



A novel implicit decision variable classification approach for high-dimensional robust multi-objective optimization in order scheduling

Youkai Xiao¹ · Wei Du¹ · Yang Tang¹

Received: 31 August 2023 / Accepted: 3 February 2024 / Published online: 2 March 2024
© The Author(s) 2024

Abstract

This paper efficiently addresses the high-dimensional robust order scheduling problem. A novel algorithm named dynamic cooperative coevolution based on an implicit decision variable classification approach (DCC/IDVCA) is developed to search for robust order schedules. To significantly reduce the computational resources required for solving the high-dimensional robust order scheduling problem, we propose decomposing the original decision variables through implicit classification methods. First, a novel estimation method is introduced to evaluate the weighted contribution of variables to robustness. This method utilizes historical information, including the variation of the overall mean effective fitness and the frequency of variables being classified into highly robustness-related subcomponents in previous cycles, for evaluating their weighted contribution to robustness. Then, based on the corresponding weighted robustness contributions, the original variables are classified into highly and weakly robustness-related variables. Finally, these two types of variables are decomposed into highly and weakly robustness-related subgroups within a dynamic cooperative coevolution framework and optimized separately. In the experimental section, the proposed algorithm is applied to two practical order scheduling problems in discrete manufacturing industry. The experimental results demonstrate that the proposed algorithm achieves competitive outcomes compared to state-of-the-art high-dimensional robust multi-objective optimization algorithms.

Keywords Implicit decision variable classification · Dynamic cooperative coevolution framework · Robust order scheduling · Weighted robustness contribution

Introduction

Order scheduling is a critical decision-making problem in the supply chain management of the manufacturing industry. The task of order scheduling is to allocate the orders received from retailers to appropriate production lines based on the delivery date of each order. Take apparel industry as an example, manufacturers face significant demand volatility in their received orders. Typically, these orders are received from

retailers close to the selling season, involving a wide range of product types with short lifecycles, resulting in increased labor costs during the production process [1]. Therefore, by making rational arrangements for order scheduling, it is possible to maximize resource utilization, reduce waiting times, improve production efficiency, and lower production costs, thereby enhancing the overall operational efficiency of the supply chain and enabling enterprises to be more competitive in the global market [2, 3]. In addition, in practical production, various disruptions often occur during order scheduling [4], including machine failures and operator illnesses, among others. As a result, the daily production quantities frequently undergo changes throughout the production process. In this case, robust order scheduling with respect to uncertain daily production quantities becomes more practical and meaningful.

The problem of robust order scheduling belongs to the category of robust optimization problems. When searching for candidate solutions capable of generating robust order sched-

✉ Wei Du
duwei0203@gmail.com

Youkai Xiao
y30210977@mail.ecust.edu.cn

Yang Tang
yangtang@ecust.edu.cn

¹ Key Laboratory of Smart Manufacturing in Energy Chemical Process, East China University of Science and Technology, No. 130 Meilong Road, Shanghai 200237, China

ules, evolutionary robust optimization has been proven to be an effective method [5, 6]. Evolutionary robust optimization refers to the utilization of evolutionary algorithms as a framework for solving robust optimization problems. In [7], the preproduction events in apparel manufacturing are taken into consideration, and a robust order scheduling approach is proposed by introducing robust multi-objective optimization to address the order scheduling problem in the apparel industry. In [8, 9], the influence of uncertain daily production quantities on order scheduling is considered, while solving high-dimensional robust order scheduling problems. The experimental results show that evolutionary robust optimization exhibits significantly superior performance in finding robust order schedules.

In practical scenarios, robust order scheduling problems are commonly characterized as high-dimensional robust optimization problems. For example, considering factors like order splitting and learning effects, an order scheduling problem with 30 orders and 6 production lines is a high-dimensional problem with over 100 decision variables [8]. As the number of decision variables increases, determining the robust region of the high-dimensional problem becomes significantly more challenging due to the “curse of dimension” [10]. This phenomenon suggests that the performance of evolutionary robust optimization approaches deteriorates as the dimensionality of the search space increases. Therefore, it is crucial to develop efficient approaches for solving high-dimensional robust order scheduling problems.

To the best of our knowledge, the research on high-dimensional robust optimization, especially high-dimensional robust multi-objective optimization, has received relatively limited attention in the field of evolutionary computation. Among these studies, a seminal method is the recently proposed decision variable classification-based framework for high-dimensional robust evolutionary multi-objective optimization, which is named constrained non-dominated sorting differential evolution based on decision variable classification (CNSDE/DVC). CNSDE/DVC classifies decision variables into highly and weakly robustness-related variables based on their contributions to the robustness of candidate solutions [8, 9]. Encouraging experimental results are reported in [8] and [9]. However, it adopts an explicit averaging technique, specifically by generating additional sample points [6], during the process of decision variable decomposition to compute the robustness indicator. The process of classifying decision variables based on their robustness requires significant computational cost, since each decision variable needs to be perturbed dimension-by-dimension and evaluated. When utilizing this algorithm to address robust order scheduling problems with higher dimensions, the computational cost incurred by decomposing decision variables is unacceptable. Hence, it is important to explore more effective

approaches to solve high-dimensional robust order scheduling problems with lower computational costs.

The implicit averaging technique [6], by utilizing previously sampled points to compute robustness metrics, presents promising prospects in the field of high-dimensional robust optimization due to its significant computational resource savings. However, the limited stability of the implicit averaging technique hinders its application. To increase the reliability of the implicit averaging technique for robust single-objective optimization problems, a novel metric called confidence measure is proposed in [11]. Nevertheless, the effectiveness of this approach has only been validated on low- to medium-dimensional robust single-objective test functions. It is noteworthy that efficient methods with lower computational costs, such as the implicit averaging technique, have not received sufficient attention in the context of high-dimensional robust order scheduling. To systematically capture the interacting variables for a more effective problem decomposition in large-scale single-objective optimization problems, a novel technique called the delta method is introduced based on the cooperative coevolution framework in [12]. This method measures the average difference of each variable across the entire population. Subsequently, decision variables are sorted in descending order according to their corresponding delta values and partitioned into pre-defined equally sized subgroups. The key point is that the delta method’s variable grouping process does not require extra fitness evaluations through perturbations or other means. Inspired by the delta method [12] and CNSDE/DVC [9], high-dimensional robust order scheduling problems can be addressed by implicitly decomposing decision variables into highly and weakly robustness-related variables without consuming additional fitness evaluations.

Based on the above discussion, this paper proposes a novel algorithm named dynamic cooperative coevolution based on the implicit decision variable classification approach (DCC/IDVCA), which tackles high-dimensional robust order scheduling problems by classifying decision variables implicitly, thereby significantly conserving computational resources. During the optimization process of DCC/IDVCA, an approach called estimation of dimensional contribution to robustness (ECR) is used to estimate the weighted contribution of each variable to robustness. Subsequently, IDVCA divides the decision variables into highly and weakly robustness-related variables based on the magnitude of their weighted contributions to the robustness of candidate solutions. This classification approach eliminates the need for extra fitness evaluations. Then, these two types of variables are decomposed into highly robustness-related subgroups and weakly robustness-related subgroups within DCC. Moreover, the sizes of these subgroups dynamically change in DCC, which helps ECR to estimate the weighted contributions of each variable to robustness more accu-

rately. Finally, different types of subgroups are optimized using different strategies in DCC. The proposed approach is applied to two order scheduling problems formulated as high-dimensional robust multi-objective optimization problems and experimental results demonstrate its effectiveness. The contributions of this work can be summarized as follows:

- We first attempt to decompose high-dimensional decision variables into highly and weakly robustness-related variables using an implicit decomposition approach named IDVCA. This initial endeavor significantly reduces the computational resources required and greatly enhances the efficiency of solving high-dimensional robust order scheduling problems.
- In the proposed DCC/IDVCA, we evaluate the weighted contribution of each variable to robustness using a non-perturbative approach named ECR. Subsequently, we allocate the variables decomposed by IDVCA to pre-defined sizes of highly and weakly robustness-related subgroups using the DCC. It is worth noting that the sizes of these subgroups dynamically change within successive cycles in the DCC, which is completely different from the existing CC frameworks based on static grouping.
- The proposed algorithm is applied to solving two high-dimensional robust order scheduling problems and compared with five state-of-the-art multi-objective evolutionary algorithms. Extensive experimental results demonstrate that DCC/IDVCA can efficiently solve high-dimensional robust order scheduling problems, particularly with substantial computational resource savings achieved through its implicit decomposition approach.

The rest of the paper is organized as follows. “[Background Information](#)” provides background information on robust multi-objective optimization and the traditional CC framework. “[The Proposed DCC/IDVCA](#)” presents the details of DCC/IDVCA. A series of comprehensive and systematic experiments are conducted in “[Experiments](#)”. Finally, “[Conclusion](#)” provides a concluding discussion.

Background information

In this section, we first introduce the background information of robust multi-objective optimization. Subsequently, to facilitate a better comprehension of the proposed DCC framework in this paper, we present the background information of the CC framework.

Robust multi-objective optimization

In [6], a novel approach was employed to introduce robustness to multi-objective optimization problems by optimizing

the mean effective objective functions instead of the original objective function. Furthermore, two categories of multi-objective robust solutions were defined in [6]. In the following paragraphs, we will provide a comprehensive explanation of these relevant definitions.

Multi-objective robust solution of Type I: A solution \mathbf{x}^* is called a multi-objective robust solution of type I, if it is the globally feasible Pareto-optimal solution to the following multi-objective minimization problem (defined about a δ -neighborhood ($\mathcal{B}_\delta(\mathbf{x})$) of a solution \mathbf{x}):

$$\left. \begin{array}{l} \text{minimize } (f_1^{\text{eff}}(\mathbf{x}), f_2^{\text{eff}}(\mathbf{x}), \dots, f_M^{\text{eff}}(\mathbf{x})), \\ \text{subject to } \mathbf{x} \in \Omega, \end{array} \right\} \quad (1)$$

where $f_i^{\text{eff}}(\mathbf{x})$ is defined as follows:

$$f_i^{\text{eff}}(\mathbf{x}) = \frac{1}{|\mathcal{B}_\delta(\mathbf{x})|} \int_{\mathbf{y} \in \mathcal{B}_\delta(\mathbf{x})} f_i(\mathbf{y}) d\mathbf{y}, \quad (2)$$

where Ω indicates the feasible search space; $\mathcal{B}_\delta(\mathbf{x})$ represents a δ -neighborhood of a solution \mathbf{x} , $|\mathcal{B}_\delta(\mathbf{x})|$ denotes the associated hypervolume of the neighborhood; $\mathbf{x} = [x_1, x_2, \dots, x_D]^T$ denotes a decision vector whose dimensionality is specified by D ; $f_i^{\text{eff}}(\mathbf{x})$ represents the i th ($1 \leq i \leq M$) mean effective objective function.

However, when it comes to practical applications, practitioners are often more interested in controlling the level of robustness according to the specific problem requirements. Consequently, the second type of robustness is defined as follows:

Multi-objective robust solution of Type II: A solution \mathbf{x}^* is called a multi-objective robust solution of type II, if it is the globally feasible Pareto-optimal solution to the following multi-objective minimization problem:

$$\left. \begin{array}{l} \text{minimize } f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_M(\mathbf{x})), \\ \text{subject to } \left\| f^{\text{eff}}(\mathbf{x}) - f(\mathbf{x}) \right\| \leq \eta, \\ \mathbf{x} \in \Omega, \end{array} \right\} \quad (3)$$

where $f^{\text{eff}}(\mathbf{x}) = (f_1^{\text{eff}}(\mathbf{x}), f_2^{\text{eff}}(\mathbf{x}), \dots, f_M^{\text{eff}}(\mathbf{x}))$; $\|\cdot\|$ applies to any appropriate norm measure; parameter η provides the practitioners with direct control over the desired level of robustness. This research has adopted the second type of robustness due to its greater practicality.

Cooperative coevolution (CC) frameworks

It is widely recognized that CC frameworks have been extensively utilized to address high-dimensional single-objective optimization problems. In the traditional CC framework, a high-dimensional decision vector is decomposed into multiple low-dimensional subcomponents using a specific

decomposition strategy. Each of these subcomponents is optimized in a round-robin manner by its corresponding subpopulation in each CC cycle. An initial effort to develop a universal divide-and-conquer framework was carried out in [13]. A genetic algorithm based on the CC framework was proposed to decompose the original D -dimensional problem into D subcomponents, each of which would be optimized by a subpopulation in a round-robin manner. It is evident that the performance of the divide-and-conquer strategy in the traditional CC framework strongly depends on the adopted variable grouping approach to suitably decompose the original high-dimensional problem. An inaccurate decomposition may result in the emergence of a pseudo-minimum [14]. Numerous decomposition strategies have been devised to partition the original high-dimensional problem into a set of subcomponents. Most of the existing decomposition methods in the traditional CC framework adopt static decomposition strategies.

Concerning the static grouping strategy, decision variables are decomposed into a multitude of subcomponents through a deterministic procedure, which remains unchanged throughout the entire optimization process. For instance, CC with variable interaction learning (CCVIL) utilizes perturbation techniques to classify interacting variables into subcomponents [15]. Another classic grouping strategy is differential grouping (DG), which adopts a deterministic interaction detection operator to group interacting variables into the same subcomponent [16]. In addition, by utilizing a parameter adaptation technique and reusing sampling points, DG2 [17] has enhanced the accuracy and efficiency of variable grouping compared to the original DG. Nonetheless, these static grouping methods will consume a substantial amount of additional computational resources. Moreover, this paper adopts a dynamic CC framework, with the specific reasons detailed in “The Proposed DCC/IDVCA”.

The proposed DCC/IDVCA

This paper proposes a high-dimensional robust EMO algorithm called DCC/IDVCA, which consists of three main parts: (1) estimation of dimensional contribution to robustness (ECR), (2) dynamic cooperative coevolution (DCC), and (3) implicit decision variable classification approach (IDVCA). The detailed pseudocode of DCC/IDVCA is listed in Algorithm 1. Line 1 shows the initialization of the population. Lines 2–12 describe the random grouping optimization phase of DCC/IDVCA. First, the original high-dimensional decision variables are randomly divided into several equally sized subgroups. Then, these subgroups are further optimized based on the CC framework. After optimizing each subgroup, the weighted contribution vector \mathbf{W} is updated using ECR

method. In this phase of DCC/IDVCA, random grouping optimization is conducted for RN iterations.

Lines 13–15 demonstrate the process of calculating the mean value of the weighted contributions to robustness for each variable, obtained after RN times of random grouping optimization. This can potentially mitigate the influence of random grouping on the accuracy of the grouping process.

Lines 17–22 describe the dynamic grouping optimization of DCC/IDVCA. Line 18 represents that IDVCA classifies the original high-dimensional variables into highly and weakly robustness-related variables based on weighted contribution vectors. Line 19 denotes that in DCC, these two types of variables are decomposed into several subgroups that are highly and weakly robustness-related. Unlike traditional optimization algorithms based on the CC framework, the size of each subgroup varies across adjacent optimization cycles. In the odd-numbered cycles of DCC, the sizes of the dynamic subgroups are all ON , whereas in the even-numbered cycles of DCC, the sizes of the dynamic subgroups are EN . This setup of subgroup sizes aids ECR in enhancing the precision of estimating the weighted contributions of variables to robustness. Finally, each subgroup of different types is optimized separately using different optimization strategies based on DCC framework.

Algorithm 1 The framework of DCC/IDVCA

Require: NP (population size), POP (current population), D (dimension size of decision variables), RN (number of random grouping optimization), ON (dynamic subgroup size during odd-numbered cycles), EN (dynamic subgroup size during even-numbered cycles)
Ensure: \mathbf{x}^* , $f(\mathbf{x})^*$ //The Pareto-optimal solutions that satisfy the robustness constraints and their corresponding variable values.

- 1: $POP = \text{Population_Initialization}(NP, D)$
- 2: **for** $i = 1 \rightarrow RN$ **do**
- 3: $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_M\} \leftarrow \text{Random Grouping}(D)$;
- 4: **for** $k = 1 \rightarrow M$ **do**
- 5: $POP \leftarrow \text{optimizer}(\mathbf{x}_k)$;
- 6: update $\mathbf{W} \leftarrow \text{ECR}(\mathbf{x}_k, f^{\text{eff}}(\mathbf{x}), f^{\text{eff}}(\mathbf{y}), \text{archive}, NP, D, m)$;
- 7: $k \leftarrow k + 1$;
- 8: update fes ;
- 9: **end for**
- 10: $\mathbf{H}(i, :) \leftarrow \mathbf{W}$;
- 11: $i \leftarrow i + 1$;
- 12: **end for**
- 13: **for** $i = 1 \rightarrow D$ **do**
- 14: $\mathbf{W}_i = \text{mean}(\mathbf{H}(:, i))$; //computation of the mean value of \mathbf{W}_i after RN rounds of random grouping optimization.
- 15: **end for**
- 16: $cycle = 1$;
- 17: **while** ($fes < FEsmax$) **do**
- 18: $\{\mathbf{DV1}, \mathbf{DV2}\} \leftarrow \text{IDVCA}(\mathbf{W}, D, \text{archive}, a, \eta)$;
- 19: $(POP, \mathbf{W}) \leftarrow \text{DCC}(\mathbf{W}, EN, ON, \mathbf{DV1}, \mathbf{DV2}, cycle)$;
- 20: $cycle = cycle + 1$;
- 21: update fes ;
- 22: **end while**

ECR

ECR estimates the weighted contribution of each variable to robustness by utilizing the historical information of the changes in f^{eff} values and the frequency with which the decision variable has been classified as highly robustness-related in previous cycles, which does not require additional computational cost. Because the present contribution-based CC (CBCC) frameworks only compute the contributions of each subcomponent, without being able to determine the contributions of individual variables within each subcomponent. Therefore, to accurately classify the decision variables in the proposed algorithm of this paper, it is necessary to design a novel method that can estimate the contributions of individual variables within each subcomponent. It is worth noting that our ultimate goal is to achieve an accurate classification of the decision variables. Therefore, if the proposed contribution calculation criterion is capable of ranking the impact of each variable on robustness, thereby accomplishing its ultimate goal, obtaining the precise contribution of each variable to robustness becomes unnecessary. Following this line of thought, we design the ECR to estimate the weighted contribution of each variable to robustness.

Algorithm 2 ECR

Require: S^c (the current subcomponent being optimized), $f^{\text{eff}}(\mathbf{x})$ (the mean effective fitness function values of individuals in the current population), $f^{\text{eff}}(\mathbf{y})$ (the mean effective fitness function values of individuals in the previous population), *archive* (an archive of the frequency with which the decision variable x has been classified as a highly robustness-related variable in previous cycles), NP (population size), D (dimension size of decision variables), m (the number of objectives)

Ensure: \mathbf{W} //The contribution vector that archives the weighted contributions of each variable to the robustness

```

1:  $\Delta F \leftarrow \frac{1}{NP} \sum_{i=1}^m \sum_{j=1}^{NP} |f^{\text{eff}}(\mathbf{x}_j^i) - f^{\text{eff}}(\mathbf{y}_j^i)|$ ;
2: for  $i = 1 \rightarrow D$  do
3:   if  $x_i \in S^c$  then
4:      $\mathbf{W}_i \leftarrow \text{archive}(i)^a \cdot \Delta F$ ;
5:   else
6:      $\mathbf{W}_i \leftarrow \mathbf{W}_i$ ;
7:   end if
8: end for

```

The pseudocode for ECR method, as shown in Algorithm 2: Line 1 calculates the average absolute change in the f^{eff} values of m objectives for all individuals in the population between two consecutive cycles, which is denoted as ΔF . The specific formula is shown in Eq. (4). Lines 2–8 indicate that if x_i belongs to S^c , the weighted contribution of x_i is updated; otherwise, the weighted contribution of x_i remains unchanged. In the pseudocode, $S^c = \{x_1, x_2, \dots, x_T\}$ rep-

resents the current subcomponent being optimized. $\mathbf{W} = (\mathbf{W}_1, \mathbf{W}_2, \dots, \mathbf{W}_D)$ denotes the contribution vector that archives the weighted contributions of each variable to the robustness, where \mathbf{W}_i tracks the contribution of the i th variable x_i to the robustness. The calculation of the weighted contribution of x_i to robustness can be summarized as Eq. (5)

$$\Delta F = \frac{1}{NP} \sum_{i=1}^m \sum_{j=1}^{NP} |f^{\text{eff}}(\mathbf{x}_j^i) - f^{\text{eff}}(\mathbf{y}_j^i)|, \tag{4}$$

where m represents the number of objectives in the robust MOPs, NP denotes the population size, and $f^{\text{eff}}(\mathbf{x}_j^i)$ and $f^{\text{eff}}(\mathbf{y}_j^i)$ refer to the mean effective fitness function values of the j th individual on the i th objective in the current and previous population, respectively

$$\mathbf{W}_i = \begin{cases} \text{archive}(i)^a \cdot \Delta F & \text{if } x_i \in S^c \\ \mathbf{W}_i & \text{otherwise,} \end{cases} \tag{5}$$

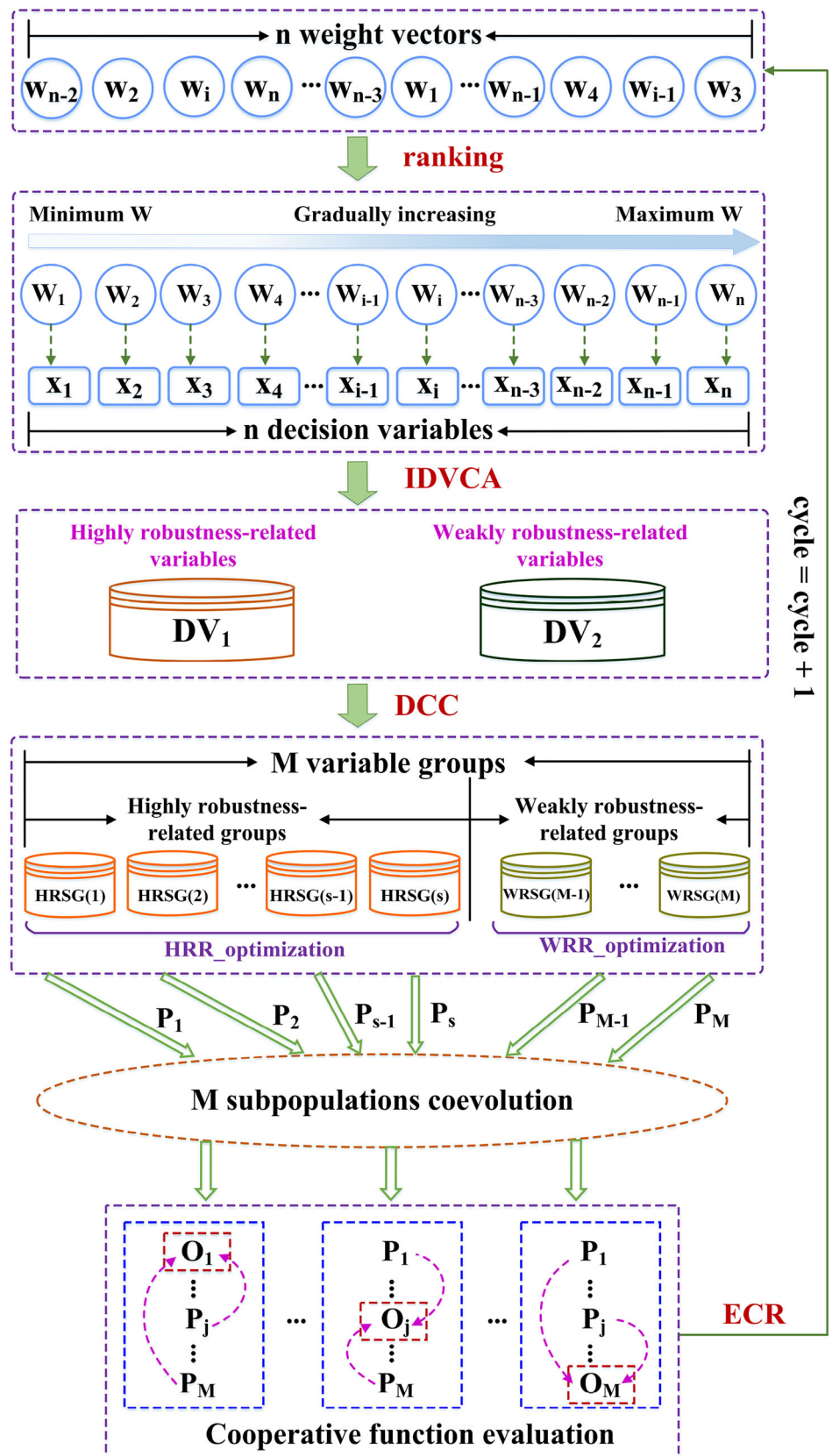
where *archive*(i) represents the number of times the decision variable x_i has been categorized as a highly robustness-related variable in previous cycles. Parameter a can introduce non-linear effects when estimating the weighted contribution of decision variables to robustness. The value of parameter a determines the intensity of this non-linear effect, and it can be set by algorithm practitioners based on a specific problem context.

In summary, the proposed ECR method is capable of ranking the impact intensity of each variable on robustness, enabling IDVCA to classify variables. As a result, obtaining accurate contribution values of each variable to robustness becomes less necessary, which is the central idea behind the development of ECR method.

DCC

In DCC, the input highly and weakly robustness-related variables are allocated to several pre-defined sizes of highly and weakly robustness-related subgroups. Following this, different optimization strategies are employed to optimize different types of subgroups. Furthermore, the sizes of these subgroups dynamically change within successive cycles in DCC, thereby improving the precision of ECR. The idea behind DCC is to treat the original high-dimensional robust multi-objective optimization problem as a series of dynamic subcomponents for optimization. Each subcomponent is dynamically constructed during the optimization process based on the weighted contribution ranking of each decision variable to the robustness of candidate solutions. The detailed pseudocode for DCC is presented in Algorithm 3.

Fig. 1 Schematic diagram of optimization based on DCC framework



Algorithm 3 DCC

Require: \mathbf{W} (the contribution vector), EN (the subgroup size during an even-numbered cycle), ON (the subgroup size during an odd-numbered cycle), $DV1$ (highly robustness-related variables), $DV2$ (weakly robustness-related variables), $cycle$ (the count of the optimization cycle)

Ensure: POP, \mathbf{W} //The current population and the updated contribution vector

```

1: sort( $\mathbf{W}$ , descending);
2: if  $cycle \% 2 = 0$  then
3:    $subgroupsize = EN$ ;
4: end if
5: if  $cycle \% 2 = 1$  then
6:    $subgroupsize = ON$ ;
7: end if
8:  $HS = \|\mathbf{DV1}\| / subgroupsize$ ;
9:  $WS = \|\mathbf{DV2}\| / subgroupsize$ ;
10:  $DR = \|\mathbf{DV2}\| \% subgroupsize$ ;
11:  $t = 1$ ;
12: for  $i = 1 \rightarrow WS$  do
13:   for  $j = 1 \rightarrow subgroupsize$  do
14:      $WRSRG(i) \leftarrow$  variable index corresponding to  $\mathbf{W}_i$ ;
15:      $t = t + 1$ ;
16:   end for
17: end for
18: if  $DR = 0$  then
19:   for  $i = 1 \rightarrow HS$  do
20:     for  $j = 1 \rightarrow subgroupsize$  do
21:        $HRSRG(i) \leftarrow$  variable index corresponding to  $\mathbf{W}_i$ ;
22:        $t = t + 1$ ;
23:     end for
24:   end for
25: end if
26: if  $DR \neq 0$  then
27:    $HS = HS + 1$ ;
28:   for  $i = 1 \rightarrow HS$  do
29:     for  $j = 1 \rightarrow subgroupsize$  do
30:        $HRSRG(i) \leftarrow$  variable index corresponding to
31:        $\mathbf{W}_{(\|\mathbf{DV2}\| - DR + t)}$ ;
32:        $t = t + 1$ ;
33:     end for
34:   end for
35: for  $i = 1 \rightarrow HS$  do
36:   POP=HRR_optimization(POP,  $HRSRG(i)$ ); /*Optimizing
37:   highly robustness-related subgroups*/
38:   update  $\mathbf{W} \leftarrow ECR$ ;
39: end for
40: for  $i = 1 \rightarrow WS$  do
41:   POP=WRR_optimization(POP,  $HRSRG(i)$ );
42:   update  $\mathbf{W} \leftarrow ECR$ ;
43: end for

```

In Algorithm 3, Line 1 indicates that DCC first sorts the contribution values of each dimension in the weighted contribution vector \mathbf{W} in ascending order. Lines 2–7 demonstrate how to determine the size of subgroups during different optimization cycles. When the count of the optimization cycle, denoted as $cycle$, is an odd number, the subgroup size is set to ON ; conversely, when it is an even number, the subgroup size is set to EN . In static grouping methods, subcomponents are predetermined using specific grouping techniques

and remain unchanged throughout the optimization process. However, in DCC, the newly constructed dynamic subcomponents may differ from the previous dynamic subcomponents. It should be noted specifically that the size of each dynamic subcomponent within the same cycle is equal, but the size of dynamic subcomponents varies across adjacent cycles. This design ensures that variables in the subcomponents of adjacent cycles are different, thereby reducing the impact of other variables in the same subcomponent on the calculation of the robustness contribution of the evaluated variables. For example, in the first cycle, a highly robustness-related variable and a weakly robustness-related variable are classified into the same subcomponent, and it is determined by ECR and IDVCA that both variables in this subcomponent are highly robustness-related. As these two variables belong to the same subcomponent, their weighted robustness contribution values are the same. Therefore, in the next cycle, the ranks of these two variables are similar. If the size of subcomponents remains unchanged across adjacent cycles, there is a high probability that these two variables will still be classified into the same subcomponent in the next cycle, which will negatively affect the accuracy of subsequent grouping. However, if the size of subcomponents varies across adjacent cycles, this impact will be greatly reduced. Lines 8–10 calculate the number of highly and weakly robustness-related subgroups within each optimization cycle, as well as the remainder of dividing the number of weakly robustness-related variables by the subgroup size.

Lines 12–17 state that the variables corresponding to the top $WS * subgroupsize$ weighted contribution rankings in \mathbf{W} are allocated sequentially to equally sized weakly robustness-related subgroups based on their ascending order of weighted contributions. $WRSRG(i)$ represents the i th weakly robustness-related subgroup. Lines 18–25 state that when the count of weakly robustness-related variables is a multiple of the subgroup size, and since the total dimension of the problem is also a multiple of the subgroup size, the count of highly robustness-related variables is also a multiple of the subgroup size. In this case, the variables corresponding to the lower $HS * subgroupsize$ weighted contribution rankings are allocated sequentially to equally sized highly robustness-related subgroups based on their ranking order. $HRSRG(i)$ represents the i th highly robustness-related subgroup. Lines 26–34 state that when the count of weakly robustness-related variables is not a multiple of the subgroup size, the variables corresponding to the rankings from $(\|\mathbf{DV2}\| - DR + 1)$ th to the last rank in terms of weighted contributions are allocated sequentially to equally sized highly robustness-related subgroups based on their ranking order. This indicates that when the count of weakly robustness-related variables is not a multiple of the subgroup size, the variables ranged from $(\|\mathbf{DV2}\| - DR + 1)$ th to the $\|\mathbf{DV2}\|$ th position based on their weighted contributions are allocated to the

highly robustness-related subgroups. Since the allocation is done in ascending order, the variables with higher rankings among the weakly robustness-related variables have relatively large weighted contributions. Therefore, classifying these variables into highly robustness-related subgroups for optimization is also considered appropriate.

Lines 35–38 show that the HRR_optimization optimizer is used to optimize each highly robustness-related subgroup. The optimizer HRR_optimization utilizes the DE/rand/1/bin strategy from the classical differential evolution (DE) algorithm. Specifically, robustness is employed as the selection criterion. Lines 39–42 show that the WRR_optimization optimizer is used to optimize each weakly robustness-related subgroup. The optimizer WRR_optimization utilizes the DE/rand/1/bin strategy from the classical differential evolution (DE) algorithm. Specifically, non-dominated ranking and crowding distance are, respectively, employed as the first and second selection criteria.

Finally, to enhance the comprehension of the optimization process based on DCC, a schematic diagram illustrating the optimization process using DCC framework has been generated, as depicted in Fig. 1. First, IDVCA categorizes the original high-dimensional decision variables into highly and weakly robustness-related variables, based on the weighted contribution vector \mathbf{W} . Afterward, these variables are further divided into the appropriate M groups (S_1, S_2, \dots, S_M). Each variable group S_j is associated with a subpopulation P_j , where each subpopulation P_j generates a subpopulation O_j by encoding solutions based on S_j using evolutionary operators. Hence, the search is performed only in a low-dimensional variable space formed by the corresponding variable group S_j , rather than the entire original large-scale variable space. This facilitates the use of the ECR approach proposed in this paper to assess the weighted contribution of each variable to robustness. The offspring solutions in O_j are then cooperated with other subpopulations to evaluate the objective function.

IDVCA

IDVCA utilizes the weighted contribution of each variable to robustness estimated by ECR and compares it with a specifically tailored self-adaptive threshold; thereby decomposing the original high-dimensional decision variables into highly and weakly robustness-related variables. It is worth noting that, in contrast to the explicit decomposition method used in CNSDE/DVC where each decision variable is perturbed and evaluated dimension-by-dimension and finally classified, IDVCA employs ECR which utilizes the historical information of the changes in f^{eff} values and the frequency with which the decision variable has been classified as highly robustness-related in previous cycles. This allows for an implicit decomposition of the variables. The processes of

implicit decomposition in IDVCA and the delta method [12] are similar as they both perform this decomposition implicitly during the optimization process. This eliminates the need for computationally costly approaches like perturbation, which requires significant computational resources for variable decomposition.

Algorithm 4 IDVCA

Require: \mathbf{W} (the contribution vector), D (dimension size of decision variables), *archive* (an archive of the frequency with which the decision variable x has been classified as a highly robustness-related variable in previous cycles), a (the constant that introduces nonlinearity to the weighted contribution.), η (the constant that controls the desired level of robustness)

Ensure: DV1, DV2 //highly and weakly robustness-related variables

```

1: for  $i = 1 \rightarrow D$  do
2:   if  $W_i > \eta \cdot \text{archive}(i)^a$  then
3:     DV1  $\leftarrow$  variable index corresponding to  $W_i$ ;
4:   end if
5:   if  $W_i \leq \eta \cdot \text{archive}(i)^a$  then
6:     DV2  $\leftarrow$  variable index corresponding to  $W_i$ ;
7:   end if
8: end for
9: Update the archive;

```

There are two key points in IDVCA approach: (1) the input consists of the weighted contribution vector \mathbf{W} obtained from ECR calculations; (2) the computation of self-adaptive threshold. The specific details of IDVCA are illustrated in Algorithm 4. Lines 2–4 indicate that if the weighted contribution value corresponding to a particular dimension of the decision variable is greater than the threshold, then this decision variable is classified as a highly robustness-related variable. Lines 5–7 indicate that if the weighted contribution value corresponding to a particular dimension of the decision variable is less than or equal to the threshold, then this decision variable is classified as a weakly robustness-related variable. The self-adaptive threshold designed in IDVCA is $\eta \cdot \text{archive}(i)^a$, where the constant η controls the desired level of robustness. The algorithm practitioners can tailor its value based on the specific problem. The constant a is applied in IDVCA to introduce a non-linear impact on the weighted contribution. The *archive* tracks the count of variables being classified into highly robustness-related variables in previous cycles. The reason for considering the impact of the *archive* when designing the threshold for IDVCA is that it takes into account the non-linear weighting of the *archive* in estimating the contribution of variables to robustness. If the threshold does not take into account the non-linear weighting of the *archive*, as optimization progresses, there will be an increasing number of variables whose weighted contributions to robustness will grow. Consequently, more variables

will have weighted contributions to robustness that exceed the threshold. This will ultimately result in a deterioration of the effectiveness of the threshold.

The process of classifying decision variables based on their robustness in CNSDE/DVC requires significant computational costs, as each decision variable needs to be perturbed dimension-by-dimension and evaluated. At the same time, the process of classifying variables based on their robustness using IDVCA significantly reduces computational costs, as it primarily relies on the historical change information of $f^{\text{eff}}(\mathbf{x})$ during the optimization process, without requiring additional computational resources for variable classification.

Comparative analysis of CNSDE/DVC and DCC/IDVCA

Similarities between CNSDE/DVC and DCC/IDVCA

The most significant similarity between CNSDE/DVC and DCC/IDVCA is that both algorithms decompose the original high-dimensional variable into highly and weakly robustness-related variables, which are then optimized separately. Both algorithms are inspired by variable property-based classification, and aim to address high-dimensional robust multi-objective optimization problems by categorizing decision variables based on their contributions to the robustness of candidate solutions.

Differences between CNSDE/DVC and DCC/IDVCA

The fundamental difference between CNSDE/DVC and the proposed DCC/IDVCA is that CNSDE/DVC adopts an explicit decomposition method to decompose decision variables, while DCC/IDVCA utilizes an implicit decomposition approach for the same purpose. CNSDE/DVC adopts an explicit decomposition method, which requires perturbing each decision variable dimension-by-dimension and then evaluating them. This classification process consumes a significant amount of computational resources. In strict terms, the computational resources solely dedicated to variable classification in DCC/IDVCA are zero. Because DCC/IDVCA adopts an implicit decomposition method, which estimates the contribution of each dimensional variable to robustness by utilizing historical information during the evolutionary process, and subsequently classifies the variables. This implicit decomposition approach can significantly reduce the consumption of computational resources and enhance the efficiency of computational resource utilization.

To provide a clearer mathematical explanation of how DCC/IDVCA significantly conserves computational resources during the variable grouping process, a detailed computational complexity analysis of both CNSDE/DVC and DCC/IDVCA is presented below. In DCC/IDVCA and

CNSDE/DVC, the common operations are non-dominated sorting, crowding-distance assignment, and crowding-degree comparison. The overall complexity of these three operations is $\mathcal{O}(M(NP)^2)$ [18]. Therefore, the computational complexity of DCC/IDVCA is $\mathcal{O}(M(NP)^2 + NP \cdot NM \cdot (1 + H))$, where M is the number of objectives, NP represents the population size, NM denotes the number of subgroups optimized based on a cooperative coevolution (CC) framework, and H represents the number of neighboring solutions sampled within the δ -neighborhood of each solution. The computational complexity of CNSDE/DVC is $\mathcal{O}(M(NP)^2 + (TN \cdot SN + NP) \cdot D \cdot (1 + H))$, where TN represents the total number of the perturbation operations repeated on selected individuals, SN represents the number of selected individuals for DVC, and D represents the dimension size of decision variables. In practice, the subgroup size of DCC/IDVCA is typically set to at least 10 or even larger. Therefore, in the computational complexity of the two algorithms, D is at least ten times larger than NM . In conclusion, the computational complexity of DCC/IDVCA is significantly lower than that of CNSDE/DVC.

Experiments

Case information

Due to the lack of high-dimensional robust multi-objective test functions, we conducted a series of experiments using order scheduling problems in the discrete manufacturing industry as cases to evaluate the effectiveness of the proposed DCC/IDVCA algorithm. The cases used in this paper have been previously employed in prior research on high-dimensional robust multi-objective optimization [8, 9], representing typical instances of high-dimensional robust multi-objective optimization problems. The task of order scheduling is to allocate m orders to n production lines in an appropriate manner. Each order consists of different types of products. These products are assigned to multiple production lines for manufacturing. Specifically, it should be noted that the production lines are dedicated to manufacturing specific products, implying that the production efficiency of a given line can only reach its highest level for a certain type of product. During production, orders can be split to facilitate flexible manufacturing. The problem also takes into account the learning effect.

An appropriate schedule for the problem implies the avoidance of encouraging either early or delayed completion of an order. The reason behind this is that completing an order before its due date will incur higher earliness penalty costs in the form of increased storage expenses while finishing an order after its due date will result in higher tardiness penalty costs due to reduced customer satisfaction. It is evident that

the two optimization objectives of an order scheduling problem can be formulated as: (1) minimizing the sum of all orders' earliness time; (2) minimizing the sum of all orders' tardiness time.

Specifically, the first optimization objective can be mathematically formulated as follows:

$$f_1 = \sum_{i=1}^n t_1(FD_i - DD_i), \quad (6)$$

where FD_i and DD_i represent the finishing date and the due date of order i ($1 \leq i \leq n$) in the schedule, respectively; and $t_1(\cdot)$ is

$$t_1(u) = \begin{cases} 0, & \text{if } u \geq 0, \\ -u, & \text{otherwise.} \end{cases} \quad (7)$$

The second optimization objective can be mathematically expressed as follows:

$$f_2 = \sum_{i=1}^n t_2(FD_i - DD_i), \quad (8)$$

where $t_2(\cdot)$ is:

$$t_2(u) = \begin{cases} 0, & \text{if } u \geq 0, \\ u, & \text{otherwise.} \end{cases} \quad (9)$$

From the above formulas, it can be observed that these two objectives are usually conflicting, which implies that obtaining a smaller f_2 (less total tardiness) solution will result in a larger f_1 (more total earliness). In this paper, the uncertainty of the order scheduling problem stems from the uncertain daily production quantities, which affects the FD_i of each order. Therefore, we aspire to acquire schedules that demonstrate robustness in the face of variations in daily production quantities.

The robust order scheduling problem in this paper is based on the definition of the second type of multi-objective robust solutions in [6], which can be transformed into a constrained bi-objective optimization problem

$$\left. \begin{array}{l} \text{minimize } f_1 = \sum_{i=1}^m t_1(FD_i - DD_i), \\ \text{minimize } f_2 = \sum_{i=1}^m t_2(FD_i - DD_i), \\ \text{s.t. } \quad \|f_1^{\text{eff}} - f_1\|_1 + \|f_2^{\text{eff}} - f_2\|_1 \leq \eta, \end{array} \right\} \quad (10)$$

where L^1 norm is utilized in the constraint and f_1 represents the total earliness of producing m orders, while f_2 represents the total tardiness of producing m orders. Moreover, $f^{\text{eff}}(\mathbf{x})$ denotes the mean effective objective that considers the impact of uncertainty, where the daily production quantities are not fixed.

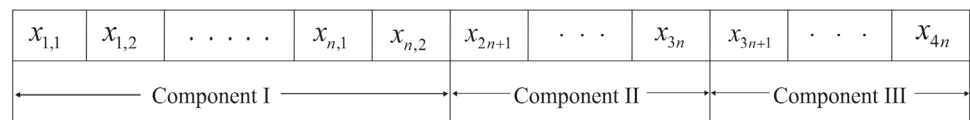
The decision variables for the scheduling problem in this study consist of three components: (1) the assignment of each order to the production line; (2) the split percentage of each order; (3) the sequence of the orders on the same production line. It should be noted that each order can be divided into at most two sub-orders, and the split percentage is selected from the range [0.2, 0.4, 0.6, 0.8]. Consequently, the length of a potential solution is four times the number of orders: $D = 4n$. Figure 2 presents the composition of decision variables for the order scheduling problem in this study. When the number of orders exceeds 25, the problem becomes high-dimensional based on the encoding scheme employed. The decision variables from the $(2n + 1)$ th to the $3n$ th represent the split percentage of each order. In comparison to other decision variables, modifying these decision variables will solely influence the size of sub-orders, without any impact on the sequence of orders across individual production lines. Maintaining the unaltered arrangement of orders within a schedule (despite potential fluctuations in suborder size) indicates that when the evolution operator modifies the decision variables between the $(2n + 1)$ th to the $3n$ th, the violation of the constraint remains generally unchanged. Therefore, the decision variables from the $(2n + 1)$ th to the $3n$ th are weakly robustness-related, while the remaining variables are highly robustness-related.

Experimental settings

In this experiment, we consider two real-world cases: one case involves an order scheduling problem with 40 orders and 6 production lines, and the other case involves an order scheduling problem with 60 orders and 6 production lines. Therefore, the dimensions of the two cases are 160 and 240, respectively. In the proposed DCC/IDVCA in this paper, the population size is $NP = 100$. We adopt a classical differential evolution algorithm (i.e., DE/rand/1/bin) [19] in the optimization process. The scaling factor and crossover probability for the differential evolution algorithm are specified as $F = 0.5$ and $CR = 0.9$, separately. The uncertainty factor γ of daily production quantities is set at 0.3 in the experiment. For each potential solution, we specify $H = 5$ as the number of neighboring points. The expected level of robustness for this problem is defined in advance as $\eta = 5$. In the dynamic grouping stage, the subcomponent size during odd-numbered cycles (denoted as ON) is set to 8, while the subcomponent size during even-numbered cycles (denoted as EN) is set to 10. The parameter a , which determines the strength of the non-linear effect applied to the weighted contribution of variable robustness, is set to 1.2. A sensitivity analysis of the parameter values for a , ON , and EN will be conducted in "Parameter sensitivity analysis".

The experiments in this paper are mainly divided into the following two parts:

Fig. 2 An illustration of the decision variables for the scheduling problem



(1) Comparison of DCC/IDVCA and CNSDE/DVC: Given that CNSDE/DVC is currently the state-of-the-art algorithm for solving high-dimensional robust MOPs, we systematically compare DCC/IDVCA with CNSDE/DVC first. The comparison between DCC/IDVCA and CNSDE/DVC can be divided into three parts: (1) Comparison of the median attainment surfaces obtained by DCC/IDVCA and CNSDE/DVC after 30 runs; (2) Comparisons of the IGD and HV values of DCC/IDVCA and CNSDE/DVC after 30 runs; (3) Comparison of their grouping accuracy. When solving the problem with $D = 160$ using the above algorithms, the FE_{max} is set to 3×10^5 ; on the other hand, when dealing with the problem of $D = 240$, the FE_{max} is set to 7×10^5 . Because the classification of variables for $D = 160$ and $D = 240$ using CNSDE/DVC typically requires approximately 2.6×10^5 and 4.6×10^5 fitness evaluations, respectively, to fairly compare the performance of the two algorithms, FE_{max} should be set significantly larger than the computational resources consumed during the CNSDE/DVC classification process.

(2) Comparison with state-of-the-art MOEAs: Since high-dimensional robust multi-objective optimization problems have so far received little attention, with no dedicated algorithms have been developed. Therefore, in the comparative studies, we select four state-of-the-art multi-objective evolutionary algorithms (MOEAs): NSGA-II [18], NSCDE [20], LMEA [21], and DGEA [22]. Among them, NSGA-II and NSCDE have been widely applied to solve robust multi-objective optimization problems in low-to-medium dimensions, while LMEA and DGEA are two recently proposed large-scale multi-objective evolutionary algorithms.

LMEA is a classic large-scale multi-objective optimization algorithm based on variable property-based classification. DGEA is a recently proposed large-scale multi-objective optimization algorithm based on adaptive offspring generation strategy. As LMEA and DGEA are not originally designed for solving high-dimensional robust optimization problems, to ensure a fair comparison, we introduce the following modifications to LMEA and DGEA specifically targeting robust optimization: (1) Replacing the tree-based non-dominated sorting method (T-ENS) with a conventional constrained non-dominated sorting method in LMEA. This modification helps LMEA to satisfy robust constraints in both diversity optimization and convergence optimization. (2) A constrained non-dominated sorting method is also applied in DGEA to rank individuals in the population. This modification helps DGEA to generate offsprings that satisfy robust

constraints while maintaining better diversity and convergence through a novel reproduction operator.

This part will include the following sets of experiments: (1) When solving the problem with $D = 160$, the five algorithms run independently 30 times with FE_{max} set to 1.5×10^5 , 2×10^5 , 2.5×10^5 , 3×10^5 , 3.5×10^5 , and 4×10^5 , respectively; (2) When solving the problem with $D = 240$, the five algorithms run independently 30 times with FE_{max} set to 2×10^5 , 2.5×10^5 , 3×10^5 , 3.5×10^5 , 4×10^5 , and 4.5×10^5 , respectively. Finally, the performance of each algorithm is compared. To ensure sufficient computational resources are allocated to each algorithm during the experiments, FE_{max} is typically set to be approximately 1000 times greater than the total dimensionality D . In the experiments of this paper, the parameter settings for CNSDE/DVC, NSGA-II, NSCDE, LMEA, and DGEA are adopted as described in their respective original papers [18, 20–22].

In the performance comparison of different algorithms, two performance metrics, namely inverted generational distance (IGD) and hypervolume (HV), are utilized to quantify all experimental outcomes. To calculate IGD, a set of reference points need to be given beforehand. In this paper, considering the requirements of the practical problem, it is assumed that the ideal total earliness and tardiness for each order should not exceed 5 days. Therefore, in the case of 40 orders, the ideal total earliness and tardiness should not exceed 200 days, while in the case of 60 orders, the ideal total earliness and tardiness should not exceed 300 days. Hence, this paper first obtains the non-dominated solutions from the combined solutions of the compared algorithms and sets the non-dominated solutions with total earliness and tardiness not exceeding 200 or 300 days as reference points. To calculate the Hypervolume (HV), taking into account the requirements of the practical problem in our experiments, we consider it is unacceptable for the total earliness or tardiness of each order to exceed 15 days. Therefore, when evaluating the HV for 40 orders, we set the reference point for each objective function as 600; when evaluating the HV for 60 orders, we set the reference point for each objective function as 900. We calculate the values of IGD and HV using PlatEMO, which is a recently designed evolutionary multi-objective optimization platform [23].

All algorithms are executed on a PC equipped with an Intel Core i7-12700 2.1-GHz processor and 16 GB RAM. When setting FE_{max} to $D \cdot 2000$, the running time of DCC/IDVCA for robust order scheduling problems with 60 orders was approximately 40 s. It is worth mentioning that order schedul-

Table 1 Performance comparison of DCC/IDVCA and CNSDE/DVC in solving scheduling problems with 40 orders. The constant η , which controls the level of robustness, is set to 5

FE_{max}	Indices	CNSDE/DVC	DCC/IDVCA
3×10^5	IGD	161.27 ± 46.51	46.74 ± 21.84
	HV	7.46E+04 ± 2.75E+04	1.36E+05 ± 1.51E+04

The highlighted entries demonstrate significant superiority. A Wilcoxon rank-sum test is conducted at a significance level of $\alpha = 0.05$

Table 2 Performance comparison of DCC/IDVCA and CNSDE/DVC in solving scheduling problems with 60 orders. The constant η , which controls the level of robustness, is set to 5

FE_{max}	Indices	CNSDE/DVC	DCC/IDVCA
7×10^5	IGD	276.28 ± 68.96	115.52 ± 69.04
	HV	7.40 E+04 ± 4.71E+04	1.99 E+05 ± 5.34E+04

The highlighted entries demonstrate significant superiority. A Wilcoxon rank-sum test is conducted at a significance level of $\alpha = 0.05$

ing is performed prior to production and can be considered as an offline scheduling. Therefore, the running time can meet the actual demands of the enterprise. Furthermore, to draw statistically valid conclusions, we conducted a Wilcoxon rank-sum test at a significance level of 0.05 to assess the significance of the differences between the results obtained by the two competing algorithms. The highlighted entries demonstrate significant superiority.

Comparison of DCC/IDVCA and CNSDE/DVC

Performance comparison in solving scheduling problems with 40 orders

To demonstrate that DCC/IDVCA can effectively address high-dimensional robust MOPs with significant computational resource savings, we compare the performance of DCC/IDVCA and CNSDE/DVC under the given constraint of FE_{max} . We first compare the performance of DCC/IDVCA and CNSDE/DVC in solving a scheduling

Fig. 3 Comparison of the median attainment surfaces obtained from 30 runs of DCC/IDVCA and CNSDE/DVC for solving scheduling problems with 40 orders under the condition of $FE_{max} = 3 \times 10^5$. The constant η , which controls the level of robustness, is set to 5

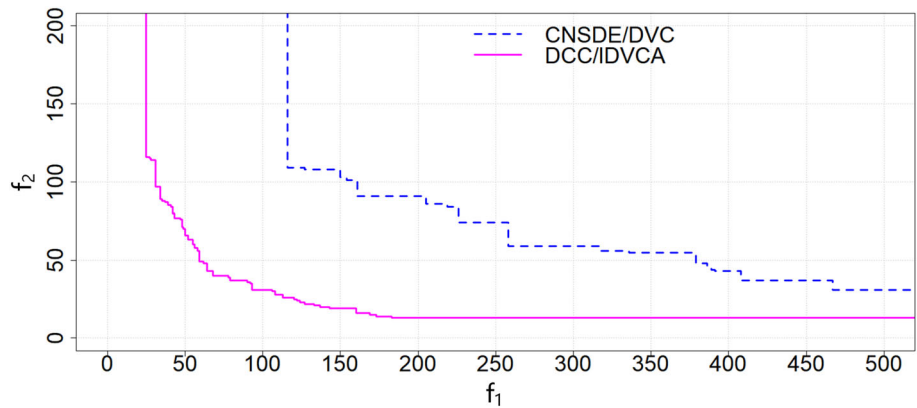
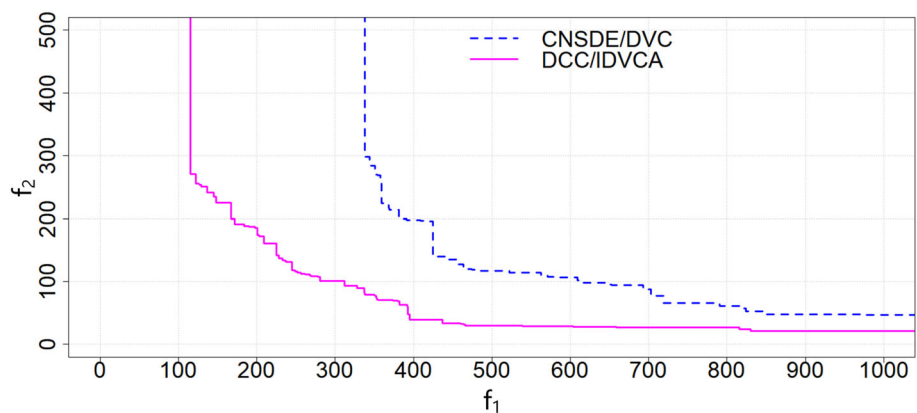


Fig. 4 Comparison of the median attainment surfaces obtained from 30 runs of DCC/IDVCA and CNSDE/DVC for solving scheduling problems with 60 orders under the condition of $FE_{max} = 7 \times 10^5$. The constant η , which controls the level of robustness, is set to 5



problem with 40 orders under the condition of $FE_{max} = 3 \times 10^5$.

Table 1 presents the IGD and HV values obtained by DCC/IDVCA and CNSDE/DVC after 30 runs when solving the scheduling problem with 40 orders under the condition of $FE_{max} = 3 \times 10^5$. It can be observed that, under the constraint of FE_{max} , DCC/IDVCA performs significantly better than CNSDE/DVC when dealing with high-dimensional robust scheduling problems.

Moreover, the median attainment surfaces obtained from DCC/IDVCA and CNSDE/DVC for a scheduling problem of 40 orders after 30 runs, under the condition of $FE_{max} = 3 \times 10^5$, are provided in Fig. 3. It can be clearly observed that under the condition of $FE_{max} = 3 \times 10^5$, DCC/IDVCA significantly improves the performance of CNSDE/DVC in handling high-dimensional robust order scheduling problems. The reasons behind the experimental result can be briefly explained as follows: CNSDE/DVC evaluates the contribution of each decision variable to robustness by perturbation, and this process incurs an extremely expensive computational cost. For example, classifying the 160 decision variables in the problem considered in this paper by CNSDE/DVC requires approximately 2.69×10^5 FEs. In contrast, DCC/IDVCA classifies decision variables implicitly, which saves a significant amount of computational resources that can be used to search for robust solutions. Therefore, under the condition of a given specific amount of computational resources, the performance of DCC/IDVCA is significantly better than that of CNSDE/DVC.

Performance comparison in solving scheduling problems with 60 orders

In this section, we compare the performance of DCC/IDVCA and CNSDE/DVC in solving the scheduling problem with 60 orders. The IGD and HV values of DCC/IDVCA and CNSDE/DVC are calculated and given in Table 2. Moreover, the median attainment surfaces obtained from DCC/IDVCA and CNSDE/DVC for a scheduling problem of 60 orders after 30 runs, under the condition of $FE_{max} = 7 \times 10^5$, are provided in Fig. 4.

It can be observed that as the dimension size of the problem increases, DCC/IDVCA continues to exhibit superior efficiency compared to CNSDE/DVC. Despite the general trend of implicit decomposition methods exhibiting lower stability, we have observed that the standard deviation of IGD and HV values obtained by DCC/IDVCA is only slightly larger than that of CNSDE/DVC. Therefore, it can be concluded that DCC/IDVCA demonstrates relatively desirable stability while significantly saving computational resources.

Comparison of grouping accuracy

In this part, we compare DCC/IDVCA with CNSDE/DVC in terms of the accuracy of high-dimensional variable decomposition based on robustness. We conduct systematic experiments on order scheduling problems with 40 and 60 orders, respectively. In the classification process of CNSDE/DVC, each decision variable needs to be perturbed dimension-by-dimension and evaluated. The classification process for the variables in the order scheduling problems with 40 and 60 orders requires approximately $FEs = 2.69 \times 10^5$ and $FEs = 5.47 \times 10^5$ fitness evaluations, respectively. Therefore, when setting the same number of fitness evaluations ($FEs = 2.69 \times 10^5$, $FEs = 5.47 \times 10^5$), we conduct independent runs of both algorithms 15 times. Subsequently, the average grouping accuracy of CNSDE/DVC and DCC/IDVCA in the two optimization stages was calculated. The experimental results are presented in Table 3.

In Table 3, the term “HR variables” refers to highly robustness-related variables, and the term “WR variables” stands for weakly robustness-related variables. DCC/IDVCA (RG) and DCC/IDVCA (DG), respectively, represent the random grouping and dynamic grouping stages of DCC/IDVCA. The grouping accuracy of DCC/IDVCA (RG) refers to the classification accuracy achieved after a specified number of iterations of optimizing through random grouping. It involves categorizing variables based on their average contributions to robustness during these periods and subsequently calculating the accuracy of the resulting classification.

From Table 3, it is evident that the grouping accuracy of CNSDE/DVC is slightly better than the grouping accuracy of DCC/IDVCA for both highly and weakly robustness-related variables in the two grouping optimization steps. However, CNSDE/DVC achieves a slight advantage by perturbing each decision variable dimension-by-dimension and evaluating it, under the premise of consuming extremely expensive computational costs. Therefore, it is difficult to consider this as an efficient method for solving high-dimensional robust MOPs, especially for higher dimensional problems. The computational cost associated with classifying decision variables is unacceptable. Meanwhile, the computational resources allocated to DCC/IDVCA are solely dedicated to the optimization process, as DCC/IDVCA does not require any additional computational resources for the variables classification process. Furthermore, from Table 3, it can be observed that both random grouping optimization and dynamic grouping optimization based on the CC framework yield a relatively high grouping accuracy for highly robustness-related variables. This implies that the optimization based on the CC framework is highly effective in estimating the robustness contribution of highly robustness-related variables to robustness. However, the random grouping optimization step of DCC/IDVCA exhibits poor performance in estimating the

Table 3 The comparison of grouping accuracy between CNSDE/DVC and DCC/IDVCA (the average accuracy of the final grouping results after 15 runs)

Algorithms	Grouping correctness rate		Grouping error rate	
	HR variables	WR variables	HR variables	WR variables
CNSDE/DVC (40 orders)	91.60%	75.30%	8.40%	24.70%
DCC/IDVCA(RG) (40 orders)	81.40%	30.00%	18.60%	70.00%
DCC/IDVCA(DG) (40 orders)	83.30%	51.10%	16.70%	48.90%
CNSDE/DVC (60 orders)	90.20%	84.80%	9.80%	15.20%
DCC/IDVCA(RG) (60 orders)	77.70%	27.10%	22.30%	72.90%
DCC/IDVCA(DG) (60 orders)	78.90%	59.80%	21.10%	40.20%

contribution of weakly robustness-related variables to robustness. The average accuracy in correctly classifying weakly robustness-related variables across 15 runs is only 30% and 27.1% for scheduling problems involving 40 and 60 orders, respectively. However, in the dynamic grouping optimization stage of DCC/IDVCA, there is a significant improvement in estimating the contribution of weakly robustness-related variables to robustness. The average accuracy in correctly classifying weakly robustness-related variables significantly increases to 51.1% and 59.8% across 15 runs, respectively. Additionally, the average error rate in misclassifying weakly robustness-related variables also shows a significant reduction. These experimental results further validate the effectiveness of the proposed IDVCA and ECR strategies in this paper.

There are two key reasons for the significant improvement in correctly classifying weakly robustness-related variables during the dynamic grouping optimization step of DCC/IDVCA: (1) During the optimization step based on the DCC framework, the size of subgroups varies across different optimization cycles. This strategy helps prevent weakly robustness-related variables, which were grouped with highly robustness-related variables during the random grouping optimization step, from being consistently assigned to the same subgroup in subsequent optimization cycles. If a highly robustness-related variable and a weakly robustness-related variable are consistently assigned to the same subgroup throughout all optimization cycles, it hinders the correct classification of these variables; (2) when estimating the weighted contribution of variables to robustness using the ECR strategy, the influence of being classified as a highly robustness-related variable in previous cycles is taken into account. Specifically, when categorizing variables with highly and weakly robustness-related, if a variable has been classified as a highly robustness-related variable more frequently in previous optimization cycles, the ECR strategy assigns a higher weighted contribution to that variable in terms of robustness. However, during the random grouping step, because the *archive* used to record the classification of being a highly robustness-related variable in previous cycles is initialized to **I**, the weighted contributions of both

highly and weakly robustness-related variables within the same subgroup are considered equal, significantly affecting the accuracy of variable grouping.

Comparison of computational resource consumption for grouping

More precisely, we present the total computational resources consumed by DCC/IDVCA during the optimization process, as well as the additional computational resources consumed by CNSDE/DVC during the variable grouping process through perturbation, which are presented in Table 4. Because DCC/IDVCA utilizes historical information during the evolution process to perform variable grouping, and does not consume additional computational resources for variable grouping. As DCC/IDVCA dynamically groups variables during the optimization process rather than using fixed grouping, the computational resources consumed by DCC/IDVCA are represented using a formula rather than a specific value.

From Table 4, we can observe that CNSDE/DVC consumes 2.69×10^5 *FES* and 5.47×10^5 *FES*, respectively, in the process of variable grouping using perturbation method when dealing with 40 and 60 orders in the order scheduling problem. Moreover, DCC/IDVCA consumes $(4.8 \times 10^4 + 600 * S * gen)$ *FES* and $(7.2 \times 10^4 + 600 * S * gen)$ *FES* in total during the grouping and optimization processes when dealing with these two order scheduling problems. In this context, *S* represents the number of subgroups in DCC/IDVCA during the optimization process, which is much smaller than the number of decision variables, and *gen* represents the generations for dynamic grouping optimization in DCC/IDVCA. There are two optimization stages in DCC/IDVCA, one for random grouping and the other for dynamic grouping optimization. In the experiments of this paper, the size of each subgroup in the random grouping stage of DCC/IDVCA was set to 10, and random grouping optimization was performed for 5 generations. Therefore, the computational resources consumed by the random grouping optimization stage in DCC/IDVCA when solving the problems with 40 and 60 orders were 4.8×10^4 *FES* and 7.2×10^4 *FES*, respectively. It is evident from Table 4 that the com-

Table 4 Comparison of computational resource consumption for grouping in solving two order scheduling problems

	CNSDE/DVC	DCC/IDVCA
40 order	$FES = 2.69 \times 10^5$	$FES = 4.8 \times 10^4 + 600 * S * gen$
60 orders	$FES = 5.47 \times 10^5$	$FES = 7.2 \times 10^4 + 600 * S * gen$

putational resources solely dedicated to variable grouping by CNSDE/DVC are significantly greater than the computational resources consumed by DCC/IDVCA during the optimization stage. As the number of orders, i.e., decision variables, increases, the advantage of DCC/IDVCA in saving computational resources during the grouping stage becomes more pronounced.

In conclusion, relative to the additional resources consumed by variable grouping in CNSDE/DVC, DCC/IDVCA does save a significant amount of computational resources during the variable grouping optimization process.

Comparison with state-of-the-art MOEAs

Performance comparison in solving scheduling problems with 40 orders

We respectively adopt the algorithms DCC/IDVCA, NSGA-II, NSCDE, LMEA, and DGEA under different conditions of FE_{max} to handle an order scheduling problem with 40 orders, and then compare the performance of these five algorithms. The IGD and HV values of DCC/IDVCA, NSGA-II, NSCDE, LMEA, and DGEA under six different FE_{max} conditions have been calculated and are presented in Fig. 5. The median attainment surfaces obtained from 30 runs of DCC/IDVCA, NSGA-II, NSCDE, LMEA, and DGEA under six different FE_{max} conditions, are provided in Fig. 6. Since DCC/IDVCA is a dynamic grouping optimization algorithm, we compared its performance with four other algorithms under six different conditions of FE_{max} to evaluate its performance at different optimization stages.

From Figs. 5 and 6, we can conclude that the performance of DCC/IDVCA under six different conditions of FE_{max} (i.e., at different optimization stages) is significantly better than that of the other five algorithms. Although the search engine of DCC/IDVCA is merely a simple original DE when compared with the other four algorithms, DCC/IDVCA performs the best among the four MOEAs. This further demonstrates the effectiveness of the proposed method.

We also find that two recently proposed large-scale multi-objective evolutionary algorithms, LMEA and DGEA, do not perform well in solving the robust order scheduling problem. Since LMEA requires perturbation to determine whether each variable is diversity-related or convergence-related, followed by interaction analysis of the convergence-related variables, and the interaction relationships among decision

variables in practical order scheduling problems are complex, the entire process described above incurs significant computational costs. The key aspect in addressing robust order scheduling problems is that the classification based on variable contributions to robustness outperforms the classification based on convergence and diversity. The DGEA based on adaptive offspring generation strategy may generate a large number of mutually non-dominated individuals during optimization when dealing with complex real-world order scheduling problems (taking into account robustness constraints). Therefore, the optimization process focuses more on enhancing algorithm diversity, ultimately leading to inadequate convergence performance of the algorithm.

Performance comparison in solving scheduling problems with 60 orders

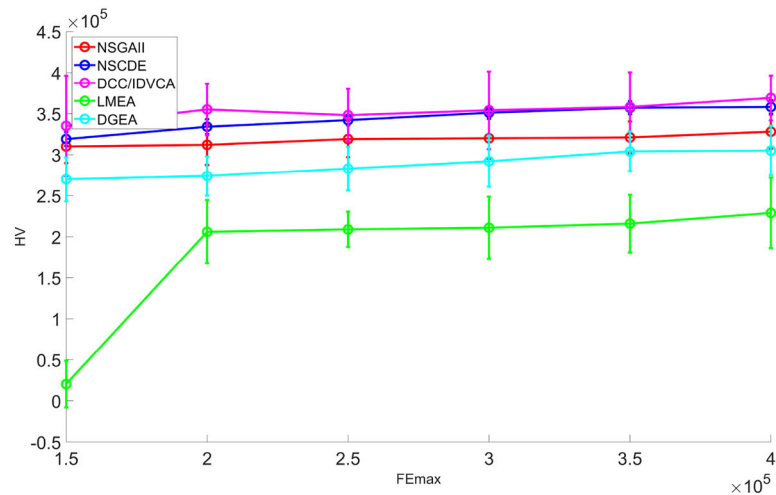
In this subsection, we aim to assess the performance of DCC/IDVCA in handling higher dimensional problems by considering additional orders. Hence, we, respectively, adopt the algorithms DCC/IDVCA, NSGA-II, NSCDE, LMEA, and DGEA under different conditions of FE_{max} to handle an order scheduling problem with 60 orders, and then compare the performance of these five algorithms. The IGD and HV values of DCC/IDVCA, NSGA-II, NSCDE, LMEA, and DGEA under six different FE_{max} conditions have been calculated and are presented in Fig. 7. The median attainment surfaces obtained from 30 runs of DCC/IDVCA, NSGA-II, NSCDE, LMEA, and DGEA under six different FE_{max} conditions are provided in Fig. 8.

From Figs. 7 and 8, we can conclude that the performance of DCC/IDVCA under six different conditions of FE_{max} (i.e., at different optimization stages) remains significantly superior to the other five algorithms, as the dimension size of the order scheduling problem increases.

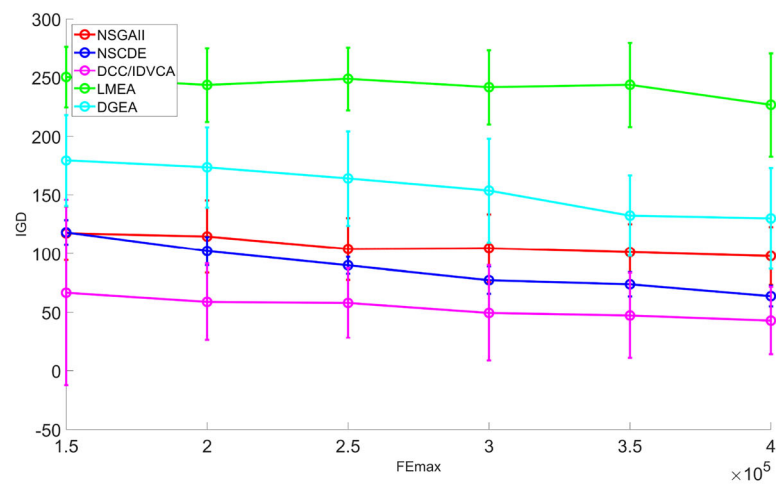
Comparison of DCC/IDVCA with random grouping and linear grouping

To investigate the effectiveness of simple grouping techniques, such as random grouping and linear grouping, in classifying decision variables based on robustness, we conducted comparative experiments between DCC/IDVCA and random grouping as well as linear grouping [24]. Linear grouping divides variables into groups based on their position in the variable space [24]. For instance, assuming that

Fig. 5 The convergence profiles with error bars of variance are presented for HV and IGD values obtained by utilizing DCC/IDVCA, NSGA-II, NSCDE, LMEA, and DGEA to solve a scheduling problem with 40 orders. The constant η , which controls the level of robustness, is set to 5



(a) The convergence profile of HV with error bars representing variance.



(b) The convergence profile of IGD with error bars representing variance.

we have 100 variables (x_0, \dots, x_{99}), we assign the decision variables to ten groups using linear grouping in order: $g_0 = \{x_0, \dots, x_9\}, \dots, g_{10} = \{x_{90}, \dots, x_{99}\}$. However, it is evident that in practical high-dimensional robust order scheduling problems, the highly and weakly robustness-related variables are unlikely to be arranged in any specific order.

Therefore, when the variables are not constructed in a certain order, linear grouping becomes ineffective. As for random grouping, it is evident from its name that the grouping is done randomly. In practical high-dimensional robust multi-objective optimization problems, there is no guarantee of the grouping accuracy for highly and weakly robustness-related variables. Furthermore, the experiments in this section utilized the CC framework with random grouping and linear grouping techniques to solve two order scheduling problems presented in this paper. To ensure sufficient computational resources are allocated to each algorithm during the exper-

iments, it is typically considered reasonable to set $FEmax$ to be at least 1000 times greater than the total dimensionality D . In this experiment, $FEmax$ is set to 2.5×10^5 for solving the order scheduling problem with 40 orders for all comparative algorithms, and it is set to 3.5×10^5 for solving the order scheduling problem with 60 orders for all comparative algorithms. The experimental results are shown in Table 5. From the experimental results, it can be observed that DCC/IDVCA achieves the best values for IGD and HV.

Parameter sensitivity analysis

In this section, we will investigate the impact of different values for the parameters a , ON , and EN . The parameter a determines the impact of the *archive* on the weighted contribution of variables to robustness. If the parameter a is set too large, it can lead to an exaggerated influence of the *archive*

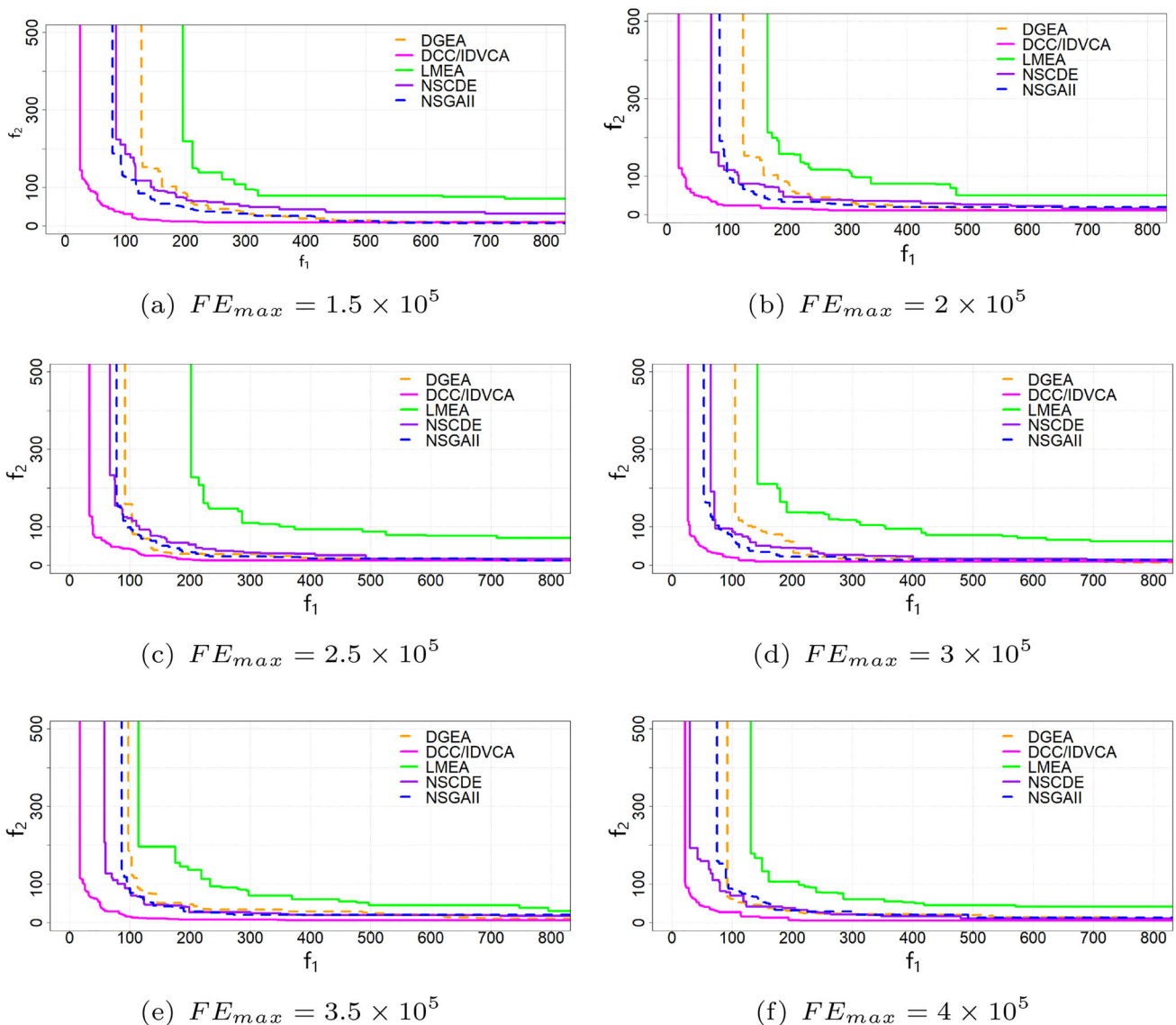


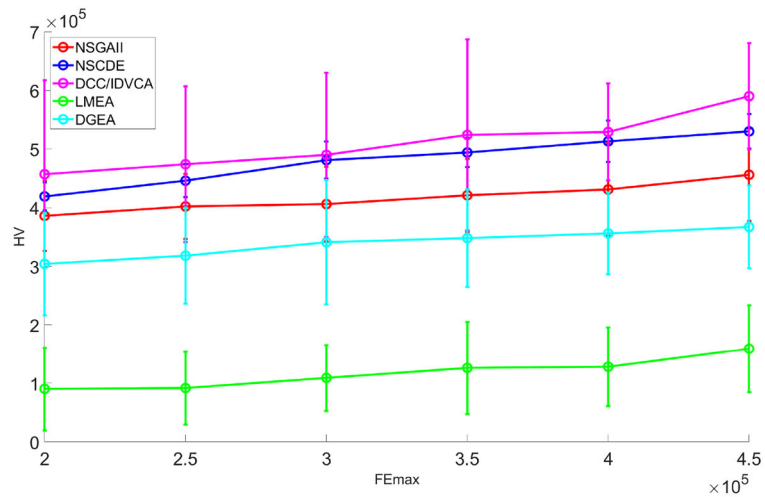
Fig. 6 Comparison of the median attainment surfaces obtained from DCC/IDVCA, NSGA-II, NSCDE, LMEA, and DGEA for solving a scheduling problem with 40 orders after 30 runs under different FE_{max} conditions. The constant η , which controls the level of robustness, is set to 5

on the weighted contribution, thereby amplifying the impact of misclassifications from previous cycles. Specifically, if a weakly robustness-related variable was mistakenly categorized as a highly robustness-related variable in the previous cycle, a large value of parameter a would greatly increase the likelihood of this variable being persistently misclassified as a highly robustness-related variable in subsequent cycles. On the contrary, a small value of a will result in a diminished impact of the *archive* on the weighted contribution. Consequently, the weighted contributions of strongly and weakly robustness-related variables within the same subgroup will be approximately equal. This has a significant impact on the precision of grouping. Typically, the parameter a is set within the range of 1–2. Hence, this paper examines the scenarios

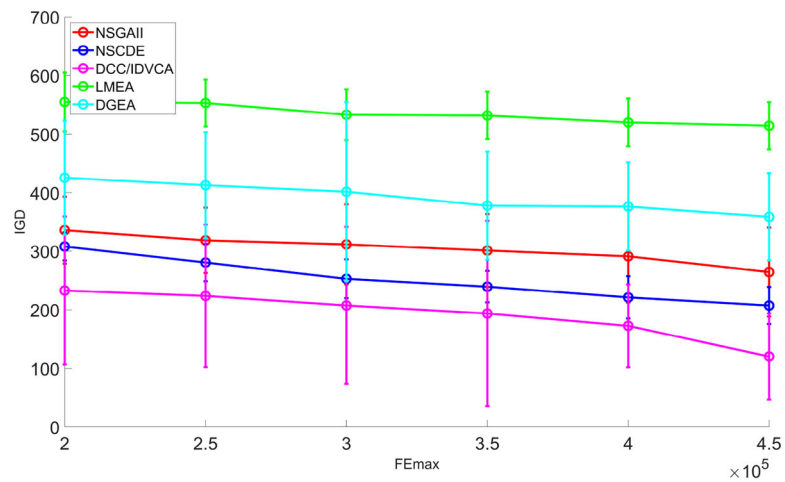
where a takes the values of 1.2, 1.6, and 2, respectively (with three values selected at intervals of 0.4 within the range of 1–2).

The parameters ON and EN represent the sizes of each subgroup during the odd-numbered cycles and even-numbered cycles, respectively, in the dynamic grouping optimization phase of DCC/IDVCA. If the parameters ON and EN are set too large, the number distribution of highly and weakly robustness-related variables within each dynamic subgroup will become more complex. Consequently, this will ultimately impact the grouping accuracy of the algorithm. On the contrary, if the parameters ON and EN are set too small, the algorithm is prone to get trapped in local optima, thus affecting the overall optimization performance of the

Fig. 7 The convergence profiles with error bars of variance are presented for HV and IGD values obtained by utilizing DCC/IDVCA, NSGA-II, NSCDE, LMEA, and DGEA to solve a scheduling problem with 60 orders. The constant η , which controls the level of robustness, is set to 5



(a) The convergence profile of HV with error bars representing variance.



(b) The convergence profile of IGD with error bars representing variance.

algorithm. The setting of dynamic subgroup sizes must meet the following two requirements: (1) The dynamic subgroup size is typically set to be 2% to 8% of the total dimensionality, the disadvantages of setting the dynamic subgroup size to be either too large or too small have been discussed above; (2) The dynamic subgroup size must be divided evenly by the total dimensionality, meaning that the dynamic subgroup size must be an integer. It is explained that there is an empirical practice in setting the size of dynamic subgroups. Setting a relatively smaller subgroup size is beneficial for improving grouping accuracy. Typically, the subgroup size ON during the first optimization (odd-numbered cycles) is set smaller than the subgroup size EN during subsequent optimizations (even-numbered cycles). This is because higher grouping accuracy in the initial optimization cycle is advantageous for the subsequent optimization process. Based on the comprehensive discussions mentioned above, there are various

combinations for setting the dynamic subgroup size of the 160-dimensional order scheduling problem in this paper, as illustrated in Table 6.

Based on the experimental results, it can be observed that the combination of dynamic subgroup size with $ON=8$ and $EN=10$ achieves the optimal IGD values when the parameter a is set to 1.2 and 1.6, while the IGD value is suboptimal when the parameter a is set to 2. Furthermore, when the parameter a is set to 2, the algorithm yields the poorest IGD values among the five combinations of dynamic subgroup sizes. However, when the parameter a is set to 1.2, the algorithm demonstrates the optimal IGD values among three combinations of dynamic subgroup sizes and the suboptimal IGD values among two combinations of dynamic subgroup sizes. Based on the comprehensive discussions presented above, it is appropriate to set a to 1.2, ON to 8, and EN to 10.

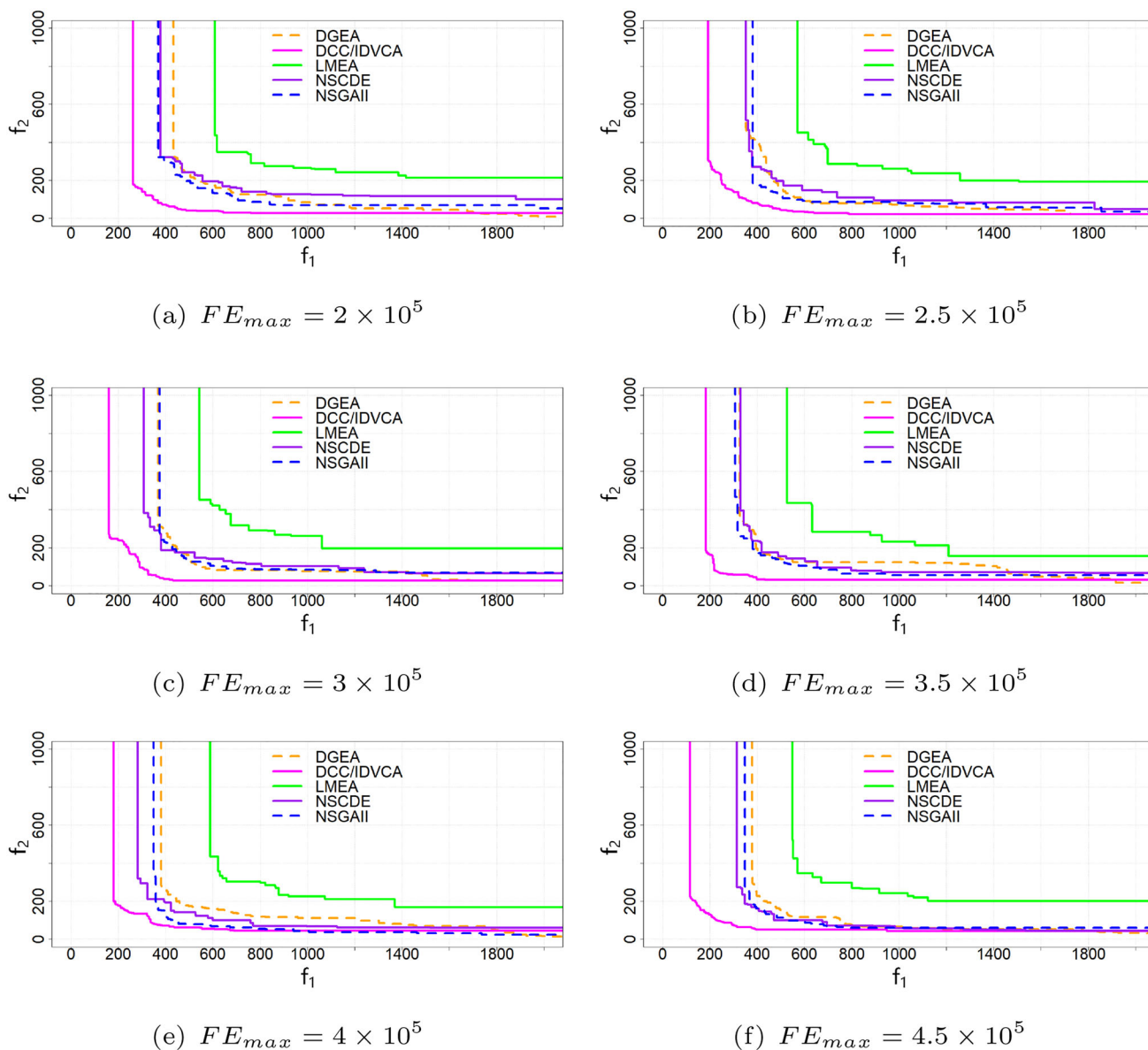


Fig. 8 Comparison of the median attainment surfaces obtained from DCC/IDVCA, NSGA-II, and NSCDE for solving a scheduling problem with 60 orders after 30 runs under different FE_{max} conditions. The constant η , which controls the level of robustness, is set to 5

Table 5 Comparison of DCC/IDVCA with random grouping and linear grouping in solving scheduling problems with 40 and 60 orders. The constant η , which controls the level of robustness, is set to 5

	Algorithms	IGD (mean \pm std)	HV (mean \pm std)
$FE_{max} = 2.5 \times 10^5$ (40 orders)	Random grouping	76.92 \pm 38.57	1.24E+05 \pm 2.14E+04
	Linear grouping	63.38 \pm 32.23	1.31E+05 \pm 1.90E+04
	DCC/IDVCA	40.74 \pm 28.16	1.36E+05 \pm 1.92E+04
$FE_{max} = 3.5 \times 10^5$ (60 orders)	Random grouping	361.24 \pm 269.42	9.08E+04 \pm 8.80E+04
	Linear grouping	289.49 \pm 165.17	1.01E+05 \pm 7.20E+04
	DCC/IDVCA	147.39 \pm 74.04	1.51E+05 \pm 4.55E+04

The highlighted entries demonstrate significant superiority. A Wilcoxon rank-sum test is conducted at a significance level of $\alpha = 0.05$

Table 6 Parameter sensitivity analysis

	$ON = 4$ $EN = 5$	$ON = 4$ $EN = 8$	$ON = 4$ $EN = 10$	$ON = 5$ $EN = 8$	$ON = 5$ $EN = 10$	$ON = 8$ $EN = 10$
$a = 1.2$	80.35 ± 28.31	75.66 ± 26.62	75.94 ± 32.48	63.53 ± 28.67	67.71 ± 30.13	54.69 ± 24.43
$a = 1.6$	75.33 ± 24.17	71.13 ± 27.97	78.32 ± 37.17	79.09 ± 36.36	70.94 ± 29.03	63.60 ± 28.64
$a = 2$	88.14 ± 36.43	77.05 ± 34.31	76.30 ± 30.99	71.65 ± 32.89	58.80 ± 32.02	67.65 ± 30.20

Conclusion

In this paper, a novel MOEA called DCC/IDVCA is proposed to efficiently address high-dimensional robust order scheduling problems. First, the historical information, including the variation of the overall mean effective fitness and the frequency of variables being classified into highly robustness-related subcomponents in previous cycles, is utilized to evaluate the weighted contribution of variables to robustness using ECR method. ECR is the first attempt to evaluate the contribution of each variable to robustness without perturbation. Based on the weighted robustness contributions of candidate solutions, the high-dimensional decision variables are classified into highly and weakly robustness-related variables using IDVCA method. Due to the utilization of historical information in the evolutionary process, IDVCA significantly conserves computational resources through the implicit decomposition of the original variables. Then, these two types of variables are decomposed into highly robustness-related subgroups and weakly robustness-related subgroups within DCC, and the sizes of these subgroups dynamically change in DCC. Finally, different types of subgroups are optimized using different strategies in DCC. In the experimental study, the proposed algorithm is applied to address two order scheduling problems in the discrete manufacturing industry. A series of comprehensive experimental results demonstrate that the introduced algorithm DCC/IDVCA significantly enhances the performance in solving high-dimensional robust order scheduling problems.

It is noteworthy that robust order scheduling is evidently more practical and meaningful. For example, robust order scheduling can provide more warehouse space for early orders in advance, while also arranging for additional overtime work for more operators to handle late orders. In the future, we intend to further enhance the stability of the implicit decomposition approach. Furthermore, we intend to develop a dedicated suite of test functions specifically designed for high-dimensional robust multi-objective optimization, addressing the absence of such specialized benchmarks. Subsequently, the performance of DCC/IDVCA will be further evaluated on these test functions as well as other real-world high-dimensional robust optimization problems. In addition, we will explore the possibility of further classifying decision variables beyond highly or weakly robustness-

related variables, as well as conduct a systematic theoretical analysis in this regard.

Acknowledgments This work was supported by National Key Research & Development Program-Intergovernmental International Science and Technology Innovation Cooperation Project (2021YFE0112800), National Natural Science Foundation of China (62173144, 62273149), National Natural Science Foundation of Shanghai (21ZR1416100), and Fundamental Research Funds for the Central Universities (222202417006).

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Şen A (2008) The us fashion industry: A supply chain review. *Int J Prod Res* 114(2):571–593
- Chan HK, Chan FT (2006) Early order completion contract approach to minimize the impact of demand uncertainty on supply chains. *IEEE Trans Indus Inform* 2(1):48–58
- Branke J, Nguyen S, Pickardt CW, Zhang M (2015) Automated design of production scheduling heuristics: A review. *IEEE Trans Evolut Comput* 20(1):110–124
- Ouelhadj D, Petrovic S (2009) A survey of dynamic scheduling in manufacturing systems. *J schedul* 12:417–431
- Jin Y, Branke J (2005) Evolutionary optimization in uncertain environments—a survey. *IEEE Trans Evolut Comput* 9(3):303–317
- Deb K, Gupta H (2006) Introducing robustness in multi-objective optimization. *Evolut Comput* 14(4):463–494
- Du W, Tang Y, Leung SYS, Tong L, Vasilakos AV, Qian F (2017) Robust order scheduling in the discrete manufacturing industry: A multiobjective optimization approach. *IEEE Trans Indus Inform* 14(1):253–264
- Du W, Tong L, Tang Y (2018) A framework for high-dimensional robust evolutionary multi-objective optimization. In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 1791–1796
- Du W, Zhong W, Tang Y, Du W, Jin Y (2018) High-dimensional robust multi-objective optimization for order scheduling: A deci-

- sion variable classification approach. *IEEE Trans Indust Inform* 15(1):293–304
10. Bellman R (1966) Dynamic programming. *Science* 153(3731):34–37
 11. Mirjalili S, Lewis A, Mostaghim S (2015) Confidence measure: A novel metric for robust meta-heuristic optimisation algorithms. *Inform Sci* 317:114–142
 12. Omidvar MN, Li X, Yao X (2010) Cooperative co-evolution with delta grouping for large scale non-separable function optimization. In: *IEEE Congress on Evolutionary Computation*, pp. 1–8. IEEE
 13. Potter MA, De Jong KA (1994) A cooperative coevolutionary approach to function optimization. In: *Parallel Problem Solving from Nature-PPSN III: International Conference on Evolutionary Computation The Third Conference on Parallel Problem Solving from Nature Jerusalem, Israel, October 9–14, 1994 Proceedings* 3, pp. 249–257. Springer
 14. Bergh F, Engelbrecht AP (2004) A cooperative approach to particle swarm optimization. *IEEE Trans Evolut Comput* 8(3):225–239
 15. Chen W, Weise T, Yang Z, Tang K (2010) Large-scale global optimization using cooperative coevolution with variable interaction learning. In: *Parallel Problem Solving from Nature, PPSN XI: 11th International Conference, Kraków, Poland, September 11–15, 2010, Proceedings, Part II* 11, pp. 300–309. Springer
 16. Omidvar MN, Li X, Mei Y, Yao X (2013) Cooperative co-evolution with differential grouping for large scale optimization. *IEEE Trans Evolut Comput* 18(3):378–393
 17. Omidvar MN, Yang M, Mei Y, Li X, Yao X (2017) Dg2: A faster and more accurate differential grouping for large-scale black-box optimization. *IEEE Trans Evolut Comput* 21(6):929–942
 18. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Trans Evolut Comput* 6(2):182–197
 19. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization* 11(4):341
 20. Tang Y, Gao H, Du W, Lu J, Vasilakos AV, Kurths J (2015) Robust multiobjective controllability of complex neuronal networks. *IEEE/ACM Trans Comput Biol Bioinform* 13(4):778–791
 21. Zhang X, Tian Y, Cheng R, Jin Y (2018) A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization. *IEEE Trans Evolut Comput* 22(1):97–112
 22. He C, Cheng R, Yazdani D (2020) Adaptive offspring generation for evolutionary large-scale multiobjective optimization. *IEEE Trans Syst Man Cybern* 52(2):786–798
 23. Tian Y, Cheng R, Zhang X, Jin Y (2017) Platemo: A matlab platform for evolutionary multi-objective optimization [educational forum]. *IEEE Comput Intell Magaz* 12(4):73–87
 24. Van Aelst S, Wang XS, Zamar RH, Zhu R (2006) Linear grouping using orthogonal regression. *Comput Stat Data Anal* 50(5):1287–1312

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.