Memetic Computing (2019) 11:89–107
https://doi.org/10.1007/s12293-018-0264-7

REGULAR RESEARCH PAPER

A fast two-objective differential evolution for the two-objective coverage problem of WSNs

Yulong Xu[1,2] · Yangdong Ye[1] · Han Zhang[3] · Wenbing Zhang[4] · Yali Lv[2]

Received: 13 April 2017 / Accepted: 1 June 2018 / Published online: 2 July 2018

Abstract
The aim of this article is to study the two-objective coverage problem of wireless sensor networks (WSNs) by means of differential evolution algorithm. Firstly, in order to reduce the computing redundancy of multi-objective optimization, namely to reduce the number of individuals which participate in non-dominated solution sorting, we introduced a fast two-objective differential evolution algorithm (FTODE). The FTODE contains a fast non-dominated solution sorted and a uniform crowding distance calculation method. The fast sorting method just handles the highest rank individuals but not all individuals in the current population. Meanwhile, during the individuals sorted, it can choose some of individuals into next generation and reduce the time complexity. The uniform crowding distance calculation can enhance the diversity of population due to it will retain the outline of optimal solution set by choosing the individual uniformly. Secondly, we use the FTODE framework to research the two-objective coverage problem of WSNs. The two objectives are formulated as: the minimum number of sensor used and the maximum coverage rate. For this specific problem, decimal integer encoding are used and a recombination operation is introduced into FTODE, which performs after initialization and guarantees at least one critical target's sensor is divided into different disjoint sets. Finally, the simulation experiment shows that the FTODE provides competitive results in terms of time complexity and performance, and it also obtains better solutions than comparison algorithms on the two-objective coverage problem of WSNs.

Keywords Differential evolution (DE) · Two-objective · Non-dominated solution sorted · Wireless sensor networks (WSNs) · Coverage problem

1 Introduction

1.1 Background

In recent years, wireless sensor networks (WSNs) have drawn enormous attention because that it can be used in various environments, such as battlefield surveillance, traffic control, animal tracking, home application, security management, medical and health care [1]. In WSNs, due to the limitation of energy and cost, there are many optimization problems such as the maximum lifetime of network, the minimum data routing, the minimum energy consumption, the maximum connectivity and coverage [2,3].

During above issues, one objective may conflict with others objectives such as between coverage and number of sensor. That is belonging to multi-objective optimization problem (MOP) [4]. In general, maintaining efficient coverage and prolonging the lifetime of WSNs is the most important issues. Many methods have been proposed to maximize the coverage, to minimize the consumption of energy, and to maximize lifetime of network simultaneously [5]. In paper [6], the authors built the balance between coverage efficiency and the capacity of the network that to get Pareto optimal solutions. In order to improve the coverage reliability of Wireless Sensor Networks, Attea et al. [7] addressed

✉ Yangdong Ye
ieydye@zzu.edu.cn

Yulong Xu
flyxyl@hactcm.edu.cn

1 School of Information Engineering, Zhengzhou University, ZhengZhou 450052, China

2 School of Information and Technology, Henan University of Traditional Chinese Medicine, Zhengzhou 450046, China

3 Intelligent Control Laboratory, College of Engineering, Peking University, Beijing 100871, China

4 Department of Mathematics, Yangzhou University, Yangzhou 225002, Jiangsu, China

the problem of improving coverage reliability of WSNs and maximizing the number of disjoint set covers (DSC) simultaneously. They transformed the definition of single-objective DSC problem into a MOP by adopting an additional conflicting objective. In literature [8], the authors aimed to cover a sensing area by deploying a minimum number of wireless sensors while maintaining the connectivity. They developed an integer linear programming model to solve the problem optimally. To study the maximum coverage deployment problem in WSNs and analyze the properties of the problem, Yoon et al. [9] has proposed an efficient genetic algorithm based on a novel normalization method. Unfortunately, there is only few reported research work concerning the coverage and number of sensor in the field of multi-objective optimization [2,3].

From above mentioned studies, it can be found that using multi-objective evolutionary algorithms to research these problems is very meaningful and valuable. Meanwhile, many multi-objective evolutionary algorithms (MOEAs) have successfully been employed to tackle MOP over the past decade. During the research results, the NSGA-II [10], the SPEA-II [11] and MOEA/D [12,13] are the most famous algorithms. Recently, the famous multi-objective and many-objective algorithms, such as IM/MOEA [14], NSGAIII [15], KnEA [16], IMOPEO-PLM [17] and MOEO [18] are presented. These algorithms can handle multi-objective or many-objective problems [19]. The literature [20] claimed that the non-parametric statistical test is concerned for comparing the performance of above different optimization algorithms. Since the Differential Evolution (DE) algorithm is proposed in [21]. Many scholars have devoted to extend their research on multi-objective Differential Evolution. According the improved individual density calculation and Pareto dominance, the multi-objective differential evolution algorithm (MODE) was proposed and achieved good result [22]. The most famous algorithm for solving multi-objective problem based on DE is differential evolution multi-objective optimization (DEMO) [23]. Its best feature is to select next generation individual on the basis of comparing with parent (target) and trial vector. If the trial vector dominates the corresponding parent vector, it replaces the parent vector; otherwise, the parent vector is retained in the current population. If the relationship of parent vector and trial vector is non-dominated, DEMO will put the trial vector into the extra archive. The test results show that the DEMO has significantly improved than NSGAII on convergence and diversity. In this year, a novel differential evolution with event-triggered impulsive control [24] and an improved NSGAIII with elimination operator [25] were presented.

Based on aforementioned analysis for multi-objective algorithms and problems of WSNs, we think there are two points need to be concerned. The first one, we consider using minimum number of sensor nodes so that it will fulfill maximum coverage of WSNs. Obviously, this is belongs to a bi-objective optimization problem. Secondly, its need introduce a suitable two-objective evolution algorithm for solving this problem. To the best of our knowledge, most of above evolution algorithms are based on the Parteo non-dominated sorted to select individuals into the next generation. These algorithms firstly sort the double size of population (SP) individuals based on the Parteo theory. After that, according the result of ranking, they select one SP individuals into next generation. However, this process has some redundant operations: (1) since the goal of sorting is just select one SP individuals into next generation, however, the above algorithms need sort double SP individuals; (2) In above algorithms, after the non-dominated sorted is finished, the operation of selecting individuals into next generation can be executed subsequently. We introduced a method to solve above computation redundancy in this work. In our method, the sorting operation only handles the highest rank individuals in current population. Meanwhile, the individuals can be chosen into the next generation during the proposed sorting operation. When the population of next generation is selected enough, the algorithm is terminated. The proposed method reduces the time complexity since the number of individuals for sorting operation is smaller. In addition, a calculation method of uniform crowding distance is given, which can retain the outline of optimal solution set since the individual is chosen uniformly.

Moreover, a fast two-objective differential evolution algorithm (FTODE) is proposed, which incorporates the introduced sorting method and uniform crowding distance into the classic DE. The simulation experiment used the standard multi-objective optimization problems ZDTl~ZDT4, ZDT6, F1–F5 for testing the performance of FTODE. Furthermore, the experimental results on optimization problems by using non-parametric statistical tests including Kruskal–Wallis test and Friedman test [26] are provided. Simulation results show that the FTODE has greatly improved in terms of time complexity and performance.

1.2 Our contribution and organization of the paper

In this paper, we proposed a fast two-objective differential evolution algorithm (FTODE) to solve the two-objective optimization problem of WSNs. The two objectives are the minimum number of sensor used and the maximum coverage rate. Our main contributions are summarized as follows:

(a) We present a fast two-objective differential evolution algorithm (FTODE), which contains a fast non-dominated solution sorted method and a uniform crowding distance calculation method.

(b) We establish a two-objective coverage problem of WSNs, namely use of minimum number of sensor nodes for maximum coverage rate.
(c) We perform extensive simulation experiment on the presented algorithm, compare and analyze results with existing and related algorithms.

The rest of this paper is organized as follows. In Sect. 2, related work is discussed emphasizing coverage problem in WSNs, Differential Evolution and multi-objective optimization. Section 3 presents the fast two-objective differential evolution algorithm. Section 4 provides the simulation experimental, the parameter settings for the investigation, and the performance of the proposed algorithm. In Sect. 5, we use the proposed FTODE to solve the two-objective coverage problem of WSNs. Finally, the conclusions and further work are discussed in Sect. 6.

2 Related works

2.1 The two-objective coverage problem in WSNs

In WSNs, one of the most critical issues is maximizing the lifetime of network while maintaining coverage and connectivity [2,3,28]. Therefore, the coverage problem influences the lifetime, the connectivity and the cost of network. In this work, the two-objective coverage problem is that using the minimum sensor nodes to coverage the maximum targets.

For the point-coverage problem of WSNs, suppose that there are a set of targets $T = \{t_1, t_2, t_3, \ldots, t_m\}$ in an $L \times W$ area, and then we randomly deploy a set of sensors $S = \{s_1, s_2, s_3, \ldots, s_n\}$ in this area to monitor the targets. All of the sensors have sleep and active mode. In the active mode, sensors can sense information of target and assume sensors have same sensing region, but in sleep mode it can't sense due to save energy. A target is said to be covered by a sensor if it lies within the sensing region of the sensor. In order to prolong the lifetime of WSNs, we need to find the maximum number of disjoint sensor covers. This problem can be solved via transformation to the DSC problem, which is defined as finding the maximum number C of disjoint complete cover sets [8,28]. The corresponding cover set C_i is satisfied following conditions:

Every cover C_i is a subset of S, $i \in [1, C_{max}]$, where the C_{max} is the upper bound of disjoint set covers number C. The maximum number of full cover subsets (C_{max}) can be used as the upper limit of the number of disjoint set covers. $C_i \in S$ and each C_i can complete coverage to all of the targets. Beyond that, for every of t_i belongs to at least one member of C_i, and for any two different covers C_i and C_j, $C_i \cap C_j = \phi$. We use the Fig. 1 as example, there are five sensors S1, S2, S3, S4, S5, and four targets t1, t2, t3, t4, every sensor have same circular sensing region.

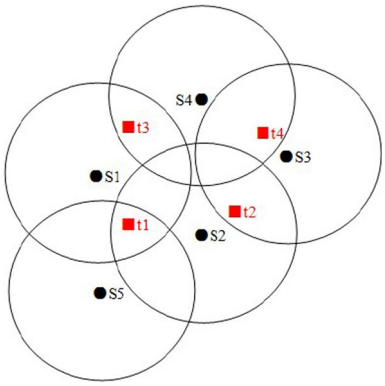

Fig. 1 A randomly deploy figure of WSNs by five sensors and four targets

From the above mentioned, we can maximize the lifetime of WSNs by maximizing the number of completely cover subsets. Meanwhile, for saving the number of sensor nodes, how using less sensor nodes to find more completely cover subsets is our main aiming. Namely, in this work the two optimization objectives are: maximum the number of completely cover subsets and minimum the number of used sensor nodes.

2.2 Differential evolution

DE is firstly proposed by Price and Store [21], with the advantage of its fast speed, less parameter, easy to implement and so on, which has become one of the most famous evolution algorithms. Because of DE is belonging to evolutionary algorithm, it have also included crossover, selection, update, and other basic structures. For saving the space of paper, the details of the DE algorithm are not provided in here and shown in the literature [21,27].

2.3 The mathematical description of multi-objective optimization

The features of multi-objective and single-objective optimization problem are different. In the single-objective optimization, the optimal solution is usually unique. However, in a multi-objective optimization, the optimal solution is a set because of the various objective functions are conflict each other. Considering the minimum of multi-objective problem, there are n decisions variables, m target variables can be expressed as Eq. (1) [10,11,14].

$$\begin{cases} Min\ y = F(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \\ \quad s.t.\quad g_i \le 0, i = 1, 2, 3, \ldots, q \\ \quad h_i = 0, j = 1, 2, 3, \ldots, p \end{cases} \tag{1}$$

where $x = (x_1, \ldots, x_n) \in X \subset R_n$ is the decision vector, X is the decision space with n-dimensional. The target vector is $y = (y_1, \ldots, y_n) \in Y \subset R_m, Y$ is the target space, are m-dimensional. $F(x)$ are m mapping functions that make decision space to the target space. The $g_i(x) \leq 0(i = 1, 2, 3, \ldots, q)$ is q inequality constraints, $h_i(x) = 0(j = 1, 2, 3, \ldots, p)$are equality constraints. The detail concept description about multi-objective is shown in [14]. In the interest of saving space, we are not discuss them in this article

However, it is should be noted that: (1) usually, in a multi-objective problem, one global optimal solution like in single-objective optimization does not exist. There are only Pareto optimal solutions in multi-objective problem. The Pareto optimal solutions are an acceptable "not bad" solution, and usually there are multiple Pareto optimal solutions. (2) All Pareto optimal solutions constitute a solution set of multi-objective optimization problem. In practical problems, according to the preferences of decision makers, they can select a suitable solution from the set as the final optimal solution. Therefore, the primary problem of multi-objective optimization is to find out Pareto optimal solutions as much as possible.

3 The fast two-objective differential evolution algorithm

The classical multi-objective evolution algorithm based on the Pareto non-dominated exist some redundant operations. To reduce redundant operations and enhance the computation speed, we present a fast sorting method based on non-dominated solutions. In addition, we introduce a uniform crowding distance calculation method. Furthermore, we combine the proposed methods to improve the performance of two-objective DE algorithm.

3.1 A fast non-dominated solution sorted method

One of the most famous multi-objective evolution algorithms is NSGAII [10]. In the canonical NSGAII, all individuals ($2N$,N is population size) are ranked and divided into different levels according to the non-dominated relation firstly. After that, based on the level (level 1 is the highest rank) in descending order, NSGAII selects some individuals into the next generation. It should be noted that the aim of selecting is just to choose excellent N individuals into the next generation. However, the NSGAII needs deal with all of $2N$ individuals.

From the foregoing, if we can reduce the number of individuals during the sorting operation, the time complexity of the algorithm will be decreased significantly. Hence, on the basis of the potential feature of sorting operation, this section proposes a fast method to sort and select the individual. This method suits to all of the multi-objective optimizations, we just describe it in the two-objective problem for concise. The method is shown on Algorithm 1.

Algorithm 1 Fast non-dominated sorting method

Input: The current population (the scale is $2N$)
Output: The next population (the scale is N)
1: **for** level r, $r = 1$ is initialized
2: **for** current population P (the scale is $2N$), each individual i in P is not allocated the level.
3: Finding 3 special individuals whose belong to the level r and storing them in a set S_r
4: The three special individuals have minimum f_1, f_2 ,and the shortest distance to the original point.
5: Sometimes, the three special individuals will repetitive, so the checkout operation is needed.
6: Three special individuals are assigned to the level r,
7: they belong to the highest level in current population
8: Making the subtraction operation for P and three special individuals and getting P'
9: **for** each individual j in P'
10: **if** the relation of j and all individuals in S_r is non-dominated
11: Assign the individual j to level r, and add it to set S_r
12: **end if**
13: **end for**
14: The set S_r stores the whole level r individuals
15: Let individuals of S_r into the next generation
16: **if** the cumulative total of the next generation more than N, the scale is overflowed
17: Just to choose some individuals into the next generation by crowded distance calculation.
18: the calculation algorithms is offered by Algorithm 2
19: **end if**
20: **if** the cumulative total of the next generation is exactly equal to N
21: Algorithm ends, the next population individuals (N) were produced
22: **end if**
23: **end for**
24: Making the subtraction operation between P and S_r, getting the set W
25: Using W to replace P ($P = W$), that is for updating P
26: r++, allocate for the next levels' individuals
27: **end for**

The main advantage of the proposed method is that it can select individual into next population during the sorting operation. Hence, the time complexity is descending due to the number of individuals for sorting operation is smaller.

In detail, before sorting and selecting the individual, the method firstly choose three special individuals. These special individuals belong to the highest level in current population. More specifically, the three individuals are the shortest distance to horizontal axis (minimum f_1), to vertical axis (minimum f_2), and to the original point, respectively. Note that these special individuals maybe repeat in some time.

These special individuals are stored in a set S_r. Then we compare the remaining individuals$i$ of population with each special individual in S_r. If the individual i non-dominate with all the special individuals in S_r, then the i is belongs to the highest level in current population and is added to the set S_r. Repeating above process for the next individual$i$, the method will find out those individuals that are belong to the current highest level.

Now, all individuals of S_r are obtained which belongs to the highest level in current population. After that the proposed method chooses all individuals of S_r into the next generation. If the number of cumulative individuals in next generation is less than N, the method will compute the next level by repeating above operation for the remainder individuals; otherwise, the crowded distance calculation is performed to select part of the current highest level individuals into next generation. When the number of the next generation individuals is just equal N, the algorithm terminates and don't processes the remaining individual to allocate level.

3.2 Theoretical basis for special individual in the fast non-dominated solution sorted method

The most important of our proposed method is how to prove three special individuals belong to the highest level of the current population. The three special individuals are shown in Fig. 2. Here, based on the Pareto dominance and sorting order theory, the analysis of special individual is given on below. For describing the problem conveniently, we use x and y represent the functions f_1 and f_2, respectively.

Theorem 1 *If the x value of individual a (x, y) is the minimum abscissa of all individuals, this individual belongs to the highest level of current population.*

Theorem 2 *If the y value of individual c (x, y) is the minimum ordinate of all individuals, the individual c belongs to the highest level of current population.*

Theorem 3 *For all individuals in the current population, if the distance from the individual b (x, y) to the original point is the minimum, the individual b belongs to the highest level of current population.*

Analysis for Theorem 1

Three special individuals are shown in Fig. 2. In current population, we assume that the minimum abscissa of all individuals is x_{min} . The coordinates of that individual is $a(x_{min}, a_y)$, so there is not exist an individual $k(k_x, k_y)$ satisfaction $k_x < x_{min}$.

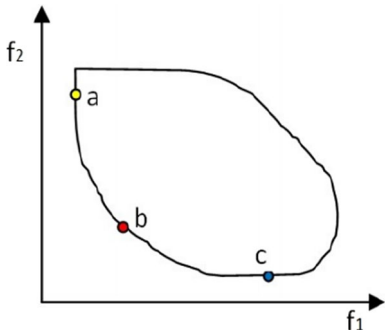

Fig. 2 Three special individuals in the proposed method

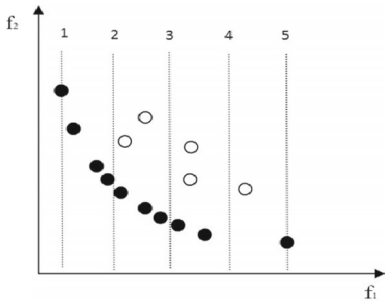

Fig. 3 A method of uniform crowding distance calculation

According to the Pareto theory, there is not exist an individual dominate the $a(x_{min}, a_y)$ in the current population. Therefore, the individual $a(x_{min,}a_y)$ belongs to the highest level distinctly. Others proof of Theorems 2 and 3 are similar above..

It should be emphasized that these special individuals may repeat each other in sometimes. Hence, the check operation for these individuals whether repeat is needed.

3.3 A uniform crowding distance calculation method

This section presents a uniform crowding distance calculation method, which is shown in Fig. 3 and Algorithm 2. When all the current highest level individuals are chosen into the next generation, the cumulative number of next generation is bigger than N, it need to select part of individuals. The presented crowded distance calculation method can evenly selects part of the current highest level individuals into next generation.

The uniformity selection is based on the distribution of the relative position in the current level individuals. This method can ensure that the individuals with a representative range are not discarded easily.

For example, there are 10 solid dots (individuals) belong to the highest level in Fig. 3. We need select 5 solid dots (individuals) into the next generation based crowding distance. We use the uniform crowding calculation method to select that. Firstly, we evenly draw 5 vertical lines between all of these individuals.

Then, find the nearest distance of 5 individuals to five lines, respectively. These 5 individuals are selected into the next generation. The detail process of this method is given in Algorithm 2.

Algorithm 2 A method of uniform crowding distance calculation

Input: k individuals
Output: m individuals
1: In the same level, select the m individuals into the next generation from k individuals, $m< k$.
2: Identify the two boundaries in the k individuals, k_1 and k_m
3: **iF** $m==2$, just select k_1 and k_m into the next generation
4: choose k_1 and k_m.
5: **else if** $m==1$, just select one of k_1 and k_m.
6: Randomly select one from k_1 and k_m into the next.
7: **end iF**
8: With the range of individual k_1 and k_m, make vertical m lines evenly.
9: **for** considering each line i
10: **for** each individual j, $j = 1: k$
11: **if** the distance between i and j is the nearest
12: Individual j is selected into the next generation.
13: If there are two individuals nearest to line i, randomly select an individual to the next generation.
14: **end if**
15: **end for**
16: **end for**
17: m individuals are chosen into the next generation

3.4 The overall framework and time complexity analysis of fast two-objective differential evolutionary algorithm

According to above fast non-dominated sorting method and uniform crowding distance calculation, we combine them with the DE to propose a fast two-objective differential evolution (FTODE). The general framework of the FTODE is shown in Algorithm 3.

Algorithm 3 Fast two-objective differential evolutionary algorithm based on non-dominated solutions sorting

Input: Parameters G, C_r, F, N, FES. etc
Output: The set of optimal solution
1: Initial population, size N, $X(0):x_1(0),\ldots,x_n(0)$;
2: Initialization parameters $G = 0$, C_r, F.etc.
3: **Begin** evolution, FES=0.
4: Mutation, obtain the variation vector V
5: Crossover, get trial vector U
6: Merging V and U, obtain P who has $2N$ individuals
7: Choose N from (P) $2N$ into the next generation
8: Using the proposed fast non-dominated solutions sorting algorithm, see the Algo 1
9: In the Algo 1, when finding the rth level individuals, directly select them into the next generation
10: **if** the total number of chosen individuals is equal to N
11: Jump into the next generation evolution
12: **end if**
13: **if** the total number of chosen individuals greater than N
14: The even crowding distance calculation is operated to the individuals in rth level. At the Algo 2
15: Select the part of individuals into the next generation.
16: if the number of next generation individuals is equal to N, jump into the next evolution.
17: **end if**
18: **End** of the evolution

The FTODE reduces the time complexity of algorithm by decreasing the number of individuals for sorting operation. The NSGAII is the most famous in all multi-objective evolutionary algorithms. However, it needs firstly rank and assign the level for all individuals ($2N$), and then select parts of individuals (N) into the next generation. In the first version of NSGA, the time complexity of non-dominated solutions sorting is $O(n^3)$. So far as to the NSGAII, the time complexity is $O(n^2)$. The reason of high time complexity is that it needs to sort and assign level for all individuals ($2N$) by non-dominated relation.

In FTODE, the best time complexity is $O(n)$. When the number of individuals in the first level is greater than or equal to the population size N, these individuals are selected into the next generation directly. If this number is greater than N, the crowding distance calculation is needed and the remaining levels of individuals are not need to be ranked. Considering the worst case, the FTODEs' time complexity is $O(n^2)$. In this condition, the algorithm needs to sort and assign level for all level individuals by their non-dominated relation.

In the Ref. [25], the authors claimed that the low selection pressure of Pareto dominance cause the MOEAs fail to handle many-objective optimization problems. Recently, Deb presented NSGAIII that also employs the uniformly distributed reference points to promote population diversity [15]. Thus, based on the above analysis, it's found that the FTODE has

Table 1 Ten bi-objective functions

Case	Range	Objective function
ZDT1	$x \in [0,1]^{30}$	$f_1(x) = x_1$ $f_2(x) = g(x)(1-\sqrt{x_1/g(x)})$ $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$
ZDT2	$x \in [0,1]^{30}$	$f_1(x) = x_1$ $f_2(x) = g(x)(1-(x_1/g(x))^2)$ $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$
ZDT3	$x \in [0,1]^{30}$	$f_1 = x_1$ $f_2(x) = g(x)(1-\sqrt{x_1/g(x)} - \frac{x_1}{g(x)}\sin(10\pi x_1))$ $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$
ZDT4	$x_1 \in [0,1]$, $x_i \in [-5,5]$, $i = 2,\ldots,10$	$f_1 = x_1$ $f_2 = g(x)(1-\sqrt{x_1/g(x)})$ $g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}(x_i^2 - 10\cos(4\pi x_i))$
ZDT6	$x \in [0,1]^{30}$	$f_1(x) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$ $f_2(x) = g(x)(1-(f_1(x)/g(x))^2)$ $g(x) = 1 + 9((\sum_{i=2}^{n} x_i)/(n-1))^{0.25}$
F1	$x \in [0,1]^{10}$	$f_1(x) = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}(x_j - x_1^{0.5(1+\frac{3(j-2)}{n-2})})^2$ $f_2(x) = 1 - \sqrt{x_1} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}(x_j - x_1^{0.5(1+\frac{3(j-2)}{n-2})})^2$ $J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\}$ and $J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$
F2	$x \in [0,1]^{30}$	$f_1(x) = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}(x_j - \sin(6\pi x_1 + \frac{j\pi}{n}))^2$ $f_2(x) = 1 - \sqrt{x_1} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}(x_j - \sin(6\pi x_1 + \frac{j\pi}{n}))^2$ $J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\}$ and $J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$
F3	$x \in [0,1]^{30}$	$f_1(x) = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}(x_j - 0.8x_1\cos(6\pi x_1 + \frac{j\pi}{n}))^2$ $f_2(x) = 1 - \sqrt{x_1} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}(x_j - 0.8x_1\sin(6\pi x_1 + \frac{j\pi}{n}))^2$ $J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\}$ and $J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$
F4	$x \in [0,1]^{30}$	$f_1(x) = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}(x_j - 0.8x_1\cos(\frac{6\pi x_1 + \frac{j\pi}{n}}{3}))^2$ $f_2(x) = 1 - \sqrt{x_1} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}(x_j - 0.8x_1\sin(6\pi x_1 + \frac{j\pi}{n}))^2$ where J_1 and J_2 are the same as $F1$
F5	$x \in [0,1]^{30}$	$f_1(x) = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}\left\{x_j - \left[0.3x_1^2\cos(24\pi x_1 + \frac{4j\pi}{n}) + 0.6x_1\right]\cos(6\pi x_1 + \frac{j\pi}{n})\right\}^2$ $f_2(x) = 1 - \sqrt{x_1} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}\left\{x_j - \left[0.3x_1^2\cos(24\pi x_1 + \frac{4j\pi}{n}) + 0.6x_1\right]\cos(6\pi x_1 + \frac{j\pi}{n})\right\}^2$ where J_1 and J_2 are the same as $F1$

high selection pressure of the Pareto-dominace relation and uniformly distributed reference points.

Generally, the individuals are distributed randomly in early stage of evolution. In order to choose N individuals into the next generation, it needs to handle more levels individuals. However, in later stage of evolution, many individuals are evaluated in the higher levels. Hence, just considering few levels, it can obtain N individuals into the next generation. In this case, the FTODE will show its high efficiency.

4 Simulation experiment for the FTODE

To verify the effectiveness of the proposed fast non-dominated sorting and uniform crowded distance calculation, we compare the FTODE with others classical algorithms in this section.

4.1 Experimental conditions and parameter settings

The simulation experiments use the standard minimum two-objective optimization problems ZDTl~ZDT4, ZDT6 [10], and F1–F5 [12] as testing functions. These functions are shown in Table 1.

Without special explanation, The FTODE uses parameters as follows: Population size $N = 100$, scaling factor $F = 0.5$, crossover probability $C_r = 0.3$, the maximum evolution generation is 250, the mutation strategy is "DE/Rand/1/bin". The parameter settings of other classic comparison algorithms are referred by original literatures. All the experiments have been

Table 2 Running time and number of comparisons for the three sorting methods, second (number of comparisons)

Algorithm	The size of population					
	100	500	1000	2000	5000	10,000
The sorting method of this paper (FTODE)	**0.045**	**0.056**	**0.096**	0.255	0.845	2.342
	(1.1e+3)	**(1.2e+4)**	**(3.8e+4)**	(1.0e+5)	(4.1e+5)	(1.2e+6)
An efficient approach to non-dominated sorting (ENS) [30]	0.063	0.078	0.102	**0.137**	**0.156**	**1.492**
	(6.6e+2)	(8.3e+3)	(2.6e+4)	**(7.2e+4)**	**(2.9e+5)**	**(8.3e+5)**
The sorting method in NSGAII [10]	0.054	0.61	2.37	10.02	67.7	310.2
	(4.9e+3)	(1.2e+5)	(4.9e+5)	(1.9e+6)	(1.2e+7)	(5.2e+7)

implemented by MATLAB software on a 3.3 GHz PC with processor core i5-4590 and 4G RAM.

It's important to note that we default use two-objective function for experiment test due to convenience. The proposed method is suitable for more than two objectives, which will be verified on many-objective optimization problem in the future work.

4.2 Running speed comparison

To verify the high efficiency of our proposed fast sorting method, this subsection compares the NSGAIIs' non-dominated sorting method [10], an efficient approach to non-dominated sorting (ENS) [30] and the proposed method on the arithmetic speed. We compare the efficiency of three methods on two-objective problem with different population scale. The computing time and number of comparison for three methods are shown in Table 2 and Fig. 4. It should be noted that in this test, the individuals are randomly generated and the three algorithms only calculated levels for all individuals of population.

The Table 2 and Fig. 4 show that the proposed method spends less computing time and number of comparison than NSGAIIs' method, but slightly more than ENSs' method. Specifically, when the number of individual (*NP*) is small, the proposed method shows better than the NSGAIIs' method, similar with ENSs' method. Moreover, during the large scale populations (*NP*), the running speed of our algorithm significantly fast than NSGAIIs' method. However, it is found that our method cannot beat the ENS on speed and number of comparison. The reason is that ENS is an excellent method, which has the best computing speed so far and published in the top journal of evolution computation [30].

4.3 Overall performance comparison

This subsection offers the overall performance comparison between the FTODE and classical algorithms, which contains two parts experiments. Firstly, we use the convergence indicator γ and diversity metric Δ to evaluate the performance of

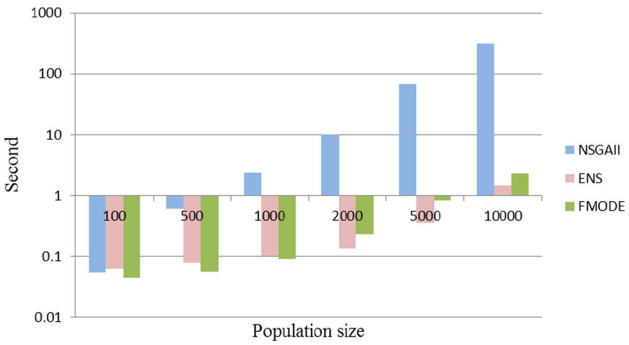

Fig. 4 Running time comparison on the two sorting methods

algorithms on problem ZDT1-4 and ZDT6 [10]. The comparing results are shown in the Tables 3 and 4.

As can be seen from the Tables 3 and 4, the FTODE is significantly better than NSGAII, SPEAII on convergence and diversity index. In addition, the Tables 3 and 4 imply that our FTODE preforms slightly better than the DEMO and similar with the IMOPEO-PLM. Specifically, FTODE converges better than all the other MOEAs on ZDT1 and ZDT2 problems. For ZDT3 and ZDT6 problem, FTODE preforms only worse than DEMO and IMOPEO-PLM in terms of γ, respectively.

Furthermore, motivated by the research results on the use of non-parametric tests for analyzing the algorithms' behavior [17,20,26], we apply non-parametric statistical test, e.g., Kruskal–Wallis and Friedman test, to compare the performance of algorithms based on convergence and diversity. Table 5 shows ranks, the statistics and related p value obtained by the Kruskal–Wallis and Friedman tests. From the Table 5, it is clear that a significant difference exists across the set of different algorithms in the terms of convergence and diversity because all the related p values are less than 0.05. FTODE achieves the best rank both in Kruskal–Wallis and Friedman in terms of convergence, and only slightly worse than IMOPEO-PLM in terms of diversity. That because of the IMOPEO-PLM is an excellent method based on extremal optimization (EO), which has effective mutation operation and mechanism of generating new population [17].

Table 3 Convergence metric γ comparison test on multi-objective minimization problem ZDT1-ZDT4 and ZDT6

	SPEAII [11]	NSGAII [10]	DEMO [23]	IMOPEO-PLM [17]	FTODE [this paper]
	Mean ± SD (rank)	Mean ± SD (rank)	Mean ± SD (rank)	Mean ± SD (rank)	Mean ± SD (rank)
ZDT1	2.43e–2±0.00e+0	3.34e–2±4.75e–3	1.08e–3±2.07e–5	1.04e–3± 8.96e–9	**7.88e–4± 3.84e–5**
	(4)	(5)	(3)	(2)	(1)
ZDT2	1.68e–1±8.15e–4	7.22e–2±3.14e–2	9.54e–4±4.65e–5	9.93e–4± 1.7e–9	**8.94e–4± 4.68e–5**
	(5)	(4)	(3)	(2)	(1)
ZDT3	1.82e–2±1.94e–5	1.12e–1±7.93e–3	**3.17e–3±5.91e–5**	3.91e–3± 7.01e–8	3.80e–3±5.09e–4
	(4)	(5)	(1)	(3)	(2)
ZDT4	4.93e+0±2.74e+0	5.13e–1±1.18e–1	1.05e–2± 8.34e–4	**1.78e–3± 9.32e–08**	1.38e–2±3.81e–2
	(5)	(4)	(2)	(1)	(3)
ZDT6	2.33e–1±4.91e–3	2.97e–1±1.31e–2	1.39e–2±1.34e–4	**8.93e–3± 2.13e–8**	1.37e–2±8.43e–4
	(4)	(5)	(3)	(1)	(2)

Table 4 Diversity metric Δ comparison test on multi-objective minimization problem ZDT1-ZDT4 and ZDT6

	SPEAII [11]	NSGAII [10]	DEMO [23]	IMOPEO-PLM [17]	FTODE (this paper)
	Mean ± SD (rank)	Mean ± SD (rank)	Mean± Std (Rank)	Mean ± SD (rank)	Mean± SD (Rank)
ZDT1	3.52e–1±8.70e–4	3.94e–1±1.85e–3	3.25e–1±3.02e–2	8.91e–2±1.23e–4	2.75e–1± 1.57e–3
	(4)	(5)	(3)	(1)	(2)
ZDT2	3.38e–1±1.75e–3	4.32e–1±4.72e–3	3.29e–1±3.25e–2	3.14e–1±9.11e–4	**2.29e–1± 2.74e–3**
	(4)	(5)	(3)	(2)	**(1)**
ZDT3	4.62e–1±5.24e–3	7.38e–1±1.98e–2	**3.07e–1±1.81e–2**	6.14e–1±2.54e–4	5.92e–1±1.39e–1
	(4)	(5)	**(1)**	(3)	(2)
ZDT4	8.23e–1±2.88e–3	7.03e–1±6.48e–2	3.59e–1±3.76e–2	7.94e–2±7.68e–4	**3.51e–1± 9.43e–2**
	(5)	(4)	(3)	(1)	**(2)**
ZDT6	1.04e+0±1.58e–1	6.67e–1±9.91e–3	4.42e–1±3.92e–2	5.61e–2±2.76e–4	**2.77e–1± 1.92e–2**
	(5)	(4)	(3)	(1)	**(2)**

Table 5 Ranks, the statistic, and related p value achieved by the Kruskal–Wallis, Friedman test for Tables 3 and 4

Algorithm	Convergence metric		Diversity metric	
	Kruskal–Wallis test	Friedman test	Kruskal–Wallis test	Friedman test
FMODE	**7**	**4.4**	7	1.8
IMOPEO-PLM [17]	**7**	4.6	**6**	**1.6**
DEMO [23]	10	2.4	11	2.6
NSGAII [10]	21	1.8	21	4.6
SPEAII [11]	20	1.8	20	4.4
Statistic	18.62	15.52	19.39	16.16
p value	9.3e–4	3.7e–3	7.0e–4	2.8e–3

Secondly, to analyze the proposed methods' performance with fast convergence, we used those novel algorithms, such as IM/MOEA [14], NSGAIII [15] and KnEA [16] for comparing on different evolution generations. Fortunately, the platform of PlatEMO [29] has integrated those algorithms, so we can employ this platform to help us completing the comparison experiment. It should be note that we use the IGD [5,6] as metric to evaluate performance of algorithms in this part test. The results are shown in the Tables 6, 7 and 8 with the evolution generation are 100, 250, and 500, respectively.

From the Table 6, it can be seen that FTODE shows the best overall performance. Especially, FTODE have the best results on ZDT1, ZDT2, ZDT3, F2, F3 and F5. When the evolution generation is 250, the results in Table 7 reveal our FTODE also has best performance on ZDT1, ZDT3, F2 and F3. However, in Table 8, when the evolution generation is 500, the MOEA/D/DE [12] performs the best result on

Table 6 IGD-Metric on multi-objective minimization problem ZDT1-ZDT4, ZDT6 and F1–F5 with 100 evolution generation

	MOEA/D/DE [12]	MOEA/D [13]	IM/MOEA [14]	NSGAIII [15]	KnEA [16]	FTODE (this paper)
	Mean error (SD)	Mean error (SD)	Mean error (SD)	Mean error (SD)	Mean error (SE)	Mean error (SE)
ZDT1	4.67e+0	1.23e–1	3.45e–1	5.37e–1	2.10e–1	**2.35e2**
	(3.17e+0)	(4.08e–2)	(1.11e–2)	(3.62e–1)	(1.16e–1)	**(8.66e–4)**
ZDT2	6.25e+0	3.41e–1	5.75e–1	1.04e+0	5.25e–1	**3.15e–2**
	(4.43e+0)	(1.75e–1)	(7.05e–2)	(2.53e–1)	(1.03e–1)	**(1.97e–3)**
ZDT3	4.06e+0	1.37e–1	3.09e–1	2.26e–1	2.16e–1	**2.26e–2**
	(2.52e+0)	(6.45e–2)	(4.53e–2)	(6.59e–2)	(1.29e–1)	**(1.80e–3)**
ZDT4	3.62e+0	4.96e–1	**1.07e–2**	6.89e–1	4.27e–1	2.58e–1
	(1.65e+0)	(1.52e–1)	**(1.42e–3)**	(3.80e–1)	(2.18e–1)	(1.16e–1)
ZDT6	**5.51e–2**	8.76e–2	2.56e+0	5.00e–1	9.78e–2	5.57e–1
	(9.76e–2)	(4.36e–2)	(1.16e–1)	(1.30e–1)	(4.36e–2)	(4.13e–2)
F1	**1.13e–2**	1.07e–1	2.01e–2	4.74e–2	1.63e–1	2.42e–2
	(2.06e–3)	(1.58e–2)	(3.85e–3)	(6.92e–3)	(2.96e–2)	(6.48e–4)
F2	1.33e–1	3.60e–1	1.87e–1	1.27e–1	1.46e–1	**9.02e–2**
	(3.57e–2)	(9.59e–2)	(3.77e–2)	(3.93e–2)	(4.01e–2)	**(4.61e–3)**
F3	1.14e–1	2.59e–1	7.82e–2	9.40e–2	3.15e–1	**4.90e–2**
	(6.42e–2)	(9.12e–2)	(1.28e–2)	(2.50e–2)	(6.99e–2)	**(3.64e–3)**
F4	8.80e–2	2.75e–1	**7.11e–2**	1.01e–1	2.76e–1	8.78e–2
	(1.23e–2)	(8.05e–2)	**(1.65e–2)**	(1.73e–2)	(6.08e–2)	(7.92e–3)
F5	7.85e–2	1.73e–1	6.54e–2	6.70e–2	2.52e–1	**6.33e–2**
	(3.88e–2)	(6.78e–2)	(7.97e–3)	(1.40e–2)	(3.47e–2)	**(3.04e–3)**

ZDT3, F1, F2 and F3; the FTODE has the best IGD on ZDT1, ZDT2, ZDT6 and F5. In other word, the FTODE shows similar performance with MOEA/D/DE but better than others four comparison algorithms when evolution generation is 500. Based on above analysis for Tables 6, 7 and 8, we can conclude that our FTODE has best performance when the evolution generation is small, and has competitive performance when the evolution generation is large. Namely, our algorithm can quickly converge to the optimal sets with less evolution generations.

In conclusion, the first experiment of this subsection shows that the proposed FTODE has competitive performance, which clearly won NSGAII, SPEAII and DEMO, similar with IMOPEO-PLM. The second part experiment claims that the FTODE has better convergence speed than those comparison algorithms.

4.4 Comparison with proposed uniform crowding distance calculation method and classic method

In this subsection, to demonstrate the efficiency of proposed uniform crowding distance calculation method, we provided the experiment tests for this method. . In the FTODE framework, we compare with Debs' crowding distance method and proposed uniform crowding distance calculation method in terms of convergence metric. . The parameters of this test are: the population size $NP = 100$, the scaling factor $F = 0.5$, the crossover probability $C_r = 0.3$, the maximum evolution generation is FES $= 250$, the mutation strategy is "DE/Rand/1/bin".

The test result for convergence is shown in Table 9, in which outside the parentheses are mean value, and in the parentheses are variances.

Clearly, the proposed uniform crowding distance calculation performs better than Debs' crowding distance calculation method on all test problems. Especially on the function ZDT1 and ZDT2, the performance of the proposed method has significantly improved than comparison algorithm. Therefore, it is proved that the proposed uniform crowding distance calculation method is effective.

4.5 Parameter setting of the FTODE

To study the effect of parameters on the performance of FTODE, this subsection uses the different crossover probability C_r, and mutation strategy for comparison and analysis. Storn and Price in [21] have indicated that a reasonable value for F is usually between 0.4 and 1, and a good initial choice of F was 0.5. Hence, we adopt the $F = 0.5$ and convergence metric for comparing the performance with different C_r and "DE/BEST/1" mutation strategy. The experiment results are provided in Tables 10 and 11.

Table 7 IGD-Metric on multi-objective minimization problem ZDT1-ZDT4, ZDT6 and F1–F5 with **250** evolution generation

	MOEA/D/DE [12]	MOEA/D [13]	IM/MOEA [14]	NSGAIII [15]	KnEA [16]	FTODE (this paper)
	Mean error (SD)	Mean error (SD)	Mean error (SD)	Mean error (SD)	Mean error (SE)	Mean error (SE)
ZDT1	2.36e–2	5.14e–2	1.78e–1	1.64e–2	3.19e–1	**3.73e–3**
	(3.93e–3)	(5.53e–2)	(5.06e–3)	(1.64e–2)	(6.65e–2)	**(1.14e–5)**
ZDT2	1.28e–1	**1.86e–2**	2.95e–1	2.27e–2	1.04e–1	2.67e–2
	(2.30e–1)	**(1.90e–2)**	(2.03e–2)	(3.11e–2)	(3.53e–2)	(3.31e–1)
ZDT3	9.64e–2	2.81e–2	1.66e–1	9.09e–2	8.39e–2	**1.08e–2**
	(4.87e–2)	(1.37e–2)	(1.08e–2)	(4.76e–2)	(5.28e–2)	**(2.03e–3)**
ZDT4	2.47e–1	3.65e–2	**6.48e–3**	1.51e–2	3.05e–1	2.79e–2
	(1.45e–1)	(2.38e–2)	**(2.48e–4)**	(2.06e–2)	(9.17e–2)	(5.43e–2)
ZDT6	**3.11e–3**	8.46e–3	2.19e+0	3.40e–3	9.18e–3	1.98e–2
	(1.81e–5)	(2.16e–3)	(1.03e–1)	(4.12e–4)	(1.59e–3)	(7.79e–4)
F1	**4.51e–3**	1.66e–2	9.67e–3	3.77e–2	2.44e–1	1.41e–2
	(2.04e–4)	(5.05e–3)	(6.43e–4)	(9.42e–3)	(3.86e–2)	(4.55e–4)
F2	1.12e–1	9.01e–2	7.66e–2	8.78e–2	2.07e–1	**7.64e–2**
	(4.33e–2)	(3.83e–2)	(9.00e–3)	(2.00e–2)	(3.47e–2)	**(6.62e–3)**
F3	5.95e–2	5.27e–2	4.99e–2	7.41e–2	4.70e–1	**4.04e–2**
	(4.45e–2)	(1.07e–2)	(1.62e–2)	(2.32e–2)	(4.15e–2)	**(5.37e–3)**
F4	6.67e–2	**3.83e–2**	4.34e–2	7.28e–2	3.90e–1	5.66e–2
	(4.18e–2)	**(4.62e–3)**	(7.05e–3)	(1.57e–2)	(1.16e–1)	(1.66e–3)
F5	5.91e–2	3.90e–2	**3.47e–2**	5.18e–2	3.95e–1	4.08e–2
	(4.03e–2)	(9.64e–3)	**(3.07e–3)**	(8.49e–3)	(4.52e–2)	(2.23e–3)

Table 8 IGD-Metric on multi-objective minimization problem ZDT1-ZDT4, ZDT6 and F1–F5 with 500 evolution generation

	MOEA/D/DE [12]	MOEA/D [13]	IM/MOEA [14]	NSGAIII [15]	KnEA [16]	FTODE (this paper)
	Mean error (SD)	Mean error (SD)	Mean error (SD)	Mean error (SD)	Mean error (SE)	Mean error (SE)
ZDT1	4.93e–3	8.38e–3	1.16e–1	3.88e–3	3.63e–1	**3.70e–3**
	(4.64e–4)	(4.45e–3)	(5.42e–3)	(9.39e–7)	(6.85e–2)	**(1.06e–5)**
ZDT2	4.40e–3	5.31e–3	1.90e–1	3.80e–3	1.54e–1	**3.72e–3**
	(5.57e–4)	(1.02e–3)	(1.28e–2)	(4.39e–7)	(1.48e–2)	**(2.41e–6)**
ZDT3	**1.07e–2**	1.97e–2	1.18e–1	2.97e–2	1.51e–1	1.16e–2
	(4.02e–5)	(1.46e–2)	(5.50e–3)	(1.38e–2)	(2.14e–1)	(1.57e–3)
ZDT4	5.82e–3	7.67e–3	5.31e–3	**4.01e–3**	3.04e–1	2.79e–2
	(1.26e–3)	(1.03e–3)	(1.68e–4)	**(7.48e–5)**	(5.82e–2)	(5.43e–2)
ZDT6	3.10e–3	4.94e–3	1.89e+0	3.18e–3	1.01e–2	**2.94e–3**
	(1.49e–7)	(7.67e–4)	(1.36e–1)	(2.16e–4)	(4.27e–3)	**(2.56e–5)**
F1	**4.01e–3**	5.11e–2	7.47e–3	2.66e–2	3.13e–1	1.05e–2
	(2.04e–5)	(4.44e–3)	(5.18e–4)	(8.24e–3)	(5.99e–2)	(7.01e–1)
F2	**8.18e–3**	2.25e–1	7.05e–2	1.38e–1	4.12e–1	7.04e–2
	(3.95e–2)	(5.96e–2)	(8.98e–3)	(7.52e–2)	(2.17e–1)	(1.73e–2)
F3	**9.27e–3**	2.05e–1	5.71e–2	5.00e–2	4.38e–1	3.53e–2
	(5.01e–2)	(6.39e–2)	(3.14e–2)	(8.11e–3)	(1.04e–1)	(7.65e–3)
F4	4.17e–2	1.91e–1	**3.56e–2**	7.87e–2	4.55e–1	4.47e–2
	(2.03e–2)	(6.46e–2)	**(1.13e–2)**	(3.93e–2)	(8.53e–3)	(5.3e–3)
F5	5.48e–2	1.42e–1	3.75e–2	4.66e–2	4.14e–1	**3.36e–2**
	(1.72e–2)	(7.58e–2)	(1.35e–2)	(9.20e–3)	(4.71e–2)	**(4.46e–3)**

Table 9 Convergence metric γ comparison test for two methods of crowding distance in FTODE

Test for two crowding distance method	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
The proposed crowding distance in FTODE	**7.88e–4**	**8.94e–4**	**3.80e–3**	**1.38e–2**	**1.37e–2**
	(3.84e–5)	**(4.68e–5)**	**(5.09e–4)**	**(3.81e–2)**	**(8.43e–4)**
The Deb's crowding distance in FTODE	1.39e–3	1.05e–3	4.76e–3	5.14e–2	1.48e–2
	(1.27e–4)	(3.69e–4)	(3.42e–4)	(6.30e–2)	(7.05e–4)

Table 10 Comparison for different parameter with "DE/rand/1" mutation in terms of convergence metric γ

	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
$F = 0.5\ C_r = 0.1$	1.15e–3	1.18e–3	3.88e–3	3.00e–1	1.77e–2
	(7.67e–5)	(4.20e–4)	(5.03e–4)	(3.90e–1)	(1.44e–3)
$F = 0.5\ C_r = 0.2$	8.49e–4	8.79e–4	**3.70e–3**	7.45e–2	**1.29e–2**
	(6.97e–5)	(3.42e–5)	**(6.63e–4)**	(8.53e–2)	**(9.94e–4)**
$F = 0.5\ C_r = 0.3$	**7.88e–4**	**7.52e–4**	3.80e–3	**1.38e–2**	1.37e–2
	(3.84e–5)	**(4.68e–5)**	(5.09e–4)	**(3.81e–2)**	(8.43e–4)
$F = 0.5\ C_r = 0.4$	8.65e–4	9.86e–4	4.05e–3	7.09e–2	1.47e–2
	(6.40e–5)	(6.22e–5)	(7.41e–4)	(1.39e–1)	(7.61e–4)
$F = 0.5\ C_r = 0.5$	1.04e–3	1.08e–3	4.39e–3	4.00e+0	1.84e–2
	(9.07e–5)	(3.64e–4)	(6.88e–4)	(8.37e–1)	(1.11e–3)
$F = 0.5\ C_r = 0.6$	1.68e–3	1.54e–3	5.13e–3	1.11e+1	2.33e–2
	(2.30e–4)	(5.01e–4)	(8.22e–4)	(1.87e+0)	(2.34e–3)
$F = 0.5\ C_r = 0.8$	7.23e–3	6.85e–3	1.07e–2	2.62e+1	6.12e–2
	(9.97e–4)	(8.34e–4)	(3.35e–3)	(1.17e+0)	(5.31e–3)

Table 11 Comparison for different parameter with "DE/best/1" mutation in terms of convergence metric γ

	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
$F = 0.5\ C_r = 0.1$	2.92e–1	1.18e+0	1.95e–1	**4.91e–1**	1.63e+0
	(1.90e–2)	(4.43e–1)	(1.12e–1)	**(2.95e–1)**	(1.63e+0)
$F = 0.5\ C_r = 0.2$	3.67e–1	5.67e–0	2.04e–1	4.97e–1	3.62e–2
	(5.52e–2)	(6.04e–1)	(7.85e–2)	(1.21e–1)	(3.13e–3)
$F = 0.5\ C_r = 0.3$	3.86e–1	9.33e–1	3.46e–1	5.93e–1	2.28e–1
	(8.01e–2)	(3.55e–1)	(1.84e–1)	(2.59e–1)	(2.63e–1)
$F = 0.5\ C_r = 0.4$	4.52e–1	1.20e+0	5.23e–1	8.14e–1	5.86e–2
	(1.25e–2)	(6.43e–2)	(2.05e–2)	(4.19e–1)	(2.81e–2)
$F = 0.5\ C_r = 0.5$	7.04e–1	1.06e+0	5.56e–1	1.25e+0	1.01e+0
	(2.01e–1)	(1.38e–1)	(7.77e–2)	(6.52e–1)	(1.01e+0)
$F = 0.5\ C_r = 0.6$	**1.20e–1**	**2.72e–1**	**6.88e–2**	1.11e+1	2.82e–2
	(8.78e–2)	**(9.68e–2)**	**(4.41e–2)**	(2.28e+0)	(3.77e–3)
$F = 0.5\ C_r = 0.7$	3.11e–1	7.03e–1	2.22e–1	2.71e+1	**2.04e–2**
	(1.34e–1)	(1.12e–1)	(1.20e–1)	(5.38e+0)	**(1.06e–2)**
$F = 0.5\ C_r = 0.8$	1.35e+0	2.38e+0	1.42e+0	2.19e+1	3.44e+0
	(1.36e–1)	(1.55e–1)	(2.32e–1)	(2.03e+0)	(7.46e–1)

From Tables 10 and 11, we find that the best performance of FTODE is when using the "DE/Rand/1" mutation strategy with the $F = 0.5$ and the $C_r = 0.3$. Hence, the $F = 0.5$ and $C_r = 0.3$ as experience values are obtained. Similarly, when using mutation strategy "DE/Current-to-best/1", the results of all the parameters combination are worse than with "DE/Rand/1/bin" mutation strategy. It should be noted that the FTODE using parameters $F = 0.5$, $C_r = 0.2$ and "DE/Rand/1" mutation also presents good performance. The reason is maybe that the small value of C_r between 0.2 and 0.3 is suitable for this method. We will deeply research the relationship between performance and these parameters in the future.

In short, through a series of parameters setting experimental for contrasting, we obtain that the suitable parameters of FTODE algorithm are: mutation strategy is "DE/Rand/1", variation factor $F = 0.5$ and crossover probability C_r is 0.3.

5 Using the FTODE to solve the two-objective coverage problem of WSNs

In this section, we use the proposed FTODE to solve the two-objective coverage problem of WSNs. In brief, the goal of issue is to use the less number of sensor nodes to cover the more target points in the WSNs.

5.1 The two-objective coverage problem of WSNs

In a WSN, the coverage problem is an important issue. In the area of $L * W$ (L is length and W is width), there are m target points $T = \{t_1, t_2, t_3, \ldots, t_m\}$ and n wireless sensor nodes $S = \{S_1, S_2, S_3, \ldots, S_n\}$.

A sensor generally has two operation modes, active and sleep mode. When in active mode, a sensor can carry out its full operations, such as sensing and communication. To maintain those operations, sensors need to consume a relatively large amount of energy. In contrast, a sensor in a sleep mode uses only a small amount of energy and can be awoken in a scheduled working interval for full operations. Hence, in the point-coverage WSNs, we can maximize the lifetime of WSNs by maximizing the number of completely cover subsets [28]. Meanwhile, for saving the cost, how using less sensor nodes to find more completely cover subsets are our two optimization objectives. Although we briefly introduced a method to solve this problem in previous work [3], we will provide more detail and systematic method for this issue in this paper.

Firstly, we define the first optimization objective is $f_1(x) = m/m_{max}$, $m \in [1, m_{max}]$. The m indicates the number of sensor nodes are used, which is the smaller the better. The m_{max} is the upper limit of m in theory. The second optimization objective is that we can obtain the number of completely cover subsets. To convert into a minimization problem, we use $f_2(x) = 1 - C/C_{max}$ as the second objective function. The C is the number of completely cover subset. The C_{max} indicates the upper limit of the number of completely cover subset. The others parameters are described detailed in the literature [28]. Hence, from above analysis, we deduce the minimization problem mode of the two-objective coverage problem of WSNs as follow Eq. (2).

$$Min \begin{cases} f_1 = \frac{m}{m_{\max}} \\ f_2 = 1 - \frac{C}{C_{\max}} \end{cases} \quad s.t. 1 < m \leq m_{max}; 1 < C \leq C_{max} \tag{2}$$

5.2 The representation of chromosomes

Intuitively, we use integer representation to encode a grouping combination of sensors. The value of a gene is produced randomly, which range is $[0, C_{\max}]$. A genes' value indicates an index of the subset that the sensor joins, thus the sensors with the same index number will form a disjoint cover set. Note that if the value of a gene is 0, means this sensor is unused or in sleep mode. Hence, a chromosome represents an allocation scheme. The purpose of optimization is to find an allocation scheme that with less sensor nodes but more completely coverage subsets.

For example, suppose a chromosome is $CH = (1, 2, 3, 2, 1, 0, 3, 1, 2)$. It means that there are nine sensors but just eight of them are used, because the 0 indicates this sensor is unused. Hence, the used number of sensors is 8, namely the function $f_1 = 8/9$. Considering the second objective, this chromosome has three disjoint subsets. The set 1contains sensor S_1, S_6 and S_8; set 2 contains sensor S_2, S_4 and S_9; the set 3 contains S_3 and S_7. If each of the above three sets can completely cover all targets, it means that we obtained the number of disjoint set cover is 3. Assume the C_{max} is 4 in theory, then the objective function $f_2 = 1 - (3/4)$.

5.3 The method of initialization recombination crossover and mutation

According to the representation of chromosomes, we randomly generate an integer between 0 and $C_{\max}$ as each gene's value (subset number) in the initialization. The sensors with same gene value mean that they shall form a subset.

The recombination operation is performed after the initialization, which guarantees at least one critical target's sensor is divided into different disjoint sets (subsets). However, there are may exist many critical targets in WSNs, and it is very difficult to scatter all of corresponding sensors into different subsets [3,28]. Thus, our recombination operation considers every chromosome, then randomly chooses a critical target

and recombines different gene values to their corresponding sensors.

Take the chromosome CH_2 as example, after initialization the chromosome CH_2 is (2, 1, 1, 1, 2), then the recombination operation will be performed. According to above analysis, there are three *critical targets* (t_2, t_3, t_4), which $t_2 \in (S_2, S_3)$, $t_3 \in (S_1, S_4)$, $t_4 \in (S_3, S_4)$. Thus, we randomly choose one *critical target* from t_2, t_3, t_4, suppose that the target t_4 is chosen and its' corresponding sensors are be scatted into different subset, such as one case is $S_3 = 2$, $S_4 = 1$. Namely the chromosome CH_2 is became $CH_2 = (2, 1, 2, 1, 2)$. It's obvious that after recombination operation the CH_2 has better fitness than before. The pseudo code of recombination is given in Algorithm 4. It is need to explain that the code is described by Matlab language and the bold words are inner functions or keywords in Matlab.

Algorithm 4 the pseudo code of recombination operation

Input: The initialization population, C_{max}
Output: The recombination population
1: **for** i=1: popsize // for every chromosome
//randomly generates C_{max} different integers and each integer value belong [1, C_{max}].
2: $sequ$ = **randperm**(C_{max});
//randomly choose one *critical target*
3: $rnum$=**randint**(1,1,[1, $numOfCritialTarget$]);
//find the corresponding sensors
4: rt= $CritialTargetMatrix$($rnum$,:);
5: **for** j = 1: C_{max} // all corresponding sensors
6: pop(i,rt(j)) = $sequ$(j); // recombination
7: **end for**
8: **end for**

Due to the mutation strategy is "DE/Rand/1" and the crossover style is binomial crossover. So, after mutation operation, the values of gene maybe become float type. Therefore, it needs to do round number operation for genes because their value should be integer type. We use the ceiling way for round number operation. If the value of gene without the range [0, $C_{\max}$], the method will regenerate its value between the [0,$C_{\max}$].

5.4 The overview flow of using FTODE to research the two-objective coverage of WSNs

This subsection provides the overview flow of using FTODE to research the two-objective coverage of WSNs. The overview is offered in Algorithm 5.

Algorithm 5 using the FTODE to solve the two-objective coverage problem of WSNs

Input: Parameters and data set
Output: The set of optimal solution
1: Load test data set.
2: Initialization parameters and population
3: Recombination operations
4: **while FES < 250**
5: Mutation operation, obtains mutation vectors V.
6: Amend operation for float types value and illegal values.
7: Binomial crossover operation gets trial vectors U.
8: Combining the V and U, obtains P (the size is $2N$)
9: From P Select N individuals into next generation
10: Compute the fitness value for two-objective
11: Using the proposed fast non-dominated solutions sorting algorithm, in the Algo 1
12: when finding the rth level individuals directly select them into the next generation
13: **if** the total number of chosen individuals is equal to N
14: Jump into the next generation evolution
15: **end if**
16: **if** the total number of chosen individuals greater than N
17: The even crowding distance calculation is operated to the individuals in rth level, which is shown in the Algo 2
18: Select the part of individuals into the next generation.
19: if the number of next generation individuals is equal to N, then jump into the next evolution.
20: **end if**
21: **FES**=**FES**+1
22: **end** while

5.5 Simulation experiment and analysis

For verification the performance of the FTODE, we use the NSGAII, FTODE and the classical DE to solve the same problem as comparison. It's worth noting that the test sets of experiment are actual data and we have not the Parteo optimal set in theory. So, we cannot use the convergence indicator to compare their performance. However, when the evolution is finished, we draw those individuals belong to the first level of all algorithms for comparison. The algorithm has more individuals belong to the first level and these individuals more closely to origin, its' performance will better.

We use six test sets for comparison experiment, the detail of test sets is shown in Table 12.

The result of comparison experiments are shown in Figs. 5, 6, 7, 8, 9 and 10. In these figures, we just draw the first levels' individual of different algorithms for comparison.

According to the Pareto-dominate principle, the Figs. 5, 6, 7, 8, 9 and 10 illustrate that the FTODEs' performance is better than these comparison algorithms on the two-objective coverage problem of WSNs. In detail, the FTODE shows better solutions than both canon DE and the NSGAII on the case

Table 12 The six test sets for comparison experiment

Index	Monitoring radius	m_{max}	Number of targets	$C_{\max}$
1	22	90	10	30
2	22	100	10	23
3	22	110	10	21
4	22	110	10	35
5	22	130	10	41
6	22	140	10	44

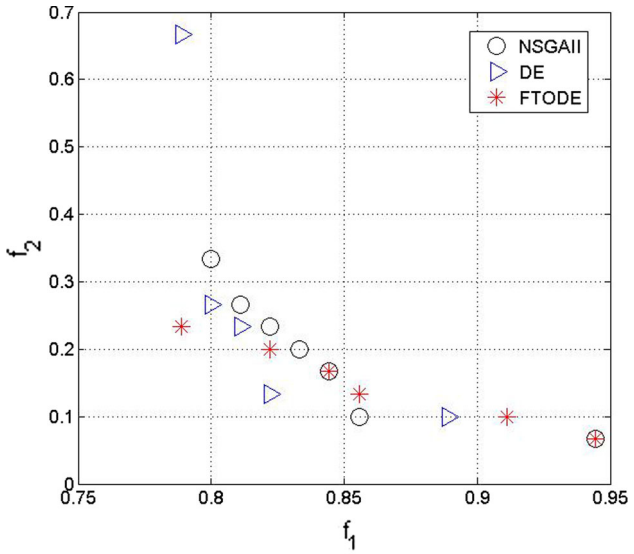

Fig. 5 The first level individuals in case 1

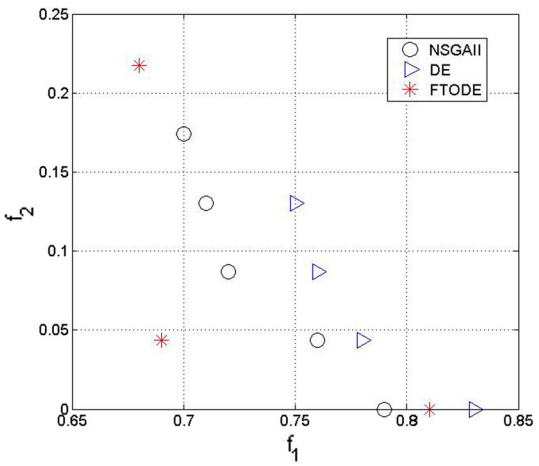

Fig. 6 The first level individuals in case 2

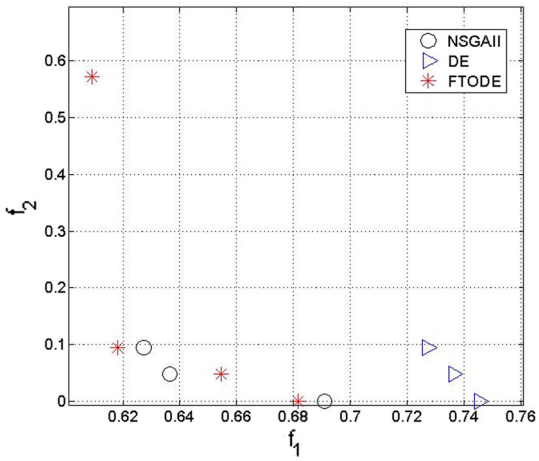

Fig. 7 The first level individuals in case 3

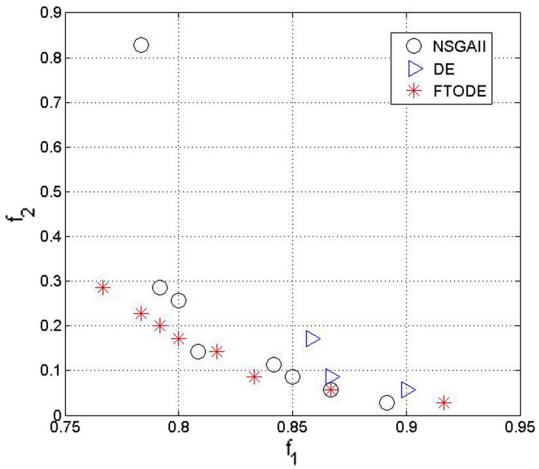

Fig. 8 The first level individuals in case 4

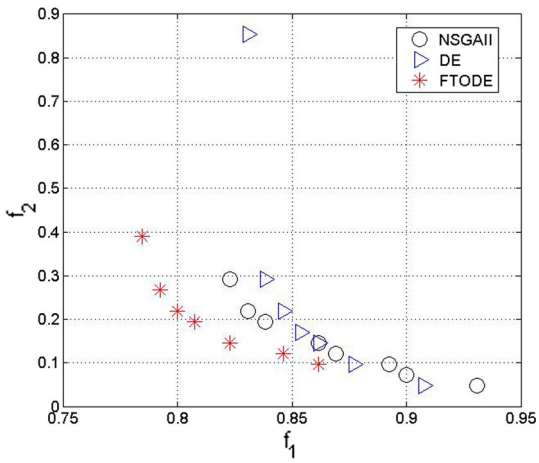

Fig. 9 The first level individuals in case 5

3. On the case 2, 4, 5, 6, the FTODE displays better performance than DE and NSGAII significantly. More specifically, in these cases, most of individuals in the FTODE dominate that in DE and NSGAII. Furthermore, the number of first level individuals in FTODE is more than that in comparison algorithms on case 4, 5, 6. On the case 1, the FTODE provides slight better performance than the DE and NSGAII.

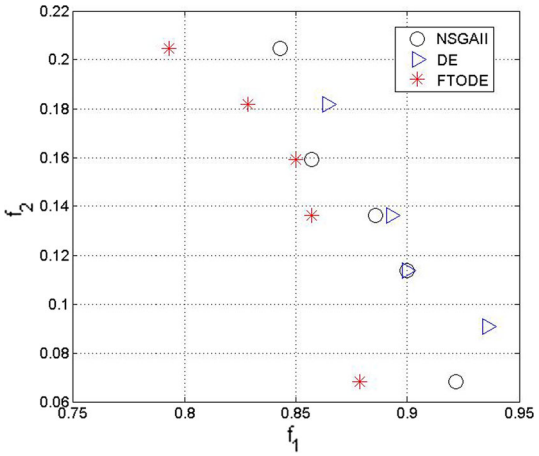

Fig. 10 The first level individuals in case 6

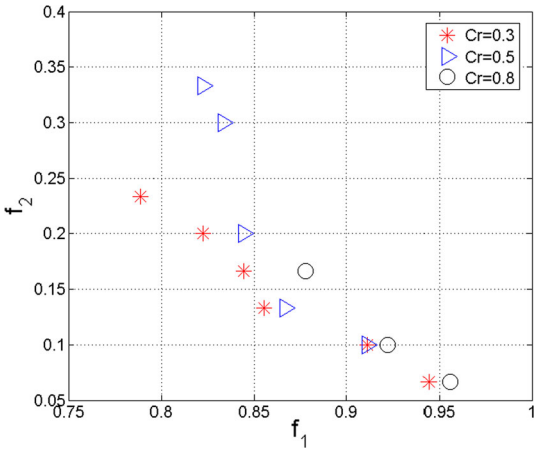

Fig. 11 The first level individuals in case 1 with various C_r

Based on above analysis, the FTODE performs the best among the three algorithms on the six test cases. Therefore, it can be drawn a conclusion that the proposed FTODE is reasonable and efficient to solve the two-objective coverage problem of WSNs.

5.6 Parameter setting of the FTODE algorithm

Price in [21] has indicated that a reasonable value for F is usually between 0.4 and 1, and a good initial choice of was 0.5. Hence, we use the $F = 0.5$ as default value with the "DE/Rand/1" mutation strategy.

The parameter C_r controls how many parameters in expectation are changed in a population member. To test the C_r influences the performance of FTODE, we use $C_r = 0.3$, 0.5, 0.8 for comparing test, respectively.

Because there are have not the optimization set in theory, we draw those individuals in the first level of algorithm for comparison when the evolution finished. The results are shown in Figs. 11, 12, 13, 14, 15 and 16.

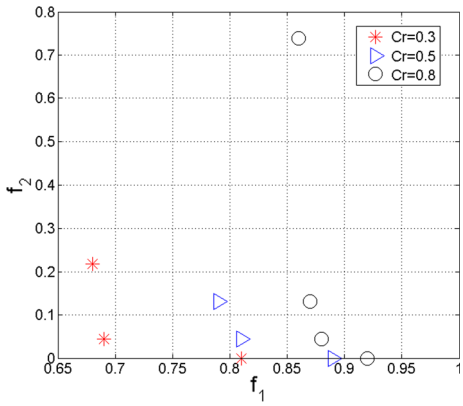

Fig. 12 The first level individuals in case 2 with various C_r

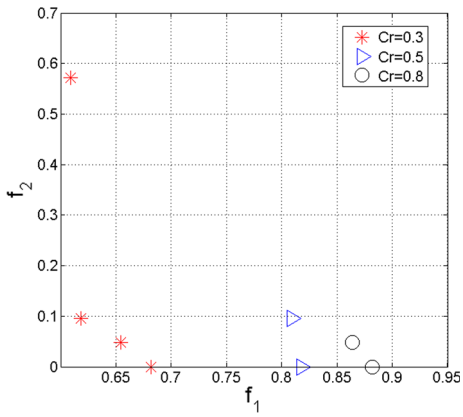

Fig. 13 The first level individuals in case 3 with various C_r

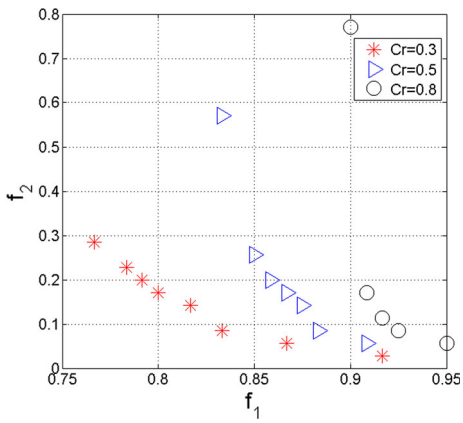

Fig. 14 The first level individuals in case 4 with various C_r

Overall, The Figs. 11, 12, 13, 14, 15 and 16 show that the FTODE provides the best performance when the $C_r = 0.3$.Specifically, on the test 1, three crossover styles have similar performance, and the $C_r = 0.3$ strategy offers slight better than others. On the test 2-6, the FTODE with $C_r = 0.3$ provides the best solutions. Especially on the test 4 and 6, it obtains significant good solution on quantity of the first level. It is note that on test 2 and 3, there are exist a solution with

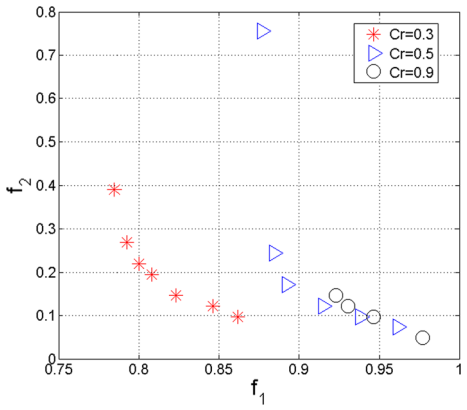

Fig. 15 The first level individuals in case 5 with various C_r

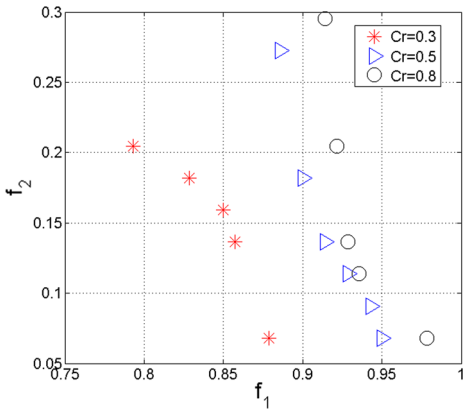

Fig. 16 The first level individuals in case 6 with various C_r

$f_2 = 0$ when $f_1 \neq 1$. That means using part number of sensors can find the maximum complete cover sets in theory. In other word, the number of sensors has some redundancy in this test. Without these redundant sensors, the algorithm also obtains the maximum complete cover sets in theory. From above analyses we can conclude that the suitable parameters setting are $F = 0.5$ and $C_r = 0.3$.

In addition, to study the various mutation strategies, with $F = 0.5$ and $C_r = 0.3$ we adopt "DE/Rand/1", "DE/Best/1" and "DE/current-to-best/1" for comparing, respectively.

Similar with others test in this subsection, we draw individuals in the first level of algorithm for comparison when the evolution finished. The results are shown in Figs. 17, 18, 19, 20, 21 and 22.

Figs 17 18, 19, 20, 21 and 22 show that the FTODE with the "DE/rand/1" mutation can obtain the best effect. Specifically, using this mutation strategy, the FTODE provides better performance than others strategies significant on test 2–4 and 6. On the test 1 and 5, all of these mutation strategies offer similar solutions. In a word, the performance of FTODE with the "DE/rand/1" mutation is the best, with the "DE/best/1" and "DE/current-to-best" mutation ranked second and third, respectively.

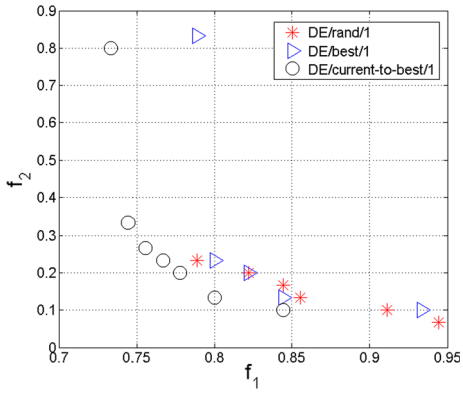

Fig. 17 The first level individuals in case 1 with various mutation strategies

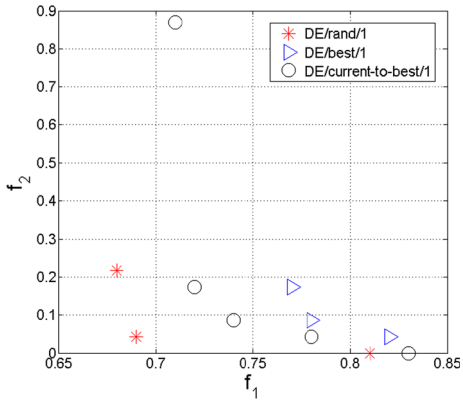

Fig. 18 The first level individuals in case 2 with various mutation strategies

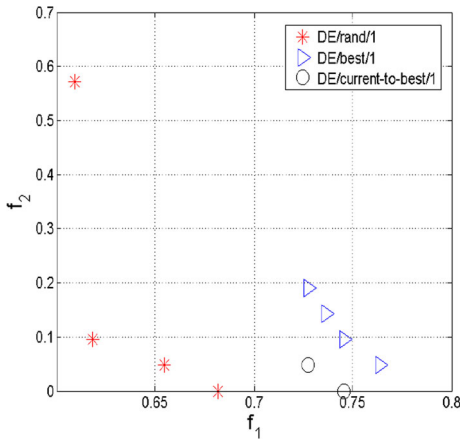

Fig. 19 The first level individuals in case 3 with various mutation strategies

6 Conclusions

In this paper, to study the two-objective coverage problem of WSNs by means of DE, we proposed an improved DE algorithm with fast non-dominated solutions sorting and uniform

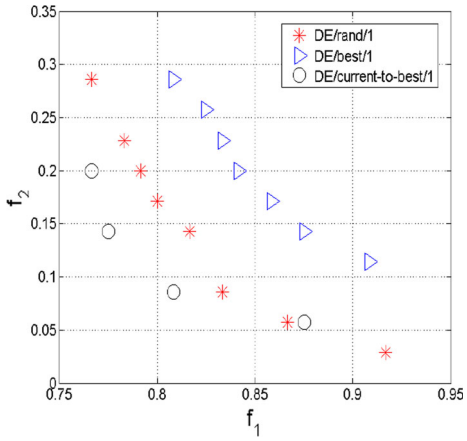

Fig. 20 The first level individuals in case 4 with various mutation strategies

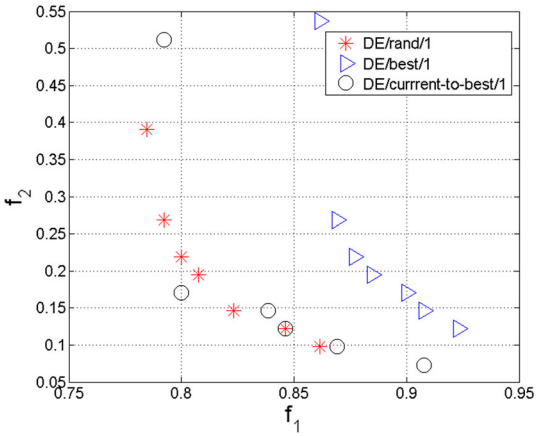

Fig. 21 The first level individuals in case 5 with various mutation strategies

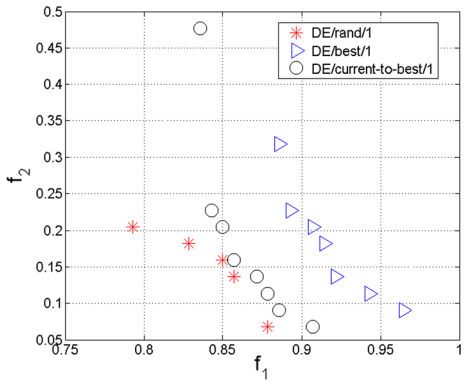

Fig. 22 The first level individuals in case 6 with various mutation strategies

crowding distance calculation. According to the principle and potential feature of non-dominated sorting solution, a fast non-dominated solutions sorting method was introduced. This method can reduce the number of individuals in sorting operation and choose individuals into next generation simultaneously. This method reduces the number of individuals for sorting process and the time complexity of algorithm. In addition, the uniform crowding distance calculation will retain the outline of optimal solution set, which can enhance the populations' diversity.

Then, we incorporated the introduced sorting method and uniform crowding distance into the DE to solve the two-objective coverage problem of WSNs. For this specific problem, the two objectives are formulated as: the minimum number of sensor used and the maximum coverage rate. The decimal integer encoding is used and a recombination operation is introduced into FTODE. Simulation results show that the proposed FTODE provides better performance than the classic comparison algorithms.

Although the FTODE is suitable for multi-objective problem, this paper just applies it in two-objective optimization. The limitation of this method is that it needs to choose too many special individuals when handling the many-objective problem. In the future research work, we will extend the proposed algorithm to study the many-objective problems in WSNs and research the relationship between performance and these parameters.

Acknowledgements The authors gratefully acknowledge the helpful comments and suggestions of editors and anonymous reviewers. This work was supported by the key scientific research project of Henan provinces' university (Nos. 15A520083 and 17B520017); the National Natural Science Foundation of China (Nos. 81703946, 61503328, 61772475 and 61502434); the doctoral research fund in Henan University of Chinese Medicine (Nos. BSJJ2015-19); the science and technology research Project of Henan province (Nos. 172102210361 and 172102310536).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Rault T, Bouabdallah A, Challal Y (2014) Energy efficiency in wireless sensor networks: a top-down survey. Comput netw 67(4):104–122
2. Fei Z, Li B, Yang S, Chen H, Hanzo L (2017) A survey of multi-objective optimization in wireless sensor networks: metrics, algorithms, and open problems. IEEE Commun Surv Tutor 19(1):550–586. https://doi.org/10.1109/COMST.2016.2610578
3. Xu YL, Wang XH, Zhang H (2016) Improved differential evolution to solve the two-objective coverage problem of wireless sensor networks. In: 2016 Chinese control and decision conference (CCDC), pp 2379–2384. https://doi.org/10.1109/CCDC.2016.7531383
4. Muhammad I, Muhammad N, Alagan A, Ashfaq A (2015) Wireless sensor network optimization: multi-objective paradigm. Sensors 15:17572–17620. https://doi.org/10.3390/s150717572
5. Chen Z, Li S, Yue W (2014) Memetic algorithm-based multi-objective coverage optimization for wireless sensor networks. Sensors 14:20500–20518

6. Parasuraman R, Fabry T, Molinari L (2014) A multi-sensor RSS spatial sensing-based robust stochastic optimization algorithm for enhanced wireless tethering. Sensors 14:23970–24003
7. Attea BA, Khalil EA, Ozdemir S, Yildiz O (2015) A multi-objective disjoint set covers for reliable lifetime maximization of wireless sensor networks. Wirel Pers Commun 81(2):819–838
8. Rebai M, Le BM, Snoussi H, Hnaien F, Khoukhi L (2015) Sensor deployment optimization methods to achieve both coverage and connectivity in wireless sensor networks. Comput Oper Res 59:11–21
9. Yoon Y, Kim YH (2013) An efficient genetic algorithm for maximum coverage deployment in wireless sensor networks. IEEE Trans Cybern 43(5):1473–1483. https://doi.org/10.1109/TCYB.2013.2250955
10. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197. https://doi.org/10.1109/4235.996017
11. Zitzler E, Laumanns M, Thiele L (2001) SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In: Proceeding of evolutionary methods design, optimization and control application to industrial problems, p 95–100.
12. Li H, Zhang Q (2009) Multi-objective optimization problems with complicated pareto sets, MOEA/D and NSGA-II. IEEE Trans Evol Comput 13(2):284–302. https://doi.org/10.1109/TEVC.2008.925798
13. Zhang Q, Li H (2007) MOEA/D: a multi-objective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput 11(6):712–731
14. Cheng R, Jin Y (2015) A multi-objective evolutionary algorithm using Gaussian process based inverse modeling. IEEE Trans Evol Comput 19(6):838–856
15. Deb K, Jain H (2014) An evolutionary many-objective optimization algorithm using reference-point-based nodominated sorting approach, part 1: solving problems with box constraints. IEEE Trans Evol Comput 18(4):577–601
16. Zhang X, Tian Y, Jin Y (2015) A knee point driven evolutionary algorithm for many-objective optimization. IEEE Trans Evol Comput 19(6):761–776
17. Zeng GQ, Chen J, Li LM (2016) An improved multi-objective population-based extremal optimization algorithm with polynomial mutation. Inf Sci 330:49–73
18. Chen MR, Lu YZ (2008) A novel elitist multiobjective optimization algorithm: multiobjective extremal optimization. Eur J Oper Res 188:637–651
19. Zeng GQ, Chen J, Dai YX et al (2015) Design of fractional order PID controller for automatic regulator voltage system based on multi-objective extremal optimization. Neurocomputing 160:173–184
20. Li LM, Lu KD, Zeng GQ, Chen MR (2016) A novel real-coded population-based extremal optimization algorithm with polynomial mutation: a non-parametric statistical study on continuous optimization problems. Neurocomputing 174:577–587
21. Storn R, Price K (1997) Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359
22. Xue F (2005) Modeling and convergence analysis of a continuous multi-objective algorithm. In: Proceedings of IEEE international conference on evolutionary computation, p 228–235
23. Robic T (2005) DEMO: differential evolution for multi-objective optimization. In: Proceedings of IEEE 3rd international conference on evolutionary multi-criterion computation, p 520–533
24. Du W, Leung S, Tang Y, Vasilakos AV (2017) Differential evolution with event-triggered impulsive control. IEEE Trans Cybern 47(1):244–257. https://doi.org/10.1109/TCYB.2015.2512942
25. Bi XJ, Wang C (2017) An improved NSGA-III algorithm based on elimination operator for many-objective optimization. Memet Comput 9:361–383. https://doi.org/10.1007/s12293-017-0240-7
26. Meza J, Espitia H, Montenegro C, Crespo RG (2016) Statistical analysis of a multi-objective optimization algorithm based on a model of particles with vorticity behavior. Soft Comput 9:3521–3536. https://doi.org/10.1007/s00500-015-1972-2
27. Chong JK (2016) A novel multi-objective memetic algorithm based on opposition-based self-adaptive differential evolution. Memet Comput 8(2):147–165. https://doi.org/10.1007/s12293-015-0170-1
28. Xu YL, Fang JA, Zhu W (2013) Differential evolution for lifetime maximization of heterogeneous wireless sensor networks. Math Probl Eng 2(1):1. https://doi.org/10.1155/2013/172783
29. Tian Y, Cheng R, Zhang XY, Jin YC (2017) PlatEMO: a matlab platform for evolutionary multi-objective optimization educational forum. IEEE Compt Intell Mag 12(4):73–87. https://doi.org/10.1109/MCI.2017.2742868
30. Zhang X, Tian Y, Cheng R, Jin Y (2015) An efficient approach to non-dominated sorting for evolutionary multi-objective optimization. IEEE Trans Evol Comput 19(2):201–213. https://doi.org/10.1109/TEVC.2014.2308305