 illustrates the comparison of the cABC-WSNs topology method with the oABC-, PSO-, and GA-WSNs topology methods.

It is obvious that the performance of fitness functions evaluation values for 10 runs using the cABC-WSNs topology and oABC-WSNs topology are faster convergence than the performance using the GA-WSNs topology and PSO-WSNs topology.

7. **Conclusion.** In this paper, a novel optimization algorithm, namely compact Artificial Bee Colony algorithm (cABC), was presented. The implementation of compact optimization algorithms is significant for the development of small-sized and low-cost embedded devices. It fits the trend of ubiquitous computing today. In this new proposed algorithm,

TABLE 4. The comparison of the proposed cABC-WSN topology with the GA-WSNs topology, the PSO-WSNs topology, and the oABC-WSNs topology in terms of quality performance evaluation and speed

Methods	Population size	Objectives	Vertices	Average function values	Consum -ption times(m)
The GA-WSNs topology (Han & Wang 2005, with cm-MST)	20	2	20	8.6023	22.086
The PSO-WSNs topology (Wenzhong et al., 2013, with cm-MST)	20	2	20	7.2091	11.086
The oABC-WSNs topology	20	2	20	5.9434	12.086
The cABC-WSNs topology	**1**	**2**	**20**	**6.0921**	**9.288**

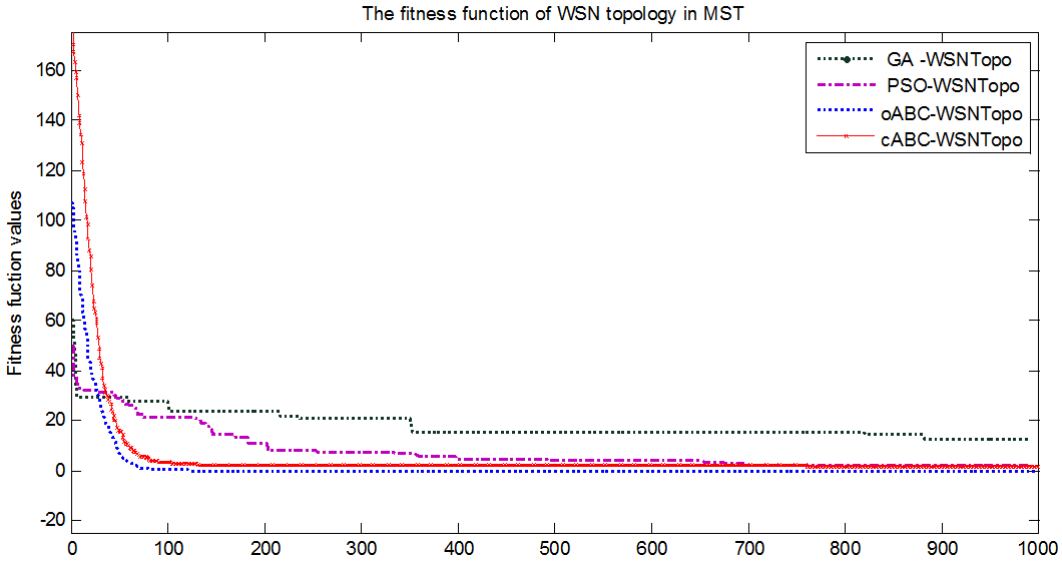

FIGURE 5. The averages of minimum value of fitness function for 10 runs obtained using the cABC-, oABC-, PSO-, and GA-WSNs topology methods.

the actual design solutions for search space for Artificial Bee Colony algorithm is replaced with a virtual population, which is a probabilistic representation of the population. This feature is essential for applications with a limited memory such as the embedded implementation in small and inexpensive devices. The performance of cABC algorithm is as good as the other compact algorithms in previous works in literature. The results of the proposed algorithm on a set of various test problems show that cABC is a valid alternative for optimization problems with a limited memory. The proposed method is also applied to solve the problem of topology control in WSNs. Compared with the GA method and the PSO method, the proposed cABC method provides the most robust structure and

lowest contention topology schemes. The experimental results show the proposed cABC as an effective memory-saving algorithm.

REFERENCES

[1] L. Davis (ed), *Handbook of genetic algorithms*, vol. 115, New York: Van Nostrand Reinhold, 1991.

[2] S. Wang, B. Yang, and X. Niu, A Secure Steganography Method based on Genetic Algorithm, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, pp. 28-35, 2010.

[3] R. J. Kuo, C. H. Chen and Y. C. Wang, An intelligent stock trading decision support system through integration of genetic algorithm based fuzzy neural network and artificial neural network, *Fuzzy Sets and Systems*, vol. 1, no. 118, pp. 21-45, 2001.

[4] R. Eberhart, J. Kennedy, Particle swarm optimization, vol. 4, pp. 1942-1948, 1995.

[5] H. Wang, H. Sun, C. Li, S. Rahnamayan, and J. S. Pan, Diversity enhanced particle swarm optimization with neighborhood search, *Information Sciences*, vol. 223, pp. 119-135, 2013.

[6] C. Sun, J. Zeng, J. Pan, S. Xue, and Y. Jin, A new fitness estimation strategy for particle swarm optimization, *Information Sciences*, vol. 221, pp. 355-370, 2013.

[7] S. M. Chen and C. Y. Chien, Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques, *Expert Systems with Applications*, vol. 38, pp. 14439-14450, 2011.

[8] C. H. Hsu, W. J. Shyr, and K. H. Kuo, Optimizing Multiple Interference Cancellations of Linear Phase Array Based on Particle Swarm Optimization, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, pp. 292-300, October 2010.

[9] S. Yuhui and R. Eberhart, A modified particle swarm optimizer, *Proc. of Evolutionary Computation Proceedings, IEEE World Congress, Computational Intelligence* , pp. 69-73, 1998.

[10] J. F. Chang and S. W. Hsu, The Construction of Stock's Portfolios by Using Particle Swarm Optimization, *Innovative Computing Information and Control ICICIC '07, Proc. of Second International Conference*, pp. 390-390, 2007.

[11] E. Mininno, F. Neri, F. Cupertino, and D. Naso, Compact Differential Evolution, *IEEE Trans. on Evolutionary Computation*, vol. 15, pp. 32-54, 2011.

[12] P. C. Pinto, A. Nagele, M. Dejori, T. A. Runkler, and J. M. Sousa, Using a Local Discovery Ant Algorithm for Bayesian Network Structure Learning, *IEEE Trans. on Evolutionary Computation*, vol. 13, no. 4, pp. 767-779, 2009.

[13] J. Y. Chouinard, K. Loukhaoukha, and M. H. Taieb, Optimal Image Watermarking Algorithm Based on LWT-SVD via Multi-objective Ant Colony Optimization, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 2, no. 4, pp. 303-319, 2011.

[14] D. Karaboga, An idea based on honey bee swarm for numerical optimization, *Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department*, vol. 200, 2005.

[15] P. W. Tsai, M. K. Khan, J. S. Pan, and B. Y. Liao, Interactive Artificial Bee Colony Supported Passive Continuous Authentication System, *IEEE, Systems Journal*, vol. 8, pp. 395-405, 2014.

[16] Y. T. Hou, S. Yi, H. D. Sherali, and S. F. Midkiff, On energy provisioning and relay node placement for wireless sensor networks, *IEEE Trans. on Wireless Communications*, vol. 4, pp. 2579-2590, 2005.

[17] P. W. Tsai, J. S. Pan, S. M. Chen, and B. Y. Liao, Enhanced parallel cat swarm optimization based on the Taguchi method, *Expert Systems with Applications*, vol. 39, pp. 6309-6319, 2012.

[18] S. C. Chu, P. W. Tsai, and J. S. Pan, Cat Swarm Optimization, *PRICAI 2006, Trends in Artificial Intelligence*, Springer Berlin Heidelberg, vol. 4099, pp. 854-858, 2006.

[19] Z. H. Wang, C. C. Chang, and M. C. Li, Optimizing least-significant-bit substitution using cat swarm optimization strategy, *Inf. Sci.*, vol. 192, pp. 98-108, 2012.

[20] M. Younis, M. Youssef, and K. Arisha, Energy-aware routing in cluster-based sensor networks, *Modeling, Analysis and Simulation of Computer and Telecommunications Systems, IEEE International Symposium*, pp. 129-136, 2002.

[21] G. Qingbo, F. Shuxing, and M. Yanhua, A Network Topology Clustering Algorithm for Service Identification, *Proc. of International Conference Computer Science and Service System (CSSS)*, pp. 1583-1586, 2012.

[22] I. F. Akyildiz, S. Weilian, Y. Sankarasubramaniam, and E. Cayirci, A survey on sensor networks, *IEEE, Communications Magazine*, vol. 40, pp. 102-114, 2002.

[23] G. Cheung, W. T. Tan, and T. Yoshimura, Real-time video transport optimization using streaming agent over 3G wireless networks, *IEEE Trans. on Multimedia*, vol. 7, no.4, pp. 777-785, 2005.

[24] P. G. Norman, The new AP101S general-purpose computer (GPC) for the space shuttle, *Proc. of the IEEE*, vol. 75, pp. 308-319, 1987.
[25] D. Abramson, J. Abela, A parallel genetic algorithm for solving the school timetabling problem, *Division of Information Technology, CSIRO*, 1991.
[26] G. R. Harik, F. G. Lobo, and D. E. Goldberg, The compact genetic algorithm, *IEEE Trans. on Evolutionary Computation*, vol. 3, pp. 287-297, 1999.
[27] F. Neri, E. Mininno, and G. Iacca, Compact Particle Swarm Optimization, *Information Sciences*, vol. 239, pp. 96-121, 2013.
[28] M. E. Aaasser and M. Ashour, Energy aware classification for wireless sensor networks routing, *Advanced Communication Technology (ICACT)*, pp. 66-71, 2013.
[29] S. C. Chu, J. F. Roddick, and J. S. Pan, A parallel particle swarm optimization algorithm with communication strategies, *Journal of Information Science and Engineering*, vol. 21, pp. 9-13, 2005.
[30] P. W. Tsai, J. S. Pan, S. M. Chen, B. Y. Liao, and S. P. Hao, Parallel Cat Swarm Optimization, *Proc. of Machine Learning and Cybernetics, International Conference*, pp. 3328-3333, 2008
[31] R. Pemantle, A survey of random processes with reinforcement, *Probability Surveys*, vol. 4, no. 25, pp. 1–79, 2007.
[32] P. Jianping, C. Lin, Y. T. Hou, S. Yi, and S. X. Shen, Optimal base-station locations in two-tiered wireless sensor networks, *IEEE Trans. on Mobile Computing*, vol. 4, pp. 458-473, 2005.
[33] C. H. Wu and Y. C. Chung, Heterogeneous wireless sensor network deployment and topology control based on irregular sensor model, *Proc. of 2nd international conference on Advances in grid and pervasive computing*, Paris, France, 2007.
[34] P. Billingsley, Probability and Measure, 1979.
[35] S. Yuhui and R. C. Eberhart, Empirical study of particle swarm optimization, *Proc. of Evolutionary Computation, CEC 99*, p. 1950–1958, vol. 3, 1999.
[36] E. Mininno, F. Cupertino, and D. Naso, Real-Valued Compact Genetic Algorithms for Embedded Microcontroller Optimization, *IEEE Trans.on Evolutionary Computation*, vol. 12, pp. 203-219, 2008.
[37] P. Puranik, P. Bajaj, A. Abraham, P. Palsodkar, and A. Deshmukh, Human Perception-based Color Image Segmentation Using Comprehensive Learning Particle Swarm Optimization, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 2, pp. 227-235, 2011.
[38] S. A. Pourmousavi, M. H. Nehrir, C. M. Colson, and W. Caisheng, Real-Time Energy Management of a Stand-Alone Hybrid Wind-Microturbine Energy System Using Particle Swarm Optimization, *IEEE Trans. on Sustainable Energy*, vol. 1, pp. 193-201, 2010.
[39] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, An application-specific protocol architecture for wireless microsensor networks, *IEEE Trans. on Wireless Communications*, vol. 1, pp. 660-670, 2002.