RESEARCH   **Open Access**

A spanning tree construction algorithm for industrial wireless sensor networks based on quantum artificial bee colony

Yuanzhen Li[*] , Yang Zhao and Yingyu Zhang

Abstract

In industrial Internet, many intelligent applications are implemented based on data collection and distribution. Data collection and data distribution in the wireless sensor networks are very important, where the node topology can be described by the spanning tree for obtaining an efficient transmission. Classical algorithms in graph theory such as the Kruskal algorithm or Prim algorithm can only find the minimum spanning tree (MST) in industrial wireless sensor networks. Swarm intelligence algorithm can obtain multiple solutions in one calculation. Multiple solutions are very helpful for improving the reliability of industrial wireless sensor networks.

In this paper, we combine quantum computing with artificial bee colony and design a spanning tree construction algorithm for industrial wireless sensor networks. Quantum computations are introduced into the onlooker bees search. Food source replacement strategy is improved. Finally, the algorithm is simulated and evaluated. The results show that the new proposed algorithm can obtain more alternative solutions and has a better performance in search efficiency.

Keywords: Industrial wireless sensor network, Minimum spanning tree, Artificial bee colony, Quantum computing

1 Introduction

WSN (wireless sensor networks) [1, 2] is an autonomous measurement and control network system that consist of a large number of ubiquitous, small sensor nodes with communication and computing capabilities that are densely deployed in unattended monitoring areas [3]. WSN is a new information acquisition and processing technology [4]. Due to its advantages of low communication cost, flexible networking, and ease of use, it has a wide application prospect in the industry, military, environment, medical, and other fields [5]. IWSN (industrial wireless sensor networks) [6–8] is used to control and monitor various industrial tasks and is an emerging application of WSN.

Due to its high flexibility, low node cost, no wiring, and relatively easy maintenance, IWSN has been favored by more and more companies [9]. A large number of wireless sensors are deployed in industrial parks [10]. These sensor nodes form a multi-hop network in the form of ad hoc networks, which are very sensitive to the equipment, production lines, and environmental information of the industrial site and transmitted to the control center in real time [11]. Through the calculation and analysis of data, the control center can monitor the operating conditions of the equipment, find problems in a timely manner, issue control commands, and reduce safety problems in the production process.

IWSN faces more challenges than ordinary wireless sensor networks [12], which have the following features: (l) The sensor node deployment of the IWSN is related to the industrial environment. It needs to be manually installed on the plant equipment that needs to be monitored [13], emphasizing reliable monitoring of designated points [14]. The nodes of the WSN are generally deployed in a random and

* Correspondence: liyuanzhen@163.com
School of Computer Science, Liaocheng University, Liaocheng, Shandong 252059, People's Republic of China

dense manner, focusing on the overall coverage of the monitoring area. (2) In IWSN, once the sensor node is installed, it is generally no longer moving, unless the node failure needs to be replaced or the plant equipment is moved [15]; in contrast, the WSN node mobility is strong. (3) In addition to sensor nodes in IWSN, there are routers, handheld devices, and other nodes [16]. Different types of nodes perform different functions, forming a heterogeneous network. The WSN generally refers to homogeneous networks, and the status of nodes is equal. (4) The factory environment is complex, and the interference is severe [17]. Therefore, the IWSN wireless network protocol design aims at high reliability and real-time performance. WSNs generally work in unattended areas. Node energy is limited, so the protocol design focuses on energy saving.

Field equipment production process data and network status data are collected periodically [18]. IWSN analyzes and processes this data to monitor factory equipment and networks. In the production process, it is often necessary to obtain some statistical information, such as the total number of network nodes and the temperature of the production site [19]. At present, the data collection of industrial wireless networks is basically centralized. That is, the original data is uniformly collected and processed by the central node. This is generally called the data collection protocol [20, 21]. The control information such as commands for monitoring the industrial field devices is performed through another protocol. This is generally referred to as data distribution protocol [22]. The data distribution process is one-to-many communication, and the data collection process is many-to-one communication. The above two processes are preferably implemented using trees to save network resources, especially the minimum spanning tree. Therefore, in industrial wireless sensor networks, how to construct an effective minimum spanning tree (MST) is a critical issue [23]. Classical algorithms in graph theory, such as Kruskal algorithm and Prim algorithm, utilize greedy strategies to generate only one MST. In IWSN, a series of spanning trees are required to improve reliability and deal with network changes [24]. Even if the found spanning tree is not optimal, but suboptimal, the found suboptimal spanning tree has important practical significance for responding to the dynamic changes of the network.
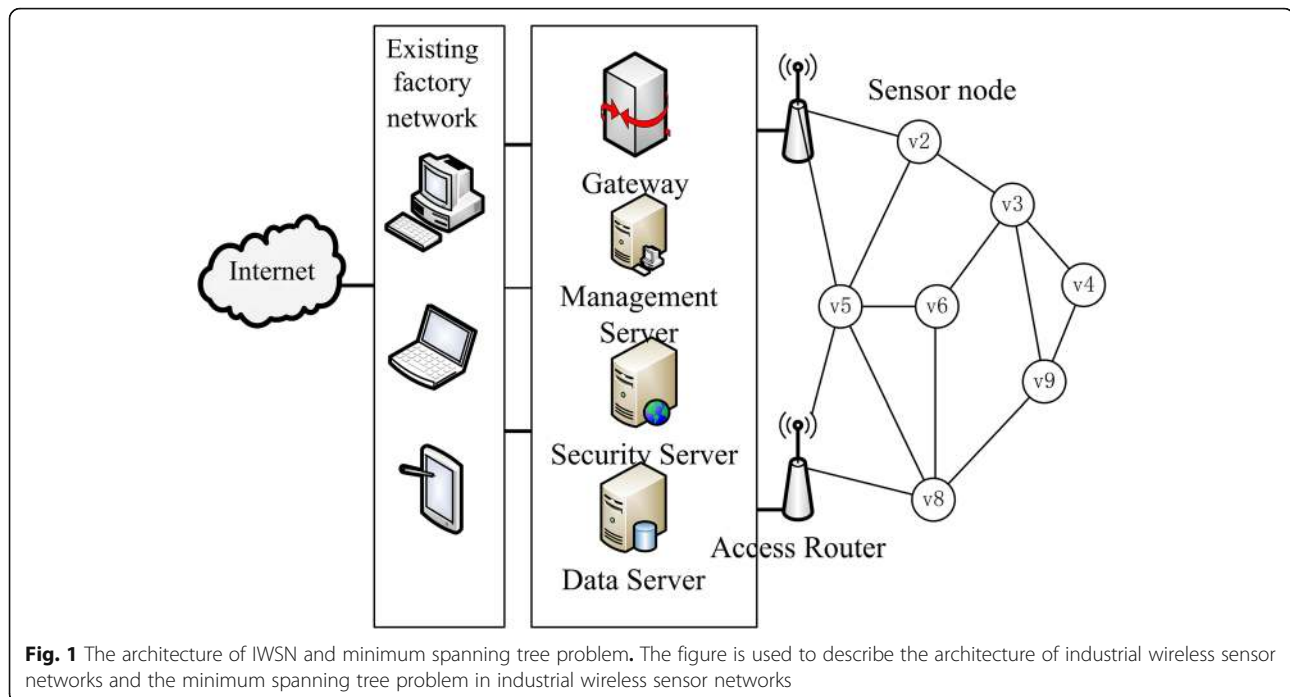
In the evolutionary process of swarm intelligence algorithm [25–27], a series of solutions are generated, which is very suitable for the solution of the spanning tree problem in Industrial Wireless Sensor Networks environment. At the same time, the swarm intelligence optimization algorithm can search without prior knowledge to find a solution to the optimization problem [28, 29]. Artificial bee colony algorithm is a kind of swarm intelligence algorithm, which was put forward by Karabogay in order to solve the problem of multivariable function optimization. Artificial bee colony algorithm is an optimization method proposed to imitate bee behavior. It is a specific application of swarm intelligence thought. Its main characteristic is that it does not need to know the special information about the problem, but only needs to compare the pros and cons of the problem. Through the local optimization behavior of individual artificial bees, the global optimal value is finally emerging in the group, and it has a faster convergence speed. Quantum computation is a new computational model that follows quantum mechanics regulation to regulate quantum information units. From the point of view of computational efficiency, due to the existence of the superposition of quantum mechanics, some known quantum algorithms are faster than conventional general-purpose computers when dealing with certain problems. In this paper, quantum computation and artificial bee colony algorithm are combined and a quantum artificial bee colony algorithm is proposed to solve multicast tree construction problem in industrial wireless sensor networks.

This paper is organized as follows: In Section 2, we first describe IWSN architecture and the minimum spanning tree problem in IWSN. We then present the proposed algorithm in Section 3. Sections 4 and 5 report the simulation results and discussion. Finally, Section 6 concludes the paper.

2 IWSN architecture and minimum spanning tree problem

Figure 1 shows the common IWSN architecture [30]. In IWSNs, sensor nodes are used as field devices to form industrial wireless networks. Nodes have the dual functions of data collection and routing. Sensor nodes distributed in the factory transmit the collected field data to servers through multi-hop routing. The access router is responsible for connecting the wireless network and the wired network and forwarding the field data from the wireless network to the wired network. There can be one or more access routers in the IWSNs. The gateway is responsible for the protocol conversion between the IWSNs and the existing factory network, so that both networks can send packets to each other. The management server is responsible for managing all network devices and in charge of storing topology information,

Fig. 1 The architecture of IWSN and minimum spanning tree problem. The figure is used to describe the architecture of industrial wireless sensor networks and the minimum spanning tree problem in industrial wireless sensor networks

node information, and neighbor relationships, as well as the link status of the entire network. In addition, the management server is also responsible for processing network information, managing network communication processes, and interacting with industrial applications. The security server [31] is responsible for the security management of the network and supports data integrity verification, data encryption, identity authentication, and replay protection. The data server stores field device configuration data, process flow data parameters, and data generated during the production process [32]. Gateways, management servers, security servers, and file servers are logically differentiated and can actually be deployed on the same network device as IWSNs controllers [33]. Finally, IWSN may need to connect to the Internet, depending on the specific conditions and needs of the factory.

A very important area of IWSN is data distribution and data collection. The effective implementation of data distribution and data collection is to use a minimal spanning tree. The minimum spanning tree problem is a basic problem in the areas of graph theory, optimization, and network optimization. Let graph $G = (N,E,W)$ be a connected undirected weighted graph, where N is the set of nodes, E is the set of edges, and W is the weight defined on the edge. $W = \sum_{e \in E} w_e$ is the weight function.

In graph theory, a tree is defined as an acyclic connected graph. If a subgraph of a connected graph G is a tree and contains all vertices of G, the subgraph T is called a spanning tree of G. If G has n vertices, its spanning tree has n vertices and $n - 1$ edges. $W_T = \sum_{e \in E_T} w_e$ is the weight function of the tree T. The spanning tree of a graph G is not unique. A graph can have many different spanning trees. Among these trees, the spanning tree with the least sum of edge weight is the minimum spanning tree (MST).

Because the minimum spanning tree problem is very important in network optimization, researchers have conducted a detailed study of this problem. At present, there are already some classical algorithms for solving the minimum spanning tree in graph theory, such as Kruskal algorithm or Prim algorithm. These algorithms can only get one solution at a time. The characteristic of industrial wireless sensor networks determines that multiple solutions are also necessary and meaningful. These different solutions are mutually complementary and backup. Swarm intelligence algorithm to obtain multiple solutions at a time just can effectively solve this requirement of industrial wireless sensor networks. This paper will use the artificial bee colony algorithm and use the idea of quantum computing to solve this problem.

3 Quantum artificial bee colony
3.1 Standard artificial bee colony
The artificial bee colony (ABC) algorithm [34], originally proposed by Karaboga in 2005, is based on the bee family's foraging behavior. The bee is a social insect. Although the behavior of individual
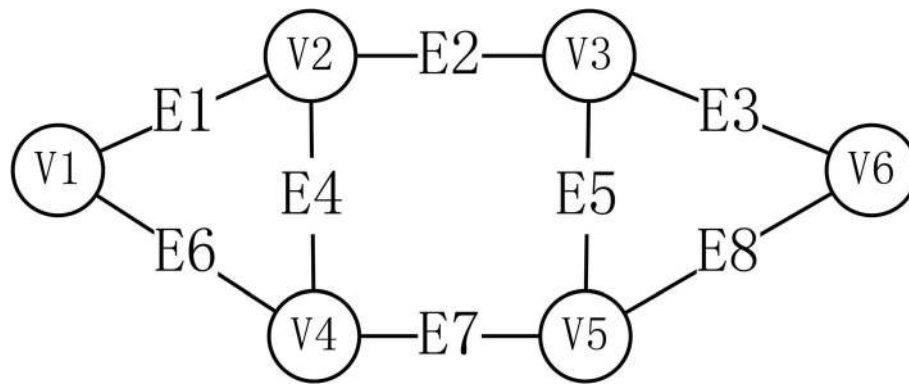
Fig. 2 Graph G. This figure is used to illustrate the coding technique used in this paper. This figure and Figs. 3 and 4 together illustrate the coding technique used in this article

insects is extremely simple, the group of individuals shows extremely complex behavior. Bees can collect nectar from food sources with great efficiency in any environment; at the same time, they can adapt to changes in the environment. As a group organism of nature, the bee colony has a more rigorous foraging system within its organization. The artificial bee colony algorithm is based on the division of labor and cooperation of different types of work groups in the bee colony, thereby more effectively searching for the global optimal solution.

The minimum search model for swarms that generate swarm intelligence includes the basic three components [35]: food sources, employed bees, and unemployed bees. There are also two basic behavioral models: recruiting bees for food sources and giving up food sources. (1) Food sources: the value of food sources is determined by many factors, such as the distance from the hive, the richness of the nectar, and the ease of

obtaining nectar. (2) Employed bees: for each food source, there is only one employed bee, that is, the number of employed bees is equal to the number of food sources. The employed bee stores information about food sources and shares this information with other bees with a certain probability.(3) Unemployed bees: their main task is to find and mine food sources. There are two types of unemployed bees: the scout bees and the onlooker bees. Scout bees search for new food sources nearby. The onlooker bees wait inside the hive and find food sources by sharing information with the employed bee.

The ABC algorithm randomly generates initial populations containing *PS* solutions (food sources). The employed bee conducts a neighborhood search on the corresponding food source, compares the new food source with the original food source, and selects a solution with a high degree of fitness as a candidate solution. When searching work finished, the employed bees share the food source information with the onlooker bees. The onlooker

Fig. 3 Example of binary coding. Binary coding is used to illustrate the coding techniques used in this paper
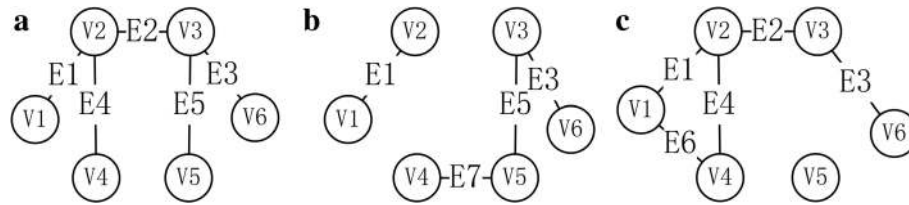
Fig. 4 Tree structure represented by a binary string as in Fig. 3. This figure is used to illustrate the coding technique used in this paper. Tree structure represented by a binary string as in Fig. 3

bees choose the food source according to probability p_i. The higher the food source's fitness value, the greater the probability of being selected.

$$p_i = \frac{f_i}{\sum\limits_{i=1}^{PS} f_i} \tag{1}$$

where f_i is the fitness value of the i-th solution x_i. Then, the onlooker bees also conduct a neighborhood search and choose a better solution. If a solution is not improved for consecutive NC cycles, it is discarded and a random solution is randomly generated by the scout bee. The main steps of the artificial bee colony algorithm are as follows.

(1) Initialize the colony population;

(2) Employed bees search for new honey source near their associated food sources;

(3) The onlooker bee to select the food source using formula (1) and search for a new honey source near the selected food source;

(4) Scout bees to search for new honey sources

(5) Memorize the best food source found so far

(6) If the maximum number of iterations is not reached, repeat the above steps (2–5). The final best honey position is the global optimal solution to be searched.

Artificial bee colony algorithm has shown good performance [36, 37] in the solution of complex optimization problems due to its advantages of simple principles, convenient implementation, good applicability, and favorable cooperation between population division and labor [38]. The original artificial bee colony algorithm is mainly aimed at solving the problem of continuous space function optimization. In order to solve many combinatorial optimization problems in practical engineering, many discrete artificial bee colony algorithms are proposed. The flow of the discretized artificial bee colony algorithm is the same as the artificial bee colony algorithm.

### 3.2 Quantum artificial bee colony algorithm

Quantum computing is a new cross discipline combining information science and quantum mechanics. Quantum computations represented by quantum algorithms have a high degree of parallelism, exponential storage capacity, and exponential acceleration of classical heuristic algorithms. It has great superiority and contains great vitality.

Quantum computing has become a frontier field for scholars from all over the world. By using quantum computing in traditional intelligent optimization, quantum computing and intelligent computing are combined. This will change the traditional optimization
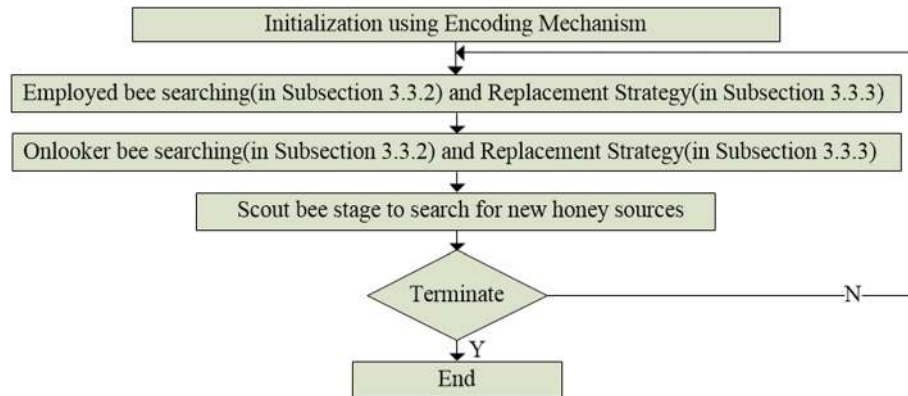
Fig. 5 Flow chart of QABCST algorithm. The flow chart of the QABCST algorithm
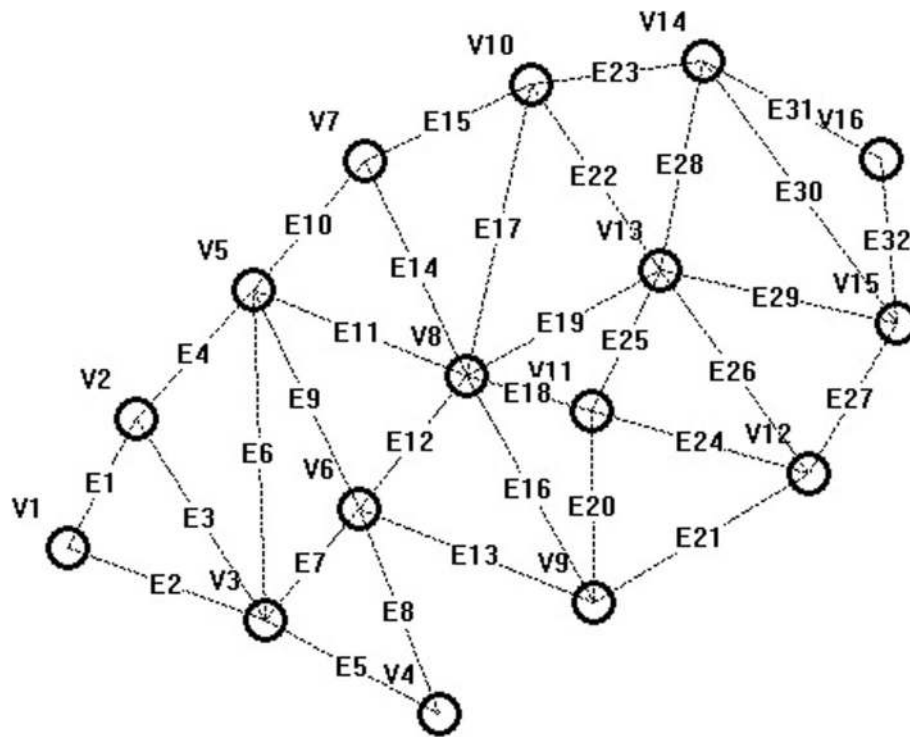
Fig. 6 An IWSN example with 16 nodes and 32 edges. An industrial wireless sensor network diagram for performance simulation. The diagram contains 32 nodes with 16 nodes

methods of intelligent computing and improve the performance of search optimization and convergence speed. Using quantum computing to improve the particle swarm algorithm, Sun et al. proposed quantum particle swarm optimization (QPSO) algorithm [39]. It is assumed that the particle's behavior has quantum characteristics. Under the role of quantum mechanics, particles no longer have limitations on the trajectory and velocity, and the ability of particles to find the optimal solution is greatly improved. Regarding the specific implementation of the algorithm, QPSO uses (Eqs. 2–7) to update the entire population.

$$mb = \frac{\sum_{i=1}^{N} pb_i}{N} \tag{2}$$

$$a = \ rand(0,1) \tag{3}$$

$$p = a * pb_i + (1-a) * gb \tag{4}$$

$$b = 1 - \frac{t}{2 * t_{\max}} \tag{5}$$

$$u = \ rand(0,1) \tag{6}$$

$$p_i = \begin{cases} p - b * \mid mb - p_i \mid * \ln\left(\frac{1}{u}\right), & u \geq 0.5 \\ p + b * \mid mb - p_i \mid * \ln\left(\frac{1}{u}\right), & u < 0.5 \end{cases} \tag{7}$$

Here, N is the population size; pb_i is the individual optimal position of the i-th particle; gb is the optimal location of the entire population; mb is the mean of best position, which is the average of the individual optimal positions of all particles; *rand(0,1)* is a function whose return value is a random decimal between [0, 1]; t is the current evolutionary generation; $t_{\max}$ the maximum evolutionary generation of the algorithm; b is called the contraction expansion coefficient, which gradually decreases with the iteration of the algorithm; p_i is the position of the i-th particle.

Inspired by the QPSO algorithm, we use a similar approach in the QABC (quantum artificial bee colony) algorithm. We only use the idea of quantum computing in the onlooker stage. A quantum representation of solutions is used to enhance the diversity of the basic ABC. In addition, the exploitive capability of the ABC is boosted through the use of the quantum interference concept.

Table 1 The coordinates of the nodes

Node	X	Y	Node	X	Y
V1	40.94	292.97	V9	309.2	1.1. 320.0
V2	75.53	226.48	V10	277.73	1.2. 55.13
V3	141.22	329.18	V11	308.81	1.3. 222.7
V4	230.90	377.83	V12	419.35	1.4. 254.59
V5	135.32	160.54	V13	343.43	1.5. 150.27
V6	189.61	272.43	V14	365.85	1.6. 43.24
V7	192.76	94.59	V15	464.2	1.7. 177.83
V8	244.29	204.32	V16	145.33	1.8. 93.51

3.3 Quantum artificial bee colony algorithm for constructing spanning trees

3.3.1 Encoding mechanism

The standard ABC algorithm for solving continuous optimization problems is not directly suitable for solving the minimum spanning tree problem. Because the minimum spanning tree problem is an optimization problem, binary representation is used. The number of edges in the graph G is denoted as $|E|$. $|E|$ bit binary code is used to represent a solution. Each binary bit corresponds to an edge in the graph G and takes value 0 or 1, where 1 indicates that the corresponding edge is contained in the spanning tree T, and 0 means the opposite. The number of nodes in the graph G is denoted as $|N|$. According to the characteristics of the spanning tree, the spanning tree contains only $|N|$ - 1 edges. In a feasible solution, only $|N|$ - 1 binary bit is 1. The other binary bits are all 0. For example, given a graph, G = (N, E)

Table 2 The weight of the edges

Edge	Vertex	Vertex	Weight	Edge	Vertex	Vertex	Weight
E1	V1	V2	74.96	E17	1.9. V8	1.10. V10	1.11. 152.89
E2	V1	V3	106.65	E18	1.12. V8	1.13. V11	1.14. 67.09
E3	V2	V3	121.91	E19	1.15. V8	1.16. V13	1.17. 112.92
E4	V2	V5	89.01	E20	1.18. V9	1.19. V11	1.20. 97.3
E5	V3	V4	102.04	E21	1.21. V9	1.22. V12	1.23. 128.11
E6	V3	V5	168.74	E22	1.24. V10	1.25. V13	1.26. 115.62
E7	V3	V6	74.58	E23	1.27. V10	1.28. V14	1.29. 88.92
E8	V4	V6	113.1	E24	1.30. V11	1.31. V12	1.32. 115.05
E9	V5	V6	124.37	E25	1.33. V11	1.34. V13	1.35. 80.28
E10	V5	V7	87.46	E26	1.36. V12	1.37. V13	1.38. 129.02
E11	V5	V8	117.44	E27	1.39. V12	1.40. V15	1.41. 88.9
E12	V6	V8	87.34	E28	1.42. V13	1.43. V14	1.44. 109.35
E13	V6	V9	128.7	E29	1.45. V13	1.46. V15	1.47. 123.87
E14	V7	V8	121.23	E30	1.48. V14	1.49. V15	1.50. 166.69
E15	V7	V10	93.69	E31	1.51. V14	1.52. V16	1.53. 103.51
E16	V8	V9	132.65	E32	1.54. V15	1.55. V16	1.56. 84.69

with $|N|$ = 6 and $|E|$ = 8 as shown in Fig. 2, where the edges and nodes are numbered in order so that each bit of a solution could be decoded to an edge. Figure 3 shows three coding schemes. The corresponding graphs for the three coding schemes are shown in Fig. 4. The number of 1 in code (b) is 4, and (b) is an infeasible string. The number of 1 in code (a) and code (c) is 5. In Fig. 4, after decoding (c), there is a loop (v1-v2-v4-v1) and isolated node $v5$. Binary string (a) is a feasible code; binary string (c) is not an infeasible code.

3.3.2 Search mechanism

In the initialization phase, for each food source, $|N|$ − *1* position is randomly selected and set to *1* and other positions are set to *0*. The calculation of fitness is calculated according to formula (8). If the binary string is a feasible code (Figs. 3a and 4a), the fitness value is $fit = \sum_{e \in E_T} w_e$. If the binary string is infeasible (Figs. 3c and 4c), the fitness is ∞.

$$fit = \begin{cases} \sum_{e \in E_T} w_e, & \text{feasible} \\ \infty, & \text{infeasible} \end{cases} \tag{8}$$

Employed bees use the following techniques when searching. A position is randomly chosen from the elements with 1 and denoted as i_1. Another position is randomly chosen from the elements with 0 and denoted as i_2. The values of positions i_1 and i_2 are interchanged. After such changes, the number of elements with 1 does not change. This technique can guarantee that the number of 1 in the binary string is constant. The fitness of the newly generated solution is calculated according to Eq. (8). The replacement strategy, described in detail in Section 3.3.3, is executed.

In the process of onlooker bee searching, the quantum computing technique introduced in Section 3.2 is used and improved. For the calculation of the average best position, we use the elite strategy. Food sources are sorted in order of fitness value from small to large. For the sorted food source, the mean value of the front half food source is calculated (formula (9)). Because the particle swarm algorithm and the artificial bee swarm algorithm are essentially different, the meaning of the variable has also changed. In order to apply to the artificial bee colony algorithm, the previous formulas (2)–(7) are modified. The new calculation methods are formulas (9)–(15).

$$nmb_j = \frac{\sum_{i=1}^{N/2} fc_{ij}}{N/2} \tag{9}$$

$$a = rand(0, 1) \tag{10}$$

$$p = a * fc_{ij} + (1-a) * gb_j \tag{11}$$

$$b = 1 - \frac{t}{2 * t_{\max}} \tag{12}$$

$$u = rand(0,1) \tag{13}$$

$$q_j = \begin{cases} p - b * \mid nmb - fc_{ij} \mid * \ln\left(\dfrac{1}{u}\right), & u \geq 0.5 \\[2ex] p + b * \mid nmb - fc_{ij} \mid * \ln\left(\dfrac{1}{u}\right), & u < 0.5 \end{cases} \tag{14}$$

$$p_j = \begin{cases} 1, & q_j \geq 0.5 \\ 0, & q_j < 0.5 \end{cases} \tag{15}$$

Here, N is the population size; fc_{ij} is the j-th component of the i-th smallest food source according to fitness; gb is the first small food source according to fitness; gb_j is the j-th component of gb; nmb is the mean after improvement; $Rand(0,1)$ is a function whose return value is a random decimal between [0, 1]; t is the current evolutionary generation; $t_{\max}$ the maximum evolutionary generation of the algorithm; b is called the contraction expansion coefficient, which gradually decreases with the iteration of the algorithm.

If p_j and the j-th component of the current food source are equal, nothing is done. Otherwise, a location, denoted as k, is randomly selected from the elements with $(1 - p_j)$. Then, the elements of position k and position j are interchanged. That is, the element at position j is assigned p_j, and the element at position k becomes $1 - p_j$. The above operation is the same as the operation of the employed bee, and it also ensures that the number of elements with 1 does not change. The search process for onlooker bees is shown in Algorithm 1.

Algorithm 1 OnlookerBees
1:   Select a food source randomly according to formula (1), which is labeled fc_i.
2:   Select a location at random for the selected food source, which is labeled j
3:   Use Equation (9)-(15) to calculate the value of p_j
4:   If p_j is not equal to fc_{ij} then
5:   a location, denoted as k, is randomly selected from the elements with $(1 - p_j)$.
6:   the elements of position k and position j are interchanged.
7:   End if

If no better food source is found in the search for employed bees and onlooker bees, the food source is not updated. If it is not updated after the set number of times, the food source is discarded. The scout bee will randomly generate a new food instead. The procedure is similar to the initialization step.

3.3.3 Replacement strategy

In the previous part of this article, we have mentioned that multiple solutions can be obtained in one calculation. Therefore, the first principle of our replacement strategy is to ensure the diversity of food sources. Under the guidance of such principles, if the newly found food source is better than the original food source, but is the same as any other food source, it will not be updated and will be discarded. In addition, as described in Section 3.3.1, the initial food source and new food source search may be infeasible. Therefore, another principle of our replacement strategy is to replace as much of the infeasible food source as possible.

After employed bee and onlooker bees search for new food sources, the specific algorithm for food source replacement is shown in Algorithm 2.

| **Algorithm 2** Replacement |
| --- |
| **1:**   The solution corresponding to the new food source is infeasible, return. Otherwise, turn to 2; |
| **2:**   If any other food source is not feasible, an infeasible food source is randomly selected and replaced, return. Otherwise, turn to 3; |
| **3:**   If the new food source is better than the original food source, the new food source replaces the original food source. |

To this position in this paper, based on quantum computing and artificial bee colony, the algorithm for solving the spanning tree of industrial wireless sensor networks has been introduced. We will use QABCST to represent this algorithm later. The main steps of QABCST are similar to the ABC algorithm described in Section 3.1. The flow chart of the QABCST algorithm is shown in Fig. 5.

The QABCST algorithm can generate multiple solutions in one calculation, and these solutions are backups of each other. The tree decoded from optimal solution can be used for data distribution or data collection. When the link failure causes the optimal tree to be unavailable, one of the backup solutions is used. Firstly, the

Table 3 Minimum spanning tree found by the Kruskal algorithm

ST	Weight	Edge series
Optimal solution	1326.409668	1,2,4,5,7,10,12,15,18,20,23,25,27,31,32
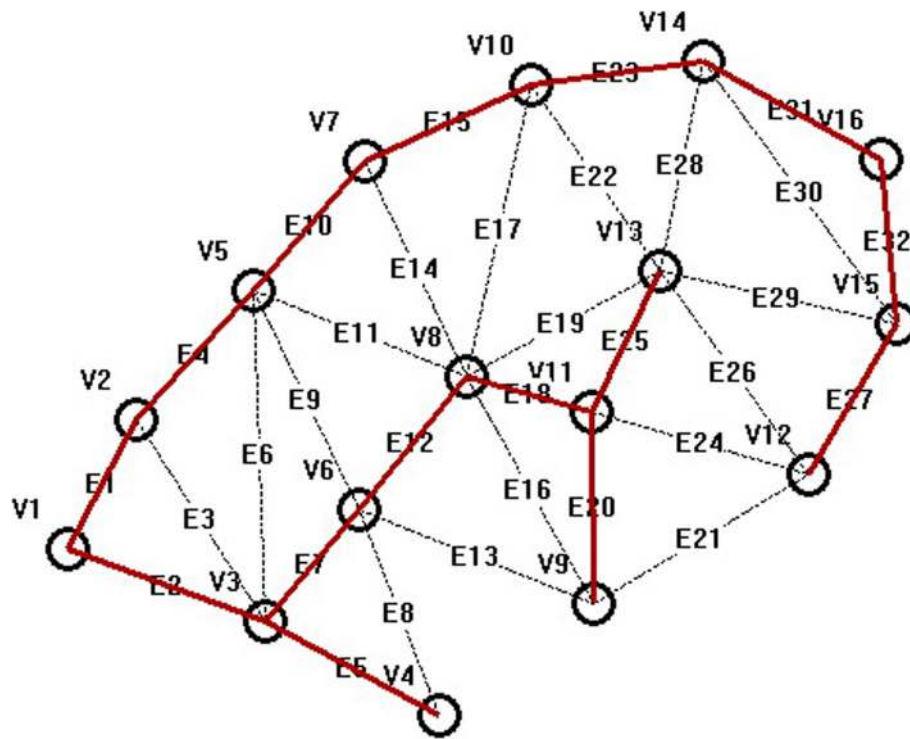
Fig. 7 Minimum spanning tree found by the Kruskal algorithm. The minimum spanning tree obtained by the Kruskal algorithm during performance simulation

candidate solutions containing the failed link can be marked as invalid. Then, choose the optimal one from the remaining solutions.

4 Experimental

In this section, simulation experiments are conducted on the newly proposed algorithm to verify the validity of our proposed algorithm.

The industrial wireless sensor network shown in Fig. 6 is used as an example to simulate. The

Table 4 Spanning tree found by BABC

ST	Weight	Edge series
1	1431.365601	1,7,8,10,11,15,18,20,24,25,27,28,29,31,32
2	1505.983398	1,3,5,7,10,12,13,15,18,23,24,28,30,31,32
3	1452.080566	1,4,5,7,8,10,13,18,23,24,25,26,27,28,31
4	1595.697266	2,4,7,8,9,10,12,16,17,19,20,21,22,23,32
5	1447.339722	1,2,5,7,9,10,18,19,20,23,25,26,27,28,31
6	1423.100586	1,2,5,7,8,10,12,15,18,20,21,23,27,28,31
7	1501.466919	1,7,8,9,10,15,17,18,20,23,24,26,27,28,32
8	1424.996460	1,5,7,10,15,18,19,20,21,23,24,25,26,27,32
9	1465.025146	2,4,8,9,10,12,13,14,15,18,23,25,27,31,32
10	1449.954712	1,2,4,5,7,10,12,16,19,22,25,27,28,31,32
Optimal solution	1326.409668	1,2,4,5,7,10,12,15,18,20,23,25,27,31,32

Table 5 Spanning tree found by QABCST

ST	Weight	Edge series
1	1647.062378	1,3,6,8,10,13,17,18,19,20,23,24,28,29,32
2	1668.069214	1,2,3,4,6,8,13,14,15,21,24,27,28,29,32
3	1411.012573	1,5,7,10,12,15,16,18,20,22,23,24,25,28,32
4	1463.207642	2,3,4,5,8,10,12,18,19,20,23,24,25,28,32
5	1424.814697	1,4,5,7,8,10,12,17,18,19,20,23,27,31,32
6	1435.052612	1,2,5,7,9,12,15,18,19,20,21,23,27,31,32
7	1442.719849	1,2,3,4,5,7,12,15,18,20,22,23,24,29,32
8	1452.438354	1,2,5,7,10,14,16,18,22,23,25,27,29,31,32
9	1525.517456	1,4,5,10,11,12,15,16,18,22,23,26,27,30,32
10	1346.926025	1,4,5,7,10,12,15,18,20,22,23,24,25,27,32
11	1396.752075	1,3,4,5,7,8,10,12,18,19,20,23,24,25,32
12	1432.078125	2,4,5,7,9,12,15,18,19,21,23,25,27,31,32
13	1476.490845	1,2,3,4,5,7,10,12,15,20,23,24,26,29,32
14	1383.834351	1,4,5,7,9,12,15,18,20,22,23,24,25,27,32
15	1360.906860	1,3,4,5,7,10,12,18,19,20,23,25,27,31,32
16	1443.100220	2,3,4,5,7,10,12,15,18,20,23,24,29,31,32
17	1495.755859	1,4,5,11,12,14,17,18,21,23,25,27,28,31,32
18	1576.713135	1,2,4,6,8,12,13,14,15,18,21,24,27,28,32
19	1455.923950	1,4,5,8,10,11,12,15,16,18,19,22,23,27,32
20	1372.569336	1,4,5,7,8,10,12,18,19,20,23,25,27,29,32
Optimal solution	1326.409668	1,2,4,5,7,10,12,15,18,20,23,25,27,31,32

industrial wireless sensor network, represented as G, has 16 nodes and 32 edges. These nodes are deployed in a 500 × 400 rectangular area. Table 1 shows the coordinates of the nodes.The weight function on each edge is the Euclidean distance between two nodes. Table 2 lists the correspondence between edges and nodes, as well as the weight of the edges.

Our new proposed algorithm will be compared with Kruskal algorithm and BABC [40] algorithm. BABC uses the basic artificial bee colony algorithm to solve the minimum spanning tree. It can also obtain multiple spanning tree construction schemes in one calculation. The population size is 20, and the algorithm loops 3000 times. All the algorithms have been coded using C++ in VS 2010. We run all the configurations on an Intel (R) Core (TM) i7-2600 CPU @ 3.40 GHz with 8.00 GB RAM in the Windows 10 Operation System.

5 Results and discussion

Each of the three algorithms runs one time, and the experimental results are compared. First, the Kruskal algorithm is used to handle this example. The Kruskal algorithm can only obtain one solution. The obtained ST (spanning tree) is shown in Table 3. The resulting spanning tree has a weight of 1326.409668. The

Table 6 Average of the 10 minimum spanning tree weights

Kruskal	BABC	QABCST
1326.409668	1328.747522	1327.755135

resulting minimum spanning tree is shown in Fig. 7. The Kruskal algorithm's calculation time is about 0.05 s.

Tables 4 and 5 are the results obtained after the BABC and QABCST algorithms are run once. As can be seen from Tables 4 and 5, the QABCST algorithm has got more solutions. This is due to the diversity of our replacement strategies. The reason for this result is the persistence of diversity in QABCST's replacement strategy. Figure 8 is an illustration of the other one (10th in Table 5) of the solutions obtained by the QABCST algorithm.

The following is a comparison of the average performance of the QABCST and BABC algorithms running multiple times. The algorithm is performed 10 times independently to obtain its average performance. Table 6 shows the average of the 10 minimum spanning tree weights. The result of Table 6 about Kruskal is the result of running once. As can be seen from the table, the QABCST algorithm has better performance than BABC.
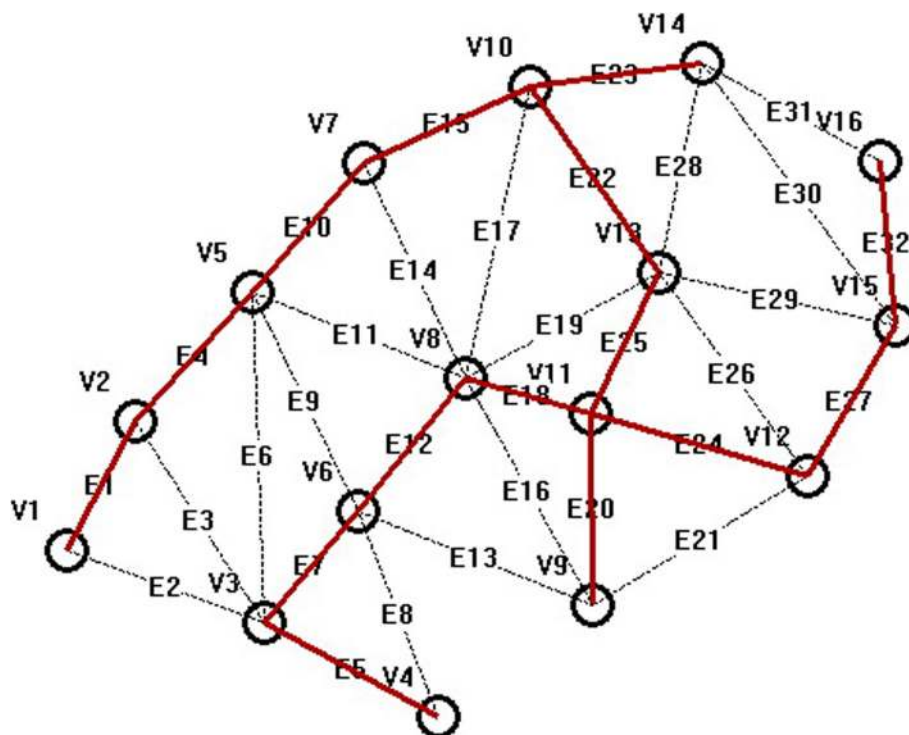
Fig. 8 Another spanning tree found by the QABCST algorithm. Another spanning tree obtained by the QABCST algorithm during performance simulation

6 Conclusion

With the rapid development of wireless sensor networks, wireless devices are increasingly deployed in industrial environments. Compared with common wireless sensor networks, industrial wireless sensor networks have higher requirements for the determinism and reliability of data communications. Therefore, it is particularly important to design reasonable mechanisms for the data aggregation/distribution of IWSNs to ensure the certainty and reliability of the data transmission process. Data collection and distribution in industrial wireless sensor networks can be described using the spanning tree problem in graph theory. Existing classical algorithms such as Kruskal and Prim algorithm can only get one solution at a time. In order to improve reliability, industrial application scenarios need to provide multiple solutions for mutual backup. This paper improves the artificial bee colony algorithm based on the idea of quantum computing and proposes a spanning tree construction algorithm for industrial wireless sensor networks based on quantum artificial bee colony. Finally, our algorithm was verified by experiments. The experimental results show that the algorithm can achieve better performance and can obtain more solutions at the same time. Future work includes increasing a priori knowledge of the network structure to improve search efficiency.

Abbreviations
MST: Minimum spanning tree; WSN: Wireless sensor networks; IWSN: Industrial wireless sensor networks; ABC: Artificial bee colony; QPSO: Quantum particle swarm optimization; QABC: Quantum artificial bee colony; ST: Spanning tree

Acknowledgements
Not applicable.

Authors' contributions
The first author conducted the experiments and wrote the first draft of the paper. Other co-authors helped to revise the paper and polished the paper. All authors read and approved the final manuscript.

Authors information
Not applicable.

Availability of data and materials
Not applicable.

Competing interests
The authors declare that they have no competing interests.

References
1. H. Cheng, Z. Su, N. Xiong, Y. Xiao, Energy-efficient node scheduling algorithms for wireless sensor networks using Markov random field model. Inf. Sci **329**(C), 461–477 (2016)
2. X. Jiang, Z. Fang, N.N. Xiong, et al., Data fusion-based multi-object tracking for unconstrained visual sensor networks. IEEE. Access. **6**, 13716–13728 (2018)
3. J. Liu, J. Wan, Q. Wang, P. Deng, K. Zhou, Y. Qiao, A survey on position-based routing for vehicular ad hoc networks. Telecommun. Syst. **62**(1), 15–30 (2016)
4. M. Wu, L. Tan, N. Xiong, Data prediction, compression, and recovery in clustered wireless sensor networks for environmental monitoring applications. Inf. Sci. **329**(SI), 800–818 (2016)
5. Y. Liu, K. Ota, K. Zhang, et al., QTSAC: an energy-efficient MAC protocol for delay minimization in wireless sensor networks. IEEE Access. **6**, 8273–8291 (2018)
6. V. C G, G. P H, Industrial wireless sensor networks: challenges, design principles, and technical approaches. IEEE Trans. Ind. Electron. **56**(10), 4258–4265 (2009)
7. D.E. Boubiche, A.S. Pathan, J. Lloret, H. Zhou, S. Hong, S.O. Amin, M.A. Feki, Advanced industrial wireless sensor networks and intelligent IoT. IEEE. Commun. Mag **56**(2), 14–15 (2018)
8. D.V. Queiroz, M.S. Alencar, R.D. Gomes, I.E. Fonseca, C. Benavente-Peces, Survey and systematic mapping of industrial wireless sensor networks. J. Netw. Comput. Appl. **97**, 96–125 (2017)
9. M. Gidlund, S. Han, E. Sisinni, A. Saifullah, U. Jennehag, From industrial wireless sensor networks to industrial Internet of things. IEEE. Trans. Ind. Inf. **14**(5), 2194–2198 (2018)
10. T. Liang, B. Zeng, J. Liu, L. Ye, C. Zou, An unsupervised user behavior prediction algorithm based on machine learning and neural network for smart home. IEEE. Access. **6**, 49237–49247 (2018)
11. J. Liu, J. Wan, B. Zeng, Q. Wang, H. Song, M. Qiu, A scalable and quick-response software defined vehicular network assisted by mobile edge computing. IEEE. Commun. Mag. **55**(7), 94–100 (2017)
12. Cheffena, industrial wireless sensor networks: channel modeling and performance evaluation. EURASIP. J. Wirel. Commun. Netw. **297** (2012)
13. C. Wang, J. Li, B. Wang, Face synthesis based on parts-based sparse component analysis face representation. Optik. Int. J. Light. Electron. Opt **140**, 843–852 (2017)
14. M. Kumar, R. Tripathi, S. Tiwari, QoS guarantee towards reliability and timeliness in industrial wireless sensor networks. Multimed. Tools. Appl. **77**(4), 4491–4508 (2018)
15. S. Wu, W. Chou, J. Niu, M. Guizani, Delay-aware energy-efficient routing towards a path-fixed mobile sink in industrial wireless sensor networks. SENSORS. **18**(3), 899 (2018)
16. J. Tan, A. Liu, M. Zhao, H. Shen, M. Ma, Cross-layer design for reducing delay and maximizing lifetime in industrial wireless sensor networks. EURASIP J. Wirel. Commun. Netw. **50** (2018)
17. M. Huang, A. Liu, N.N. Xiong, et al., A low-latency communication scheme for mobile wireless sensor control systems. IEEE. Trans. Syst. Man. Cybern. Syst. **49**(2), 317–332 (2019)
18. W. Zhang, J. Chang, F. Xiao, et al., Design and analysis of a persistent, efficient, and self-contained WSN data collection system. IEEE. Access. **7**, 1068–1083 (2019)
19. J. Tan, W. Liu, T. Wang, et al., An adaptive collection scheme-based matrix completion for data gathering in energy-harvesting wireless sensor networks. IEEE. Access. **7**, 6703–6723 (2019)
20. H. Zheng, W. Guo, N. Xiong, A Kernel-based compressive sensing approach for mobile data gathering in wireless sensor network systems. IEEE. Trans. Syst. Man. Cybern. Syst. **48**(12), 2315–2327 (2018)
21. X. He, S. Liu, G. Yang, et al., Achieving efficient data collection in heterogeneous sensing WSNs. IEEE. Access. **6**, 63187–63199 (2018)
22. K. Huang, Q. Zhang, C. Zhou, N. Xiong, Y. Qin, An efficient intrusion detection approach for visual sensor networks based on traffic pattern learning. IEEE Trans. Syst. Man. Cybern. Syst **47**(10), 2704–2713 (2017)
23. H. Cheng, Y. Chen, N. Xiong, et al., Layer-based data aggregation and performance analysis in wireless sensor networks. J. Appl. Math. **502381** (2013)
24. S. Montero, J. Gozalvez, M. Sepulcre, Neighbor discovery for industrial wireless sensor networks with mobile nodes. Comput. Commun. **111**, 41–55 (2017)
25. Z. Zheng, J. Li, Optimal chiller loading by improved invasive weed optimization algorithm for reducing energy consumption. Energ. Buildings. **161**, 80–88 (2018)
26. J. Li, Q. Pan, S. Xie, An effective shuffled frog-leaping algorithm for multi-objective flexible job shop scheduling problems. Appl. Math. Comput. **218**(18), 9353–9371 (2012)

27. H. Sang, Q. Pan, J. Li, et al., Effective invasive weed optimization algorithms for distributed assembly permutation flowshop problem with total flowtime criterion. Swarm. Evol. Comput. **44**(6), 64–73 (2019)

28. Z. Zheng, J. Li, P. Duan, Optimal chiller loading by improved artificial fish swarm algorithm for energy saving. Math. Comput. Simul. **155**(SI), 227–243 (2019)

29. H. Sang, Q. Pan, P. Duan, et al., An effective discrete invasive weed optimization algorithm for lot-streaming flowshop scheduling problems. J. Intell. Manuf. **29**(6), 1337–1349 (2018)

30. J. Zhao, Y. Qin, D. Yang, J. Duan, Reliable graph routing in industrial wireless sensor networks. Int. J. Distrib. Sens. Netw. **9**(12), 758217 (2013)

31. J. Akerberg, M. Gidlund, T. Lennvall, J. Neander, M. Bjorkman, Efficient integration of secure and safety critical industrial wireless sensor networks. EURASIP. J. Wirel. Commun. Netw. **100** (2011)

32. J. Li, P. Duan, H. Sang, et al., An efficient optimization algorithm for resource-constrained steelmaking scheduling problems. IEEE. Access. **6**, 33883–33894 (2018)

33. C. Pei, Y. Xiao, W. Liang, X. Han, Trade-off of security and performance of lightweight block ciphers in Industrial Wireless Sensor Networks. EURASIP. J. Wirel. Commun. Netw. **117** (2018)

34. J. Li, Q. Pan, P. Duan, An improved artificial bee colony algorithm for solving hybrid flexible flowshop with dynamic operation skipping. IEEE. Trans. Cybern. **46**(6), 1311–1324 (2016)

35. K.Z. Gao, P.N. Suganthan, Q.K. Pan, et al., Artificial bee colony algorithm for scheduling and rescheduling fuzzy flexible job shop problem with new job insertion. Knowl. Based. Syst. **109**, 1–16 (2016)

36. Y.Y. Han, Q.K. Pan, J.Q. Li, et al., An improved artificial bee colony algorithm for the blocking flowshop scheduling problem. Int. J. Adv. Manuf. Technol. **60**, 1149–1159 (2012)

37. Y. Han, J.J. Liang, Q. Pan, et al., Effective hybrid discrete artificial bee colony algorithms for the total flowtime minimization in the blocking flowshop problem. Int. J. Adv. Manuf. Technol. **67**, 397–414 (2013)

38. J. Li, Q. Pan, Solving the large-scale hybrid flow shop scheduling problem with limited buffers by a hybrid artificial bee colony algorithm. Inf. Sci. **316**, 487–502 (2015)

39. Jun S, Wenbo X, Bin F, in Proceedings of 2005 IEEE International Conference on Systems, Man and Cybernetics. Adaptive parameter control for quantum-behaved particle swarm optimization on individual level (IEEE 2005), pp. 3049-3054.

40. X. Zhang, X. Zhang, A binary artificial bee colony algorithm for constructing spanning trees in vehicular ad hoc networks. Ad. Hoc. Networks. **58**(4), 198–204 (2017)

Publisher's Note