Journal of Manufacturing Systems 83 (2025) 470–482

Contents lists available at ScienceDirect

Journal of Manufacturing Systems

journal homepage: www.elsevier.com/locate/jmansys

Technical paper

Causal inference-based dynamic optimization for disassembly line balancing with uncertain component states

Yilin Fang *, Zhiyao Li, Kai Huang, Junyufeng Chen, Ling Gui, Xinyi Chen

School of Information Engineering, Wuhan University of Technology, Wuhan, China

ARTICLE INFO

Keywords:
Disassembly line balancing
Dynamic optimization
Evolutionary algorithm
Causal inference
Uncertain component states

ABSTRACT

Efficient resource recovery from end-of-life (EOL) products requires well-structured disassembly processes. A disassembly line with multiple workstations provides a systematic framework, assigning specific tasks to each station. The disassembly line balancing problem (DLBP) aims to optimize task allocation to improve performance indicators such as cycle time. However, in practical scenarios, the states of EOL product components are often uncertain during the disassembly process. To tackle this challenge, we propose a DLBP with uncertain component states (DLB-UCS) model, which incorporates component state changes as dynamic factors during EOL product disassembly. Unlike conventional DLBP models, DLB-UCS supports real-time adaptation to uncertain changes, making it more consistent with industrial conditions. To solve this problem, we develop a causal inference-based dynamic multi-objective evolutionary algorithm (CI-DMOEA) that simultaneously minimizes the total disassembly cycle time and the number of robotic units required. In particular, a causal feature selection technique based on conditional independence testing is used for efficient initial population generation, enhancing adaptability in dynamic environments. Extensive comparative experiments on eight disassembly scenarios against three state-of-the-art dynamic multi-objective optimization algorithms show that the proposed CI-DMOEA demonstrates superior performance and responsiveness.

1. Introduction

With the rapid advancement of global manufacturing, product life cycles have shortened considerably, resulting in a growing volume of end-of-life (EOL) products [1]. Disassembly plays a crucial role in systematically separating reusable parts, components, and materials from EOL products, thereby enabling efficient resource recovery and remanufacturing [2]. Among different approaches, disassembly lines have proven to be the most efficient and scalable solution for large-scale operations [3]. Disassembly line balancing (DLB) refers to the process of assigning various disassembly tasks to different workstations in a way that optimizes key performance indicators, such as disassembly cycle time, thereby improving the overall efficiency of the line while minimizing idle time and resource waste [4]. In a typical disassembly line, EOL products are transported to the initial workstation, where operators commence the disassembly operations. Workstations are sequentially connected, and products are passed along the line until all disassembly tasks are completed.

In real-world disassembly environments, various uncertainties often lead to uncertain or partially certain conditions of components in EOL products, resulting in significant variability in disassembly times across different products [5]. Current research on uncertainty in DLB problem (DLBP) primarily focuses on modeling uncertain disassembly task times caused by product conditions, human factors, and environmental variability [6], while paying limited attention to the deeper consequences and internal representations of such uncertainties. However, these factors are particularly critical when the internal conditions of EOL products cannot be fully observed in advance. Existing studies commonly rely on inspection systems to acquire complete information about product conditions prior to entering the disassembly line, thereby transforming uncertainty into deterministic inputs for DLB [7]. For large and complex EOL products, however, it is often infeasible to obtain comprehensive internal condition using current inspection technologies. Consequently, developing efficient DLB strategies under uncertain component conditions, where disassembly decisions must be made without access to full information, remains a critical and unresolved challenge in the field.

The presence of uncertain component conditions in disassembly processes essentially transforms the DLBP into a dynamic multi-objective optimization problem (DMOP), where both the task structure and performance objectives evolve over time. In this context, it becomes

* Corresponding author.
E-mail address: fangspirit@whut.edu.cn (Y. Fang).

https://doi.org/10.1016/j.jmsy.2025.09.018
Received 16 June 2025; Received in revised form 19 August 2025; Accepted 29 September 2025
Available online 6 October 2025

critical to develop optimization strategies that can adapt to dynamic changes triggered by progressively changed component states. To address such challenges, dynamic optimization algorithms (DOAs) [8] have been widely applied to solve various DMOPs [9]. In recent years, a variety of DOAs have been proposed to handle time-varying objectives and solution landscapes. These algorithms can generally be categorized into two main types [10]: diversity introduction approaches [11], which aim to preserve or enhance population diversity to maintain adaptability across environmental changes; and prediction-based approaches [12], which seek to exploit historical information and environmental patterns to forecast future Pareto-optimal solutions (POSs) and guide the search process accordingly. However, most existing methods fail to consider the causal relationships between dynamic environments and the predicted populations, resulting in a lack of causal reasoning in solving dynamic optimization problems.

To address these challenges, this paper introduces a DLBP with uncertain component states (DLB-UCS) model, which explicitly accounts for state uncertainty during disassembly. In this model, the condition of internal components can only be observed after outer components are removed. As disassembly progresses, component states are gradually revealed, necessitating continuous updates to the disassembly plan to adapt to the dynamically changing environment. To solve the proposed DLB-UCS, we develop a causal inference-based dynamic multi-objective evolutionary algorithm (CI-DMOEA). The algorithm adapts task assignments as new component states are revealed. In addition, the algorithm employs seed variables to filter causal features, predicts these features, and generates an initial population for the next environment accordingly. The main contributions of this paper can be summarized as follows:

(1) A DLB-UCS model is proposed, which incorporates component state changes as dynamic factors during the disassembly of EOL products. Compared to traditional disassembly models, our model enables real-time adaptation to uncertain state changes, better reflecting practical industrial scenarios.
(2) A CI-DMOEA is developed that leverages causal feature screening and prediction to construct guided initial populations, enhancing adaptability to dynamically changing component states.
(3) The effectiveness of the proposed model and algorithm is validated through comparative experiments on eight disassembly scenarios. Results demonstrate that the proposed approach outperforms three state-of-the-art dynamic multi-objective optimization algorithms in both adaptability and optimization quality.

The remainder of this paper is organized as follows. Section 2 reviews related work on uncertain DLB, DOAs, and causal inference methods. Section 3 formulates the DLB-UCS and illustrates with a representative example. Section 4 presents the details of the proposed CI-DMOEA. Section 5 compares the performance of our approach against three state-of-the-art algorithms across eight problem instances. Finally, Section 6 concludes the paper and outlines future research directions.

2. Literature review

2.1. Uncertain DLB

In real-world disassembly environments, numerous uncontrollable factors such as the actual condition of EOL products, demand fluctuations, and the state of individual components introduce significant uncertainties into the disassembly process. Yuan et al. [13] proposed a multi-objective fuzzy disassembly scheduling model with fuzzy processing time and environmental cost, and developed an improved multi-objective scheduling fruit fly optimization algorithm (MSFOA) to address the challenges posed by uncertain disassembly times. Gao et al. [14] proposed a data-driven method for selective disassembly planning under uncertainty, using uncertainty metrics and a trapezium cloud model to predict disassemblability and optimize disassembly sequences. Liu et al. [15] studied a stochastic multi-product DLBP with workforce assignment, proposing a distributionally robust formulation with Value-at-Risk (CVaR) constraints. Fang et al. [16] proposed a dynamic optimization algorithm, which integrated meta-learning with multi-objective optimization to rapidly generate high-quality initial solutions and accelerate the search for the Pareto Front (PF). Zhang et al. [7] proposed a stochastic multiobjective DLB framework, utilizing a hybrid metaheuristic algorithm that combined the water cycle algorithm (WCA) and simulated annealing (SA). Guo et al. [17] used a hybrid VNS-NSGA II algorithm that integrated variable neighbourhood search (VNS) into NSGA II to solve a multi-objective stochastic hybrid production line balancing problem. Guo et al. [18] proposed a human–robot collaborative DLB model, solving it with an improved multi-objective shuffled frog leaping algorithm (MSFLA). As shown in Table 1, we summarize the uncertainties in the relevant literature along with their associated impacts and the innovative methods used to address them.

However, these strategies often overlook the inherent characteristic of real disassembly environments where component states cannot be fully known in advance. This study focuses on the dynamic state changes of components during the disassembly process and establishes a dynamic model accordingly. To the best of our knowledge, no prior studies have addressed this research problem in the existing literature.

2.2. Dynamic optimization algorithms

A DMOP is a specialized and complex variant of multi-objective optimization, characterized by time-varying objective functions, constraints, or parameters. Multi-objective evolutionary optimization algorithms are commonly employed to address such problems. In the study of memory-based optimization methods, Azzouz et al. [22] proposed an adaptive population management approach that integrates a memory mechanism, local search, and stochastic strategies, while Chen et al. [23] designed novel mating and environmental selection operators to enhance algorithmic performance. Methods based on population diversity can be categorized into two types: diversity preservation and diversity introduction. Among them, Ma et al. [24] developed a strategy based on objective space partitioning to ensure even distribution of new populations, while Ahrari et al. [11] employed adaptive mutation operators to dynamically regulate population diversity. The multiple swarm approach divides the search space into several subregions for parallel optimization through spatial decomposition, particularly effective for multimodal and competitive-peak problems. Liu et al. [25] proposed a coevolutionary multi-swarm particle swarm optimizer that employs an objective decomposition strategy, assigning a dedicated swarm to each objective. In contrast, Gong et al. [26] developed a co-evolutionary framework for dynamic interval multi-objective problems that groups decision variables based on variable similarity metrics. These approaches significantly improve algorithm performance in complex optimization scenarios through diverse strategic approaches.

In the study of DMOPs, prediction-based approaches have attracted significant attention due to their capability to forecast environmental changes using historical data. Cao et al. [27] proposed a novel prediction strategy based on support vector regression (SVR) to generate initial populations for Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [28] in dynamic multiobjective optimization problems exhibiting nonlinear temporal correlations. Liang et al. [10] proposed DMOEA-DVC, which classifies decision variables and applies prediction, maintenance, and diversity strategies to improve dynamic optimization performance. Chen et al. [29] proposed HPPCM, a hybrid prediction and precision-controlled mutation strategy for dynamic multiobjective optimization. Jiang and Yang [30] developed the SGEA algorithm, which combines the fast response capability of steady-state algorithms with the diversity advantage of generational algorithms, leveraging historical information to locate new PF. Chen et al. [31] integrated fuzzy inference with single-step prediction techniques to

Table 1
Summary of related studies on uncertainty in DLBP.

Articles	Uncertainty	Impact	Method
Zhang et al. [7]	Stochastic parameters	Task times	WCA + SA
Yuan et al. [13]	Disassembly process	Task times	MSFOA
Gao et al. [14]	Feasible disassembly	Task times	Data-driven
Liu et al. [15]	Assigned worker	Task times	CVaR constraints
Fang et al. [16]	Dynamic environments	Task times	Meta-learning + MOO
Guo et al. [17]	Disassembly process	Task times	VNS-NSGA II
Guo et al. [18]	Human–Robot collaboration	Task times	MSFLA
He et al. [19]	Integrated problem	Demands	SAA, L-shaped
Tian et al. [20]	Disassembly process	Task times	NGO
Xu et al. [21]	Depreciation	Failure risk	PIMBO
this paper	Component states	Task times	CI-DMOEA

estimate the location and direction of new solutions by analyzing historical POS trajectories. Rong et al. [32] grouped the populations and designed a multiplex prediction strategy, Zhang et al. [33] embedded polynomial fit prediction into the RM-MEDA algorithm to enhance dynamic optimization capability. Sun et al. [34] proposed a two-stage prediction strategy to handle separate centroid and streaming predictions, integrating a Kalman filter with elite solution information. Zheng et al. [35] combined pivot point prediction with diversity strategies to address both predictable and unpredictable DMOPs, Yu et al. [36] further proposed relevance-guided hierarchical prediction methods, which classify populations into subgroups of high, medium, and low relevance via clustering analysis, enabling adaptation to various prediction models. These methods significantly improve the adaptability of the algorithms in dynamic environments through the strategies of historical data mining, model fitting, and hierarchical prediction, and provide diversified solutions for DMOP solving.

Current research on predictive-based DOAs primarily focuses on improving optimization performance. However, various uncertainties exist in practical disassembly processes. Although existing algorithms can achieve good performance under specific environmental conditions, they fail to effectively generate populations adapted to new environments when environmental parameters change abruptly. To address this issue, this study integrates a causality-based feature screening method with DMOPs to predict the initial population when a new environment arises.

2.3. Causal inference methods

Causal analysis research can be broadly categorized into two primary directions [37]: causal discovery and causal effect estimation. Causal discovery aims to uncover causal relationships among variables by leveraging data-driven approaches, while causal effect estimation focuses on quantifying the impact of specific interventions. Existing causal inference methods are generally classified into three categories [38]: functional causal model-based methods [39], which construct causal models by assuming functional relationships between variables to infer causal structures; score-based search methods [40], which evaluate the quality of different causal structures using scoring functions and search algorithms to identify the best causal graph; and constraint-based methods [41], which rely on independence constraints in the data to infer causal relationships, often utilizing conditional independence tests.

Causal inference methods exhibit distinct advantages in achieving stable prediction. Rojas-Carulla et al. [42] employed causally inspired transfer learning techniques to enhance prediction performance, Peters et al. [43] applied causal models for improved inference. Kuang et al. [44] proposed a decorrelation-weighted regression algorithm to enhance prediction stability through sample reweighting, Shen et al. [45] further extended the approach by incorporating multi-environment data to perform variable clustering-based de-correlation. Kuang et al. [46] proposed an algorithm based on conditional independence testing to filter out non-causal correlated features and reduce spurious correlations using seed variables. In most existing studies that utilize predictive models and strategies, the resulting predictions often suffer from compromised stability and interpretability due to biases in the sample data, making them less applicable to the dynamic and variable industrial environments addressed in this study, in contrast to causal feature screening strategies, which can identify stable causal variables and are therefore better suited for such environments. Inspired by these studies, which provide new perspectives for developing interpretable predictive frameworks, this paper introduces a causal feature screening strategy prior to prediction in order to more effectively address the challenges posed by dynamic industrial environments.

Table 2
The indices of DLB-UCS.

Notation	Definition
a, b	Index of artificial nodes
n, h	Index of normal nodes
r	Index of robots
w, j	Index of workstations
ℓ	Index of environments

3. The DLB with uncertain component states

3.1. Problem description

In this study, the primary dynamic factor is the uncertainty in the condition of product components, causing fluctuations in the processing times of disassembly tasks. Damaged components generally require longer processing times than those in normal condition, potentially rendering the initial disassembly plan invalid. As illustrated in Fig. 1, this dynamic variability calls for an adaptive approach. To address this challenge, each disassembly task is characterized by two attributes: an informational attribute (certain or uncertain) and a physical attribute (normal or damaged).

Unlike conventional methods that rely on pre-disassembly inspection to obtain deterministic component state information, our model assumes that the true state of each component is revealed only during disassembly. This eliminates the need for complete global inspection and accounts for real-world limitations such as structural obstructions or undetectable damage. As the disassembly progresses, the system continuously updates component states in real time. Upon detecting condition changes, the disassembly plan is adjusted accordingly. This dynamic response mechanism enhances the applicability of the model to complex industrial scenarios and improves its capacity to handle uncertainty during execution.

3.2. Mathematical formulation

The indices, sets, and variables in DLB-UCS are given in Tables 2, 3, and 4, respectively.

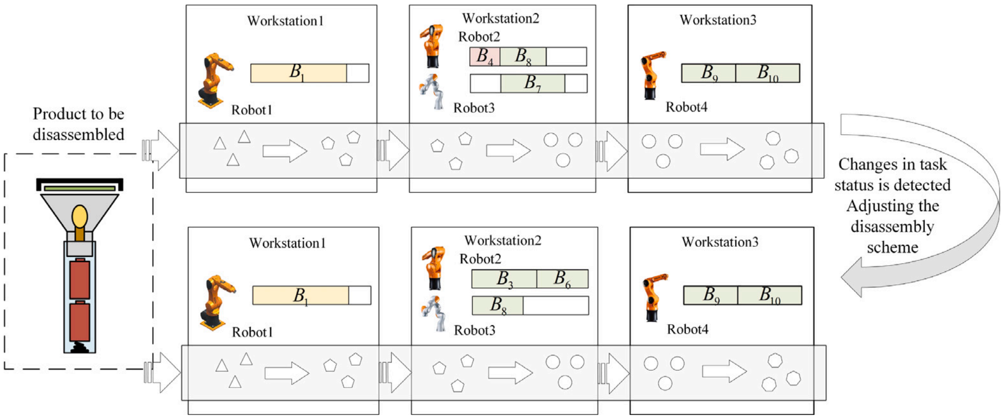


Fig. 1. DLB with uncertain component states.

Table 3
The sets and parameters of DLB-UCS.

Notation	Definition	Notation	Definition
A_a	Artificial node a	B_n	Normal node n
A	Set of artificial node indices	B	Set of normal node indices
P	Set of component indices	R	Set of robot indices
R_j	Set of robots on the j^{th} workstation	W	Set of workstation indices
DT	Set of all selected tasks for disassembly	$S_B(A_a)$	Set of immediate successors of A_a
$P_B(A_a)$	Set of immediate predecessors of A_a	$P_A(B_n)$	Set of immediate predecessors of B_n
$S_A(B_n)$	Set of immediate successors of B_n	$P_{all}(B_n)$	All normal predecessors of B_n in DT
$S_{all}(A_a)$	All artificial successors of A_a	$P_{all}(A_a)$	All artificial predecessors of A_a
$S_{all}(B_n)$	All normal successors of B_n in DT	φ	A big positive number
STA_n	Status of task B_n	R_{max}	Max number of robots per workstation
t_{nr}	Processing time of B_n by r^{th} robot	A_{ing}	Parts being dismantled when environment changes
W_{ing}	Workstations performing disassembly during environment change	B_{ing}	Tasks being executed when environment changes
B_{ed}	Tasks that have been executed	R_n	Robot performing task B_n
R_{ing}	Robots performing tasks when environment changes	ST_n	Start time of task B_n
T_b	Time when the environment changes	R_b	Robot that originally executed the task with changed state

Table 4
The variables of DLB-UCS.

Notation	Definition
Z_n	1, if B_n is selected; 0, otherwise
W_{rj}	1, if the j^{th} workstation uses the r^{th} robot; 0, otherwise
X_{nrj}	1, if task B_n is assigned to the r^{th} robot on the j^{th} workstation; 0, otherwise
Y_{nh}	1, if task B_n is executed before task B_h on the same robot; 0, otherwise
S_n	Starting time of B_n

The objective functions are presented in formulae (1) and (2), where formula (1) aims to minimize the cycle time and formula (2) aims to minimize the number of robots used.

$$F_1 = \min CT(\ell) \tag{1}$$

$$F_2 = \min \sum_r \sum_j W_{rj} \tag{2}$$

Formulae (3) to (26) present the constraints of the proposed mathematical model. Formula (3) defines the cycle time constraint, ensuring that the completion time of all tasks at each workstation does not exceed the specified cycle time.

$$S_n(\ell) \cdot Z_n + \sum_{j \in W} \sum_{r \in R} t_{nr}(\ell) \cdot X_{nrj} \le CT(\ell), \quad \forall n \in B \tag{3}$$

Formulae (4) and (5) ensure that each decision selects one and only one successor node for execution from the disassembly candidates, thereby establishing a unique disassembly sequence. Formula (6) ensures that each task is assigned to exactly one robot.

$$\sum_{B_n \in S_B(A_a)} Z_n = 1, \quad \forall A_a \in A_{ing} \tag{4}$$

$$\sum_{B_n \in S_B(A_b)} Z_n = \sum_{B_n \in P_B(A_b)} Z_n, \quad \forall A_a \in A_{ing}, A_b \in S_{all}(A_a) \tag{5}$$

$$\sum_{j \in W} \sum_{r \in R} X_{nrj} = Z_n, \quad \forall n \in B \tag{6}$$

Formula (7) ensures that the tasks are prioritized in relation to each other. Formula (8) ensures that the selected subsequent task is assigned to a workstation with an index greater than or equal to that of its preceding task.

$$\sum_{B_n \in P_B(A_b)} \sum_{j \in W} \sum_{r \in R} X_{nrj} \ge \sum_{B_n \in S_B(A_b)} \sum_{r \in R} X_{nrw},$$
$$\forall a \ne 0, a \in A, w \in W, A_b \in S_{all}(A_a), A_a \in A_{ing} \tag{7}$$
$$\sum_{B_n \in P_B(A_b)} \sum_{j \in W} \sum_{r \in R} j \cdot X_{nrj} \le \sum_{B_n \in S_B(A_b)} \sum_{j \in W} \sum_{r \in R} j \cdot X_{nrj},$$

$$\forall a \neq 0, a \in A, A_b \in S_{all}(A_a), A_a \in A_{ing} \tag{8}$$

Formula (9) specifies that when two tasks with a precedence relationship are assigned to the same workstation, the start time of the subsequent task must be no earlier than the end time of the preceding task. Formulae (10) and (11) ensure that, for two tasks without a precedence relationship assigned to the same workstation, the interval between their start times is at least the duration of the first executed task.

$$S_n(\ell) - S_h(\ell) + \varphi\left(1 - \sum_{r \in R} X_{hrj}\right) + \varphi\left(1 - \sum_{r \in R} X_{nrj}\right) \geq \sum_{r \in R} t_{nr}(\ell) \cdot X_{nrj},$$
$$\forall j \in W, a \neq 0, a \in A, B_h \in DT \cap P_B(A_a), B_h \in DT \cap S_B(A_a) \tag{9}$$
$$S_h(\ell) - S_n(\ell) + \varphi(1 - X_{hrj}) + \varphi(1 - X_{nrj}) + \varphi(1 - Y_{nh}) \geq t_{nr}(\ell) \cdot X_{nrj},$$
$$\forall B_n \in DT, B_h \in DT - P_{all}(B_n) \cup S_{all}(B_n), j \in W, r \in R \tag{10}$$
$$S_n(\ell) - S_h(\ell) + \varphi(1 - X_{nrj}) + \varphi(1 - X_{nrj}) + \varphi Y_{nh} \geq t_{hr}(\ell) \cdot X_{nrj},$$
$$\forall B_n \in DT, B_h \in DT - P_{all}(B_n) \cup S_{all}(B_n), j \in W, r \in R \tag{11}$$

Formulae (12) and (13) define the set DT and characterize the range of $P_{all}(B_n)$ and $S_{all}(B_n)$.

$$DT = \{B_n \mid Z_n = 1, n \in B\}, \quad |DT| = |P| - 1 \tag{12}$$

$$P_{all}(B_n) \subset DT, \quad S_{all}(B_n) \subset DT, \quad \forall n \in B \tag{13}$$

Formulae (14) through (24) define constraints related to task rescheduling under dynamic environmental changes. Formula (14) ensures that executed and ongoing tasks are to be selected again, and formula (15) ensures that a successor task is to be selected for execution from a part to be disassembled that has not yet begun to be disassembled among the parts to be disassembled.

$$Z_n = 1, \quad \forall B_n \in B_{ed} \cup B_{ing} \tag{14}$$

$$\sum_{B_n \in S_B(A_a)} Z_n = 1, \quad \forall A_a \in A_{ing} - P_A(B_h), B_h \in B_{ing} \tag{15}$$

Formula (16) ensures that the robot currently assigned to a workstation remains assigned to it. Formula (17) maintains consistency of workstation and robot assignments for tasks in execution. Formula (18) defines that when an executed task is re-assigned to a workstation, it should be assigned to either the current workstation or the workstation on which the task has been executed. Formula (19) ensures that in the disassembly task set, the tasks that have not been executed yet should be arranged on the current workstation or the workstation that has not executed the disassembly task when the workstation is arranged.

$$W_{rj} = 1, \quad \forall r \in R_j, j \in W_{ing} \tag{16}$$

$$X_{nrj} = 1, \quad \forall B_n \in B_{ing}, r = R_n, j = W_{ing} \tag{17}$$

$$\sum_{j \in W} \sum_{r \in R} j \cdot X_{nrj} \leq W_{ing}, \quad \forall B_n \in B_{ed} \tag{18}$$

$$\sum_{r \in R} \sum_{j \in W} j \cdot X_{nrj} + \varphi(1 - X_{nrj}) \geq W_{ing}, \quad \forall B_n \in DT - B_{ed} - B_{ing} \tag{19}$$

Formulae (20) and (21) ensure that when rescheduling a task that has already been executed and completed, if the task is scheduled to be executed on the current workstation, the end time of the task is prior to this moment if it is assigned to a robot that is not currently working, or the end time of the task is prior to the start time of the task that is currently being executed if it is assigned to a robot that is executing the task. Formula (22) ensures that when a task being executed is rescheduled, the start time of that task must be the same as the start time of that task in the previous environment.

$$T_b(\ell) - S_n(\ell) + \varphi(1 - X_{nrj}) \geq t_{nr}(\ell), \quad \forall B_n \in B_{ed}, j = W_{ing}, r = R_j - R_{ing} \tag{20}$$

$$ST_n(\ell) - S_h(\ell) + \varphi(1 - X_{hrj}) \geq t_{hr}(\ell),$$
$$\forall B_n \in B_{ing}, B_h \in B_{ed}, j = W_{ing}, r = R_n \tag{21}$$

$$S_n(\ell) = ST_n(\ell), \quad \forall B_n \in B_{ing} \tag{22}$$

Formulae (23) and (24) enforce that when rescheduling tasks that have not yet been performed in the disassembly task set, if the task is scheduled on the current workstation, the start time of the task is after this point in time if it is assigned to a robot that is not currently working, and the start time of the task is after the end time of the task that is currently being performed if it is assigned to a robot that is currently executing the task.

$$S_n(\ell) - S_h(\ell) + \varphi(1 - X_{nrj}) \geq t_{hr}(\ell),$$
$$\forall B_h \in B_{ing}, B_n \in DT - B_{ed} - B_{ing}, j = W_{ing}, r = R_b \tag{23}$$
$$S_n(\ell) + \varphi(1 - X_{nrj}) \geq T_b(\ell), \quad \forall B_n \in DT - B_{ed} - B_{ing},$$
$$j = W_{ing}, r = R_j - R_{ing} \tag{24}$$

Formulae (25) and (26) define the bounds of the decision variables.

$$Z_n, W_{rj}, X_{nrj}, Y_{nh} \in \{0, 1\} \tag{25}$$

$$S_n(\ell) \geq 0 \tag{26}$$

3.3. Illustrative example

To validate the proposed DLBP-UCS model, a small-scale case study is conducted using CPLEX optimization software. The experiment is based on a Task AND/OR Graph (TAOG) [47], a widely used model representing disassembly sequences and precedence relationships, augmented with state attributes as shown in Fig. 2. The product consists of 35 ordinary nodes (disassembly tasks) and 20 artificial nodes (components), among which 8 tasks have uncertain initial states. The system includes 3 serially connected workstations, each allowing 1 – 5 robots, with 15 robots in total. Tasks with known informational attributes are assumed stable throughout execution, whereas those with uncertain attributes may be identified as damaged during disassembly, leading to increased execution times. This modeling approach enables simulation of realistic, state-driven environmental changes during the process.

As illustrated in Fig. 3, the initial disassembly sequence is $DT: \{B_1, B_4, B_5, B_{13}, B_{14}, B_{21}, B_{24}, B_{29}, B_{30}, B_{31}, B_{35}\}$, in which eight robots are deployed across three workstations with a cycle time of seven units. During execution, task B_{21} is detected to be damaged, and its processing time is updated to 20, triggering a dynamic rescheduling. Based on completed tasks $\{B_1, B_4, B_5, B_{13}\}$ and the ongoing task B_{14}, the revised optimal sequence becomes $DT: \{B_1, B_4, B_5, B_{13}, B_{14}, B_{20}, B_{24}, B_{26}, B_{30}, B_{31}, B_{35}\}$. The robot assignments and workstation configurations for already executed and ongoing tasks remain unchanged, while tasks B_{21} and B_{29} are replaced by B_{20} and B_{26} to accommodate the new environment.

4. Causal inference-based dynamic optimization for DLB

4.1. Procedure of algorithm

The proposed CI-DMOEA framework consists of two core phases. In the first stage, it reallocates solutions for tasks whose states have changed between the previous and current environments. This reallocation rigorously enforces real-world constraints, requiring that the disassembly sequence preceding the changed tasks remain unaltered. Only the remaining solution segment is reallocated to adapt to the new environment. In the second stage, the DOA applies conditional independence tests to assess the statistical relationships between a predefined seed variable and other feature variables. This analysis enables the identification of causal features to be employed in the prediction phase. These causal features are then used to forecast their values in the current environment by leveraging historical data from prior environments. Based on the predicted feature vectors, an initial population for the new environment is generated, which enhances the

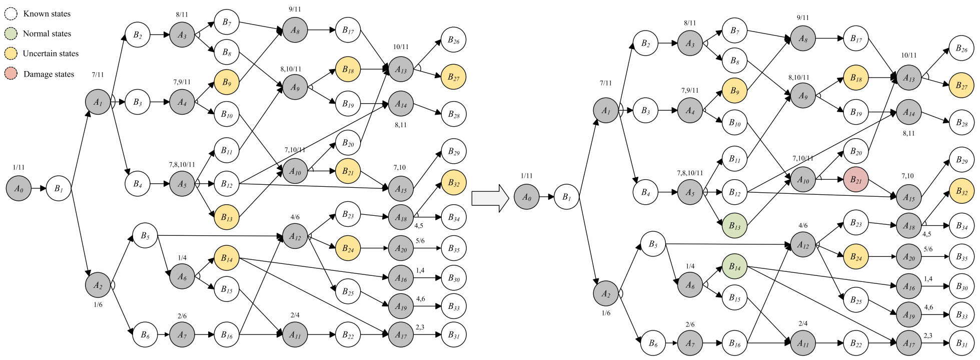


Fig. 2. The TAOG of the product before and after the start of disassembly.

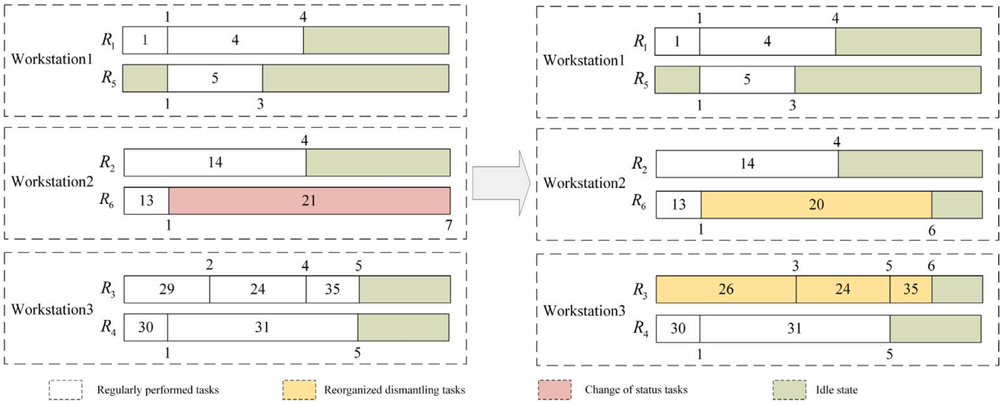


Fig. 3. Disassembly program developed before and after the start of disassembly.

ability of the algorithm to rapidly locate the PF in the current dynamic context. The overall procedure is outlined in Algorithm 1.

Algorithm 1 CI-DMOEA

```
Input: DLB-UCS in $\ell$ environments $(F(\ell), \ell = 0, 1, \ldots)$, MOEA algorithm
Output: POSs $(\mathrm{POS}(\ell), \ell = 0, 1, \ldots)$ of DLB-UCS in $\ell$ environments
1: $\ell \leftarrow 0$
2: init_P ← initialize()
3: $\mathrm{POS}(\ell) \leftarrow$ MOA(init_P, $F(\ell)$)
4: while $\ell < 10$ do
5:     while ! checkEnvirChange() do
6:         wait()
7:     end while
8:     $\ell \leftarrow \ell + 1$
9:     if $\ell \le 3$ then
10:        rein_POP($\ell$) ← POS($\ell - 1$) {Solution re-generation}
11:        init_P ← initialize()
12:        POS($\ell$) ← MOA(init_P, $F(\ell)$) ∪ rein_POP($\ell$)
13:    else
14:        rein_POP($\ell$) ← POS($\ell - 1$) {Solution re-generation}
15:        Data($n$) ← causeSelect(changeIndex, POS(0), …, POS($\ell - 1$)) {Screening for causal characteristics using seed variables}
16:        guid_P ← predict(Data($n$)) {Prediction-based generation}
17:        init_P ← guid_P
18:        POS($\ell$) ← MOA(init_P, $F(\ell)$) ∪ rein_POP($\ell$)
19:    end if
20: end while
21: return POSs $(\mathrm{POS}(\ell), \ell = 0, 1, \ldots)$
```

The algorithm begins by initializing the environment index ℓ to 0 and generating the initial population init_P for the first environment, shown in line 1–2. The initial POS(0) is then obtained by applying the Multi-Objective Algorithm (MOA) to init_P, shown in line 3. The algorithm continuously monitors the environment for changes. When a change is detected, the environment index is incremented, shown in line 8. For early-stage environments $\ell \le 3$, a subpopulation rein_POP(ℓ) is regenerated from the previous POS, and a randomly initialized population is used to update POS(ℓ) via MOEA evolution, shown in lines 10–12. For later-stage environments $\ell > 3$, rein_POP(ℓ) is first regenerated from the previous POS. Then, causal feature selection is performed using historical POSs to obtain Data(n), shown in line 15, followed by prediction-based generation of a guided population guid_P, shown in line 16. The initial population init_P is updated with guid_P, shown in line 17, and MOA evolution produces the final POS(ℓ) for the current environment, shown in line 18. Finally, the populations rein_POP and the evolved population are merged to produce POS(ℓ) for the current environment. The causal and non-causal filtering using historical POSs is performed only when the environment index exceeds 3. This is because reliable identification of causal relationships requires sufficient historical data, which is lacking in the early stages ($\ell \le 3$). Delaying the causal filtering until more environments have been observed ensures more accurate causal modeling and avoids introducing noise due to unstable inference, thereby improving the quality of the predicted guided population guid_P. This design enables CI-DMOEA to maintain high optimization performance while effectively adapting to dynamic environmental changes.

4.2. Method for solution re-generation

Upon detecting component state changes during disassembly, solution components associated with the affected environment may become infeasible or excluded from the POS. To maintain solution feasibility and effectiveness under dynamic conditions, affected individuals require re-generation through environmental adaptation. This re-generation process employs an encoding scheme with three interdependent vectors: Robot-Workstation Assignment Vector (RW), Task-Robot

Assignment Vector (TR), Task Sequence Vector (TS) These vectors collectively represent the allocation and execution structure across robots, workstations, and disassembly tasks, enabling structured yet flexible adaptation. The specific encoding/decoding, crossover, and mutation operations follow the implementation in [16]. Algorithm 2 details the stepwise solution re-generation procedure.

Algorithm 2 Solution Re-generation Method

```
Input: $RW, TR, TS$; $T_b$; $DT$
Output: $RW', TR', TS'$
1: Identify $R_b, W_{ing}$ via $TR, RW$
2: Reset $RW[i] \leftarrow -1$ for $i > W_{ing}$
3: reassign robots to update $RW'$
4: Determine $B_{ed}$ and $B_{ing}$ tasks based on $T_b$
5: Compute $A_{ing} = A_{pro} \setminus A_{suc}$
6: for each part in $A_{ing}$ do
7:     Generate subtask sequence
8: end for
9: Reset $TR[t] \leftarrow -1$ for unexecuted tasks $t$
10: Sort $DT$ by weight; update $TR'$
11: Update $TS'$
12: return $RW', TR', TS'$
```

(1) Identify the solution containing the change factor, and then modify it based on the original configuration.

(2) Using the TR and RW encodings and the dynamic task index from the original solution, identify the robot index R_b and the current workstation index W_{ing} responsible for the dynamic task. Initialize all positions in the RW vector corresponding to subsequent workstation indices to -1, then reassign robots to these workstations, completing the re-generation of the RW encoding (line 1).

(3) Identify the set of completed tasks on the preordered workstations of the current workstation W_{ing}. Simultaneously, extract tasks assigned to W_{ing} whose end time is earlier than T_b. These two sets together constitute the completed task set B_{ed}. Additionally, from the tasks assigned to W_{ing}, select those whose start time is before T_b and end time is after T_b, forming the ongoing task set B_{ing} (line 2–4).

(4) Filter out the two ordinary node sets B_{ed} and B_{ing} from all ordinary nodes connected to the direct preorder artificial nodes to construct the preorder artificial node set A_{pro}. Similarly, extract successor artificial nodes to form the successor node set A_{suc}. The set difference between A_{pro} and A_{suc} yields the set A_{ing}, representing the parts to be processed at the current time. For each element in A_{ing}, generate the corresponding subtask disassembly program, and combine these programs to form the new disassembly sequence (line 5–8).

(5) Reset the values in the TR encoding corresponding to unexecuted tasks. Analyze and rank the tasks listed in the DT based on their weights, and assign the highest-weighted tasks to robots, specifying their start and end times (line 9–10).

(6) Update TS codes for new solutions that have been allocated (line 11).

4.3. Initial population prediction based on causal features screening

Traditional machine learning methods may rely on spurious correlations in the training data for prediction, which are inherently unstable across different environments. To enhance the robustness of predictive models, this study adopts a conditional independence testing method [46] to identify and exclude non-causal features. Feature variables can be categorized into causal (C) and non-causal (N) variables, with the latter further divided into correlated (L) and independent (I) types. In practice, the distinction between L and I is made by testing each variable's conditional independence with the target variable Y given the seed causal variable C_0, representing the disassembly task affected by component state changes. Variables conditionally independent of Y given C_0 (high p-values) are considered independent non-causal variables (I) and removed, while those showing dependence (low p-values), including L and C, are retained.

In the proposed DLB-UCS model, environmental changes are primarily triggered by changes in the states of certain specific components. When the state of a specific disassembled component changes, the associated execution time of its corresponding disassembly task also varies, which significantly impacts the selection of the disassembly plan. Therefore, this paper selects the task index associated with such a change as the seed variable C_0, and constructs a causal feature vector comprising the seed variable and multiple causal features identified through it. In each environment, labeled data $\{X_\ell, Y_\ell\} = \{x_1^\ell, x_2^\ell, \ldots, x_n^\ell, y^\ell\}$, comprising the disassembly task order vector and the corresponding solution in the POS, is recorded. When a new environment is encountered, historical data from previous environments is utilized as training data for causal feature selection.

Algorithm 3 Population Prediction Algorithm Based on Causal Feature Screening

```
Input: Historical environment datasets $\{X, Y\} = \{(X_0, Y_0), (X_1, Y_1), \ldots, (X_{\ell-1}, Y_{\ell-1})\}$, seed variable $C_0$
Output: $guid\_P(\ell)$
for each $(X_i, Y_i) \in (X, Y)$ do
    $C \leftarrow \text{selectC}(C_0, X)$
    for each $C_i \in C$ do
        $pv_i \leftarrow \text{CI-test}(C_i \perp\!\!\!\perp C_0 \mid Y)$
    end for
end for
$C_{\text{rank}} \leftarrow \text{Ranking}(C, PV)$
$D(k) \leftarrow C_{\text{rank}}$
$\{D_0, D_1, \ldots, D_{\ell-1}\} \leftarrow D(k)$
for $i = 1$ to $N$ do
    for $j = 1$ to $k$ do
        $\{(x_i, y_i)\}_{i=1}^{\ell-q-1} \leftarrow \{((v_{i,j}^0, \ldots, v_{i,j}^{q-1}), v_{i,j}^q), \ldots, ((v_{i,j}^{\ell-q-1}, \ldots, v_{i,j}^{\ell-2}), v_{i,j}^{\ell-1})\}$
        $\{(x_i, y_i)\}_{i=1}^{\ell-q-1} \xrightarrow{\text{train}} f_{\text{SVR}}$
        $v_{i,j}^\ell \leftarrow f_{\text{SVR}}(v_{i,j}^{\ell-q}, \ldots, v_{i,j}^{\ell-1})$
    end for
    $DT_i(\ell) \leftarrow \text{selectByPredic}(v_{i,1}^\ell, v_{i,2}^\ell, \ldots, v_{i,k}^\ell)$
    $ind_i(\ell) \leftarrow DT_i(\ell)$
    $guid\_P(\ell) \leftarrow ind_i(\ell)$
end for
return $guid\_P(\ell)$
```

The matrix $\mathbf{X}$ encapsulates the historical data of all variables in the disassembly task order vector, where each column represents a variable and each row corresponds to a sample. Let n denote the number of variables, and $\mathbf{x}_i \in \mathbf{X}$ represent the historical data of a single variable across all environments. Each row encodes task selections of each task in a solution; for instance, if task 4 is selected and task 6 is not, their respective values are 1 and 0. The response variable Y comprises binary labels (0 or 1) assigned based on non-dominated sorting and crowding distance rules. Specifically, the top 60% of the sorted solutions are labeled 1, while the bottom 40% are labeled 0.

The screening algorithm calculates the conditional independence test for each variable X_i conditional on Y and the seed variable C_0, respectively, and sorts the calculated p-values pv_i of each variable to select the top k variables, i.e., screening out the latter $n-k$ variables that are more likely to be independent and non-causal variables.

In this paper, a prediction method based on SVR [27] is employed to forecast causal features after an environmental change, aiming to generate a high-quality initial population. The SVR predictors are trained to capture temporal patterns in historical data. Specifically, for each causal feature $v_{i,j}$, the input is a time-series vector $(v_{i,j}^{\ell-q}, \ldots, v_{i,j}^{\ell-1})$ from the past q environments, and the output is the predicted value $v_{i,j}^\ell$

Table 5
The details of the 8 product instances.

Instance	Number	Number of tasks	Task index with uncertain status
Model1 [47]	P1	23	3, 6, 9, 12, 16
Model2 [48]	P2	20	3, 8, 14
Model3 [49]	P3	10	2, 3
Model4 [50]	P4	13	3, 5, 8
Model5 [51]	P5	22	4, 11, 14, 18
Model6 [51]	P6	30	4, 12, 19, 21, 25
Model7 [52]	P7	37	4, 7, 11, 13, 19, 22, 26
Model8 [50]	P8	32	5, 8, 13, 19, 20

in the new environment. Let $\{D_0, D_1, \dots, D_{\ell-1}\}$ denote the historical causal feature datasets, where each $D_\ell \in \mathbb{R}^{N\times k}$ contains k features for a population of size N in the ℓ-th environment.

Based on the predicted causal feature values, features are categorized into selected and non-selected categories, depending on their likelihood of being chosen. An artificial node is selected using the predicted seed variable, and a complete disassembly sequence is constructed via backward traversal. This results in a guided population tailored to the new environment, which is used to initialize the optimization process. The detailed steps of the population prediction based on causal feature screening are outlined in Algorithm 3, where lines 3–5 select causal features based on the seed variable C_0; lines 6–8 perform conditional independence tests to compute p-values for feature ranking; lines 9–11 rank and select the top-k causal features; lines 12–16 train SVR models using historical data and predict feature values in the new environment; and lines 17–20 construct the guided population used for initializing optimization.

5. Computational experiments

5.1. Test instances

In this paper, eight test cases are generated by assigning state attributes to the TAOG model for eight real-world examples. The detail of these 8 product instances is shown in Table 5.

This experiment considers a total of 25 available robots, imposing an maximum capacity of 5 robots per workstation distributed over 4 workstations. The number of robots assigned to each workstation can vary within the specified range, allowing for unallocated robots. Based on this, ordinary nodes with unpredictable states are stochastically assigned normal or damaged states during the disassembly process, generating eight distinct test cases. The component state is classified into two conditions: normal and damaged. The processing times for these two states of the normal node correspond to the two processing times t_{nr} and $\left[1.4t_{nr},\ 2.5t_{nr}\right]$, which are randomly generated within the given interval.

5.2. Compared algorithms and performance indicators

This paper evaluates the effectiveness of the proposed CI-DMOEA using standard performance indicators and compares the experimental results with three state-of-the-art DOAs: MOEA/D-SVR [27], DMOEA-DVC [10], and HPPCM [29]. The three algorithms address DOA by integrating prediction, decision variable classification, and hybrid mutation strategies to improve adaptation and diversity in changing environments.

To investigate the performance and stability differences between the proposed CI-DMOEA and the comparison algorithms, we conduct comparative experiments. Each algorithm was independently executed 10 times for each test problem. parameter settings are: crossover probability of 0.6, mutation probability of 0.4, population size of 100, and the number of evolutionary iterations per run was 50. At the end of each run, performance metrics of each algorithm are recorded to form the dataset for experimental analysis. All algorithms are implemented using C++ in Visual Studio platform and executed on a Windows 11 operating system with Intel(R) Core(TM) i5-8250U CPU @ 1.60 GHz and 8 GB RAM.

We adopt four commonly used performance indicators [29]: mean inverted generational distance (MIGD), mean hypervolume (MHV), Stability_IGD, and Stability_HV. MIGD measures the average IGD values across all environments, reflecting the overall convergence of the obtained solutions to the true POS, where smaller values indicate better convergence. MHV evaluates the average hypervolume over all environments, and larger values suggest better convergence and diversity of the obtained PF. Stability_IGD and Stability_HV represent the variances of IGD and HV over time, respectively, which indicate the stability of algorithm performance in dynamic environments. Lower values of these stability indicators imply better robustness of the algorithm. To ensure fair comparison, the solution set obtained by each algorithm is normalized based on the approximate PF, which is constructed by extracting the non-dominated solutions from the union of all runs. The reference point for HV calculation is set to 1.1 times the nadir point of this approximate front [53].

5.3. Performance on test instances

To evaluate the impact of the parameter k in CI-DMOEA on performance, 10 independent experiments were conducted, each involving 10 dynamic environments generated by tasks with uncertain states. The parameter k, which determines the number of selected causal features, was tested at 20%, 40%, 60%, 80%, and 100% of the task sequence length. Table 6 summarizes the mean and variance of four metrics under different k values. The results indicate that setting k to 60% achieves optimal performance on MIGD and MHV in six instances, improved Stability_IGD in five instances, and enhanced Stability_HV in three instances. Across different instances, the number of selected causal features critically influences whether non-causal variables are included or whether sufficient causal variables are retained. Including weakly causal variables can reduce solution stability, whereas excluding many strongly causal variables may degrade solution performance. Overall, selecting approximately half of the task sequence length as causal features enables CI-DMOEA to generate more stable solutions in most cases.

To evaluate the performance of CI-DMOEA, we conduct a series of experiments, in which MIGD and MHV metrics are recorded over ten independent runs for CI-DMOEA and three baseline algorithms across different instances and dynamic environments. Table 7 presents the mean and variance of MIGD values obtained from these runs for CI-DMOEA and the comparison algorithms across eight test cases. The results show that in 7/8 test cases (P1-P5, P7-P8), CI-DMOEA achieves a lower mean MIGD than the baseline algorithms, indicating the superior convergence performance of CI-DMOEA.

Similarly, Table 8 presents the mean and variance of MHV obtained from ten independent runs of CI-DMOEA and the comparison algorithms across eight test instances. Analysis of the data shows that in four test cases (P3, P4, P6 and P8), CI-DMOEA demonstrates superior performance in terms of MHV. In terms of MHV, the results indicate that the proposed CI-DMOEA outperforms the comparison algorithms.

Fig. 4 illustrates the outcomes of the Friedman rank sum test applied to the MIGD and MHV metrics, comparing CI-DMOEA against other

Table 6
Performance of CI-DMOEA under different k values for instances.

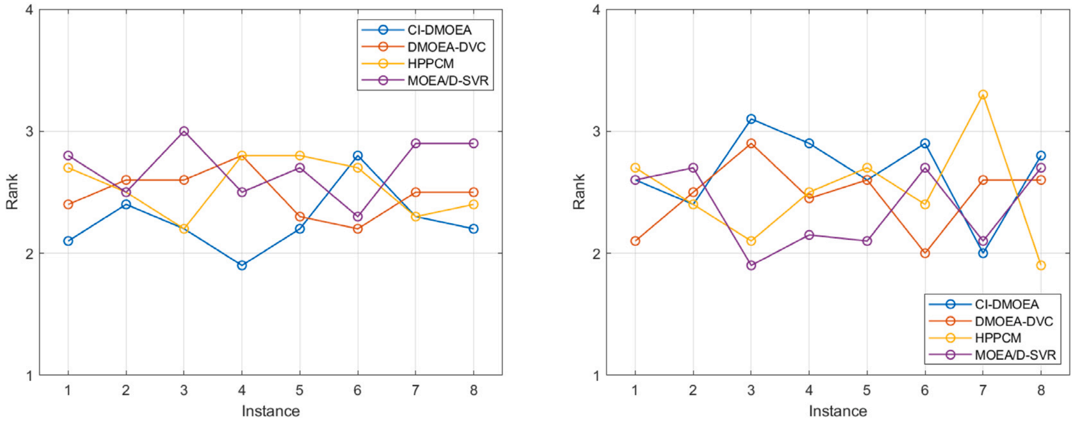
Instance	Parameter values	MIGD	MHV	Stability_IGD	Stability_HV
P1	20%	1.0530E−1 [2.09E−4]	6.9032E−1 [1.56E−4]	6.5908E−2 [1.27E−4]	1.0438E−1 [3.07E−4]
	40%	1.0258E−1 [1.34E−4]	6.8998E−1 [2.33E−4]	6.4227E−2 [7.30E−5]	1.0745E−1 [4.05E−4]
	60%	**9.2726E−2** **[3.77E−4]**	**6.9925E−1** **[3.84E−4]**	**5.4103E−2** **[1.08E−4]**	1.0306E−1 [3.43E−4]
	80%	1.0037E−1 [2.39E−4]	6.9629E−1 [2.36E−4]	5.9523E−2 [9.50E−5]	**9.8864E−2** **[2.86E−4]**
	100%	1.0168E−1 [3.03E−4]	6.8558E−1 [2.13E−4]	6.2050E−2 [9.90E−5]	1.0664E−1 [3.94E−4]
P2	20%	7.7355E−2 [9.60E−5]	6.6457E−1 [1.91E−4]	3.8913E−2 [1.06E−4]	1.1143E−1 [9.06E−4]
	40%	6.9132E−2 [1.93E−4]	6.6807E−1 [3.26E−4]	3.8445E−2 [3.29E−4]	1.1619E−1 [9.90E−4]
	60%	**6.8002E−2** **[7.20E−5]**	6.7486E−1 [1.79E−4]	3.3880E−2 [2.80E−5]	1.0134E−1 [2.00E−4]
	80%	7.0957E−2 [5.00E−5]	6.7598E−1 [1.60E−4]	**3.3254E−2** **[1.90E−5]**	**1.0092E−1** **[1.80E−4]**
	100%	7.1284E−2 [3.80E−5]	**6.7850E−1** **[1.45E−4]**	3.5436E−2 [2.20E−5]	1.0213E−1 [7.60E−5]
P3	20%	7.4735E−2 [2.11E−4]	6.0947E−1 [4.88E−4]	7.6034E−2 [2.68E−4]	1.6913E−1 [3.71E−4]
	40%	7.3771E−2 [2.39E−4]	6.0958E−1 [6.00E−4]	**6.6050E−2** **[2.20E−4]**	1.6807E−1 [2.09E−4]
	60%	7.3041E−2 [6.40E−5]	**6.0978E−1** **[7.92E−4]**	7.0950E−2 [7.30E−5]	**1.6239E−1** **[3.13E−4]**
	80%	**6.7260E−2** **[2.90E−4]**	5.9840E−1 [4.51E−4]	6.7963E−2 [1.10E−4]	1.6618E−1 [4.21E−4]
	100%	7.8856E−2 [4.11E−4]	5.9617E−1 [7.09E−4]	6.9448E−2 [3.70E−5]	1.7045E−1 [1.61E−4]
P4	20%	9.8905E−2 [4.02E−4]	4.6677E−1 [9.40E−5]	1.2932E−1 [9.39E−4]	1.7621E−1 [1.29E−4]
	40%	9.0639E−2 [1.04E−3]	4.6877E−1 [8.04E−4]	1.1390E−1 [1.33E−3]	1.6785E−1 [7.63E−4]
	60%	**8.9663E−2** **[6.95E−4]**	**4.7777E−1** **[4.30E−4]**	**1.0923E−1** **[1.08E−3]**	**1.6208E−1** **[9.01E−4]**
	80%	1.0258E−1 [2.75E−4]	4.6827E−1 [2.97E−4]	1.3213E−1 [9.15E−4]	1.7386E−1 [3.17E−4]
	100%	1.1392E−1 [5.62E−4]	4.6167E−1 [4.34E−4]	1.4107E−1 [8.14E−4]	1.8562E−1 [3.38E−4]
P5	20%	8.6623E−2 [6.90E−5]	6.6105E−1 [2.84E−4]	4.8360E−2 [1.30E−4]	1.1598E−2 [1.25E−4]
	40%	8.4068E−2 [1.60E−4]	6.6577E−1 [1.40E−4]	4.1791E−2 [6.00E−5]	1.1096E−2 [9.20E−5]
	60%	**8.3357E−2** **[1.93E−4]**	**6.6634E−1** **[4.24E−4]**	**4.0716E−2** **[2.10E−5]**	1.0950E−2 [6.30E−5]
	80%	8.4364E−2 [6.40E−5]	6.6405E−1 [3.89E−4]	4.2114E−2 [4.50E−5]	1.0947E−2 [9.30E−5]
	100%	8.7970E−2 [5.20E−5]	6.5274E−1 [9.70E−5]	4.5484E−2 [7.80E−5]	**1.0923E−2** **[3.10E−5]**
P6	20%	1.8497E−1 [2.05E−3]	6.1625E−1 [2.07E−3]	1.3562E−2 [1.73E−3]	2.2047E−1 [1.88E−3]
	40%	1.7812E−1 [1.95E−4]	6.1683E−1 [4.49E−4]	1.1969E−2 [6.67E−4]	**2.0285E−1** **[7.12E−4]**
	60%	**1.7696E−1** **[4.44E−4]**	**6.3673E−1** **[1.08E−3]**	**1.0548E−2** **[4.06E−4]**	2.0453E−1 [1.07E−3]
	80%	1.8108E−1 [4.51E−4]	6.2019E−1 [2.06E−3]	1.2062E−2 [6.39E−4]	2.0912E−1 [9.49E−4]
	100%	1.7846E−1 [6.62E−4]	6.2986E−1 [2.35E−3]	1.2301E−2 [9.83E−4]	2.0787E−1 [8.31E−4]
P7	20%	1.8507E−1 [1.87E−3]	6.9555E−1 [2.32E−3]	1.1659E−1 [1.41E−3]	1.6088E−1 [2.19E−3]
	40%	1.7533E−1 [1.43E−3]	7.0500E−1 [1.40E−3]	**1.1487E−1** **[7.61E−4]**	1.5025E−1 [1.36E−3]
	60%	**1.6095E−1** **[9.83E−4]**	**7.2245E−1** **[2.09E−3]**	1.1691E−1 [5.64E−4]	**1.4718E−1** **[1.12E−3]**
	80%	1.9511E−1 [9.52E−4]	7.1337E−1 [1.48E−3]	1.2998E−1 [1.22E−3]	1.8718E−1 [3.53E−3]
	100%	1.9541E−1 [7.54E−4]	1.9541E−1 [1.25E−3]	1.3762E−1 [2.55E−3]	1.7159E−1 [2.17E−3]
P8	20%	7.5156E−2 [7.40E−5]	6.7322E−1 [5.70E−5]	4.1835E−2 [6.70E−5]	1.0852E−1 [7.50E−5]
	40%	7.2279E−2 [5.90E−5]	6.7228E−1 [1.68E−4]	3.7637E−2 [5.90E−5]	**9.9352E−2** **[2.13E−4]**
	60%	**7.0116E−2** **[7.20E−5]**	**6.8273E−1** **[7.90E−5]**	**3.6368E−2** **[1.30E−5]**	1.0305E−1 [7.70E−5]
	80%	7.6264E−2 [7.70E−5]	6.7160E−1 [2.96E−4]	4.0022E−2 [7.60E−5]	1.0398E−1 [1.83E−4]
	100%	7.3053E−2 [1.21E−4]	6.7311E−1 [1.27E−4]	4.3450E−2 [9.80E−5]	1.1241E−1 [6.60E−5]

Table 7
Mean and variance of MIGD for CI-DMOEA and comparison algorithms across test instances.

Instance	MOEA/D-SVR	HPPCM	DMOEA-DVC	CI-DMOEA
P1	1.0443E−1	1.0693E−1	1.0495E−1	**9.4270E−2**
	[2.49E−4]	[2.79E−4]	[4.48E−4]	**[1.83E−4]**
P2	7.0775E−2	7.0216E−2	7.0774E−2	**6.8002E−2**
	[3.80E−5]	[3.02E−4]	[1.04E−4]	**[7.20E−5]**
P3	7.9278E−2	8.2151E−2	7.7475E−2	**7.4847E−2**
	[7.40E−5]	[4.82E−4]	[3.63E−4]	**[5.93E−4]**
P4	1.0705E−1	1.1059E−1	1.1311E−1	**9.1666E−2**
	[6.40E−4]	[6.13E−4]	[3.33E−4]	**[4.47E−4]**
P5	9.0791E−2	8.5627E−2	8.9793E−2	**8.3357E−2**
	[1.07E−4]	[1.17E−4]	[2.32E−4]	**[1.93E−4]**
P6	1.7111E−1	**1.6494E−1**	1.7689E−1	1.7696E−1
	[6.41E−4]	**[1.41E−3]**	[6.03E−4]	[4.44E−4]
P7	1.8845E−1	1.8735E−1	1.8888E−1	**1.7964E−1**
	[1.97E−3]	[1.56E−3]	[8.74E−4]	**[1.65E−3]**
P8	7.5989E−2	7.2287E−2	7.9726E−2	**7.0116E−2**
	[1.59E−4]	[6.10E−5]	[1.09E−4]	**[7.20E−5]**

Table 8
Mean and variance of MHV for CI-DMOEA and comparison algorithms across test instances.

Instance	MOEA/D-SVR	HPPCM	DMOEA-DVC	CI-DMOEA
P1	6.9561E−1	**7.1045E−1**	6.9300E−1	6.9846E−1
	[6.74E−4]	**[4.30E−4]**	[4.69E−4]	[1.40E−4]
P2	**6.7862E−1**	6.7738E−1	6.7757E−1	6.7486E−1
	[8.50E−5]	[6.27E−4]	[2.35E−4]	[1.79E−4]
P3	5.7573E−1	5.8177E−1	6.0586E−1	**6.1420E−1**
	[1.41E−3]	[9.09E−4]	[3.36E−4]	**[4.92E−4]**
P4	4.6327E−1	4.6347E−1	4.6127E−1	**4.7237E−1**
	[2.37E−4]	[3.02E−4]	[4.80E−4]	**[2.46E−4]**
P5	6.5866E−1	**6.6723E−1**	6.5232E−1	6.6634E−1
	[1.26E−4]	**[1.45E−4]**	[9.11E−4]	[4.24E−4]
P6	6.2253E−1	6.3014E−1	6.1891E−1	**6.3673E−1**
	[6.41E−4]	[3.01E−3]	[2.10E−3]	**[1.08E−3]**
P7	7.1032E−1	**7.3050E−1**	7.1089E−1	6.9801E−1
	[8.08E−4]	**[1.38E−3]**	[1.47E−3]	[1.90E−3]
P8	6.7871E−1	6.7287E−1	6.6731E−1	**6.8273E−1**
	[1.34E−4]	[1.71E−4]	[3.68E−4]	**[7.90E−5]**



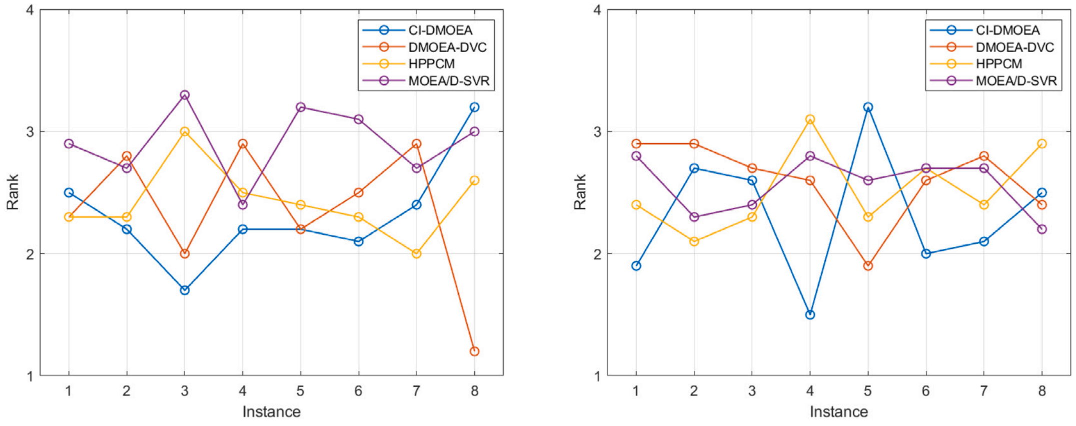
(a) MIGD

(b) MHV

Fig. 4. Friedman rank sum test results of CI-DMOEA and comparison algorithms for two performance metrics under different instances.

algorithms across various test cases. The results indicate that, with respect to the MHV metric, CI-DMOEA exhibits a statistically significant advantage in four test cases and ranks among the top two in six cases. Regarding MIGD, CI-DMOEA demonstrates a significant advantage in five test cases. These findings suggest that CI-DMOEA achieves superior convergence performance and demonstrates broad applicability across diverse dynamic optimization scenarios.

To evaluate the stability performance of CI-DMOEA, we compare it with three baseline algorithms by computing Stability_IGD and Stability_HV across ten dynamic environments, with each experiment independently repeated ten times. Table 9 presents the mean and variance of Stability_IGD obtained from multiple independent runs for CI-DMOEA and the comparison algorithms across eight test instances. The experimental results show that the proposed CI-DMOEA algo-



(a) Stability_IGD

(b) Stability_HV

Fig. 5. Friedman rank sum test results of CI-DMOEA and comparison algorithms for two performance metrics under different use instances.

Table 9
Mean and variance of Stability_IGD for CI-DMOEA and comparison algorithms across test instances.

Instance	MOEA/D-SVR	HPPCM	DMOEA-DVC	CI-DMOEA
P1	6.3598E−2 [9.70E−5]	5.9106E−2 [3.50E−5]	6.2246E−2 [1.79E−4]	**5.7412E−2** **[9.95E−5]**
P2	3.7147E−2 [4.40E−5]	3.5798E−2 [7.70E−5]	3.6909E−2 [7.80E−5]	**3.3880E−2** **[2.80E−5]**
P3	8.0770E−2 [7.00E−5]	7.4643E−2 [1.11E−4]	6.9215E−2 [7.50E−5]	**6.5697E−2** **[1.33E−4]**
P4	**1.2425E−1** **[8.26E−4]**	1.3075E−1 [9.13E−4]	1.4092E−1 [6.00E−4]	1.2471E−1 [1.20E−3]
P5	4.9153E−2 [3.28E−4]	4.9389E−2 [3.70E−5]	4.5943E−2 [5.90E−5]	**4.0716E−2** **[2.10E−5]**
P6	1.2804E−1 [3.03E−4]	1.2485E−1 [6.33E−4]	1.2534E−1 [1.13E−3]	**1.0548E−2** **[4.06E−4]**
P7	1.2051E−1 [5.93E−4]	1.3849E−1 [2.12E−3]	1.4957E−1 [1.73E−3]	**1.1666E−1** **[1.83E−3]**
P8	3.8010E−2 [4.70E−5]	4.1194E−2 [9.20E−5]	4.0048E−2 [7.90E−5]	**3.6368E−2** **[1.30E−5]**

rithm achieves superior performance in terms of Stability_IGD in 7/8 test cases (P1-P3, P5-P8). Overall, the data analysis indicates that CI-DMOEA demonstrates better stability compared to the baseline algorithms.

Table 10 presents the mean and variance of the Stability_HV metric obtained from ten independent runs of CI-DMOEA and the comparison algorithms across eight test instances. The results show that the proposed CI-DMOEA outperforms the comparison algorithms in 6/8 test cases (P1, P4-P8). These experimental results indicate that the proposed CI-DMOEA exhibits a notable advantage in terms of stability performance compared to the baseline algorithms.

Fig. 5 presents the results of the Friedman rank sum test for Stability_IGD and Stability_HV metrics, comparing CI-DMOEA with other algorithms across different test cases. The results show that, in terms of Stability_HV, CI-DMOEA demonstrates a significant advantage in four test cases and ranks among the top two in five cases. For Stability_IGD, CI-DMOEA ranks first in six out of eight cases. These results statistically validate that the strategy of selecting causal features based on seed variables and generating the initial population for new environments using the predicted causal feature vectors effectively enhances the stability of the algorithm. This approach enables CI-DMOEA to consistently search for the PF when encountering new environments.

The experimental results demonstrate that CI-DMOEA, by utilizing seed variables that induce environmental changes to filter out non-causal variables, reduces the interference of confounding factors, enhances prediction stability, and generates a high-quality initial population when new environments emerge, thus improving responsiveness to dynamic environments. In summary, CI-DMOEA performs better in dynamic DLB at the component level, effectively improving solution quality. It demonstrates enhanced stability in responding to environmental changes and is more adaptable to dynamic industrial environments.

6. Conclusion

This study tackles the DLB-UCS problem, a practical industrial challenge in the context of EOL product recycling. To handle the dynamic nature of component condition changes during disassembly, a CI-DMOEA is developed. The proposed method adaptively reallocates disassembly tasks in response to environmental changes while preserving solution feasibility. By identifying task-relevant causal features through conditional independence testing and utilizing historical data for prediction, CI-DMOEA efficiently generates high-quality initial populations suited to new environments. Experimental results on multiple benchmark scenarios demonstrate that the algorithm achieves superior performance and adaptability compared to three leading dynamic optimization approaches.

Future research should aim to extend the DLB-UCS framework to support a wider range of EOL product types and industrial applications. Incorporating more detailed sources of uncertainty and complex task interdependencies could further improve the model realism. In addition, applying the framework to hybrid disassembly lines that simultaneously handle multiple product families would be a valuable

Table 10
Mean and variance of Stability_HV for CI-DMOEA and comparison algorithms across test instances.

Instance	MOEA/D-SVR	HPPCM	DMOEA-DVC	CI-DMOEA
P1	1.0910E−1	1.2301E−1	1.1539E−1	**1.0205E−1**
	[3.79E−4]	[3.57E−4]	[6.28E−4]	**[4.12E−4]**
P2	**1.0008E−1**	1.0413E−1	1.0985E−1	1.0134E−1
	[4.80E−5]	[2.12E−4]	[5.41E−4]	[2.00E−4]
P3	**1.6133E−1**	1.6868E−1	1.8045E−1	1.7632E−1
	[3.66E−4]	[2.99E−4]	[1.60E−4]	[2.79E−4]
P4	1.8791E−1	1.8341E−1	1.8748E−1	**1.7297E−1**
	[2.35E−4]	[5.97E−4]	[2.39E−4]	**[2.77E−4]**
P5	1.1361E−1	1.1342E−1	1.1077E−1	**1.0950E−1**
	[2.52E−4]	[8.60E−5]	[2.27E−4]	**[6.30E−5]**
P6	2.1238E−1	2.3278E−1	2.2030E−1	**2.0453E−1**
	[1.02E−4]	[5.13E−4]	[9.88E−4]	**[1.07E−3]**
P7	1.8304E−1	1.6152E−1	1.8449E−1	**1.5945E−1**
	[2.56E−3]	[3.56E−3]	[3.07E−3]	**[1.51E−3]**
P8	1.0444E−1	1.0572E−1	1.0319E−1	**1.0305E−1**
	[5.40E−5]	[2.17E−4]	[4.90E−5]	**[7.70E−5]**

direction. These advancements could contribute to the development of more intelligent and adaptive disassembly planning systems in practical applications.

CRediT authorship contribution statement

Yilin Fang: Writing – review & editing, Validation, Supervision, Funding acquisition. **Zhiyao Li:** Writing – review & editing, Validation, Supervision. **Kai Huang:** Writing – review & editing, Validation, Supervision. **Junyufeng Chen:** Writing – review & editing, Formal analysis. **Ling Gui:** Writing – original draft, Investigation. **Xinyi Chen:** Writing – review & editing, Writing – original draft, Investigation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the Major Special Projects of National Development and Reform Commission-The project of 30000 t/a High-performance carbon fiber of Zhongfu Shenying Carbon Fiber Lianyungang Co., Ltd. and the National Natural Science Foundation of China under Grant 52075402.

References

[1] Yin T, Zhang Z, Wu T, Zeng Y, Zhang Y, Liu J. Multimanned partial disassembly line balancing optimization considering end-of-life states of products and skill differences of workers. J Manuf Syst 2023;66:107–26.

[2] Tian G, Sheng H, Zhang L, Zhang H, Fathollahi-Fard AM, Zhang X, Feng Y. Enhancing end-of-life product recyclability through modular design and social engineering optimiser. Int J Prod Res 2024;1–19.

[3] Tian G, Zhang C, Fathollahi-Fard AM, Li Z, Zhang C, Jiang Z. An enhanced social engineering optimizer for solving an energy-efficient disassembly line balancing problem based on bucket brigades and cloud theory. IEEE Trans Ind Informatics 2022;19(5):7148–59.

[4] Özceylan E, Kalayci CB, Güngör A, Gupta SM. Disassembly line balancing problem: a review of the state of the art and future directions. Int J Prod Res 2019;57(15-16):4805–27.

[5] Jiang S, Zou J, Yang S, Yao X. Evolutionary dynamic multi-objective optimisation: A survey. ACM Comput Surv 2022;55(4):1–47.

[6] He J, Chu F, Dolgui A, Anjos MF. Multi-objective disassembly line balancing and related supply chain management problems under uncertainty: Review and future trends. Int J Prod Econ 2024;272:109257.

[7] Zhang X, Tian G, Fathollahi-Fard AM, Pham DT, Li Z, Pu Y, Zhang T. A chance-constraint programming approach for a disassembly line balancing problem under uncertainty. J Manuf Syst 2024;74:346–66.

[8] Mavrovouniotis M, Li C, Yang S. A survey of swarm intelligence for dynamic optimization: Algorithms and applications. Swarm Evol Comput 2017;33:1–17.

[9] Nguyen TT, Yang S, Branke J. Evolutionary dynamic optimization: A survey of the state of the art. Swarm Evol Comput 2012;6:1–24.

[10] Liang Z, Wu T, Ma X, Zhu Z, Yang S. A dynamic multiobjective evolutionary algorithm based on decision variable classification. IEEE Trans Cybern 2020;52(3):1602–15.

[11] Ahrari A, Elsayed S, Sarker R, Essam D, Coello CAC. A heredity-based adaptive variation operator for reinitialization in dynamic multi-objective problems. Appl Soft Comput 2021;101:107027.

[12] Yu K, Zhang D, Liang J, Qu B, Liu M, Chen K, Yue C, Wang L. A framework based on historical evolution learning for dynamic multiobjective optimization. IEEE Trans Evol Comput 2023;28(4):1127–40.

[13] Yuan G, Yang Y, Tian G, Fathollahi-Fard AM. Capacitated multi-objective disassembly scheduling with fuzzy processing time via a fruit fly optimization algorithm. Environ Sci Pollut Res 2022;1–18.

[14] Gao Y, Lou S, Zheng H, Tan J. A data-driven method of selective disassembly planning at end-of-life under uncertainty. J Intell Manuf 2023;1–21.

[15] Liu X, Chu F, Zheng F, Chu C, Liu M. Distributionally robust and risk-averse optimisation for the stochastic multi-product disassembly line balancing problem with workforce assignment. Int J Prod Res 2022;60(6):1973–91.

[16] Fang Y, Liu F, Li M, Cui H. Domain generalization-based dynamic multiobjective optimization: A case study on disassembly line balancing. IEEE Trans Evol Comput 2023;27(6):1851–65.

[17] Guo J, Pu Z, Du B, Li Y. Multi-objective optimisation of stochastic hybrid production line balancing including assembly and disassembly tasks. Int J Prod Res 2022;60(9):2884–900.

[18] Guo X, Fan C, Zhou M, Liu S, Wang J, Qin S, Tang Y. Human–robot collaborative disassembly line balancing problem with stochastic operation time and a solution via multi-objective shuffled frog leaping algorithm. IEEE Trans Autom Sci Eng 2023.

[19] He J, Chu F, Dolgui A, Zheng F, Liu M. Integrated stochastic disassembly line balancing and planning problem with machine specificity. Int J Prod Res 2022;60(5):1688–708.

[20] Tian G, Zhang X, Fathollahi-Fard AM, Jiang Z, Zhang C, Yuan G, Pham DT. Hybrid evolutionary algorithm for stochastic multiobjective disassembly line balancing problem in remanufacturing. Environ Sci Pollut Res 2023;1–16.

[21] Xu G, Zhang Z, Li Z, Guo X, Qi L, Liu X. Multi-objective discrete brainstorming optimizer to solve the stochastic multiple-product robotic disassembly line balancing problem subject to disassembly failures. Math 2023;11(6):1557.

[22] Azzouz R, Bechikh S, Said LB. A dynamic multi-objective evolutionary algorithm using a change severity-based adaptive population management strategy. Soft Comput 2017;21:885–906.

[23] Chen Q, Ding J, Yang S, Chai T. A novel evolutionary algorithm for dynamic constrained multiobjective optimization problems. IEEE Trans Evol Comput 2019;24(4):792–806.

[24] Ma X, Yang J, Sun H, Hu Z, Wei L. Multiregional co-evolutionary algorithm for dynamic multiobjective optimization. Inform Sci 2021;545:1–24.

[25] Liu R, Li J, Mu C, Jiao L, et al. A coevolutionary technique based on multi-swarm particle swarm optimization for dynamic multi-objective optimization. European J Oper Res 2017;261(3):1028–51.

[26] Gong D, Xu B, Zhang Y, Guo Y, Yang S. A similarity-based cooperative co-evolutionary algorithm for dynamic interval multiobjective optimization problems. IEEE Trans Evol Comput 2019;24(1):142–56.

[27] Cao L, Xu L, Goodman ED, Bao C, Zhu S. Evolutionary dynamic multiobjective optimization assisted by a support vector regression predictor. IEEE Trans Evol Comput 2019;24(2):305–19.

[28] Zhang Q, Li H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput 2007;11(6):712–31.

[29] Chen Y, Zou J, Liu Y, Yang S, Zheng J, Huang W. Combining a hybrid prediction strategy and a mutation strategy for dynamic multiobjective optimization. Swarm Evol Comput 2022;70:101041.

[30] Jiang S, Yang S. A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization. IEEE Trans Evol Comput 2016;21(1):65–82.

[31] Chen D, Zou F, Lu R, Wang X. A hybrid fuzzy inference prediction strategy for dynamic multi-objective optimization. Swarm Evol Comput 2018;43:147–65.

[32] Rong M, Gong D, Zhang Y, Jin Y, Pedrycz W. Multidirectional prediction approach for dynamic multiobjective optimization problems. IEEE Trans Cybern 2018;49(9):3362–74.

[33] Zhang Q, He X, Yang S, Dong Y, Song H, Jiang S. Solving dynamic multi-objective problems using polynomial fitting-based prediction algorithm. Inform Sci 2022;610:868–86.

[34] Sun H, Ma X, Hu Z, Yang J, Cui H. A two stages prediction strategy for evolutionary dynamic multi-objective optimization. Appl Intell 2023;53(1):1115–31.

[35] Zheng J, Zhou F, Zou J, Yang S, Hu Y. A dynamic multi-objective optimization based on a hybrid of pivot points prediction and diversity strategies. Swarm Evol Comput 2023;78:101284.

[36] Yu K, Zhang D, Liang J, Chen K, Yue C, Qiao K, Wang L. A correlation-guided layered prediction approach for evolutionary dynamic multiobjective optimization. IEEE Trans Evol Comput 2022;27(5):1398–412.

[37] Yao L, Chu Z, Li S, Li Y, Gao J, Zhang A. A survey on causal inference. ACM Trans Knowl Discov from Data (TKDD) 2021;15(5):1–46.

[38] Spirtes P, Zhang K. Causal discovery and inference: concepts and recent methodological advances. Appl Inform 2016;3(1):3.

[39] Zhang K, Wang Z, Zhang J, Schölkopf B. On estimation of functional causal models: general results and application to the post-nonlinear causal model. ACM Trans Intell Syst Technol (TIST) 2015;7(2):1–22.

[40] Ramsey J, Glymour M, Sanchez-Romero R, Glymour C. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. Int J Data Sci Anal 2017;3:121–9.

[41] Triantafillou S, Tsamardinos I. Constraint-based causal discovery from multiple interventions over overlapping variable sets. J Mach Learn Res 2015;16(1):2147–205.

[42] Rojas-Carulla M, Schölkopf B, Turner R, Peters J. Invariant models for causal transfer learning. J Mach Learn Res 2018;19(36):1–34.

[43] Peters J, Bühlmann P, Meinshausen N. Causal inference by using invariant prediction: identification and confidence intervals. J R Stat Soc Ser B Stat Methodol 2016;78(5):947–1012.

[44] Kuang K, Xiong R, Cui P, Athey S, Li B. Stable prediction with model misspecification and agnostic distribution shift. In: Proceedings of the AAAI conference on artificial intelligence, vol. 34, 2020, p. 4485–92.

[45] Shen Z, Cui P, Liu J, Zhang T, Li B, Chen Z. Stable learning via differentiated variable decorrelation. In: Proceedings of the 26th acm sigkdd international conference on knowledge discovery & data mining. 2020, p. 2185–93.

[46] Kuang K, Wang H, Liu Y, Xiong R, Wu R, Lu W, Zhuang Y, Wu F, Cui P, Li B. Stable prediction with leveraging seed variable. IEEE Trans Knowl Data Eng 2022;35(6):6392–404.

[47] Koc A, Sabuncuoglu I, Erel E. Two exact formulations for disassembly line balancing problems with task precedence diagram construction using an AND/OR graph. Iie Trans 2009;41(10):866–81.

[48] Lambert A. Optimal disassembly of complex products. Int J Prod Res 1997;35(9):2509–24.

[49] Tang Y, Zhou M, Zussman E, Caudill R. Disassembly modeling, planning, and application. J Manuf Syst 2002;21(3):200–17.

[50] Lambert AJ. Optimizing disassembly processes subjected to sequence-dependent cost. Comput Oper Res 2007;34(2):536–51.

[51] Lambert A. Linear programming in disassembly/clustering sequence generation. Comput Ind Eng 1999;36(4):723–38.

[52] Ma Y-S, Jun H-B, Kim H-W, Lee D-H. Disassembly process planning algorithms for end-of-life product recovery and environmentally conscious disposal. Int J Prod Res 2011;49(23):7007–27.

[53] Li M, Yao X. Quality evaluation of solution sets in multiobjective optimisation: A survey. ACM Comput Surv (CSUR) 2019;52(2):1–38.