Hindawi
Discrete Dynamics in Nature and Society
Volume 2022, Article ID 2021535, 12 pages
https://doi.org/10.1155/2022/2021535

Research Article

Intelligent Warehouse Robot Scheduling System Using a Modified Nondominated Sorting Algorithm

Jia Ma,[1] **Shujun Yang**,[2] **and Hao Jing**[1]

[1]*College of Economics and Management, Shenyang Aerospace University, Shenyang 110136, China*
[2]*College of Software, Northeastern University, Shenyang 110819, China*

Correspondence should be addressed to Shujun Yang; yangshujun@stumail.neu.edu.cn

Received 29 March 2022; Revised 24 April 2022; Accepted 13 May 2022; Published 15 June 2022

Academic Editor: Shi Cheng

In intelligent warehouse, the problem of transporting goods in intelligent warehouse is becoming increasingly complex, and the traditional way of automatically guiding vehicles (AGVs) is inefficient, so automated robot systems are introduced into intelligent warehouses. In this paper, a task assignment model for robots is presented with the transportation problem of robots in intelligent warehouse as the research background. To solve the robot task assignment problem in intelligent warehouse, a novel Pareto-based multiobjective optimization algorithm (MOEA) is proposed, and the aggregation function is invoked to replace the crowding distance; the brain storm operator is used for crossover and mutation. Finally, the ability of the algorithm to solve the benchmark test problem suite and real-world problems is experimentally confirmed.

1. Introduction

Recently, the logistics industry is facing more fierce competition, forcing the logistics industry to adopt cheap and efficient automatic robots to replace the traditional automatic guidance vehicles (AGVs) and adopt new intelligent robot scheduling system to reduce operating costs and improve storage efficiency [1]. From the perspective of warehouse management, it is important to quickly transport goods in the warehouse to reduce the production and transaction cycles. In other words, the basic task of intelligent warehouse multirobot scheduling system is to store, transport, and extract goods [2]. Therefore, the task allocation program needs to allocate tasks rationally and effectively. In the process of multirobot task scheduling, a group of tasks need to be scheduled in an optimal way (for example, allocation and execution).

At present, there are some metaheuristic algorithms to handle the robot allocation problems. Whale optimization algorithm (WOA) is used to handle mobile robot assignment in the intelligent manufacturing system [3]. Hyperheuristic algorithms based on stigmergy and flocking are used to solve the robot search problem in the unmanned space [4]. A generalized graph-based heuristic algorithm is used to solve a dynamic route planning problem during robot movement [5]. *Sam* uses metaheuristic algorithm to provide carton manufacturers with an optimal robot transportation strategy for cases with more autonomous robots [6]. *Faiza* proposes a (Grey wolf optimization and particle swarm optimization) PSO-GWO algorithm to solve the problem of mobile robot avoidance of obstacles [7]. *Liu* proposes two dynamic order planning algorithms to reduce the order delay problem in smart warehouses [8]. *Tan* solves the vertical picking problem in the warehouse using the PSO algorithm [9].

However, most recent work on task assignment in intelligent warehouse has focused only on single-objective minimization, i.e., it has considered only the minimization of overall time and not the minimization of individual autonomous robot time [10]. For this reason, we build optimization problems based on the characteristics of robot task assignment in intelligent warehouse, which includes two objective functions; the first objective function is minimization of overall time for the robots to perform tasks and the other objective function is to maximize the time a single robot spends performing tasks. Therefore, this is a

multiobjective optimization problem (MOP) [11], where multiple objectives need to be optimized simultaneously. Meanwhile, with the increase of orders in warehouse, the scheduling of multiple robots in the intelligent warehouse will become more difficult [12].

For real-world MOPs with multiple attributes, the multiobjective optimization algorithm (MOEA) appears [13–15]. Among these MOEAs, Pareto-based method has always been an effective method to solve MOPs [16], and the most representative algorithms are NSGA-II [17] and SPEA2 [18]. In these algorithms, nondominated sorting has great advantages in solving 2,3-objective optimization problems, it allows nondominated stratification of the population and then selects individuals which satisfy the conditions to the next generation.

In summary, a novel algorithm which uses nondominated sorting and maximin aggregation function was proposed for the task allocation problem in intelligent warehouse. In this algorithm, nondominated sorting is used to quickly select solutions, and maximin function and one-by-one strategy are used to maintain the uniform distribution of the population. Finally, brain storm operator is used to generate new individuals [19]. Some effective optimization properties of maximin aggregation function will be introduced in the following sections. For the proposed algorithm, the maximin aggregation function can better replace the crowdedness distance to evaluate the contribution of individuals in the same nondominated layer to the new population, the one-by-one comparison strategy can more accurately select a more suitable new solution from the candidate solutions, and the randomness of the brain storm optimization operator helps the algorithm to jump out of the local optimum and enhance the global search ability of the algorithm.

The contributions of this paper are as follows:

(i) The task assignment problem for the intelligent warehouse incorporating minimization of overall multirobot time and maximization of individual robot time is proposed. For this task scheduling model, a different crossover mutation operator is adopted.

(ii) For this multirobot scheduling problem, we propose a new Pareto-based algorithm, which also uses maximin function and one-by-one comparison strategy as a supplement, and uses Brain Storm operator to select the parents for crossover and mutation.

(iii) The results of simulation experiments on ZDT and DTLZ benchmark test suites and task assignment problem in intelligent warehouse verify the excellent performance of the algorithm in solving MOPs.

The other sections are as follows. Section 2 presents a mathematical model for robot task assignment in the intelligent warehouse. In Section 3, related technologies and the proposed algorithm MB-NSGA-II are shown. The experimental results are summarized and analyzed in Section 4. Finally, Section 5 gives the conclusion.

2. Problem Formulation

2.1. Background Description. For the intelligent warehouse, a batch of goods after delivery to the warehouse, the manager will provide a list of tasks to the intelligent warehouse system based on demand [20]. What the task assignment in the intelligent warehouse needs to do is to reasonably allocate this group of task sequences to the autonomous robots. Among them, the tasks are divided into the following categories: (1) Inbound task, transporting goods to the warehouse; (2) Transportation task is to move goods from one shelf to another according to the needs of the manager; (3) Outbound task, transporting goods out of the warehouse.

The two-dimensional plane diagram of the intelligent warehouse is shown in Figure 1, in which the neatly arranged look grid represents the shelves. The white cells in the shelves indicate that the shelves are empty because there are no goods stored on them. The automatic robot walks in the aisle to reach the position of the shelf to transport goods. In this way, the information such as the shelf of the intelligent warehouse and the walking range of the robot can be clearly displayed. In the example Figure 1, all goods enter the warehouse from the bottom right and leave the warehouse from the top left.

For the robot task assignment issues, the two objective functions of the task assignment model are minimizing the total time for robots to perform tasks and minimizing the time for a single robot to perform tasks, respectively. Assuming that there are m intact robots that can move freely and n tasks to be assigned in the intelligent warehouse, the robots can assign inbound tasks, transportation tasks, and outbound tasks, respectively. When a robot completes a series of tasks, it will encounter the following situations. For the inbound task and transportation task, the coordinates of task t_i are assumed to be (x_i, y_i). When the time consumption between task and task is not considered, the time consumption of robot executing task t_i is $TC_{\text{inbound}}(t_i)$. Therefore, time required for outbound task $TC_{\text{outbound}}(t_i)$ is

$$\text{TC}_{\text{outbound}}(t_i) = \left(\left|x_i - x_{\text{in}}\right| + \left|y_i\right|\right). \tag{1}$$

The time consumption of the inbound task $TC_{\text{inbound}}(t_i)$ is

$$\text{TC}_{\text{inbound}}(t_i) = \left(\left|x_i\right| + \left|y_i - y_{\text{out}}\right|\right). \tag{2}$$

Unlike the inbound task and the outbound task, the transportation task is different. In the transportation task, assuming that task t_i needs to transport goods from (x_i, y_i) to (m_i, n_i), its formula is as follows:

$$\text{TC}_{\text{trans}}(t_i) = \left(\left|x_i - m_i\right| + \left|y_i - n_i\right|\right). \tag{3}$$

In the robot task scheduling model, in addition to the time consumption of autonomous robot executing tasks, there is also the time consumption between tasks. Assuming that the destination (x_i, y_i) of a robot executing task t_i and the starting point of the next task z_i to be executed by the robot is (m_i, n_i), the time consumption between two tasks is

$$C_{\text{between}}(t_i, z_i) = \left(\left|m_i - x_i\right| + \left|n_i - y_i\right|\right). \quad (4)$$

Therefore, the multirobot task assignment problem is transformed into two objective functions: maximizing the time consumption of a single robot and minimizing the total time for robots to perform tasks. Maximizing the time consumption of a single robot refers to the maximum time consumption MRC of a single robot among the robots completing the task, as shown in the following formula:

$$\begin{aligned} \text{MRC} &= \max \text{TC}_{\text{total}}(r_i), \\ \text{TC}_{\text{total}}(r_i, S_i) &= \sum_{j=1}^{k} \text{TC}\left(t_{i,j}\right) + \sum_{n=2}^{k} C_{\text{between}}\left(t_{i,n-1}, t_{i,n}\right), \end{aligned} \quad (5)$$

where TC_{total} denotes the total time consumed by the robot r_i to execute a task sequence S_i: $t_{i1} \longrightarrow t_{i2} \longrightarrow t_{i3} \cdots \longrightarrow t_{ik}$, t_{ik} denotes the i_{th} robot performing the k_{th} task. $C_{between}$ $(t_{i,n-1}, t_{i,n})$ represents the time consumed by the i_{th} robot when executing the task, k denotes the total number of tasks performed by the i_{th} robot, and n denotes the n_{th} task in execution.

Total robot time consumption minimization (MTC) is the second objective function; we need to minimize the total time for robots to perform tasks, and the formula is shown below:

$$\text{MTC} = \sum_{i=1}^{N} \text{TC}_{\text{total}}(r_i, S_i), \quad (6)$$

where N denotes the total number of robots in the intelligent warehouse.

All the above time consumption functions can be calculated using the Manhattan distance, since the speed of the robot in the warehouse is set to 1 m/s. The total distance traveled by the robots is also equal to the total time consumed by the robots.

2.2. Generation of Offspring. In the iterative optimization of discrete decision variables, some special crossover and mutation methods are used to form new offspring.

To solve the robot task assignment problem, we redesigned chromosome part and divided the chromosome into two parts. The first part is a set of task sequences, and the other part contains the number of tasks undertaken by each robot, and the sum of numbers is the total number of tasks in the task list. Figure 2 illustrates the relationship between the first part of the chromosome and the second part of the chromosome. As shown in Figure 2, robot 1 performs three tasks, robot 2 performs two tasks, 4 and 5, and robot 3 performs the remaining tasks.

For discrete decision variables, the conventional crossover and mutation operators [21] are not appropriate. Therefore, for this particular chromosome, two crossover operators are used. For these two parts of the chromosomes, we use order crossover [22] and simulated binary crossover.

Mutation operation can usually increase the random exploration ability of the algorithm, preventing algorithms from falling into local optimal. For discrete decision variables, we use slight mutation to replace polynomial mutation. The mutation process is shown in Figure 3. Firstly, we select a point from these points as the mutation point, then select a substring from the chromosome, and finally insert the substring behind the mutation point to form a new string of chromosomes.

3. The Proposed Method

In recent research, most of the literature studies tend to solve MOPs based on Pareto-optimal, allowing the algorithm to keep approaching the true Pareto optimal front through nondominated sorting and many improved Pareto-based methods [23]. In addition, some population diversity conservation mechanisms ensure that populations remain well distributed in the objective space after nondominated sorting.

3.1. Construction of Nondominated Solution Set. To reduce the high time complexity of constructing nondominated solution sets in NSGA, NSGAII proposed a new nondominated sorting method to select solutions for the new population. Given the good performance of NSGAII in dealing with MOPs, here we use this hierarchical nondominated solution set construction scheme to construct new populations.

In the nondominated sorting process, the nondominated individuals are first selected into the first stratum, then, the second stratum is the set of nondominant individuals obtained after removing the first stratum individuals from the population, and so on. In the final selection, individuals in the first stratum are considered first, and then individuals in the second stratum is considered until the new population size is satisfied. Specific details can be found in reference [17]. Algorithm 1 is the process of constructing the nondominated solution set.

3.2. Maximin Fitness Function. In the continuous development of multiobjective optimization, some fitness functions are used as indicators to evaluate individuals in the population, among which the maximin function is applied to multiobjective optimization with its own characteristics. The formula is as follows:

$$\text{fitness}^i = \max_{j \neq i}\left(\min_k\left(f_k(x_i) - f_k\left(x_j\right)\right)\right), \quad (7)$$

where k denotes objective from 1 to m, i represents the i_{th} individual, and j denotes any individual except i_{th} individual. And the properties of the maximin function are as follows [24, 25]:

(1) The maximin fitness can reflect the dominance between individuals. The maximin value greater than, equal to, or less than zero means that the individual is a dominated individual, the individual is weakly dominated, and individual is nondominated in the population, respectively.

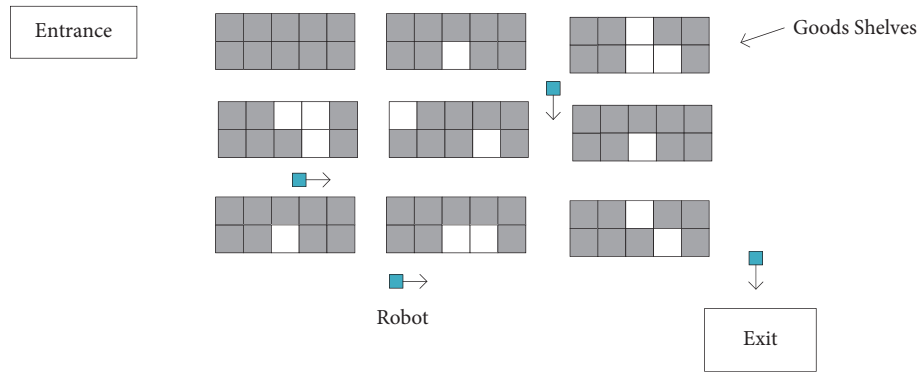

Figure 1: Floor plan of intelligent warehouse.

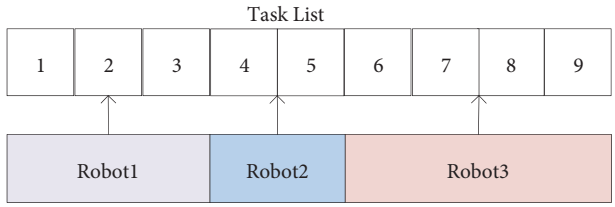

Figure 2: Example diagram of task assignment.

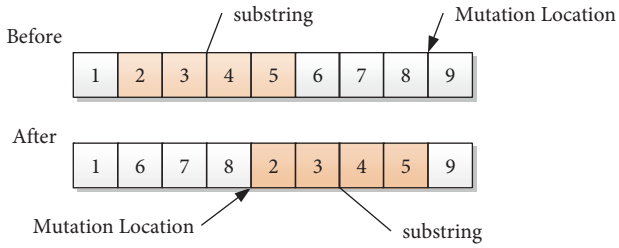

Figure 3: Example diagram of slight mutation.

(2) The maximin fitness reflects the presence or absence of clustering around an individual, as shown in Figure 4.

(3) The maximin function can be used to reflect the relationship between the dominated individual and the nondominated front. (see Figure 4).

3.3. One-by-One Comparison Strategy. The contribution of individual to the population can be effectively assessed by the maximin aggregation function. However, if only the maximin aggregation function is used to select suitable individuals, then the selection process may encounter the equivalence selection dilemma, that is, some individuals have equal maximin fitness, and there is no way to select a better individual from them, and then it is necessary to use a one-by-one comparison strategy to select a better individual from these individuals. The following figure shows an example of using the one-by-one comparison strategy to further select individuals [26].

Through one-by-one comparison strategy, the problem of equivalence selection can be solved efficiently. Suppose that three individuals are selected from the four individuals in Figure 5. First, the maximin aggregation function is used to evaluate these individuals. A with the minimum value is selected from these individuals to become the first candidate individual, then continue to select individuals from the remaining individuals through the maximin aggregation function, and the individual D is selected into the new population. Next, individuals B and C are compared with the new population by maximin function, and we select individual B to join the new population. Finally, the new population is formed by individuals A, B, D. As can be seen in the figure, the selected individuals remain well distributed.

3.4. Brain Storm Operator. Unlike most multiobjective algorithms, we use brain storm operator as a strategy for selecting parents, which increases the ability of the algorithm in terms of exploration and makes it less prone to fall into local optima. We demonstrate this part of the process with Algorithm 2.

3.5. Proposed Algorithm: MB-NSGA-II. The flow and framework of the algorithm are described below. In the first step, the population P is randomly initialized. And the brain storm operator is used to cluster parent population and then select appropriate chromosomes as parent chromosomes based on suitable conditions and generate the new population Q by crossover and mutation. After this, parent P and offspring Q are combined to form P'. Finally, the new population is regenerated by comprehensive selection. Keep cycling through the above steps until the *MAX_ Fitness Evaluations* is reached. The general framework of MB-NSGA-II is shown in Algorithm 4.

3.5.1. Normalization. In multiobjective optimization algorithms, normalization can effectively solve the problem that different objectives have widely varying ranges of values [27]. The formula is as follows:

$$f_m^{'}(x) = \frac{f_m(x) - z_m^{\text{lower}}}{z_m^{\text{upper}} - z_m^{\text{lower}}}, \tag{8}$$

where z_m^{lower} denotes the lower bound of the m_{th} objective function, z_m^{upper} denotes the upper bound of the m_{th} objective

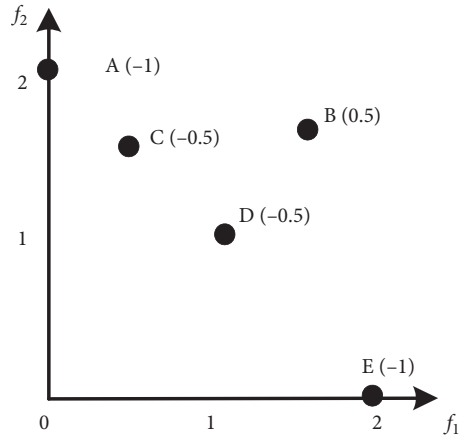

Figure 4: Properties of the maximin function.

Input: P (population)
Output: $P_1, P_2, \ldots, P_n$ (stratification results)
(1) $\forall p \in P, S_p = 0, D_p = \varnothing, i = 1$;// S_p denotes the size of set of solutions that dominate p, D_p i S denotes the set of solutions dominated by p
(2) **for** $\forall p \in P$
(3) **for** $\forall q \in P$
(4) if $(p \succ q)$ then $D_p = D_p \cup q$
(5) elseif $(q \succ p)$ then $S_p = S_p + 1$
(6) **end** for q
(7) if $(S_p = 0)$ then $P_1 = P_1 \cup p$
(8) **end** for p
(9) **while** $(P_i \neq \varnothing)$ // P_i indicates the number of non-dominated layers
(10) {POP= $\varnothing$;
(11) for $\forall p \in P_i$
(12) for $\forall q \in s_p$, $n_q = n_q - 1$;
(13) if $(n_q = 0)$ POP = POP $\cup q$
(14) end for p
(15) $i = i + 1$;
(16) P_i = POP ;
(17) **end** for while
(18) **end**
(19) **return** $\mathbf{P_1}, \mathbf{P_2}, \ldots, \mathbf{P_n}$

Algorithm 1: Nondominated sorting.

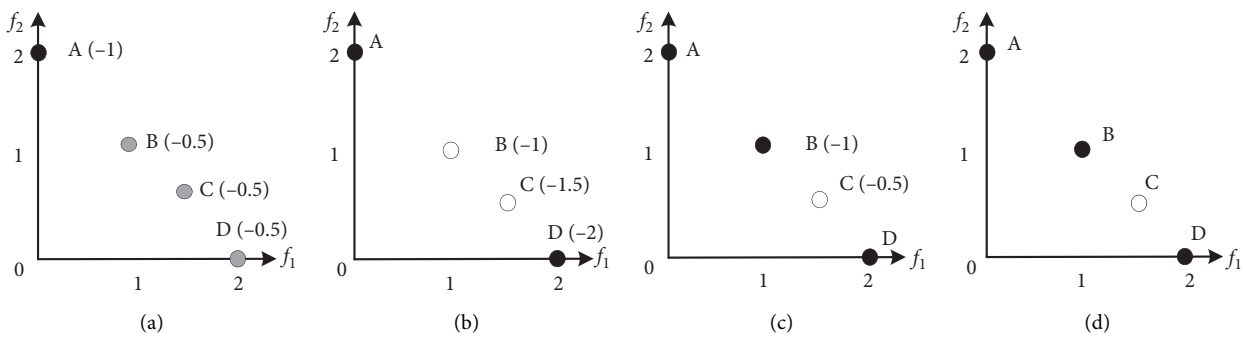

Figure 5: Diagram of the process of solving the equivalence selection problem by one-by-one comparison strategy.

(1) Objective normalization (P)
(2) **for** $i = 1: S - 1$
(3) **Clustering**: Dividing the population into m clusters using the k-means method.
(4) $p_{\text{gen}} = \text{rand}(0, 1)$
(5) **if** $p_{\text{gen}} < p_{\text{fixed}}$
(6) $p_{\text{one}} = \text{rand}(0, 1)$ //Select a random cluster from m clusters
(7) **if** $p_{\text{one}} < p_{\text{one-fixed}}$//$p_{\text{one-fixed}}$ is a predetermined value
(8) Select the clustering center and the individual Pop_i as parent generation of population
(9) **else**
(10) Select a random individual and the individual Pop_i as parent generation of population
(11) **end**
(12) **end**
(13) Compare the newly generated individuals with the present ones by maximin function, where the better one is preserved.
(14) **end**

Algorithm 2: Brain storm operator (P, S).

function, m means this is the m_{th} objective function, and $m \in \{1, 2, \ldots, M\}$.

3.5.2. Comprehensive Selection. As shown in Algorithm 3, comprehensive selection consists of the following steps. First, nondominated sorting for the population *Pop* is executed to divide *Pop* into strata from 1 to n. If there is only one layer of nondominated layer, the maximin fitness value of the population and the ideal point is calculated, and individuals are selected into the new population one by one. Then, individuals were selected layer by layer starting from the first layer until the new population size is satisfied. When proceeding to stratum i, if the predetermined S is less than the number of individuals in the i_{th} stratum, the set of individuals P_s is selected using a one-by-one comparison strategy (Line 8 Algorithm 5), and finally P_s is merged with *Pop* to obtain the new population *Pop*.

3.5.3. One-by-One Comparison Strategy. The one-by-one comparison strategy is shown in Algorithm 5. One-by-one comparison strategy is used to compare individuals at the same stratum and select a subset of individuals from them. Firstly, let P_n be empty. First, each objective of the population P'' is normalized by (7).Then, if the population size of S is larger than *size*, comparing individual fitness values in S using the maximin aggregation function, select the individual with the smallest maximin fitness from it to join the *MF*, and so on, until P_n reaches the new population size.

3.6. Computational Complexity Analysis of MB-NSGA-II. To study the computational cost consumption of the algorithm, we summarize and discuss the time complexity of one generation in Algorithm 3. In addition to crossover and mutation operators, computational complexity is affected by the nondominated sorting and one-by-one comparison strategy.

First, normalization requires a computational complexity of $O(mN)$. And the nondominated sorting's computational complexity in Algorithm 1 is $O(mN^2)$. The complexity of calculating the maximin fitness value is $O(mN)$. The step of one-by-one comparison strategy needs $O(mN^2)$. In a word, the overall complexity of the algorithm is $O(mN^2)$ in one generation.

4. Experimental Study

The experimental part contains two sections; the first section discusses and analyzes the effectiveness of MB-NSGA-II on the benchmark test suites; the second part describes how we use our proposed algorithm to solve practical problems in intelligent warehouse.

4.1. Experimental Results on Benchmarks. This section provides the experimental results of MB-NSGAII with two other start-of-art multiobjective algorithms, NSGAII and IBEA, on the benchmark test suites DTLZ [28] and ZDT [29]. On DTLZ, each test instance was experimented on the 2,3,5- objectives, and on ZDT, the experiments were run on the 2-objective. Each algorithm needs to be run 20 times on each test issue. We can discuss the results of this experiment in terms of comprehensive performance indicator IGD.

4.1.1. Results and Discussion on ZDTs and DTLZs. The ZDT test suite contains six test problems with different characteristics, each test instance involving a feature that would cause the algorithm to have difficulty converging during optimization (e.g., multimodality). Through these test instances, it is possible to analyze which problems the algorithm excels in. DTLZ designs the problems through a systematic approach, and its decision variables and objectives are scalable, facilitating the algorithm to experiment in solving MaOPs. DTLZ increases the difficulty of the test problem by introducing manageable difficulties, facilitating the algorithm to experiment on test problems with any number of objectives. To more fully validate the effectiveness of several compared algorithms in solving MOPs, we chose to conduct experiments on 2,3, and 5 objectives. M in Table 1 indicates the objective number. The numbers of decision variable are set, as shown in Table 1. Details can be found in reference [28,29].

Input: *Off* (Offspring population), S (population size)
Output: *Pop* (new population)
(1) $P_1, P_2, \ldots, P_n$ = **Nondominated sorting** (*Off*)
(2) Pop = $\varnothing$;
(3) if $n = 1$
(4) Pop = **One-by-one comparison strategy** (Off, z^*, S);
(5) else
(6) **while** m
(7) Pop = Pop $\cup$ P_i
(8) $i = i+1$
(9) **end for while**
(10) P_s = **One-by-one comparison strategy** (S_i, Pop, $S - |\text{Pop}|$) ;
(11) Pop = $P'' \cup P_s$
(12) **end for if**
(13) **return *Pop***

Algorithm 3: Comprehensive Selection (Off, S).

(1) Initialization: Initial population P containing N randomly individuals
(2) **while** *Fitness Evaluations* < *MAX_Fitness Evaluations* **do**
(3) Q = **Brain Storm Operator** (P, s) ;
(4) *Off* = $P \cup Q$
(5) *Pop* = **Comprehensive _Selection** (*Off*, S);
(6) **end for while**
(7) **return *Pop***

Algorithm 4: General Framework of MB-NSGA-II.

Input: S: individuals in stratum I, P'': new population, size: number of individuals needed for new population
Output: P_s: the set of individuals provided to the new population
(1) $P_n = \varnothing$;
(2) Objective normalization (Q);
(3) **if** $|S|$>size
(4) **for** $i = 1$: size-1
(5) $MF = \text{argmin}_{i=\{1,2,\ldots,|S|\}} \{\text{fitness}^i\}$;//individual with minimal maximin fitness between individuals in S_i and the new population P
(6) $P_n = P_n, \leq MF$;
(7) **end for**
(8) **end if**
(9) **Return P_s**

Algorithm 5: One-by-one comparison strategy (S, P'', size).

To more visually reflect the ability of the algorithm on the benchmark test instances, we choose the comprehensive evaluation indicator HV and IGD as the basis for judging the performance of the algorithms. These two metrics are composed of different formulas, so the final population obtained by the algorithm can be evaluated from different perspectives. The formula and detailed description can be found in [30,31].

4.1.2. Experimental Settings

(i) Population size and termination conditions: we use population of size 100 for DTLZ with 2,3 objectives and 200 for DTLZ with 5 objectives. For ZDTs, we select population of size 100 for the three algorithms. For DTLZ1-4 and ZDT1-6, the algorithms will end after 500 generations.

(ii) Crossover and mutation: in this experiment, binary crossover and polynomial mutation were used as operators for crossover and mutation. Set 1.0 as the crossover probability and set $1/D$ as the probability of mutation, where D is the number of decision variables.

(iii) Selection of indicator: for IBEA, we select I_ε^+ as a performance indicator to evaluate the strengths and

Table 1: Setting of decision variables.

Problem	Number of decision variables
ZDT1-3	30
ZDT4,6	10
ZDT5	80
DTLZ1	$M+4$
DTLZ2-4	$M+9$

Table 2: Experimental results of IGD on ZDTs (mean and standard deviation).

Problem	M	NSGAII	IBEA	MBNSGAII
ZDT1	2	4.7879*e*-3 (1.31*e*-4) -	4.0901*e*-3 (1.01*e*-4) -	3.9627*e*-3 (6.92*e*-5)
ZDT2	2	4.9504*e*-3 (2.24*e*-4) +	8.4212*e*-3 (7.74*e*-4) -	5.3033*e*-3 (1.32*e*-4)
ZDT3	2	5.4102*e*-3 (1.31*e*-4) -	1.6020*e*-2 (8.05*e*-4) -	5.0404*e*-3 (1.24*e*-4)
ZDT4	2	5.0667*e*-3 (4.10*e*-4) +	2.2761*e*-2 (1.59*e*-2) -	6.3474*e*-3 (2.30*e*-4)
ZDT5	2	5.1086*e*-1 (8.31*e*-2) +	1.8513*e*+0 (4.39*e*-1)=	1.6593*e*+0 (7.13*e*-1)
ZDT6	2	3.7200*e*-3 (1.20*e*-4)=	4.4425*e*-3 (1.14*e*-4) -	3.7006*e*-3 (5.32*e*-5)

weaknesses of individuals. The details of the indicator can be seen in [32].

4.1.3. Analysis of Experimental Results. Comparison data of MB-NSGA-II, NSGA-II, and IBEA on DTLZ and ZDT are presented in Tables 2–5. These tables show the IGD and HV means and standard deviations of these compared algorithms on the benchmark test problems, and highlights the algorithm that worked best on these test problems. The Wilcoxon rank sum test with a significance level of 0.05 was used for the standard deviation analysis when conducting the experiment. The symbols "+," "−," and "=" are used to indicate that the algorithms are significantly better, significantly worse, or not significantly different between the compared algorithms and MB-NSGA-II.

In Table 2 and Table 3, MB-NSGA-II has an outstanding performance on both DTLZ test problems and ZDT test problems, and it ranks first on 11 out of 18 test instances, and it is also competitive on other test instances. In particular, it achieved the best results on DTLZ1 for both 2,3,5 objectives, and also ranked first on DTLZ3 for two objective problems. Of course, NSGA-II also had good results, obtaining the best results on the six problems. In Table 3, MB-NSGAII ranked first in performance on 8 test problems, while NSGAII values achieved good results on three problems. It can be seen that MB-NSGA-II achieved the best results on all test problems on DTLZ1 with hyperplane PF, which is a significant improvement over NSGAII. On DTLZ2, MB-NSGAII achieves the best results on 5 objectives, which also proves the effectiveness of maximin aggregation function in high-dimensional objective space. For DTLZ3, which is easily trapped in the local optimum and difficult to converge to the global optimum, MB-NSGAII achieves the best results on all three test problems except for the 2-objective. On DTLZ4, obtained after modifying the DTLZ2 function, MB-NSGAII achieves the best results on the 2-objective and 5-objective. Next two tables show the results of the algorithm under HV performance indicators. MB-NSGA-II has outstanding performance on most test instances. It can be seen from this that the validity of using the aggregation function to evaluate the populations. Of course, IBEA and NSGA-II also achieved the first in a few problems. The results in the table also prove that the strategy used in MB-NSGA-II is effective compared to NSGA-II.

4.2. Results and Discussion on Task Assignment Model. To confirm the ability of the algorithm to solve real-world problems, we built the robot assignment problem in the warehouse and solved it using the algorithm MB-NSGA-II. We complement the intelligent warehouse's situation mentioned above by first setting a two-dimensional coordinate system with the coordinates of the shipping gate of the warehouse as (0,100) and the coordinates of the incoming gate of the warehouse as (100,0). The intelligent warehouse has a certain number of autonomous robots in perfect condition and equal shipping speed of 1 m/s. All tasks to be performed in the warehouse are generated randomly. For a uniform comparison, we set the population of 500 for the compared algorithms and the termination condition to 200 generations.

To fully validate the performance of the algorithm in solving the task scheduling model, we assume that 5 or 10 robots are used to solve 100 randomly generated tasks; and 10 or 20 robots are used to solve 500 randomly generated tasks.

For real multiobjective optimization problems, we usually use the comprehensive performance indicator HV to measure the merit of the algorithm. To use HV as an indicator, the maximum values on each objective are selected as a reference points. In this real-world MOP, we select the maximum value that can be reached on each objective separately, and the specific details of HV can be found in [30]. Tables 6 and 7 show the HV values that the algorithm can obtain on 100 and 500 tasks, respectively. From the tables, the results show that MB-NSGAII obtains the maximum HV on all these tasks. That is, the populations

Table 3: Experimental results of IGD on several DTLZs (mean and standard deviation).

Problem	M	NSGAII	IBEA	MBNSGAII
DTLZ1	2	2.2068*e*-3 (7.35*e*-5) -	7.9965*e*-2 (7.99*e*-3) -	2.0230*e*-3 (5.45*e*-5)
	3	2.8447*e*-2 (1.33*e*-3) -	1.6616*e*-1 (2.51*e*-2) -	2.2376*e*-2 (3.68*e*-4)
	5	1.7146*e*-1 (1.25*e*-1) -	1.8372*e*-1 (2.16*e*-2) -	5.1833*e*-2 (3.37*e*-4)
DTLZ2	2	5.0483*e*-3 (1.75*e*-4) +	1.6671*e*-2 (1.60*e*-3) -	8.5238*e*-3 (2.92*e*-4)
	3	7.3058*e*-2 (2.57*e*-3) =	8.4162*e*-2 (2.51*e*-3) -	7.3973*e*-2 (1.57*e*-3)
	5	2.0563*e*-1 (4.67*e*-3) -	1.9251*e*-1 (1.54*e*-3) -	1.7906*e*-1 (2.44*e*-3)
DTLZ3	2	7.1268*e*-3 (1.39*e*-3) +	3.3913*e*-1 (8.73*e*-3) -	9.8870*e*-3 (1.10*e*-3)
	3	1.4200*e*-1 (2.06*e*-1) =	4.7891*e*-1 (4.51*e*-3) -	7.8834*e*-2 (2.64*e*-3)
	5	7.5535*e*-1 (7.31*e*-1) -	5.9365*e*-1 (9.76*e*-3) -	1.8301*e*-1 (2.58*e*-3)
DTLZ4	2	1.5241*e*-1 (3.11*e*-1) -	4.5174*e*-1 (3.75*e*-1) -	8.1849*e*-2 (2.32*e*-1)
	3	1.5811*e*-1 (2.77*e*-1) +	8.2918*e*-2 (2.68*e*-3) =	2.6157*e*-1 (2.41*e*-1)
	5	2.0630*e*-1 (3.41*e*-3) -	2.1267*e*-1 (7.27*e*-2) -	1.8232*e*-1 (1.93*e*-3)

Table 4: Experimental results of HV on several DTLZs (mean and standard deviation).

Problem	M	NSGAII	IBEA	MBNSGAII
DTLZ1	2	5.8121*e*-1 (3.60*e*-4) -	3.9969*e*-1 (1.87*e*-2) -	5.8162*e*-1 (3.86*e*-4)
	3	8.2082*e*-1 (5.02*e*-3) -	4.7837*e*-1 (5.50*e*-2) -	8.3698*e*-1 (1.86*e*-3)
	5	6.9302*e*-1 (3.39*e*-1) -	7.5172*e*-1 (4.51*e*-2) -	9.7599*e*-1 (1.40*e*-3)
DTLZ2	2	3.4654*e*-1 (1.97*e*-4) -	3.4613*e*-1 (2.17*e*-4) -	3.4731*e*-1 (1.09*e*-4)
	3	5.2551*e*-1 (3.99*e*-3) -	5.5477*e*-1 (1.48*e*-3) -	5.5689*e*-1 (9.61*e*-4)
	5	6.7753*e*-1 (7.73*e*-3) -	8.0904*e*-1 (1.36*e*-3) +	8.0024*e*-1 (2.49*e*-3)
DTLZ3	2	3.4146*e*-1 (2.73*e*-3) =	1.6879*e*-1 (3.86*e*-3) -	3.4251*e*-1 (2.27*e*-3)
	3	4.5747*e*-1 (1.56*e*-1) -	2.3645*e*-1 (9.09*e*-3) -	5.4725*e*-1 (3.20*e*-3)
	5	4.0235*e*-1 (3.46*e*-1) -	3.8077*e*-1 (6.39*e*-3) -	8.0179*e*-1 (4.10*e*-3)
DTLZ4	2	2.9554*e*-1 (1.08*e*-1) -	1.9305*e*-1 (1.32*e*-1) -	3.2164*e*-1 (8.11*e*-2)
	3	4.8619*e*-1 (1.39*e*-1) =	5.5574*e*-1 (9.75*e*-4) =	4.7330*e*-1 (1.09*e*-1)
	5	6.8256*e*-1 (8.53*e*-3) -	8.0150*e*-1 (2.60*e*-2) =	8.0780*e*-1 (1.73*e*-3)

Table 5: Experimental results of HV on ZDTs (mean and standard deviation).

Problem	M	NSGAII	IBEA	MBNSGAII
ZDT1	2	7.1925*e*-1 (1.76*e*-4) -	7.2017*e*-1 (1.36*e*-4) -	7.2031*e*-1 (8.51*e*-5)
ZDT2	2	4.4399*e*-1 (2.03*e*-4) -	4.4410*e*-1 (1.58*e*-4) -	4.4485*e*-1 (6.70*e*-5)
ZDT3	2	5.9938*e*-1 (8.34*e*-5) -	5.9816*e*-1 (1.17*e*-4) -	5.9973*e*-1 (4.53*e*-5)
ZDT4	2	7.1800*e*-1 (1.09*e*-3) =	7.0312*e*-1 (9.85*e*-3) -	7.1735*e*-1 (5.54*e*-4)
ZDT5	2	8.1676*e*-1 (1.20*e*-2) =	8.1119*e*-1 (9.16*e*-3)	8.1451*e*-1 (3.26*e*-4)
ZDT6	2	3.8826*e*-1 (1.18*e*-4) =	3.8766*e*-1 (1.17*e*-4) -	3.8832*e*-1 (5.60*e*-5)

Table 6: Experimental results of HV when robots perform 100 tasks.

Number of robots	IBEA	NSGA-II	MB-NSGAII
5	0.1379	0.1426	0.1445
10	0.2514	0.2341	0.2549

Table 7: Experimental results of HV when robots perform 500 tasks.

Number of robots	IBEA	NSGA-II	MB-NSGAII
10	0.1388	0.1374	0.1452
20	0.1805	0.1804	0.1853

Table 8: Average time consumption of robots when using 5 robots for 100 tasks.

	IBEA	NSGA-II	MB-NSGAII
Robot 1	3429.3	3358.3	3424.3
Robot 2	3589.9	3705	3719.2
Robot 3	3524.4	3520.3	3418.6
Robot 4	3424.3	3103.9	3031.7
Robot 5	3316.4	3301.3	3128

Table 9: Average time consumption of robots when using 10 robots for 500 tasks.

	IBEA	NSGA-II	MB-NSGAII
Robot 1	8894.5	9296.3	8227.2
Robot 2	8677.2	8882.1	8693.7
Robot 3	8516.4	7974.5	8681.3
Robot 4	8094.8	7616.1	7594.8
Robot 5	9015.4	8169	8170.3
Robot 6	7878	7454.1	7684.7
Robot 7	8511.7	7680.7	7364.1
Robot 8	8244.3	8189.8	9170
Robot 9	8665.4	9522.6	9233.9
Robot 10	8739.7	8210	8620.1

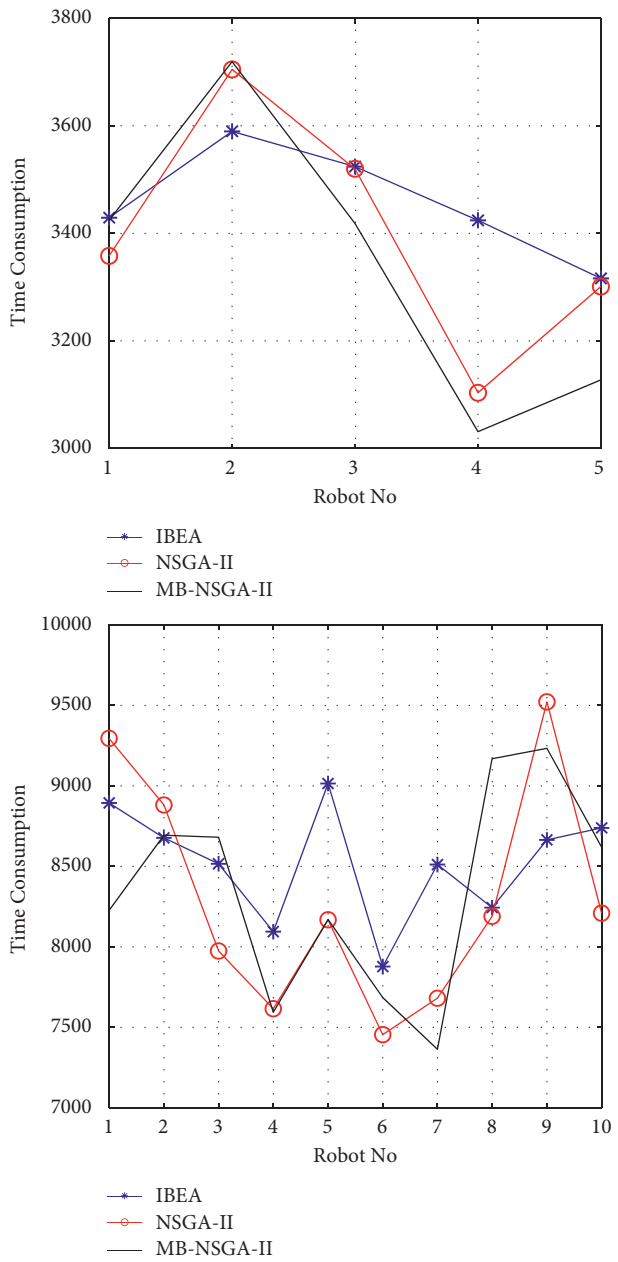

Figure 6: Schematic diagram of the average time consumed by robots to perform tasks.

obtained by MB-NSGAII have better convergence and distribution. Through the above experiments, it can be proved that MB-NSGAII can give managers better solutions.

Tables 8 and 9 show the average time consumption of robots when solving 100 tasks with 5 robots and 500 tasks with 10 robots, respectively. Since each robot runs at a speed of 1 m/s, their average running distance can also be represented by Tables 8 and 9. As can be seen from the tables, the time consumption of each robot is close to each other and load balancing is achieved. Figure 6 shows the average time consumption of the robot when performing 100 tasks and 500 tasks.

5. Conclusion

In this paper, we have proposed a corresponding mathematical model for robot task assignment problem in an intelligent warehouse and solve it by MB-NSGAII, which uses nondominated sorting, maximin aggregation function, and brain storm operator. MB-NSGA-II is validated on the DTLZ and ZDT test suites, and it achieves satisfactory results in solving the real-world MOP with robot task assignment.

Next, we will continue to investigate the use of maximin aggregation function and brain storm operator to solve MOPs. We will also use this approach to solve more real-world MOPs.

Data Availability

The data are not available due to the nature of this research as participants of this study did not agree for their data to be shared publicly.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by Liaoning Social Science Planning Fund Key Project (L20CGL012); Liaoning Science and Technology Department Science Career Public Welfare Research Fund (2021JH4/10100026); the Fundamental Research Funds for the Central Universities No. N2117005; the Joint Funds of the Natural Science Foundation of Liaoning Province und Grant 2021-KF-11-01. It is noted that, unlike earlier work [33], this paper provides some improvements: a new 2-objective mathematical model, i.e., minimum total time consumed by all robots and maximum time consumed by a single robot, for the robot task scheduling problem, and then propose a novel nondominated sorting algorithm to solve it.

References

[1] X. Yao, C. Yuanyuan, Z. Li, and S. Malin, "Green efficiency performance analysis of the logistics industry in China: based on a kind of machine learning methods," *Annals of Operations Research*, vol. 308, no. 1, pp. 727–752, 2022.

[2] H. Qin, J. Xiao, D. Ge et al., "JD.com: operations research algorithms drive intelligent warehouse robots to work," *INFORMS Journal on Applied Analytics*, vol. 52, no. 1, pp. 42–55, 2022.

[3] M. Petrović, M. Zoran, and J. Aleksandar, "A novel methodology for optimal single mobile robot scheduling using whale optimization algorithm," *Applied Soft Computing*, vol. 81, 2019.

[4] M. G. C. A. Cimino, D. Minici, M. Monaco, S. Petrocchi, and G. Vaglini, "A hyper-heuristic methodology for coordinating swarms of robots in target search," *Computers & Electrical Engineering*, vol. 95, Article ID 107420, 2021.

[5] W. Chi, Z. Ding, J. Wang, G. Chen, and L. Sun, "A generalized voronoi diagram-based efficient heuristic path planning method for RRTs in mobile robots," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 5, pp. 4926–4937, 2021.

[6] S. Mosallaeipour, M. G. Nejad, S. M. Shavarani, and R. Nazerian, "Mobile robot scheduling for cycle time optimization in flow-shop cells, a case study," *Production Engineering*, vol. 12, no. 1, pp. 83–94, 2018.

[7] F. Gul, W. Rahiman, S. S. N. Alhady, A. Ali, I. Mir, and A. Jalil, "Meta-heuristic approach for solving multi-objective path planning for autonomous guided robot using PSO-GWO optimization algorithm with evolutionary programming," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 7, pp. 7873–7890, 2021.

[8] Y. Liu, B. Y. Ooi, and D. Qin, "Dynamic order-based scheduling algorithms for automated retrieval system in smart warehouses," *IEEE Access*, vol. 9, pp. 158340–158352, 2021.

[9] Z. Tan, H. Li, and X. He, "Optimizing parcel sorting process of vertical sorting system in e-commerce warehouse," *Advanced Engineering Informatics*, vol. 48, Article ID 101279, 2021.

[10] D. Guo, Z. Lyu, W. Wu, R. Y. Zhong, Y. Rong, and G. Q. Huang, "Synchronization of production and delivery with time windows in fixed-position assembly islands under Graduation Intelligent Manufacturing System," *Robotics and Computer-Integrated Manufacturing*, vol. 73, Article ID 102236, 2022.

[11] L. Ma, X. Wang, X. Wang, L. Wang, Y. Shi, and M. Huang, "TCDA: truthful combinatorial double auctions for mobile edge computing in industrial internet of things," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[12] X. Wu and M. Zhang, "An intelligent algorithm for AGV scheduling in intelligent warehouses," *Lecture Notes in Computer Science*, vol. 12689, pp. 163–173, 2021.

[13] L. Ma, M. Huang, S. Yang, R. Wang, and X. Wang, "An adaptive localized decision variable analysis approach to large-scale multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, 2021.

[14] H. Ishibuchi, T. Noritaka, and N. Yusuke, "Evolutionary manyobjective optimization: a short review," in *Proceedings of the CEC 2008*, pp. 2419–2426, IEEE, Hong Kong, China, 1 Jun. 2008.

[15] L. Ma, N. Li, Y. Guo et al., "Learning to optimize: reference vector reinforcement learning adaption to constrained many-objective optimization of industrial copper burdening system," *IEEE Transactions on Cybernetics*, 2021.

[16] L. S. Batista, F. Campelo, F. G. Guimarães, and J. A. Ramírez, "A comparison of dominance criteria in many-objective optimization problems," in *Proceedings of the CEC 2011*, pp. 2359–2366, IEEE, New Orleans, LA, U.S.A, 5 June 2011.

[17] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *Proceedings of the Parallel*

Problem Solving from Nature PPSN VI, pp. 849–858, Springer, Paris, France, 18 January 2000.

[18] M. Emmerich, N. Beume, and B. Naujoks, "An EMO algorithm using the hypervolume measure as selection criterion," *Lecture Notes in Computer Science 2005*, Springer, pp. 62–76, Berlin, Heidelberg.

[19] Y. Shi, "Brain storm optimization algorithm," *Lecture Notes in Computer Science*, Springer, pp. 303–309, Berlin, Heidelberg.

[20] H. Zhang, Z. Guo, W. Zhang et al., "Layout design for intelligent warehouse by evolution with fitness approximation," *IEEE Access*, vol. 7, pp. 166310–166317, 2019.

[21] K. Deb, K. Sindhya, and T. Okabe, "Self-adaptive simulated binary crossover for real-parameter optimization," in *Proceedings of the GECCO 2007*, pp. 1187–1194, ACM, London, England, U.K, 7 July 2007.

[22] L. Davis, "Applying adaptive algorithms to epistatic domains," in *Proceedings of the IJCAI 1985*, pp. 162–164, ACM, San Francisco, CA, U.S.A, 2 August 1985.

[23] L. Ma, S. Cheng, and Y. Shi, "Enhancing learning efficiency of brain storm optimization via orthogonal learning design," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 11, pp. 6723–6742, 2021.

[24] E. J. P. Solteiro, P. B. D. M. Oliveira, and J. A. T. Machado, "Multi-objective MaxiMin sorting scheme," *Lecture Notes in Computer Science 2005*, Springer, pp. 165–175, Berlin, Heidelberg.

[25] X. Li, "Better spread and convergence: particle swarm multiobjective optimization using the maximin fitness function," *Genetic and Evolutionary Computation - GECCO 2004 2004*, Springer, pp. 117–128, Berlin, Heidelberg.

[26] Y. Liu and D. J. Y. Gong, "A many-objective evolutionary algorithm using a one-by-one selection strategy," *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2689–2702, 2017.

[27] Q. He, X. Wang, Z. Lei, M. Huang, Y. Cai, and L. Ma, "TIFIM: a two-stage iterative framework for influence maximization in social networks," *Applied Mathematics and Computation*, vol. 354, pp. 338–352, 2019.

[28] K. Deb, T. Lothar, L. Marco, and Z. Eckart, "Scalable test problems for evolutionary multi-objective optimization," *Evol. Multi. Opti, Theoretical Advances and Applications*, Springer, pp. 105–145, London, U.K.

[29] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[30] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 29–38, 2006.

[31] C. A. C. Coello and Corte, "Solving multiobjective optimization problems using an artificial immune system," *Genetic Programming and Evolvable Machines*, vol. 6, no. 2, pp. 163–190, 2005.

[32] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," *PPSN VIII Lecture Notes in Computer Science*, Springer, pp. 832–842, Berlin, Germany.

[33] S. Yang, Y. Zhang, L. Ma et al., "A novel maximin-based multi-objective evolutionary algorithm using one-by-one update scheme for multi-robot scheduling optimization," *IEEE Access*, vol. 9, pp. 121316–121328, 2021.