RFID Reader Anticollision Based on Distributed Parallel Particle Swarm Optimization

Bin Cao, *Member, IEEE*, Yu Gu, *Member, IEEE*, Zhihan Lv, *Senior Member, IEEE*, Shan Yang, Jianwei Zhao, and Yujie Li, *Member, IEEE*

***Abstract*—The deployment of a very large number of readers in a limited space may increase the probability of collision among radio-frequency identification (RFID) readers and reduce the dependability and controllability of Internet-of-Things (IoT) systems. Intelligent computing technologies can be used to realize intelligent management by scheduling resources to circumvent collision issues. In this article, an improved RFID reader anticollision model is constructed by modifying the measure index, introducing a constraint function, and simultaneously considering collisions among readers and between readers and tags. The dense deployment of large numbers of readers increases the number of variables to be encoded, resulting in a high-dimensional problem that cannot be effectively and efficiently solved by traditional algorithms. Accordingly, distributed parallel cooperative co-evolution particle swarm optimization (DPCCPSO) is proposed. The inertia weight and learning factors are adjusted during evolution, and an improved grouping strategy is presented. Moreover, various combinations of random number generation functions are tested. For improved efficiency, DPCCPSO is implemented with distributed parallelism. Experimental verification shows that the proposed novel algorithm exhibits superior performance to existing state-of-the-art algorithms, particularly when numerous RFID readers are deployed.**

***Index Terms*—Cooperative co-evolution, grouping method, parameter adjustment, radio-frequency identification (RFID) reader anticollision.**

Manuscript received July 17, 2020; revised September 23, 2020; accepted October 7, 2020. Date of publication October 23, 2020; date of current version February 19, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61976242, Grant 61876059, and Grant 61902203; in part by the Opening Project of Guangdong Province Key Laboratory of Computational Science at the Sun Yat-sen University under Grant 2018002; in part by the Ministry of Science and Technology of the People's Republic of China under Grant 2017YFB1400100; and in part by the Key Research and Development Plan-Major Scientific and Technological Innovation Projects of Shandong Province under Grant 2019JZZY020101. *(Corresponding authors: Yu Gu; Zhihan Lv.)*

Bin Cao, Shan Yang, and Jianwei Zhao are with the State Key Laboratory of Reliability and Intelligence of Electrical Equipment, Hebei University of Technology, Tianjin 300130, China, and also with the School of Artificial Intelligence, Hebei University of Technology, Tianjin, 300401, China (e-mail: caobin@scse.hebut.edu.cn; 201532103005@stu.hebut.edu.cn; 201422102003@stu.hebut.edu.cn).

Yu Gu is with the School of AutoMation, Guangdong University of Petrochemical Technology, Maoming 525000, China, also with the Beijing Advanced Innovation Center for Soft Matter Science and Engineering, Beijing University of Chemical Technology, Beijing 100029, China, and also with the Department of Biochemistry, Chemistry and Pharmacy, Goethe-University, 60438 Frankfurt, Germany (e-mail: guyu@mail.buct.edu.cn).

Zhihan Lv is with the School of Data Science and Software Engineering, Qingdao University, Qingdao 266071, China (e-mail: lvzhihan@qdu.edu.cn).

Yujie Li is with the School of Information Engineering, Yangzhou University, Yangzhou 225127, China (e-mail: yzyjli@gmail.com).

Digital Object Identifier 10.1109/JIOT.2020.3033473

I. Introduction

RADIO-FREQUENCY identification (RFID) [1], [2] is one of the most important components of the Internet of Things (IoT) [3]–[5]. Through intelligent edge computing and artificial intelligence technologies, RFID systems can achieve intelligent monitoring and management and are increasingly used in vehicle networks [6]–[8].

Unmanned delivery vehicles have potential applications in libraries, express delivery, and other industries. The use of RFID to efficiently and effectively manage a parking lot of unmanned delivery vehicles is an important issue in the application of this technology [4].

The dense deployment of numerous RFID readers within the same area presents a large probability of reader collision. These collisions can reduce the identification rate and affect the reliability of the system [9]. In addition, the large number of variables makes the corresponding optimization problem difficult for many algorithms based on resource scheduling strategies and soft computing methods. Therefore, the motivations for this article are twofold: 1) to study and improve the RFID anticollision model to better facilitate simulation and 2) to propose an algorithm to effectively and efficiently optimize the high-dimensional problem.

For the first motivation, we adopt the RFID reader-to-reader anticollision model proposed by Li *et al.* [10], whereby the RFID reader anticollision problem is transformed into a resource scheduling problem. However, this model considers only reader-to-reader collisions and has limited practical application value. Accordingly, we improve this model by introducing reader-to-tag collisions, modifying the measure index and introducing a constraint condition.

Then, to solve the model via metaheuristic algorithms [11], based on the Spark-based parallel cooperative co-evolution particle swarm optimization (PCCPSO) [12], a distributed PCCPSO (DPCCPSO) is proposed. Specifically, the inertia weight and learning factors are nonlinearly adjusted to prevent the solution from falling into local optima, and several random number generation functions are investigated for the updating procedure. Variables are grouped using an improved strategy that combines improved differential grouping (DG2) [13] and dynamic grouping. Additionally, the message passing interface (MPI) is used to implement DPCCPSO in a distributed parallel environment to reduce the computational time and increase the efficiency of the algorithm.

The organization of this article is as follows. Section II discusses related studies. In Section III, the RFID reader collision

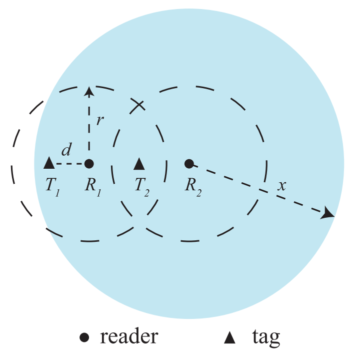

Fig. 1. Instance of the collision problem.

problem and the improved anticollision model are presented. The proposed DPCCPSO is described in Section IV. The proposed algorithm is evaluated using numerical experiments in Section V. The proposed scheme is discussed in Section VI. Finally, the conclusions are presented in Section VII.

II. Related Studies

A. Methods for Solving RFID Reader Collisions

Algorithms based on coordinated control have been developed to solve the collision problem. The core concept of these algorithms is the use of the "listen before talk" mechanism to monitor the state of neighboring readers to reduce collisions. In [14], a neighbor friendly reader anticollision (NFRA) protocol was proposed for the dense deployment of readers. However, the NFRA protocol is a single-channel protocol.

The geometric distribution reader anticollision (GDRA) protocol, which is based on the NFRA protocol, was subsequently developed [15]; it considers multiple communication channels and uses the Sift geometric probability distribution function to select a possible frequency channel for each reader in a contention frame. Nevertheless, the GDRA sets the output power of each reader to its maximum value.

Other algorithms reduce collisions by resource allocation optimization. An anticollision model that simultaneously considers reader positions, time slots, frequency channels, and the transmitting power was proposed [10]. The artificial immune network algorithm with hybrid encoding for resource allocation (AINetHE-RA) was proposed to tackle this model. However, for relatively large numbers of readers, e.g., more than 20, AINetHE-RA performs poorly. Some scholars have used swarm intelligence-based algorithms. Chen *et al.* [16] developed a deployment optimization model that considers reader positions and the transmitting power and proposed a novel multiswarm optimizer (PS^{2}O) to solve this model.

B. Particle Swarm Optimization

Particle swarm optimization (PSO) [17]–[19] is relatively simple compared to other algorithms and thus has potential applications in many fields [20]–[23]. A cooperative coevolving PSO (CCPSO) [24] was proposed that integrates random grouping [25], [26] and adaptive weighting to effectively optimize high-dimensional optimization problems. In addition to random grouping, other methods include differential grouping (DG) [13], DG2 [13], graph-based DG [27]–[29], and dynamic grouping [30]. In [30], PSO was improved by incorporating dynamic grouping into CCPSO, denoted as CCPSO2. PCCPSO, which was proposed in [12], integrates global PSO (GPSO) and local PSO (LPSO) while incorporating some of the advantages of CCPSO2. PCCPSO has been verified by experiments and exhibits superior performance for optimizing high-dimensional (or large-scale) problems.

III. RFID Reader Anticollision Model

A. Collision Problem

The collision problem consists of two situations: 1) signal collision between readers and tags (reader-to-tag collision) and 2) signal collision among readers (reader-to-reader collision). Fig. 1 shows two readers R_1 and R_2 and two tags T_1 and T_2. r is the interrogation radius of R_1, d is the distance between R_1 and T_1, and x is the interference radius of R_2. T_2 is located in the overlap of the reading regions of R_1 and R_2. If R_1 and R_2 simultaneously send signals to T_2, a reader-to-tag collision will cause communication between the two readers and the tag to fail. In addition, T_1 is located in R_1's interrogation range and R_2's interference range. If R_1 interrogates T_1 and R_2 is also activated, R_1 will be disturbed. Thus, the interrogation radius of the readers will be reduced. If the interference from R_2 is too high to make r less than d, the reading/writing operation of R_1 will not succeed.

B. Anticollision Model

In this article, reader-to-reader and reader-to-tag collisions are simultaneously considered based on the RFID reader-to-reader anticollision model proposed by Li *et al.* [10]. In the optimization problem of RFID reader deployment and scheduling, we consider two objectives: 1) *throughput* and 2) *load balance*. With respect to actual application scenarios, we also propose a constraint condition for reader positions.

1) Throughput: The decision vector comprises three components, as in [10]: 1) the frequency channels; 2) the transmitting power at each time slot; and 3) the position of each reader. The number of variables N_{var} is given by

$$N_{\text{var}} = 2(N_{\text{slot}} \cdot N_{\text{reader}} + N_{\text{reader}}) \tag{1}$$

where N_{reader} denotes the reader number and N_{slot} denotes the time slot number. To communicate with tag k at time slot t, the signal-to-interference-plus-noise ratio of reader i $\text{SINR}_i(t, x_{i,k})$ should not be less than the desired minimum value $\text{SINR}_{\min}$, here, $x_{i,k}$ is the distance from reader i to tag k. Correspondingly, at time slot t, the radius of reader i's interrogation area is formulated as

$$r_i^t = \arg\max_{x_{i,k}} SINR_i\big(t, x_{i,k}\big) \geq SINR_{\min} \tag{2}$$

where r_i^t represents the interrogation range of reader i at time slot t. The detailed formula for $\text{SINR}_i(t, x_{i,k})$ [10] is given in (3), shown at the bottom of the next page, where $\alpha_{b\omega}$, β_{mask}, E_{tag}, G_T, G_R, P_0, h, and γ are parameters, $P_i(t)$ denotes the working power of reader i at time slot t, $\omega_i(t)$ denotes whether or not reader i is working at time slot t (1 for operational and 0 for idle), $CH_i(t)$ denotes the working channel of reader i at time slot t, $d_{i,j}$ denotes the distance between readers i and j, and Noise_i denotes the noise power of reader i

In this article, *Throughput* refers to the quantity of tags effectively interrogated by readers in an identification round, which includes N_{slot} time slots. This objective function is calculated as follows:

$$\text{maximize } f_1 = \text{Throughput} = \sum_{k=1}^{N_{\text{tag}}} \delta_k \tag{4}$$

where N_{tag} denotes the tag number, the locations of which are randomly determined, and δ_k represents the identification status of tag k in the identification round, which is formulated as follows:

$$\delta_k = \begin{cases} 1, & C_k^t = 1, \quad \text{s.t. } \exists\, t \in \mathbf{SS} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where $\mathbf{SS} = \{1, 2, \ldots, N_{\text{slot}}\}$, and C_k^t indicates the identification status of tag k at time slot t, which is formulated as follows:

$$C_k^t = \begin{cases} 1, & x_{i,k} \le r_i^t, x_{j,k} > r_j^t, \ \text{s.t. } \exists\, i \in \mathbf{SR} \wedge \forall j \in \mathbf{SR}, j \neq i \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $\mathbf{SR} = \{1, 2, \ldots, N_{\text{reader}}\}$, $x_{i,k}$ ($x_{j,k}$) denotes the distance from reader i (j) to tag k, and r_i^t (r_j^t) denotes the interrogation range of reader i (j) at time slot t.

2) Load Balance: In the RFID system network, a balanced reader cost for monitoring tags results in better performance than the unbalanced case [31]. The use of the objective load balance in the optimization model of the RFID system deployment in [16] yielded good deployment results in experiments. Therefore, load balance is also considered as an objective in the model proposed here, which is formulated as

$$\text{minimize } f_2 = L = \sqrt{\sum_{i=1}^{N_{\text{reader}}} \left(N_i - \frac{\text{Throughput}}{N_{\text{reader}}} \right)^2} \tag{7}$$

where N_i denotes the number of tags served by reader i in an identification round and is calculated as follows:

$$N_i = \sum_{k=1}^{N_{\text{tag}}} \varphi_{i,k} \tag{8}$$

where $\varphi_{i,k}$ indicates whether or not reader i can effectively identify tag k in the identification round, which is determined as follows:

$$\varphi_{i,k} = \begin{cases} 1, & x_{i,k} \le r_i^t, x_{j,k} > r_j^t, \ \text{s.t. } \exists\, t \in \mathbf{SS}\ \forall j \in \mathbf{SR}, j \neq i \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

3) Constraint Condition: In the model proposed by Li *et al.* [10], reader locations can be randomly selected in the deployment region. However, in some real situations, reader locations are fixed or selected from a set of predefined candidates. We consider these actual application scenarios of the RFID system by restricting the reader locations.

Assume that the predefined set of candidate locations is as follows:

$$\mathbf{R} = \left\{ (y_1, z_1), (y_2, z_2), \ldots, \left(y_p, z_p\right) \right\} \tag{10}$$

where p is the number of candidate locations of set $\mathbf{R}$.

Then, the constraint condition on the reader locations is given as follows:

$$\left(\text{pos}_{2\times i-1}, \text{pos}_{2\times i}\right) \in \mathbf{R} \quad \forall\, i = 1, 2, \ldots, N_{\text{reader}} \tag{11}$$

where $(\text{pos}_{2\times i-1}, \text{pos}_{2\times i})$ is the deployment location of reader i.

4) Formulation of the RFID Reader Anticollision Model: Considering the two objectives *throughput* and *load balance*, we can obtain the following:

$$\text{maximize } F = w_1 f_1 + w_2 f_2 \tag{12}$$

where w_1 and w_2 are the weight factors satisfying $w_1 > 0$, $w_2 < 0$, and $w_1 + |w_2| = 1$.

Considering the restriction imposed on the reader locations, the objective function of the RFID reader anticollision model is formulated as follows:

$$\text{maximize } F = \sum_{m=1}^{2} w_m f_m$$
$$\text{s.t.} \begin{cases} w_1 > 0, w_2 < 0 \\ w_1 + |w_2| = 1 \\ \left(\text{pos}_{2\times i-1}, \text{pos}_{2\times i}\right) \in \mathbf{R} \\ i = 1, 2, \ldots, N_{\text{reader}}. \end{cases} \tag{13}$$

IV. Distributed Parallel Cooperative Co-Evolution PSO

A. Cooperative Co-Evolution PSO

In this article, GPSO [32] and LPSO [33] from PCCPSO are combined. GPSO operates under the guidance of the global optimal information and exhibits fast convergence when solving unimodal problems, but it can easily fall into local optima. LPSO operates under the guidance of the neighborhood optimal information and circumvents falling into local optima, but it has a slow convergence speed. A strategy that combines the advantages of the two algorithms is used. The detailed combination strategy is presented in the work of Cao *et al.* [12]. The following equations are used to update the velocity and position within GPSO:

$$\begin{aligned} v_i\left(N_g + 1\right) = {} & \omega v_i\left(N_g\right) + c_1 \text{rand}_1 \left(p_i^h\left(N_g\right) - p_i\left(N_g\right) \right) \\ & + c_2 \text{rand}_2 \left(p^g\left(N_g\right) - p_i\left(N_g\right) \right) \end{aligned} \tag{14}$$

$$\text{SINR}_i\left(t, x_{i,k}\right) = \frac{\alpha_{b\omega} E_{\text{tag}} P_i(t) G_T G_R \left(P_0 x_{i,k}{}^{-\gamma}\right)^2}{h G_T G_R \sum_{j=1, j\neq i}^{N_{\text{reader}}} \omega_j(t) \beta_{\text{mask}}\left(\left|CH_i(t) - CH_j(t)\right|\right) P_0 d_{i,j}{}^{-2\gamma} + \text{Noise}_i} \tag{3}$$

$$p_i(N_g + 1) = p_i(N_g) + v_i(N_g + 1) \tag{15}$$

where N_g represents the current generation number, ω denotes the inertia weight, c_1 and c_2 denote coefficients, p_i^h denotes particle i's personal best position, p^g denotes the global best position, p_i denotes particle i's current position, v_i denotes particle i's current velocity, and rand$_1$ and rand$_2$ are real numbers randomly generated in the range of [0, 1].

Chen *et al.* [34] tested three strategies to decrease the inertia weight of PSO, among which nonlinearly decreasing the inertia weight was experimentally shown to be the best strategy. Thus, we adopt this strategy and employ similar decreasing or increasing forms for w, c_1, and c_2, respectively, the formulas of which are as follows:

$$\omega = \left(\omega^i - \omega^f\right) \times \left(\frac{N_t}{N_T}\right)^2 + 2\left(\omega^f - \omega^i\right) \times \frac{N_t}{N_T} + \omega^i \tag{16}$$

$$c_1 = \left(c_1^i - c_1^f\right) \times \left(\frac{N_t}{N_T}\right)^2 + 2\left(c_1^f - c_1^i\right) \times \frac{N_t}{N_T} + c_1^i \tag{17}$$

$$c_2 = \left(c_2^i - c_2^f\right) \times \left(\frac{N_t}{N_T}\right)^2 + 2\left(c_2^f - c_2^i\right) \times \frac{N_t}{N_T} + c_2^i \tag{18}$$

where N_t and N_T denote the current and overall numbers of fitness evaluations (FEs), respectively, and ω^i, ω^f, c_1^i, c_1^f, c_2^i, and c_2^f are six constants.

During optimization, the particle velocities and positions will gradually tend toward those of the global optimal particle, which may cause the population to fall into a local optimum. To address this issue, we reset the particle position and velocity [35]. After updating the particle positions, the distances of each particle position to the global optimal position are calculated. If the distance between p_i and p_g is smaller than the specified threshold σ, the particle position and velocity will be reset. The distance between p_i and p_g is calculated as follows:

$$d_i = \left|p_i - p_g\right|, i = (1, 2, \ldots, \mathrm{NP}) \tag{19}$$

where NP is the number of particles. If $d_i < \sigma$, the position and velocity of particle i will be reset as follows:

$$v_i = \mathrm{rand}_3 \times (v_{\max} - v_{\min}) \tag{20}$$

$$p_i = \mathrm{rand}_4 \times (B_U - B_L) \tag{21}$$

where $v_{\max}$ ($v_{\min}$) denotes the maximum (minimum) particle velocity, B_U (B_L) denotes the upper (lower) bound of the particle position, and rand$_3$ and rand$_4$ are random numbers within [0, 1].

The following formula is used to update the particle position [30] within the LPSO algorithm:

$$p_i(N_g + 1) = \begin{cases} p_i^h(N_g) + \mathrm{rand}_5\left|p_i^h(N_g) - l_i(N_g)\right|, & \text{if rand} \le th_p \\ l_i(N_g) + \mathrm{rand}_6\left|p_i^h(N_g) - l_i(N_g)\right|, & \text{otherwise} \end{cases} \tag{22}$$

where l_i is the optimal position in the ith particle's neighborhood, th$_p$ is the threshold, and rand$_5$ and rand$_6$ are random numbers within [0, 1].

After the position of each particle has been updated by the GPSO and LPSO procedures, it is determined if the updated location of each reader belongs to the predefined candidate location set. If the new position satisfies the constraint, the fitness of the new position is evaluated; otherwise, the updated location will be changed to the closest location in the candidate location set before evaluation. Based on the fitness value of each new position, the corresponding individual optimum and global optimum will be checked and updated.

B. Improved Grouping Method

The grouping method in a high-dimensional optimization problem is a key factor that affects the performance of the optimization algorithm. The DG2 method is a modified DG method in which decision variables are grouped based on the interaction analysis. The magnitude of the roundoff error may differ from component to component for some imbalanced functions. In the process of detecting the interaction between each pair of variables, DG2 dynamically calculates a threshold parameter instead of using a single global threshold parameter that is predefined by the user. That is, DG2 is parameter-free. DG2 is more accurate than DG and performs well for imbalanced functions. CCPSO2 is used to incorporate dynamic grouping based on a predefined set of several possible group size values. At each generation, the algorithm first determines whether the group size needs to be changed. If the evolution of the population is stagnant, a new group size is randomly selected from the specified group size set and used to group the decision variables. Otherwise, the decision variables are randomly grouped, with the group size remaining the same. The experimental result proves that a combination of reasonable group sizes increases the effectiveness of the algorithm.

In CCPSO2, at each generation, the random grouping method is utilized, but its grouping accuracy is not high. Though the grouping accuracy of DG2 is higher than that of the random grouping method, it may not be completely correct, and some errors and inaccuracy remain. In addition, if the DG2 method is used to group variables, the grouping information remains unchanged during the entire optimization process. Therefore, misclassified interacting variables cannot be considered within the same group during the entire optimization process. Therefore, we combine DG2 and dynamic grouping methods to obtain the advantages of both methods.

The main workflow of the improved grouping strategy is given as follows.

1) Before the iterative optimization, DG2 groups the decision variables to produce the following initial grouping result: $\mathbf{G} = \{\mathbf{G}_1, \mathbf{G}_2, \ldots, \mathbf{G}_{n_{\mathrm{ini}}}\}$, where n_{ini} is the initial number of groups.
2) During optimization, the group size s is chosen randomly from a set $\mathbf{S}$, (where $\mathbf{S}$ contains several possible group size values).
3) If $n_{\mathrm{ini}} > \mathrm{Dimension}/s$, merge $\mathbf{G}$ according to Algorithm 1; otherwise, the population is optimized according to $\mathbf{G}$.
4) If the evolutionary degree of the population is not significant, the procedure is repeated from step 2); otherwise, it is repeated from step 3).

Algorithm 1 randomly arranges $\mathbf{G}_1, \mathbf{G}_2, \ldots, \mathbf{G}_{n_{\mathrm{ini}}}$ before merging. Therefore, a different merged grouping result may be obtained at each merging. As the grouping results of the

Algorithm 1: Merge Groups

```
  Input: initial grouping result $\mathbf{G} = \{\mathbf{G}_1, \mathbf{G}_2, \ldots, \mathbf{G}_{n\,ini}\}$;
         initial number of groups $n_{ini}$; selected group size $s$.
  Output: merged groups $\mathbf{MG} = \{\mathbf{MG}_1, \mathbf{MG}_2, \ldots, \mathbf{MG}_{n_{mer}}\}$;
          the number of groups $n_{\text{mer}}$ in $\mathbf{MG}$.
1 Randomly arrange $\mathbf{G}_1, \mathbf{G}_2, \mathbf{G}_3, \ldots, \mathbf{G}_{n_{\text{ini}}}$, defined as:
  $\mathbf{G}' = \{\mathbf{G}'_1, \mathbf{G}'_2, \mathbf{G}'_3, \ldots, \mathbf{G}'_{n\,ini}\}$;
2 $k = 1$, $i = 1$, $\mathbf{MG} = \{\phi\}$;
  /* Main loop */
3 while $i < n_{ini}$ do
4     $\mathbf{MG}_k = \{\phi\}$;
5     while $sizeof(\mathbf{MG}_k) < s$ && $i < n_{ini}$ do
6         $\mathbf{MG}_k = \{\mathbf{MG}_k, \mathbf{G}'_i\}$;
7         $i++$;
8     $k++$;
9 $n_{mer} = k$;
```

TABLE I
PROBABILITY DENSITY FUNCTIONS OF DISTRIBUTIONS AND THE FORMULA FOR THE SINUS MAP

Function name	Density function or formula	Parameters
Cauchy	$f(x) = \frac{1}{\pi\gamma\left[1+(\frac{x-x_0}{\gamma})^2\right]}$	$-\infty < x < +\infty$ $\gamma = 2,\ x_0 = 0$
Gaussian	$f(x) = \frac{1}{\sqrt{2\pi}\sigma} exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$	$-\infty < x < +\infty$ $\mu = 0,\ \sigma = 1$
Lévy	$f(x) = \begin{cases} \sqrt{\frac{c}{2\pi}}\,(x-\mu)^{-1.5}\, e^{-\frac{c}{2(x-\mu)}}, & x \geq \mu \\ 0, & x < \mu \end{cases}$	$\mu = 0,\ c = 1$
Beta	$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{(\alpha-1)}(1-x)^{(\beta-1)}$	$0 < x < 1$ $\Gamma(x) = \int_0^{+\infty} t^{x-1}e^{-t}dt$ $\alpha = 2,\ \beta = 5$
Exponential	$f(x) = \begin{cases} \lambda e^{-\lambda x}, & x > 0 \\ 0, & x \leq 0 \end{cases};$	$\lambda = 2$
Sinus	$X_{n+1} = 2.3 \times X_n^{2\sin(\pi X_n)}$	$X_0 = 0.3133$

DG2 method may not be completely correct, two interacting variables may be placed in different groups. The strategy of dynamically changing the group size and randomly merging the initial grouping result can increase the probability of placing interacting variables in the same group. In addition, when the merging is performed, the interacting decision variables from the initial grouping result always remain in the same group, which is a superior result to that obtained with the bare-bones random grouping.

C. Combinatorial Testing of Generation Functions

The random numbers used in the formulas for updating the particle position and velocity are crucial factors for the performance of the algorithm. Therefore, we test the random number generation distribution functions.

1) Description of Random Number Generation Functions: The utilized functions include the Cauchy, Gaussian, Lévy, Beta, and exponential [36] distributions, as well as a chaotic map [37]. The probability density functions for the abovementioned distributions and the formula for the Sinus map can be found in Table I.

2) Testing and Selecting Generation Functions: The four random numbers in (14) and (22) are important factors that affect the performance of the algorithm. To determine the best generation function combination for rand_1, rand_2, rand_5, and rand_6, we need to evaluate the above six distribution functions. DPCCPSO uses the Lévy distribution to generate random numbers to update the particle position in LPSO, achieving good results. Therefore, based on our prior knowledge and the orthogonal experiment design, we design six combinations, as listed in Table II.

TABLE II
GENERATION FUNCTION DETAILS AND FRIEDMAN TEST RESULTS OF THE SIX COMBINATIONS

No.	$rand_1$	$rand_2$	$rand_5$	$rand_6$	Rank
C1	*Lévy*	*Lévy*	*Cauchy*	*Gauss*	2.0
C2	*Beta*	*Lévy*	*Exponential*	*Sinus*	4.8
C3	*Exponential*	*Lévy*	*Sinus*	*Beta*	5.8
C4	*Cauchy*	*Exponential*	*Beta*	*Sinus*	3.0
C5	*Gauss*	*Lévy*	*Exponential*	*Beta*	4.4
C6	*Lévy*	*Beta*	*Exponential*	*Lévy*	1.0

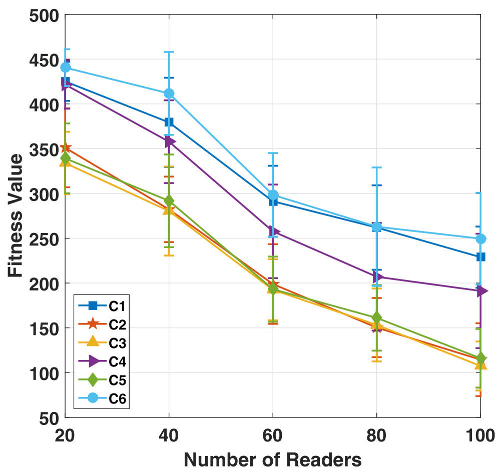

Fig. 2. Test results of the six combinations.

With respect to the anticollision model described in Section III, we compare the six combinations for reader numbers of 20, 40, 60, 80, and 100. As shown in Fig. 2, *combination six* outperforms the other combinations for all of the test cases. We also conduct the Friedman test with respect to the six combinations, as shown in the last column of Table II. As can be seen in the table, *combination six* outranks the other combinations. Therefore, *combination six* is used in this article to generate random numbers for the position and velocity updating procedures.

D. Parallelism Implementation

We reduce the time consumption for optimizing the high-dimensional problem by using MPI to implement the proposed algorithm in a distributed parallel environment. A coarse-grained parallel strategy is used, in which each process represents a particle and all particles are optimized in parallel. Particle and grouping information sharing is realized by communication among processes through MPI. For simplicity, we omit the details of the parallelism.

E. Complexity Analysis

First, the DG2 method is utilized to group variables. In this process, the main calculation burden is the FE, the number

TABLE III
PARAMETER SETTINGS OF THE EXPERIMENTATION

Notation	Description	Value
N_{tag}	tag number	1000
N_{reader}	reader number	20, 40, 60, 80, and 100
	candidate deployment locations	30, 60, 80, 100, and 120
N_{var}	variable number	200, 400, 600, 800, and 1000
N_{slot}	time slot number	4
	channel number	10
(w_1, w_2)	weight factors	$(0.8, -0.2)$
	other model parameters	detailed in [10]
NP	particle number	40
	CPU cores utilized	40
N_T	total number of FEs	$N_{var} \times 10^3$

TABLE IV
COMPARISON OF THE TEST RESULTS OF DPCCPSO, PCCPSO, AND AINETHE-RA

Reader Number	DPCCPSO	PCCPSO [12]	AINetHE-RA [10]
20	**440.5310**	364.1108	229.1091
40	**411.7091**	316.6659	105.7067
60	**298.4147**	221.8843	43.8780
80	**262.9034**	197.3505	32.7636
100	**249.5425**	182.1630	19.1445

of which is $N_T^{\text{grp}} = [(N_{\text{var}}(N_{\text{var}}+1))/2]+1$ [13]. Thus, the computational complexity is $N_T^{\text{grp}} \times O(f)$, here, $O(f)$ denotes the complexity of the optimization problem [i.e., (13)], which relates to the variable number and specific function formulas. Then, during optimization, to generate a new individual, for each variable in the considered group, PSO is utilized for evolution, and the needed calculations can be considered to be proportional to the variable number, which are $(N_{\text{var}}/n_{\text{mer}}) \times O(\text{PSO})$, here, n_{mer} denotes the group number after group merging as in Algorithm 1, and $O(\text{PSO})$ denotes the complexity of PSO to update one variable. With respect to the entire evolution process, the complexity will be $(N_T - N_T^{\text{grp}})[(N_{\text{var}}/n_{\text{mer}}) \times O(\text{PSO}) + O(f)]$. Finally, the complexity of the algorithm is $N_T^{\text{grp}} \times O(f) + (N_T - N_T^{\text{grp}})[(N_{\text{var}}/n_{\text{mer}}) \times O(\text{PSO}) + O(f)]$. For high-dimensional complex problems, $O(f) \gg (N_{\text{var}}/n_{\text{mer}}) \times O(\text{PSO})$, and the simple version of the algorithm complexity is $N_T \times O(f)$. As the distributed parallelism is employed, with parallel DG2 and evolution, the optimization time will be $[(N_T \times O(f))/N_{\text{MPI}}] + O(\text{Comm})$, here, N_{MPI} denotes the number of MPI processes and $O(\text{Comm})$ denotes the time complexity for communication among the MPI processes.

V. EVALUATIONS

A. Experimental Setup

We perform an experiment to simulate the application of RFID in a parking lot of unmanned delivery vehicles. The parking lot is a $100 \times 100 \times 5$ m^3 cuboid. Every unmanned delivery vehicle is labeled with a RFID tag, which stores the vehicle information. We assume that tags are randomly distributed over the 2-D floor of the deployment scenario. The reader locations in the parking lot are selected from a set of preassigned candidate locations. The parameter settings are listed in Table III. We validate the proposed algorithm by comparison with AINetHE-RA [10] and PCCPSO [12]. The Tianhe-2 supercomputer is the experimental platform for PCCPSO and the proposed DPCCPSO, and two spatially separated nodes with 24 cores each are utilized for each algorithm. AINetHE-RA is tested on MATLAB R2015a in Windows 7 with a 32-core Intel Xeon CPU E5-2665. The parameter settings of AINetHE-RA can be found in the work of Li *et al.* [10]. For each N_{tag} N_{reader} setting, we run each algorithm 20 times.

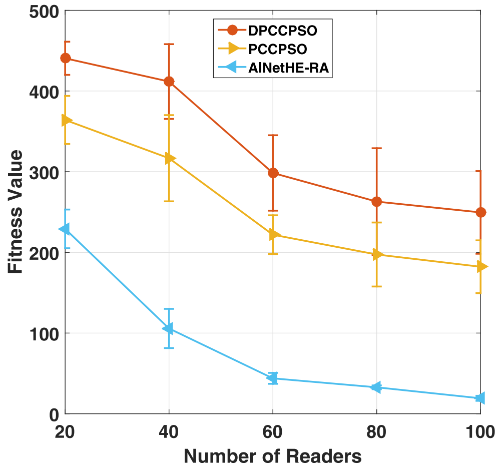

Fig. 3. Optimization results generated by DPCCPSO, PCCPSO [12], and AINetHE-RA [10].

B. Comparison of Effectiveness of Algorithms

For the three algorithms, five test instances corresponding to the reader numbers of 20, 40, 60, 80, and 100 are considered. As shown in Table IV, the proposed DPCCPSO outperforms PCCPSO and AINetHE-RA for all cases.

Fig. 3 illustrates the optimization results obtained using AINetHE-RA, PCCPSO, and DPCCPSO. As shown, the proposed DPCCPSO is superior to PCCPSO and AINetHE-RA for all five cases. PCCPSO ranks the second and AINetHE-RA is much worse. Considering the dimensions of the five kinds of tests, we conclude that for high-dimensional problems, AINetHE-RA is not effective, and much higher fitness values are obtained with PCCPSO and DPCCPSO. Additionally, the various strategies used in DPCCPSO result in superior performance to PCCPSO for all cases.

We analyze whether the differences among the three algorithms are significant by performing the Wilcoxon and Friedman tests on the results. As shown in Table V, DPCCPSO outranks the other two algorithms. The R+ values exceed the R-values, and the P-values are all below 0.05, indicating that DPCCPSO is significantly superior to AINetHE-RA and PCCPSO.

Fig. 4 illustrates the deployment effect of the optimal solutions obtained by DPCCPSO, PCCPSO, and AINetHE-RA (the reader number is shown in parentheses). The black triangles represent the candidate locations for readers, the blue and red dots represent the tags that are uncovered and effectively covered, respectively, and the circles represent the covering regions of the readers in an identification round. The four time slots are shown in different colors, with black, blue,

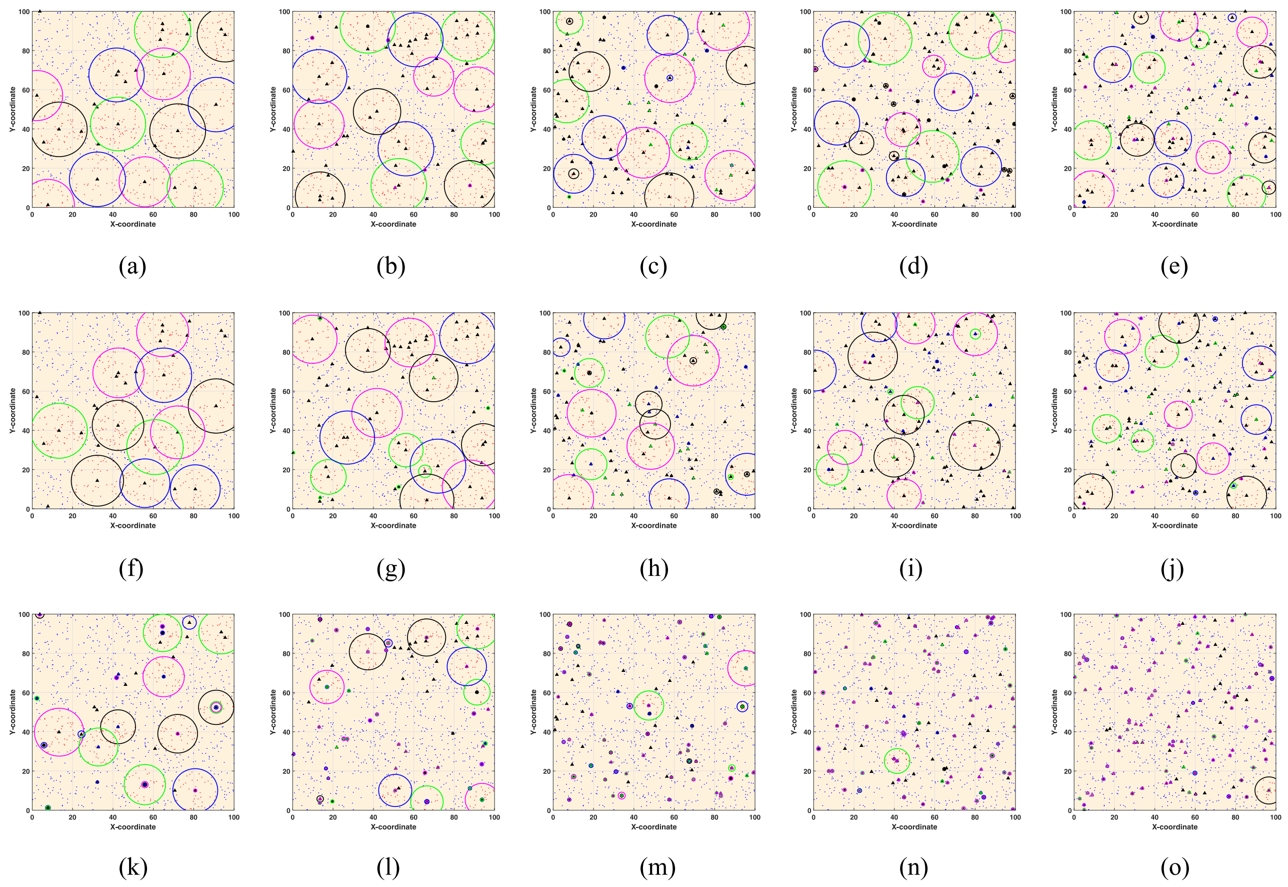

Fig. 4. Test results obtained by DPCCPSO, PCCPSO, and AINetHE-RA. (a) DPCCPSO (20). (b) DPCCPSO (40). (c) DPCCPSO (60). (d) DPCCPSO (80). (e) DPCCPSO (100). (f) PCCPSO (20). (g) PCCPSO (40). (h) PCCPSO (60). (i) PCCPSO (80). (j) PCCPSO (100). (k) AINetHE-RA (20). (l) AINetHE-RA (40). (m) AINetHE-RA (60). (n) AINetHE-RA (80). (o) AINetHE-RA (100).

TABLE V
WILCOXON AND FRIEDMAN TEST RESULTS OF DPCCPSO WITH PCCPSO AND AINETHE-RA

Algorithm	Ranking	R+	R-	p-value	$\alpha = 0.05$
DPCCPSO	1	\	\	\	\
PCCPSO [12]	2	15.0	0.0	0.030971	Yes
AINetHE-RA [10]	3	15.0	0.0	0.030971	Yes

TABLE VI
STATISTICS FOR COMPUTING TIMES OF PARALLEL AND SERIAL VERSIONS

N_{var}	parallel version	serial version
200	3.53 E+00 s	3.52 E+01 s
400	9.76 E+00 s	1.32 E+02 s
600	2.14 E+01 s	3.07 E+02 s
800	3.82 E+01 s	5.33 E+02 s
1000	5.66 E+01 s	8.07 E+02 s

green, and magenta representing time slots 1–4, respectively. As shown in Fig. 4, there are more effectively identified tags in the first row than in the following rows. In addition, the quantity of tags effectively interrogated by each reader is relatively balanced in the first two rows, but not in the third row. In summary, DPCCPSO outperforms the other two algorithms for optimizing the improved reader anticollision model.

C. Time Consumption Comparison

We evaluate the efficiency of the algorithm by comparing the time consumptions of the parallel and serial implementations of DPCCPSO. As previously mentioned, the Tianhe-2 supercomputer is the experimental platform, and 40 cores are used for the parallel program. We use one core for the serial program, and the other parameters are the same as for the parallel program. Table VI shows the test results for the computing time. We record the time consumptions of the parallel and serial programs for solving the anticollision problems with different variables. For each case, each algorithm only runs five times. The corresponding speedups for the five cases are 9.97, 13.52, 14.35, 13.95, and 14.26, and the corresponding efficiencies are approximately 24.9%, 33.8%, 35.9%, 34.9%, and 35.7% of the ideal speedup (i.e., 40), respectively.

VI. DISCUSSION

If the RFID readers are densely deployed, which is common in the IoT era and smart cities, the proposed scheme can effectively and efficiently determine the deployment locations and scheduling of RFID readers to maximize the throughput and load balance. The application scenarios include the IoT [38], factories [39], industry [40], chemical analysis [41], etc. However, the scheme requires global information, such as candidate locations and tag locations. For localized algorithms, the interrogation behavior of each RFID reader can be adjusted using local information. In addition, the positions and behaviors of RFID readers are fixed after deployment, which reduces the flexibility. Therefore, localized algorithms can be designed and combined to enhance the flexibility.

In the proposed scheme, the objective function is the weighted sum of two components. Nevertheless, the weight factors are difficult to determine, and the optimization of the sum cannot guarantee the simultaneous optimization of all components. Via multiobjective optimization [17], each component can be considered as an objective and optimized in a self-organized manner, resulting in a diversified population.

VII. Conclusion

To enhance the dependability and controllability of IoT systems, RFID reader anticollision is studied in this article. Collisions among densely deployed RFID readers can deteriorate system performance. For practical simulation purposes, we develop an improved anticollision model by simultaneously considering reader-to-reader and reader-to-tag collisions, modifying the measure index, and introducing a constraint condition. The proposed DPCCPSO is used to solve the high-dimensional anticollision optimization problem. The experimental results have demonstrated that the proposed DPCCPSO significantly outperforms two state-of-the-art algorithms: AINetHE-RA and PCCPSO. The reduction in the number of RFID reader collisions under intensive reader deployment can enhance the identification rate and reliability of RFID systems in parking lots and improve the management level of smart parking lots based on RFID technology. Future work will include comparing our model to more state-of-the-art anticollision algorithms. In addition, we will consider more factors, such as protocols, traffic, and other components to make the model more realistic.

References

[1] J. Lai *et al.*, "TagSort: Accurate relative localization exploring RFID phase spectrum matching for Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 1, pp. 389–399, Jan. 2020.

[2] A. T. Mobashsher, A. J. Pretorius, and A. M. Abbosh, "Low-profile vertical polarized slotted antenna for on-road RFID-enabled intelligent parking," *IEEE Trans. Antennas Propag.*, vol. 68, no. 1, pp. 527–532, Sep. 2020.

[3] J. Li and M. Chen, "Multiobjective topology optimization based on mapping matrix and NSGA-II for switched industrial Internet of Things," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1235–1245, Jun. 2016.

[4] F. Al-Turjman and A. Malekloo, "Smart parking in IoT-enabled cities: A survey," *Sustain. Cities Soc.*, vol. 49, Aug. 2019, Art. no. 101608.

[5] Z. Liu, X. Liu, J. Zhang, and K. Li, "Opportunities and challenges of wireless human sensing for the smart IoT world: A survey," *IEEE Netw.*, vol. 33, no. 5, pp. 104–110, Oct. 2019.

[6] B. Groza and P.-S. Murvay, "Security solutions for the controller area network: Bringing authentication to in-vehicle networks," *IEEE Veh. Technol. Mag.*, vol. 13, no. 1, pp. 40–47, Jan. 2018.

[7] L. Wan *et al.*, "Approximate and sublinear spatial queries for large-scale vehicle networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1561–1569, Feb. 2018.

[8] X. Cheng, J. Lu, and W. Cheng, "A survey on RFID applications in vehicle networks," in *Proc. Int. Conf. Identification Inf. Knowl. Internet Things (IIKI)*, Oct. 2015, pp. 146–151.

[9] L. Zhang, R. Ferrero, F. Gandino, and M. Rebaudengo, "Evaluation of single and additive interference models for RFID collisions," *Math. Comput. Model.*, vol. 58, nos. 5–6, pp. 1236–1248, 2013.

[10] Z. Li, J. Li, and C. He, "Artificial immune network-based anti-collision algorithm for dense RFID readers," *Expert Syst. Appl.*, vol. 41, no. 10, pp. 4798–4810, 2014.

[11] B. Cao, J. Zhao, Z. Lv, Y. Gu, P. Yang, and S. K. Halgamuge, "Multiobjective evolution of fuzzy rough neural network via distributed parallelism for stock prediction," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 5, pp. 939–952, Feb. 2020.

[12] B. Cao *et al.*, "Spark-based parallel cooperative co-evolution particle swarm optimization algorithm," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, 2016, pp. 570–577.

[13] M. N. Omidvar, M. Yang, Y. Mei, X. Li, and X. Yao, "DG2: A faster and more accurate differential grouping for large-scale black-box optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 6, pp. 929–942, Dec. 2017.

[14] J.-B. Eom, S.-B. Yim, and T.-J. Lee, "An efficient reader anticollision algorithm in dense RFID networks with mobile RFID readers," *IEEE Trans. Ind. Electron.*, vol. 56, no. 7, pp. 2326–2336, May 2009.

[15] M. V. Bueno-Delgado, R. Ferrero, F. Gandino, P. Pavon-Marino, and M. Rebaudengo, "A geometric distribution reader anti-collision protocol for RFID dense reader environments," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 2, pp. 296–306, Apr. 2013.

[16] H. Chen, Y. Zhu, K. Hu, and T. Ku, "RFID network planning using a multi-swarm optimizer," *J. Netw. Comput. Appl.*, vol. 34, no. 3, pp. 888–901, 2011.

[17] J. D. Ser *et al.*, "Bio-inspired computation: Where we stand and what is next," *Swarm Evol. Comput.*, vol. 48, pp. 220–250, Aug. 2019.

[18] B. Cao *et al.*, "Distributed parallel particle swarm optimization for multi-objective and many-objective large-scale optimization," *IEEE Access*, vol. 5, pp. 8214–8221, 2017.

[19] X. Ma *et al.*, "A survey on cooperative co-evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 23, no. 3, pp. 421–441, Jun. 2019.

[20] Q. Liu, W. Cai, J. Shen, Z. Fu, X. Liu, and N. Linge, "A speculative approach to spatial–temporal efficiency with multi-objective optimization in a heterogeneous cloud environment," *Security Commun. Netw.*, vol. 9, no. 17, pp. 4002–4012, 2016.

[21] Z. Zhou, Q. J. Wu, F. Huang, and X. Sun, "Fast and accurate near-duplicate image elimination for visual sensor networks," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 2, pp. 1–12, 2017.

[22] J. Shen, S. Chang, J. Shen, Q. Liu, and X. Sun, "A lightweight multi-layer authentication protocol for wireless body area networks," *Future Gener. Comput. Syst.*, vol. 78, pp. 956–963, Jan. 2016.

[23] B. Wang, X. Gu, L. Ma, and S. Yan, "Temperature error correction based on BP neural network in meteorological wireless sensor network," *Int. J. Sensor Netw.*, vol. 23, no. 4, pp. 265–278, 2017.

[24] X. Li and X. Yao, "Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2009, pp. 1546–1553.

[25] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Inf. Sci.*, vol. 178, no. 15, pp. 2985–2999, 2008.

[26] M. N. Omidvar, X. Li, Z. Yang, and X. Yao, "Cooperative co-evolution for large scale optimization through more frequent random grouping," in *Proc. IEEE Congr. Evol. Comput.*, 2010, pp. 1–8.

[27] B. Cao, J. Zhao, Y. Gu, Y. Ling, and X. Ma, "Applying graph-based differential grouping for multiobjective large-scale optimization," *Swarm Evol. Comput.*, vol. 53, Mar. 2020, Art. no. 100626.

[28] Y. Ling, H. Li, and B. Cao, "Cooperative co-evolution with graph-based differential grouping for large scale global optimization," in *Proc. 12th Int. Conf. Nat. Comput. Fuzzy Syst. Knowl. Disc. (ICNC-FSKD)*, 2016, pp. 95–102.

[29] B. Cao, S. Fan, J. Zhao, P. Yang, K. Muhammad, and M. Tanveer, "Quantum-enhanced multiobjective large-scale optimization via parallelism," *Swarm Evol. Comput.*, vol. 57, Sep. 2020, Art. no. 100697.

[30] X. Li and X. Yao, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 2, pp. 210–224, Jul. 2012.

[31] Q. Dong, A. Shukla, V. Shrivastava, D. Agrawal, S. Banerjee, and K. Kar, "Load balancing in large-scale RFID systems," *Comput. Netw.*, vol. 52, no. 9, pp. 1782–1796, 2008.

[32] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE Int. Conf. Evol. Comput. World Congr. Comput. Intell.*, 1998, pp. 69–73.

[33] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 2, 2002, pp. 1671–1676.

[34] G. Chen, J. Jia, and Q. Han, "Study on the strategy of decreasing inertia weight in particle swarm optimization algorithm," *J. Xi'an Jiaotong Univ.*, vol. 40, no. 1, pp. 53–56, 2006.

[35] Z. Yu, L. Xiao, H. Li, X. Zhu, and R. Huai, "Model parameter identification for lithium batteries using the coevolutionary particle swarm optimization method," *IEEE Trans. Ind. Electron.*, vol. 64, no. 7, pp. 5690–5700, Jun. 2017.

[36] R. A. Krohling and L. dos Santos Coelho, "PSO-E: Particle swarm with exponential distribution," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2006, pp. 1428–1433.

[37] A. B. Ozer, "CIDE: Chaotically initialized differential evolution," *Expert Syst. Appl.*, vol. 37, no. 6, pp. 4632–4641, 2010.

[38] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 3, no. 1, pp. 70–95, Nov. 2016.

[39] J. Feng, F. Li, C. Xu, and R. Y. Zhong, "Data-driven analysis for RFID-enabled smart factory: A case study," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 1, pp. 81–88, Jan. 2020.

[40] F. Bibi, C. Guillaume, N. Gontard, and B. Sorli, "A review: RFID technology having sensing aptitudes for food industry and their contribution to tracking and monitoring of food products," *Trends Food Sci. Technol.*, vol. 62, pp. 91–103, Sep. 2017.

[41] P. Kassal, M. D. Steinberg, and I. M. Steinberg, "Wireless chemical sensors and biosensors: A review," *Sensors Actuators B Chem.*, vol. 266, pp. 228–245, Aug. 2018.