Journal of Intelligent Manufacturing (2023) 34:1065–1083
https://doi.org/10.1007/s10845-021-01837-5

Energy-time tradeoffs for remanufacturing system scheduling using an invasive weed optimization algorithm

Wenjie Wang[1,2] · Guangdong Tian[1,2] · Gang Yuan[3,4] · Duc Truong Pham[5]

Received: 18 May 2021 / Accepted: 31 August 2021 / Published online: 9 September 2021

Abstract
This article studies the scheduling problem for a remanufacturing system with parallel disassembly workstations, parallel flow-shop-type reprocessing lines and parallel reassembly workstations. The problem is formulated as a multi-objective optimization problem which contains both energy consumption and makespan to be addressed using an improved multi-objective invasive weed optimization (MOIWO) algorithm. Two vectors regarding workstation assignment and operation scheduling jointly form a solution. A hybrid initialization strategy is utilized to improve the solution quality and the Sigma method is adopted to rate each solution. A novel seed spatial dispersal mechanism is introduced and four designed mutation operations cooperate to enhance search ability. A group of numerical experiments and a practical case involving the disassembly of transmission devices are carried out and the results validate the effectiveness of the MOIWO algorithm for the considered problem compared with existing methods.

Keywords Remanufacturing system · Disassembly · Shop scheduling · Energy consumption · Multi-objective invasive weed optimization

Introduction

Rapid development of industrial civilization brings a sharp increase in the amount of end-of-life (EOL) products, which leads to our serious environmental concerns (Joshi & Gupta, 2019; Tian et al., 2018). Deciding how to collect, upgrade and handle these EOL products in an environment-friendly approach is an important and pressing issue (Yu & Lee, 2018). When products reach their EOL states, they can be treated in a number of approaches such as repair, recycling, reuse, remanufacturing, and disposal (Heese et al., 2005). Among them, remanufacturing is regarded as one of the advanced EOL options due to its characteristics of high profits, energy saving, and environmental friendliness (Liu et al., 2020; Parkinson & Thompson, 2003). According to British Standard BS 8887 (Part 2), remanufacturing refers to the process of "returning a product to at least its original performance with a warranty which is equivalent (to) or better than that of the newly manufactured product." Remanufacturing is important to the economy, the environment and society, providing skilled employment for people as well as reducing raw material and energy usage (Jiang et al., 2019; Wang et al., 2019; Tian et al., 2017).

A typical remanufacturing system for managing EOL products is composed of three core subsystems, i.e., disassembly, reprocessing, and reassembly (Guide, 2000; Lund, 1984). Its general process can be described as follows: an EOL product is taken apart into its constituent components with essential classification/inspection operations at a disassembly shop (Tian et al., 2019; Wang et al., 2021; Yuan

✉ Guangdong Tian
tiangd2013@163.com; tgd1232001@aliyun.com

[1] Key Laboratory of High Efficiency and Clean Mechanical Manufacture (Ministry of Education), National Demonstration Center for Experimental Mechanical Engineering Education, School of Mechanical Engineering, Shandong University, Jinan 250061, People's Republic of China

[2] State Key Laboratory of Fluid Power and Mechatronic Systems, Zhejiang University, Hangzhou 310027, People's Republic of China

[3] College of Biological and Agricultural Engineering, Jilin University, Changchun 130022, People's Republic of China

[4] Department of Industrial Systems Engineering and Management, National University of Singapore, Singapore 119077, Singapore

[5] College of Mechanical Engineering, University of Birmingham, Birmingham COB152TT, UK

et al., 2020; Zhu et al., 2020), and then the components are recovered to like-new conditions by various advanced processing technologies at a reprocessing shop (Kerin & Pham, 2019). Eventually, the new ones are gathered and assembled into remanufactured products at a reassembly shop. These three shops are interdependent and it is necessary to operate them closely to realize an efficient remanufacturing system (Fu et al., 2021). Regarding the structure of the three shops, there are many distinct system configurations. In our article, we consider the one with parallel disassembly workstations, parallel flow-shop-type reprocessing lines, and parallel reassembly workstations. This configuration is common in many practical remanufacturing systems, for example, in automotive part remanufacturing systems.

The scheduling problem in our studied type of remanufacturing systems is to decide the sequence and allocation of EOL products to be worked on parallel disassembly workstations, sequence of separated components to be processed at each workstation of parallel reprocessing lines, and sequence and allocation of products to be assembled on parallel reassembly workstations, so as to achieve several specific goals. Theoretically, the scheduling problem can be regarded as an ordinary two-stage assembly flow shop scheduling problem with the additional scheduling problem of a disassembly shop. However, the considered type of remanufacturing systems in this article is distinct from hybrid flow shops due to that the reprocessing shop is composed of several flow-shop-type reprocessing lines.

The scheduling problem in remanufacturing systems has attracted the attention of many scholars in the world. Guide (1995) applied a production control method, named drum-buffer-rope, to a Naval Aviation Depot that remanufactured a wide range of parts from aircraft to avionics components. Later, Guide et al. (1997), Guide and Srivastava (1997) and Daniel and Guide (1997) extended his work and examined a group of priority dispatching rules to find out one that best supported the drum-buffer-rope method used in the manufacturing system. Gungor and Gupta (2001) utilized a branch-and-bound algorithm to produce disassembly sequence plans for EOL products remanufacturing. Tian et al. (2018) extended their work and studied the disassembly sequence planning problem with fuzzy information on component quality and operational cost. Further, to realize a cost effective and profitable disassembly, Kalayi and Gupta (2013) studied the disassembly line balancing problem, i.e., assign tasks to workstations for EOL products to be disassembled and aim at realizing several objectives, and proposed the ant colony optimization algorithm to address it. Ozceylan et al. (2019) summarized the previous work on disassembly line balancing problem and pointed out that future researches ought to consider sustainability issues such as, energy usage, green technology and so on.

According to Yu and Lee (2018), the scheduling problem in remanufacturing systems was divided into two categories based on the type of reprocessing shop: flow-shop-type ones and job-shop-type ones. Chikhi et al. (2014) addressed a two-stage flow shop scheduling problem in a robotic cell, where dedicated machines were set at the reprocessing stage and a common machine was set at the assembly stage. Jolai et al. (2012) considered an m-machine identical parts robotic cell scheduling problem with swap and load lock, and provided a new robot move cycle that was proved to be better that the classical ones. Foumani and Jenab (2013) examined the robotic cell scheduling problem with two workstations and considered two different configurations, i.e., free-pick up category and non-wait category. Stanfield et al. (2006) established a network model to represent a reprocessing flow shop and adopted a heuristic approach to minimize work-in-process inventory and maximize production system utilization. Kim et al. (2015) studied remanufacturing systems with one disassembly workstation, flow-shop-type reprocessing lines and a group of reassembly workstations. To minimize total tardiness of EOL products processed in the remanufacturing system, priority scheduling rules were used and experimental results indicated that the rule combination approach where each subsystem adopted a different priority rule performed better than using a single priority rule in all subsystems. Soon after, Kim et al. (2017) studied another kind of remanufacturing systems that have one disassembly workstation, several flow-shop-type reprocessing lines and one reassembly workstation. Three solution algorithms were applied to obtain the minimum completion time. Recently, Yu and Lee (2018) studied remanufacturing systems with a job-shop-type reprocessing shop. In the systems, components obtained from a disassembly shop were grouped into different job families and must meet component matching requirements before being sent to a reassembly shop.

Pedrielli et al. (2018) took the manufacturing system as an unusual type of queueing system and developed a discrete event optimization methodology to establish integrated models. To capture the physical aspects from practical manufacturing systems, Lugaresi et al. (2021) studied a real-time rescheduling problem under a lab-scale environment.

From the literature review, the scheduling models and solution algorithms in remanufacturing systems are extensively addressed. However, their applications are system-specific to some extent. Besides, awareness of energy conservation in the remanufacturing industry and scheduling is increasing (Foumani & Smith-Miles, 2019; Fu et al., 2019). According to Fang et al. (2011), industrial production takes up about 50% of the total energy usage in the world. Further, remanufacturing processes can consume approximately 23% of the energy of production. Energy usage is regarded as one of the most considerable factors that contribute to environmental deterioration, such

as resource depletion and global warming (Lu et al., 2017; Yuan et al., 2020; Yang et al., 2020). Therefore, in this article, we treat the remanufacturing system scheduling problem with energy usage and time consideration. To the best of the authors' knowledge, no previous work has been conducted on this configuration and its energy and time-oriented scheduling objectives are worthwhile to be studied in the aspects of scheduling theory and actual application.

To represent the scheduling problem, a multi-objective mathematical model is established for the purpose of simultaneously minimizing the energy consumption and the makespan. Since a two-stage assembly flow shop scheduling problem with total flow time consideration is already NP-hard, so is the problem studied in the paper. The methods for solving shop scheduling problems can be classified into three categories: exact methods (Fattahi et al., 2014), heuristic algorithms (Thornton & Hunsucker, 2004), and meta-heuristics algorithms (An et al., 2020). Among them, meta-heuristics algorithms are commonly used due to their excellent characteristics of simple implementation, fast convergence and strong search ability for optimal solutions (Mousavi et al., 2019; Natarajan et al., 2020; Zhang et al., 2020).

The Invasive Weed Optimization (IWO) algorithm proposed by Mehrabian and Lucas (2006), is an effective population-based meta-heuristics algorithm. It is inspired by a natural phenomenon from agriculture and simulates the colonization of invasive weeds. Due to its remarkable performance in robustness and self-adaptation (Sang et al., 2018), the IWO algorithm has been successfully applied in no-idle flow shop scheduling (Zhou et al., 2014), permutation flow-shop scheduling (Chen et al., 2013), inventory routing (Jahangir et al., 2019), optimal design of PID controller (Misaghi & Yaghoobi, 2019) and many other fields. To the best of the authors' knowledge, the IWO algorithm has not been utilized in the studied problem. Moreover, the scheduling algorithm for this problem is still in its infancy. Fresher and more effective solution methods are desired. Regarding the successful application of IWO algorithm verified in literature, a multi-objective invasive weed optimization (MOIWO) algorithm is proposed for addressing the multi-objective remanufacturing system scheduling problem considering energy efficiency and productivity.

Compared with previous work, this study makes the following three contributions.

(1) Investigating the scheduling of remanufacturing systems with parallel disassembly workstations, parallel flow-shop-type reprocessing lines and parallel reassembly workstations, where energy and time-oriented objectives are optimized simultaneously.
(2) Formulating a multi-objective mathematical model with several constraints to represent this NP-hard scheduling problem and developing an improved MOIWO algorithm to solve it.
(3) Through numerical experiments and a case study, demonstrating the feasibility and effectiveness of the proposed algorithm, compared with non-dominated sorting genetic algorithm II (NSGA-II) and multi-objective evolutionary algorithm based on decomposition (MOEA/D), in addressing scheduling problems in remanufacturing systems.

The structure of this manuscript is as follows. Section 2 describes remanufacturing systems and illustrates the scheduling problem mathematically. The MOIWO algorithm is discussed in Sect. 3. Numerical experiments and a case study are conducted and their results are presented in Sect. 4. Section 5 concludes this work and outlines our future study lines.

Problem formulation

System description

As explained in Sect. 1, the studied type of remanufacturing systems is composed of parallel disassembly workstations, several flow-shop-type reprocessing lines and parallel reassembly workstations. Its basic procedure is described as: a group of EOL products are decomposed into constituent components on parallel disassembly workstations (DWs) in a disassembly shop, then the separated components (Cs) are reprocessed at several parallel reprocessing lines (RLs) in a reprocessing shop and eventually the remanufactured components are reassembled into remanufactured products on parallel reassembly workstations (AWs) in a reassembly shop. The DWs/AWs are identical and thus EOL products can be disassembled/reassembled with a same processing time on any DW/AW. Note that components must be processed at their corresponding RLs in which the necessary reprocessing processes are done on its serial workstations (Kim et al., 2015), due to the fact that each line is dedicated to processing one kind of component. Besides, components are serial number-specific because they must be matched on AWs. Further, disassembly operations, reprocessing operations and reassembly operations cooperate with each other. In other words, the reassembly operation of a remanufactured product can be executed only after all of its components are reprocessed at the parallel RLs.

In general, a remanufacturing system can be regarded as a three-stage hybrid flow shop, but it is different from the usual hybrid flow shop since it has several parallel RLs in its second stage (i.e., in the reprocessing shop).

Theoretically, the scheduling in the reprocessing shop can be taken as an extension of permutation flow shop scheduling problem. Note that the processing times could be distinct even for the EOL products of the same category with respect to the conditions of EOL products (Kim et al., 2015).

Figure 1 describes an example of studied remanufacturing system with two parallel DWs, three flow-shop-type RLs and two parallel AWs. Two EOL products, i.e., P_1 and P_2 with different statuses (use environment or serve time) are remanufactured through the three serial subshops, where C_{ij} refers to component j of EOL product i. As seen from Fig. 1, EOL products P_1 and P_2 are both with three components. EOL product P_1 can be disassembled on DW_1 or DW_2, and then its constituent components C_{11}, C_{12}, C_{13} can only be reprocessed through RL_1, RL_2, and RL_3, respectively. Finally, those remanufactured components can choose AW_1 or AW_2 to reassemble them into remanufactured P_1. EOL product P_2 has the same routing condition as P_1. Specifically, P_1 and P_2 are firstly taken apart into their constituent components (C_{11}, C_{12}, C_{13} and C_{21}, C_{22}, C_{23}) on DW_1 or DW_2; the components are reprocessed through one of RL_1, RL_2, and RL_3; and the remanufactured and new components are sent to a reassembly shop where AW_1 and AW_2 are waiting to put them together. Figure 2 presents an example schedule of the remanufacturing system described in Fig. 1. It can be found that EOL products P_1 and P_2 are allocated to DW_1 and DW_2, respectively, and their reprocessing sequences are $C_{23} \rightarrow C_{13}$, $C_{22} \rightarrow C_{12}$, and $C_{21} \rightarrow C_{11}$ on RL_1, RL_2, and RL_3, respectively. Finally, P_1 and P_2 are allocated to AW_1 and AW_2, respectively.

Problem description

The problem studied in this paper can be summarized as follows: Given a set of EOL products, the scheduling problem is to decide the sequence and allocation of EOL products to be worked on parallel DWs, the sequence of separated components to be processed at each workstation of parallel RLs, and the sequence and allocation of products to be assembled on parallel AWs, so as to minimize both energy consumption and makespan.

This paper touches upon a static and deterministic type of remanufacturing system scheduling problem, i.e., all necessary parameters, such as the count of EOL products, the count of workstations in the three shops, processing power and processing time, are determined and given in advance. Moreover, the assumptions are as follows: (1) preemption among products/components is not allowed, i.e., once a product/component starts to be worked, it must be completed without any interruption; (2) a workstation can only work a single product/component at a specific time and a product/component can only be worked by a single workstation at a specific time; (3) buffers between two consecutive processing stages are sufficiently large; (4) transportation times of products/components among workstations are negligible; (5) random failure or preventive maintenance of the workstations are not considered; (6) setup times are sequence-independent and are integrated into processing time.

The notations used in this article are summarized as below.

(1) i: index of EOL products, $i \in I = \{1, 2, \ldots, m\}$.

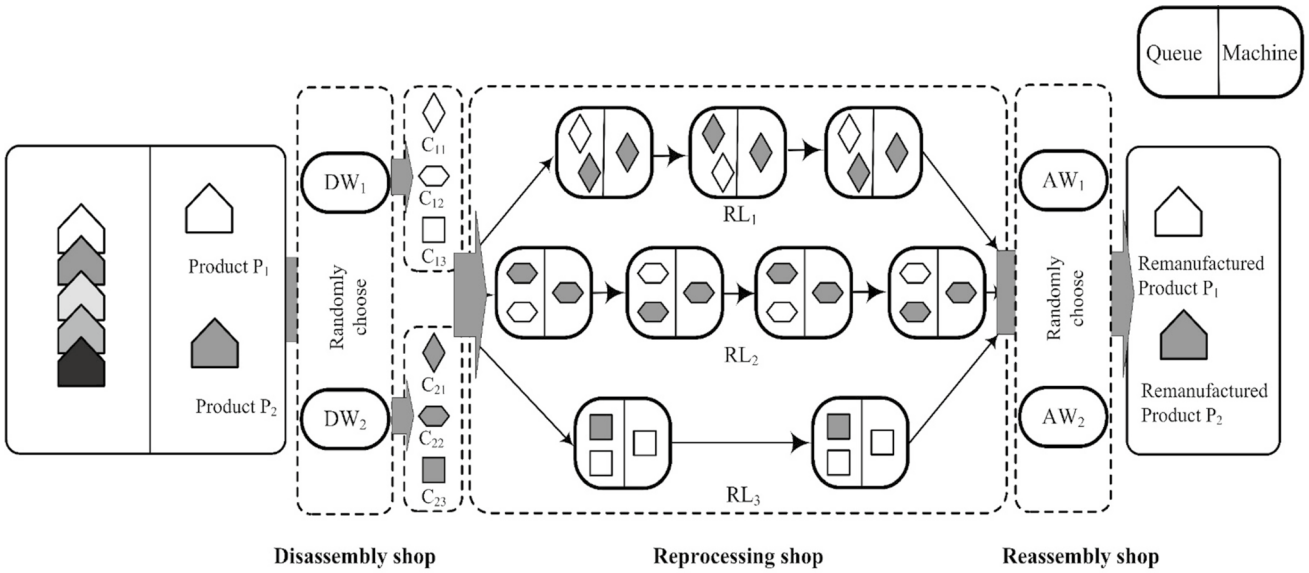

Fig. 1 Material flows in the studied type of remanufacturing systems: An example

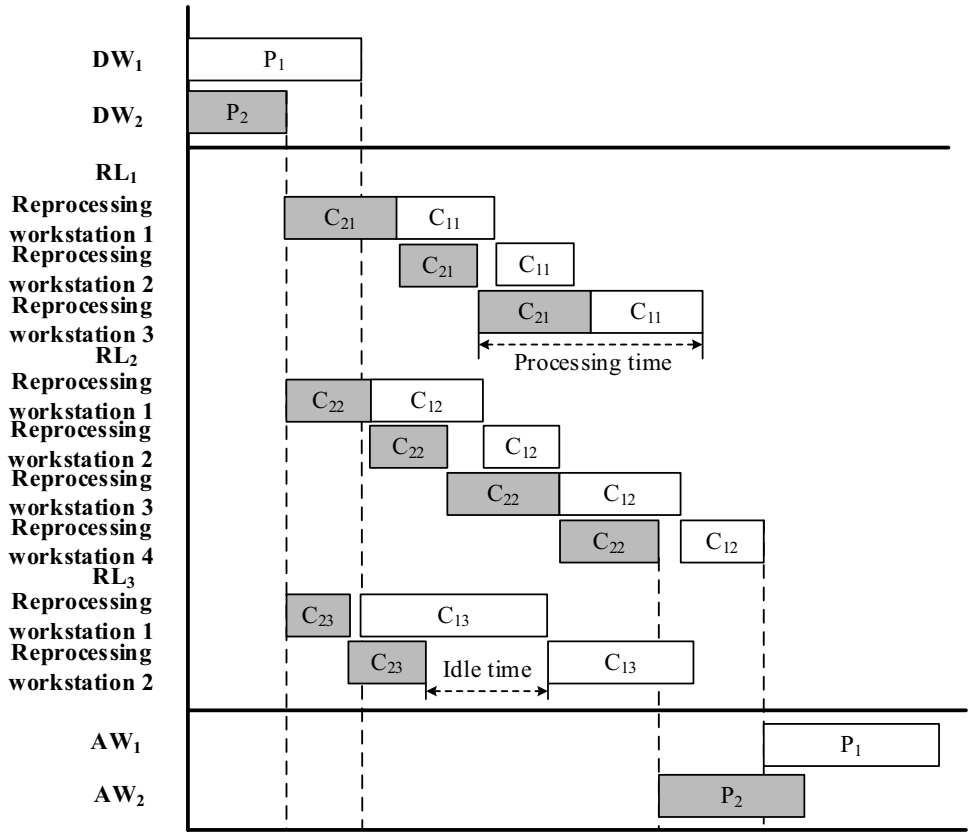

Fig. 2 An example remanufacturing system schedule

(2) j: index of components/reprocessing lines, $j \in J = \{1, 2, \ldots, n\}$.
(3) k: index of reprocessing workstations in RL_j, $k \in M_j = \{1, 2, \ldots, |M_j|\}$.
(4) g: index of DWs, $g \in M_D = \{1, 2, \ldots, p\}$.
(5) r: index of AWs, $r \in M_A = \{1, 2, \ldots, q\}$.
(6) t_i^D: processing time of disassembling EOL product i.
(7) t_{ijk}^R: processing time of working component j of EOL product i at reprocessing workstation k.
(8) t_i^A: processing time of reassembling EOL product i.
(9) B_i^D: beginning time of disassembling EOL product i.
(10) F_i^D: finishing time of disassembling EOL product i.
(11) B_{ijk}^R: beginning time of reprocessing component j of EOL product i at reprocessing workstation k.
(12) F_{ijk}^R: finishing time of reprocessing component j of EOL product i at reprocessing workstation k.
(13) F_i^R: finishing time of reprocessing all components of EOL product i.
(14) B_i^A: beginning time of reassembling EOL product i.
(15) F_i^A: finishing time of reassembling EOL product i.
(16) L: a very large positive number.
(17) xx_{ig}: a binary variable. If EOL product i is disassembled on DW_g, $xx_{ig} = 1$; otherwise $xx_{ig} = 0$.
(18) $x_{ii'}$: a binary variable. If EOL product i is disassembled before EOL product i', $x_{ii'} = 1$; otherwise $x_{ii'} = 0$.
(19) $y_{ii'}$: a binary variable. If components of EOL product i is reprocessed before those of EOL product i', $y_{ii'} = 1$; otherwise $y_{ii'} = 0$.
(20) zz_{ir}: a binary variable. If EOL product i is reassembled on AW_r, $zz_{ir} = 1$; otherwise $zz_{ir} = 0$.
(21) $z_{ii'}$: a binary variable. If EOL product i is reassembled before EOL product i', $z_{ii'} = 1$; otherwise $z_{ii'} = 0$.

Scheduling objectives

Based on the above descriptions, major energy consumption (MEC) and makespan $C_{\max}$ in the studied type of remanufacturing systems are taken as its optimization objectives, which are described as follows:

$$f_1 = \text{MEC} \tag{1}$$

$$f_2 = C_{\max} \tag{2}$$

where $C_{\max}$ is calculated as:

$$C_{\max} = \max_{i \in I} \left\{ F_i^{\text{A}} \right\}. \tag{3}$$

It is worth mentioning that the considered MEC focuses on the energy consumption for processing products/components on workstations, while energy consumption from

other auxiliary parts, such as lighting, ventilation and so on are not considered in our work. It should be noted that the energy consumption rates of workstations may vary with respect to their types and conditions (Albertelli et al., 2016). To illustrate MEC in the studied type of remanufacturing systems, the following notations on power are defined.

(1) p_i^D: disassembly power for disassembling product i (kW).
(2) p_{ijk}^R: processing power for working component j of EOL product i at workstation k (kW).
(3) p_i^A: reassembly power for reassembling product i (kW).
(4) p_{ojk}^R: idle power of reprocessing workstation k in RL_j (kW).
(5) p_o^A: idle power of parallel reassembly workstations (kW).

The MEC consists of three parts, i.e., disassembly shop energy consumption E_D, reprocessing shop energy consumption E_R, and reassembly shop energy consumption E_A, which is formulated as:

$$\mathrm{MEC} = E_\mathrm{D} + E_\mathrm{R} + E_\mathrm{A}. \tag{4}$$

Note that once the allocation and sequence of EOL products are determined, the parallel DWs continue to work until the last product in their respective processing sequences is disassembled. Thus energy consumption caused by the idle state of DWs need not be considered. That is why there is no symbol representing idle power of parallel DWs.

Directly, E_D, E_R and E_A are calculated as:

$$E_\mathrm{D} = \sum_{i\in I} p_i^\mathrm{D} \times t_i^D \tag{5}$$

$$E_\mathrm{R} = \sum_{i\in I}\sum_{j\in J}\sum_{k\in M_j} p_{ijk}^\mathrm{R} \times t_{ijk}^\mathrm{R} + \sum_{j\in J}\sum_{k\in M_j} p_{ojk}^\mathrm{R} \times t_{ojk}^\mathrm{R} \tag{6}$$

$$E_\mathrm{A} = \sum_{i=1}^{m} p_i^\mathrm{A} \times t_i^\mathrm{A} + \sum_{r=1}^{q} p_\mathrm{o}^\mathrm{A} \times t_{\mathrm{or}}^\mathrm{A} \tag{7}$$

where t_{ojk}^R refers to the idle time of the k-th reprocessing workstation in RL_j and is expressed as $t_{ojk}^\mathrm{R} = \sum_{i\in\{2,\ldots,m\}} F_{ijk}^\mathrm{R} - F_{i-1,jk}^\mathrm{R} - t_{ijk}^\mathrm{R}$. t_{or}^A means the idle time of RW_r and is calculated as $t_{\mathrm{or}}^\mathrm{A} = \sum_{f=2}^{|PN_r|} F_f^\mathrm{A} - F_{f-1}^\mathrm{A} - t_f^\mathrm{A}$, where $|PN_r|$ refers to the element count in set PN_r that stores the products reassembled on RW_r. F_f^A is the f-th smallest of finishing time of reassembling those products. Note that an example of processing time and idle time of reprocessing workstation is marked in Fig. 2.

Model formulation

Based on the above descriptions, a multi-objective mathematical model is established for the scheduling problem in the studied type of manufacturing systems:

$$\min f_1 = \mathrm{MEC} \tag{8}$$

$$\min f_2 = C_{\max} \tag{9}$$

s.t.

$$\sum_{g\in M_D} xx_{ig} = 1, \forall i \in I \tag{10}$$

$$x_{ii'} + x_{i'i} \le 1, i, i' \in I \tag{11}$$

$$B_i^D \ge 0, \forall i \in I \tag{12}$$

$$F_i^D = B_i^D + t_i^D, \forall i \in I \tag{13}$$

$$B_{i'}^D - F_i^D + L \times \left(3 - x_{ii'} - xx_{ig} - xx_{i'g}\right) \ge 0, \forall i, i' \in I, g \in M_D \tag{14}$$

$$B_{ij1}^R \ge F_i^D, \forall i \in I, j \in J \tag{15}$$

$$y_{ii'} + y_{i'i} \le 1, i, i' \in I \tag{16}$$

$$F_{ijk}^R = B_{ijk}^R + t_{ijk}^R, \forall i \in I, j \in J, k \in M_j \tag{17}$$

$$B_{ijk}^R - F_{ij,k-1}^R \ge 0, \forall i \in I, j \in J, k \in \left\{2, \ldots, \left|M_j\right|\right\} \tag{18}$$

$$B_{i'jk}^R - F_{ijk}^R + L \times \left(1 - y_{ii'}\right) \ge 0, i, i' \in I, j \in J, k \in M_j \tag{19}$$

$$F_i^R = \max\left\{F_{ijh_j}^R\right\}, \forall i \in I, j \in J \tag{20}$$

$$B_i^A - F_i^R \ge 0, \forall i \in I \tag{21}$$

$$\sum_{r\in M_A} zz_{ir} = 1, \forall i \in I \tag{22}$$

$$z_{ii'} + z_{i'i} \le 1, i, i' \in I \tag{23}$$

$$F_i^A = B_i^A + t_i^A, \forall i \in I \tag{24}$$

$$B_{i'}^A - F_i^A + L \times \left(3 - z_{ii'} - zz_{ir} - zz_{i'r}\right) \ge 0, \forall i, i' \in I, r \in M_A \tag{25}$$

$$x_{ii'} \in \{0,1\}, y_{ii'} \in \{0,1\}, z_{ii'} \in \{0,1\}, \forall i, i' \in I, j \in J, k \in M_j \quad (26)$$

$$xx_{ig} \in \{0,1\}, zz_{ir} \in \{0,1\}, \forall i \in I, g \in M_D, r \in M_A \quad (27)$$

The optimization goal is to minimize both MEC and $C_{\max}$, as expressed in Eqs. (8) and (9). Equation (10) specifies that each EOL product can only be disassembled at one DW. Equation (11) specifies the sequence of disassembling EOL products. Equation (12) guarantees that no products can begin to be worked on DWs before time zero. Equation (13) describes the relationship between the beginning time and finishing time of disassembling each product. Equation (14) illustrates that no two products are disassembled on a same DW simultaneously, i.e., disjunctive constraints. Equation (15) specifies the beginning time of the first reprocessing operation of all components and guarantees that a series of reprocessing operations can be conducted on components until their relevant disassembly operations finish. Equation (16) specifies the sequence of reprocessing components of EOL products. Equation (17) describes the relationship between the beginning time and finishing time of reprocessing each component on reprocessing workstations. Equation (18) defines the time relation between two successive reprocessing processes. Equation (19) guarantees that no two components can be processed on a same workstation at RLs. Equation (20) stipulates the completion times of reprocessing all the components of EOL products. Equation (21) stipulates that an EOL product cannot be reassembled until all of its corresponding components finish their necessary reprocessing operations. Equations (22) and (23) stipulate the sequence and allocation of products to be worked on parallel AWs, which resembles the disassembly stage. Equation (24) describes the relationship between the beginning time and finishing time of reassembling each product on AWs. Equation (25) illustrates that no two products are reassembled on a same DW simultaneously. Finally, Eqs. (26) and (27) define the conditions of corresponding variables.

Note that these three sub-scheduling problems in disassembly, reprocessing, reassembly shops could be taken as parallel-machine, permutation flow shop and parallel-machine scheduling models. An improved MOIWO algorithm will be presented in next section to resolve this NP-hard problem.

Solution algorithm

The general invasive weed optimization algorithm

The IWO is a stochastic search algorithm developed by Mehrabian and Lucas (2006), which simulates the natural behavior of weeds in colonizing and finding appropriate place for growth and reproduction. IWO makes use of some interesting properties of weeds, such as strong invasiveness, fast reproduction, selective distribution and competitive exclusion. The general IWO works on the following four basic phases: initialization, reproduction, spatial dispersal and competitive exclusion.

To be specific, the IWO algorithm starts with initializing a weed population of size PS_0, where each weed represents a solution. And then, in the reproduction phase, each weed is allowed to produce a certain number of seeds and the seed number depends on its relative fitness in the population. Afterwards, in the spatial dispersal phase, the new produced seeds are scattered over the search space near to their respective parent weeds by a normal distribution with mean zero and varying standard deviation adaptive to iteration index. Obviously, the number of weeds in the colony will reach its maximum limit $PS_{\max}$ by fast reproduction, and the population will enter the competitive exclusion phase. In this phase, an elimination mechanism is activated where the weeds with lower fitnesses are abandoned by the population to reach the limit $PS_{\max}$ and the survived seeds will go for the next generation. The above four phases repeat until the predefined termination condition is achieved.

Flow chart of MOIWO

Like many other population-based meta-heuristics, the general IWO algorithm is designed for the single-objective and continuous optimization problems. To solve multi-objective and discrete optimization problems, some improvements are needed. Based on the characteristics of the studied problem, an improved MOIWO algorithm is proposed and its procedure is shown in Fig. 3. MOIWO is composed of the following five core phases: weed initialization, weed reproduction, seed spatial dispersal, plant competitive and external Pareto archive set.

Weed initialization

(1) Solution Encoding and Decoding: Based on the model established in Sect. 2, a solution in MOIWO is encoded in two vectors through the segment encoding method: the workstation assignment vector X_w and the operation scheduling vector X_p. Such encoding approach has low decoding complexity and is easy to understand, which gains practical applications (Cai et al., 2020; Tang et al., 2019). As shown in Fig. 4, a solution is denoted as $X_i = \{X_w, X_p\} = \{d_1, d_2, \ldots, d_m, a_1, a_2, \ldots, a_m, Pd_1, Pd_2, \ldots, Pd_m, Pa_1, Pa_2, \ldots, Pa_m\}$, where $X_w = \{d_1, d_2, \ldots, d_m, a_1, a_2, \ldots, a_m\}$ and $X_p = \{Pd_1, Pd_2, \ldots, Pd_m, Pa_1, Pa_2, \ldots, Pa_m\}$. In vector X_w, d_i and a_i ($i = 1, 2, \ldots, m$) represent the workstation index used by product P_i in the disassembly shop and reassembly shop, respectively. Recall that m refers to the count of EOL products to be scheduled.

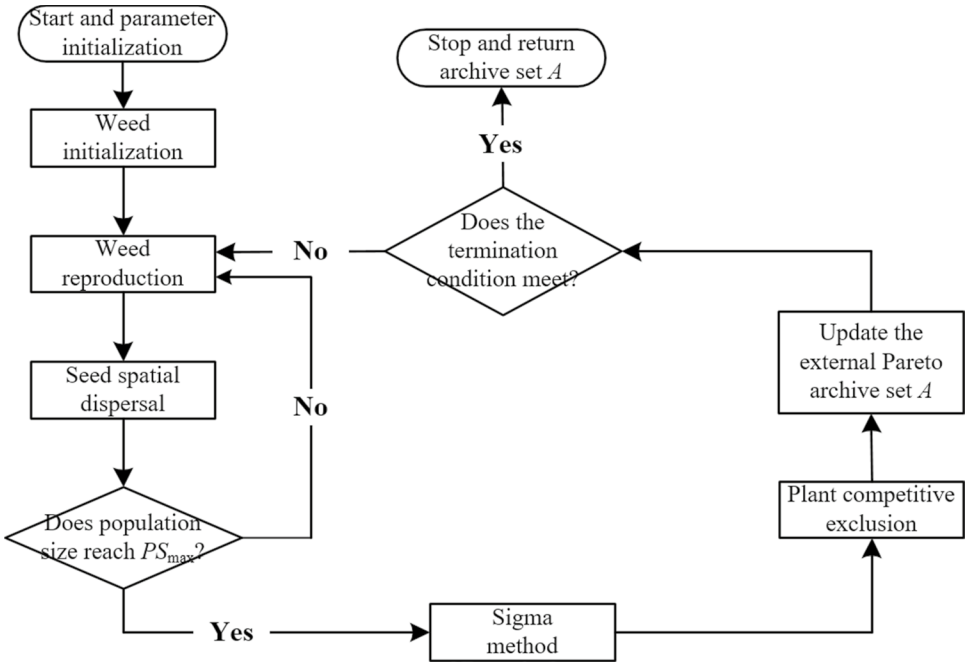

Fig. 3 Flow chart of MOIWO algorithm

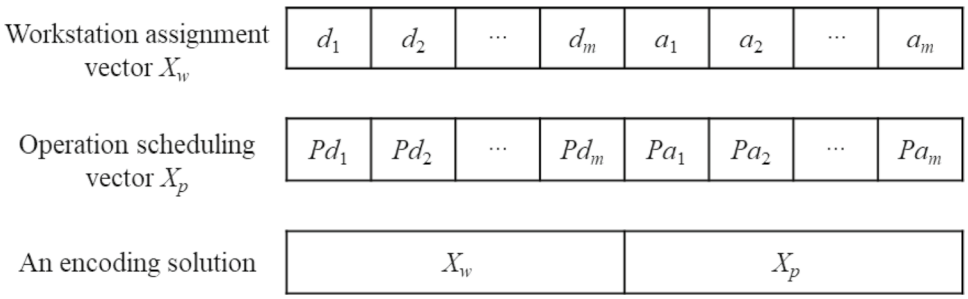

Fig. 4 Encoding scheme of a solution/a weed

In vector X_p, Pd_i and Pa_i represent the processing priority of product P_i in the disassembly shop and the reassembly shop, respectively, and a smaller number indicates a higher processing priority. d_i and a_i in vector X_w are randomly taken from $\{1, 2, \ldots, p\}$ and $\{1, 2, \ldots, q\}$, respectively. Recall that p and q refer to the count of DWs and AWs to process EOL products, respectively. The integers from 1 to product count m are rearranged and then assigned to Pd_i and Pa_i, where $Pd_i \neq Pd_j$, and $Pa_i \neq Pa_j$.

For example, a scheduling scheme can be denoted as $X_i = \{2\ 1\ 3\ 2\ 1\ 2\ 2\ 1|3\ 2\ 4\ 1\ 4\ 2\ 1\ 3\}$. The first element in X_w is 2, denoting that DW_2 is applied to disassemble product P_1 (for $d_1 = 2$); the eighth element in X_w is 1, denoting that AW_1 is employed to reassemble product P_4 (for $a_4 = 1$). Products P_1 and P_4 share the same disassembly workstation DW_2 (for $d_1 = d_4 = 2$), and product P_4 will be processed first since it has a smaller priority value than product 1 (for $Pd_4 = 1 < Pd_1 = 3$).

The sequence and allocation of EOL products to be disassembled/reassembled on the parallel DWs/AWs can be directly determined from the encoding scheme. However, the encoding scheme does not reveal the sequence of components to be worked at RLs. To address this problem, the well-accepted first come first serve (FCFS) heuristic rule is utilized based on the characteristics of the studied problem (Kim et al., 2017). Due to the close relationship between the disassembly shop and the reprocessing shop, priority values in the disassembly stage will be reused in the reprocessing stage for convenience. As a result, a component/job with the earliest release time and the lowest processing priority value will be reprocessed first at its specified reprocessing line.

(2) Initial Population/Weeds Construction: The initial weeds size of MOIWO is denoted as PS_0 and the population size will increase with iteration, but there exists an upper limit subject to environmental capacity. In the process of generating initial population *Population*(0), two generation approaches, i.e., the random generation strategy and the balanced allocation strategy are utilized to produce well-satisfied solutions. The random generation strategy refers to that the range of the individual vector elements is known,

and the random value is taken based on the upper and lower limits of the range. This strategy is simple to implement and can produce feasible solutions in short times. The balanced allocation strategy refers to distributing EOL products evenly to the DWs and AWs, which can help improve the disassembly efficiency and also enable the components to enter reprocessing/reassembly shop earlier, thus gaining a good performance in both reducing energy consumption and improving processing efficiency. Therefore, *Population*(0) is initialized through the above two strategies evenly so as to improve the solution efficiency and quality.

Weed reproduction

(1) Pareto Domination and Sigma Method: The studied problem is with two optimization objectives, i.e., MEC and $C_{\max}$, as shown in Eqs. (4) and (3). Our goal is to minimize they two simultaneously, thus the optimization problem is a minimization multi-objective optimization problem (MOP). When it comes to a MOP, a solution X is thought to be dominated by another solution X' if solution X performs better than solution X' in each objective, which is denoted as $X \prec X'$. X^* is thought to be a Pareto-optimal solution once no other solutions can dominate it (Feng et al., 2019). Usually, a group of Pareto-optimal solutions which forms a Pareto optimal set will be obtained for a MOP (Wang et al., 2020b; Zhang et al., 2019). The curve that represents the Pareto optimal set in the objective space is said to be Pareto front (PF).

In the basic IWO, the count of seeds each weed can produce depends on its fitness and the highest and lowest fitness of the whole population (Mehrabian & Lucas, 2006). Thus, the calculation of individuals' fitness is essential for the whole algorithm. Due to the fact that there are two optimization objectives in our work, it is with difficulties in calculating their fitness. The NSGA-II adopts the local crowding distance for assessing chromosomes in the same rank (Deb et al., 2002), yet it cannot give quantitative values for all the chromosomes that are in different ranks. To address this problem, we adopt the Sigma method proposed by Enayatifar et al. (2013) to work out the fitness with a sigma value. From the literature, the Sigma method which can qualify every individual has been successfully employed in solving a number of MOPs (Li et al., 2018a). It contains the following two steps:

Step 1: Use the fast non-dominated sorting method on weeds/individuals to divide them into different ranking sets. Note that the first set consists of Pareto-optimal solutions which form PF. The second ranking set contains solutions which are only dominated by several solutions in the first set and this procedure will continue until each solution gets their corresponding rankings.

Step 2: Calculate the sigma values for all weeds by:

$$\mathrm{Sigma}_i = \sum_{l=1}^{n_{obj}} \left[f_l(i) \Big/ \sum_{j=1}^{N_{\mathrm{rank}}} f_l(j) \right] + \left(\mathrm{rank}_i - 1\right) \times 2 \tag{28}$$

where n_{obj} represents the count of objectives in a MOP, obviously, is 2 in this paper. $f_l(i)$ refers to the fitness value of the l-th objective of the i-th solution, and N_{rank} denotes the count of solutions with the same rank_i. Obviously, the rankings of solutions in PF are set to 1 and solution with a lower sigma value is preferred. Figure 5 presents the procedure of Sigma method.

(2) Seed Number Determination: As basic IWO presents, the count of seeds that a weed allowed to reproduce is determined by a prefixed maximum and minimum and increases linearly (Mehrabian & Lucas, 2006). In the MOIWO algorithm, we replace fitness values in basic IWO with sigma values due to the studied problem is mathematically a MOP. And since the lower the better principle on operating sigma values, the count of reproduced seeds $n(x_i)$ of weed x_i is calculated as follows:

$$n(x_i) = \mathrm{floor}\left(s_{\min} + \frac{\mathrm{Sigma}_i - \mathrm{Sigma}_{\max}}{\mathrm{Sigma}_{\min} - \mathrm{Sigma}_{\max}} \times \left(s_{\max} - s_{\min}\right) \right) \tag{29}$$

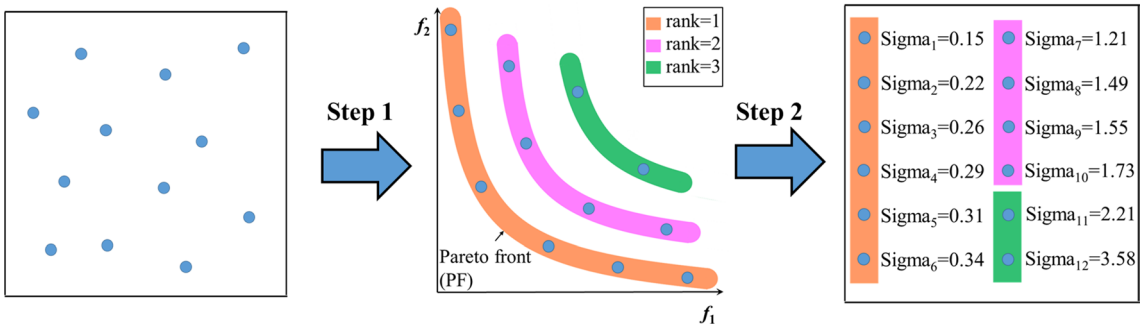

Fig. 5 Procedure of the Sigma method

where floor () is a round-down function. s_{max} is the maximum count of seeds that a weed can reproduce and s_{min} is the minimum count of seeds that a weed can reproduce, which are prefixed in MOIWO.

Seed spatial dispersal

(1) Distance Based Spatial Dispersal: Due to the fact that our studied problem is discrete and a solution is represented by the workstation assignment vector and the processing priority vector but not a real-valued vector, thus real-valued-based calculation equation on spatial dispersal in Mehrabian and Lucas (2006) cannot be used. Sang et al. (2018) consider the distance of the seeds spread from their parent weed under normal distribution with mean equal to zero. The distance is measured as the smallest execution times of mutation operations required to convert one sequence into another sequence. For example, a mutation operation on sequence $\pi=(\pi_1, \pi_2, \ldots, \pi_m)$, denoted as $\kappa(\pi, i, j)$, $i, j \in \{1, 2, \ldots, m\}$ and $i \neq j$, generates a sequence π' by swapping the elements in positions i and j from π. Let $\pi^0=(3, 2, 1, 4, 5)$ and $\pi^e=(4, 2, 5, 1, 3)$ be two sequences. Figure 6 shows the procedure of how we use $\kappa(\pi, i, j)$ to convert π^0 to π^e. It can be seen that the mutation operation $\kappa(\pi, i, j)$ should be performed at least three times whatever execution order is employed. Thus, the distance from π^0 to π^e is equal to three. Let $\pi^a=(3, 2, 2, 1, 1)$ and $\pi^b=(1, 2, 3, 2, 1)$ be another two sequences. It can be seen that the mutation operation $\kappa(\pi, i, j)$ should be performed at least two times whatever execution order is employed, which means that the distance from π^a to π^b is equal to two. And this kind of distance measure approach will be adopted in this paper.

Then, the formula of standard deviation (SD) for the normal distribution is needed to produce the specific distance and an alternative one proposed in Sang et al. (2018) is adopted, which is designed as follows:

$$\sigma_{iter} = \tan\left(\frac{\pi}{4} \times \frac{iter_{max} - iter}{iter_{max}}\right) \times (\sigma_0 - \sigma_f) + \sigma_f \qquad (30)$$

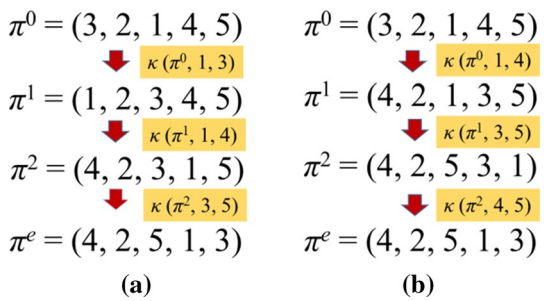

Fig. 6 The distance based seed spatial dispersal: example. **a** Distance $(\pi^0, \pi^e)=3$. **b** distance $(\pi^0, \pi^e)=3$

where iter is the index of current iteration and σ_{iter} is its SD value. σ_0 and σ_f are the initial SD value and final SD value, respectively. tan () is a usual tangent function, the factor $\pi/4$ guarantees that σ_{iter} is equal to σ_0 at the first iteration.

Consider a weed with $X=\{2\ 1\ 3\ 2\ 1\ 2\ 2\ 1|3\ 2\ 4\ 1\ 4\ 2\ 1\ 3\}$. Regarding the encoding scheme, weed X has two parts that are the same as π^a in structure (i.e., [2 1 3 2] and [1 2 2 1]) and two parts that are the same as π^0 in structure (i.e., [3 2 4 1] and [4 2 1 3]). Suppose that $\sigma_0=5$, $\sigma_f=0.1$, $iter_{max}=50$, and current iteration is iter = 20. The corresponding SD is calculated $\sigma_{iter}=\tan(0.875\times(50-20)/50)\times(5-0.1)+0.1=0.145$. Therefore, the distance from π^0 obeys the normal function $N(0, 0.145^2)$. Suppose the generated random number is 0.6, then the distance is determined by ceil (0.6) = 1, where ceil () is the usual round-up function that rounds the element to its nearest integers toward positive infinity. By conducting a mutation operation $\kappa(\pi, 1, 3)$ on X once, we have $X'=\{3\ 1\ 2\ 2\ 2\ 2\ 1\ 1|4\ 2\ 3\ 1\ 1\ 2\ 4\ 3\}$and X' is a seed of weed X. The seed generation procedure repeats until all of a weed's seeds are obtained. Due to the structural differences between X_w and X_p, mutation operations should be carefully determined. Next, we will introduce the mutation operations designed and adopted in this article.

(2) Mutation Operations: Four kinds of mutation operations are specially designed to produce seeds from their parent weeds. As Fig. 7 shows, two of them are for workstation assignment vector X_w while the rest are for operation scheduling vector X_p. Various mutation operations and stochastic execution times ensure the population diversity and search depth. The general process of four mutation operations will be discussed one by one next.

For generating a new workstation assignment vector from X_w, two mutation operations, i.e., random mutation and ± 1 mutation are designed. Random mutation: randomly choose a position h in X_w, suppose the element in position h is x, then replace x with x' which is generated from a uniform distribution on [1, M], where M represents the count of parallel workstations and varies with respect to the shop type (M equals p for disassembly shop, while M equals q for reassembly shop). ± 1 mutation: randomly select a position h in X_w and suppose its corresponding element is x. Randomly produce a number in the interval 0 and 1. If the generated number is smaller than 0.5, then produce a new element x' by $x'=x-1$; else $x'=x+1$ is activated. Note that this mutation operation may produce infeasible solutions and those solutions need to be adjusted. The adjustment process is similar to Tian et al. (2016) and is described as: assign M to x' if $x+1>M$ and assign 1 to x' if $x-1<0$.

Considering the uniqueness of elements in operation scheduling vector X_p, two mutation operations, i.e., swap mutation and insert mutation are introduced to generate a new operation scheduling vector. Swap mutation: randomly

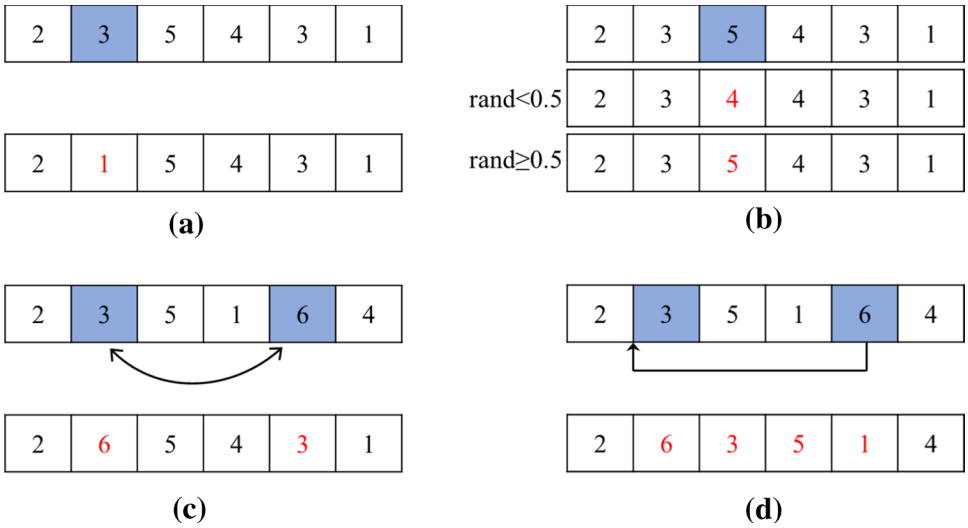

Fig. 7 Four mutation operations. **a** Random mutation. **b** ± 1 mutation. **c** Swap mutation. **d** Insert mutation

decide two different positions in X_p and exchange their corresponding elements. Insert mutation: randomly decide two different positions h and v in X_p, where $h < v$; and then insert the v-th element before the h-th element.

It should be referred that we define the unit distance as executing, one of the dedicated two mutation operations on X_w and one of the other dedicated two mutation operations on X_p. Obviously, four combinations can be generated and they will be called randomly but evenly in runs of MOIWO.

Plant competitive exclusion

After going through some iterations, the count of weeds in the colony will exceeds its maximum limit $PS_{\max}$ by fast reproduction. Therefore, an elimination mechanism for discarding weeds with poor fitness, called plant competitive exclusion is activated. It works as follows: after all weeds reproduce their seeds, we put weeds and their seeds together and apply a screening mechanism to remove identical solutions. Then the Sigma method is used again to rank them with respect to their sigma values. Finally, the top $PS_{\max}$ solutions (recall that a smaller sigma value indicates a better solution, see Sect. 3.4. Weed Reproduction) will survive and are selected as potential weeds for next iteration. For convenience, the colony size is kept unchanged in the runs of MOIWO, i.e., $PS_{\max} = PS_0$.

Pareto archive set

In MOIWO, an external archive set A is used to store all Pareto-optimal solutions that make up PF in the objective space. After the execution of plant competitive exclusion, PS_{max} solutions are produced and updated, then they will be compared with the solutions in A through Pareto dominance, and finally, A is updated iteratively by removing its solutions that are dominated by iteratively produced solutions and adding those non-dominated produced ones (Tian et al., 2016). When the index of predetermined maximum iterations reaches (i.e., the termination condition meets), the algorithm will be out of action and export final Pareto-optimal solutions.

Experimental results and discussion

In this section, we evaluate the performance of MOIWO by conducting a series of numerical experiments. MOIWO is programmed in MATLAB 2018a software and executed on an Intel(R) Core(TM) i7-8700 (3.20 GHz/8.00 GB RAM) PC with a Windows 10 operating system. Its parameters include $PS_0 = PS_{\max} = 60$, $G_{\max} = 500$, $s_{\max} = 4$, $s_{\min} = 1$, $\sigma_0 = 3$, and $\sigma_f = 0.5$, which are determined from a series of preliminary tests.

Instances and chosen algorithms

To test the performance of the proposed MOIWO, a set of experimental instances is designed based on characteristics of the studied scheduling problem. Several relative parameters such as count of EOL products, count of components, count of DWs, count of AWs, count of workstations in RLs, processing times, processing powers and idle powers should be determined. Experimental instances are designed on the basis of Kim et al., (2015, 2017), as described as below.

The count of EOL products has five levels, $m \in \{10, 15, 20, 40, 60\}$, the count of components has three levels, $n \in \{3, 5, 7\}$, and the count of DWs/AWs is with two levels, $p/q \in \{3/2, 4/3\}$. Specially, when m belongs to $\{10, 15, 20\}$, each EOL product contains two quantity levels of component $\{3, 5\}$. However, when m belongs to $\{40, 60\}$, each

EOL product will contain another two quantity levels component {5, 7}. Therefore, we produce 20 test instances by the different combinations of m, n and p/q, as is presented in Table 1.

In the studied problem, there are serial workstations in each RL to process specific components. For each test instance, the count of workstations at RLs are produced from $DU(2, 4)$ if m belongs to {10, 15, 20} and produced from $DU(4, 6)$ if m belongs to {40, 60}. $DU(a, b)$ represents the discrete uniform distribution with range $[a, b]$. Note that reprocessing times, disassembly times and reassembly times may be distinct even for products of the same type according to the actual conditions of EOL products and their constituent components, which are produced from $DU(100, 300)$, $DU(120, 180)$, and $DU(150, 220)$ (unit: s), respectively. In addition, processing powers of DWs, processing powers and idle powers of reprocessing workstations in RLs and AWs are produced by $DU(15, 25)$, $DU(5, 20)$, $DU(1, 5)$, $DU(20, 30)$, and $DU(5, 10)$ (unit: kW), respectively.

The results obtained from MOIWO are compared with two well-accepted optimization algorithms NSGA-II (Deb et al., 2002) and MOEA/D (Zhang & Li, 2007) which are based on the Pareto rule and the decomposition approach, respectively. To guarantee a fair comparison, parameters of these different algorithms must be set consistent. For comparison algorithm NSGA-II, tournament selection is determined as the selection operator and crossover and mutation operators are selected from Chen et al. (2020). Differential mutation and polynomial mutation are incorporated into MOEA/D.

Performance metrics

Several performance metrics that can be employed to assess the results of multi-objective optimization algorithms. In the experiment, three following evaluation metrics are selected finally. Note that PF_{obtain} and PF_{true} refer to the PF obtained by a specific algorithm and the PF gotten from PFs among all runs of different algorithms after Pareto dominance, respectively.

Table 1 Comparison of NSGA-II, MOEA/D and MOIWO

CP[a]	CC[b]	CD/CA[c]	GD			Δ			IGD		
			NSGA-II	MOEA/D	MOIWO	NSGA-II	MOEA/D	MOIWO	NSGA-II	MOEA/D	MOIWO
10	3	3/2	2.44E−02	6.20E−02	**1.54E−02**	6.74E−01	8.57E−01	**6.52E−01**	1.65E−02	2.90E−02	**1.14E−02**
		4/3	3.23E−02	6.27E−02	**1.68E−02**	6.91E−01	7.87E−01	**6.61E−01**	2.22E−02	3.17E−02	**1.29E−02**
	5	3/2	2.16E−02	4.36E−02	**1.41E−02**	7.25E−01	8.18E−01	**7.13E−01**	1.65E−02	2.65E−02	**1.23E−02**
		4/3	3.73E−02	6.69E−02	**1.98E−02**	**6.76E−01**	7.45E−01	7.13E−01	2.58E−02	3.70E−02	**1.81E−02**
15	3	3/2	4.97E−02	8.82E−02	**1.40E−02**	7.70E−01	**7.67E−01**	7.85E−01	2.96E−02	8.82E−02	**1.40E−02**
		4/3	3.53E−02	8.47E−02	**1.19E−02**	8.29E−01	8.34E−01	**6.82E−01**	2.42E−02	8.47E−02	**1.39E−02**
	5	3/2	3.47E−02	6.16E−02	**1.25E−02**	7.35E−01	8.17E−01	**7.14E−01**	2.20E−02	2.79E−02	**1.12E−02**
		4/3	5.37E−02	9.70E−02	**1.34E−02**	8.74E−01	8.09E−01	**6.87E−01**	3.13E−02	3.87E−02	**1.36E−02**
20	3	3/2	4.01E−02	7.13E−02	**8.70E−03**	8.02E−01	**7.49E−01**	8.02E−01	2.19E−02	2.59E−02	**9.80E−03**
		4/3	5.73E−02	9.70E−02	**7.70E−03**	9.20E−01	**8.42E−01**	9.47E−01	1.96E−02	2.27E−02	**7.70E−03**
	5	3/2	3.97E−02	6.08E−02	**9.00E−03**	8.70E−01	8.28E−01	**8.10E−01**	2.48E−02	2.51E−02	**9.30E−03**
		4/3	3.67E−02	8.66E−02	**9.40E−03**	8.20E−01	7.65E−01	**7.22E−01**	2.26E−02	2.77E−02	**9.90E−03**
40	5	3/2	3.42E−02	7.11E−02	**6.30E−03**	9.33E−01	8.06E−01	**7.50E−01**	2.22E−02	2.19E−02	**6.80E−03**
		4/3	4.19E−02	6.41E−02	**5.40E−03**	9.14E−01	**6.79E−01**	8.78E−01	2.14E−02	2.22E−02	**7.50E−03**
	7	3/2	3.73E−02	6.52E−02	**9.70E−03**	9.93E−01	8.01E−01	**7.92E−01**	2.31E−02	2.27E−02	**1.11E−02**
		4/3	4.60E−02	6.83E−02	**8.90E−03**	8.93E−01	**7.48E−01**	7.93E−01	2.24E−02	2.27E−02	**8.70E−03**
60	5	3/2	3.17E−02	7.79E−02	**7.90E−03**	9.56E−01	**8.14E−01**	8.25E−01	2.05E−02	2.27E−02	**8.50E−03**
		4/3	2.95E−02	6.11E−02	**7.50E−03**	9.75E−01	**7.81E−01**	8.88E−01	1.65E−02	1.68E−02	**6.00E−03**
	7	3/2	2.87E−02	5.60E−02	**1.02E−02**	9.97E−01	**8.18E−01**	9.00E−01	3.01E−02	3.00E−02	**2.01E−02**
		4/3	4.53E−02	7.16E−02	**1.29E−02**	9.61E−01	8.34E−01	**8.17E−01**	2.36E−02	2.47E−02	**1.02E−02**
Avg			3.79E−02	7.09E−02	**1.11E−02**	8.50E−01	7.95E−01	**7.77E−01**	2.28E−02	3.24E−02	**1.12E−02**

[a]Count of EOL products

[b]Count of components (parallel RLs)

[c]Count of DWs/AWs

(1) Generational Distance (GD) (Li et al., 2018b): this metric is used to measure how close PF_{obtain} is to PF_{true}, formulated as:

$$GD = \frac{1}{N}\sqrt{\sum_{i=1}^{N} d_i^2} \tag{31}$$

where N refers to the size of PF_{obtain}, d_i refers to the Euclidean distances between the i-th solution in PF_{obtain} and its nearest point in PF_{true}. Usually, a smaller GD value indicates an algorithm is with better convergence.

(2) Spread (Δ) (Deb et al., 2002): this metric is to illustrate the diversity of solutions in PF_{obtain}, formulated as:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} \left| d_i - \overline{d} \right|}{d_f + d_l + (N-1)\overline{d}} \tag{32}$$

where $\overline{d}$ refers to the average of all distances d_i, d_f and d_l represent the Euclidean distances between extreme solutions in PF_{obtain} and boundary solutions in PF_{true}. The smaller Δ is, the better the PF_{obtain} is in terms of distribution and diversity. To eliminate the influence of dimensions, a normalization process is utilized.

(3) Inverted Generation Distance (IGD) (Li et al., 2018b): this a comprehensive metric to reflect both convergence and diversity, formulated as:

$$IGD = \frac{1}{N^*}\sqrt{\sum_{i=1}^{N^*} d_i^{*2}} \tag{33}$$

where N^* represents the count of solutions in PF_{true}. As a variation of the GD indicator, d_i^* in IGD refers to Euclidean distances between the i-th solution in PF_{true} and its nearest point in PF_{obtain}. Similar to the GD indicator, a smaller IGD is preferred.

Experimental results

The results obtained from NSGA-II, MOEA/D, and MOIWO are presented and analyzed below. As mentioned above, ten independent runs of each instance on each algorithm are executed to get the average of metrics GD, Δ, and IGD. Obtained results are presented in Table 1 and the optimal ones will be shown in bold. The PF comparisons of NSGA-II, MOEAD and MOIWO in the twelfth instance are presented in Fig. 8. The horizontal axis refers to the C_{max}, while the vertical axis refers to the MEC. The magenta "◇" points indicate the PF of NSGA-II, the blue "○" points signify the PF of MOEAD, the red "☆" points represent the PF of MOIWO.

These experimental results tell that the GD values and IGD values of MOIWO are smaller than those of NSGA-II and MOEA/D, which indicates good solving ability of MOIWO. As the problem scale increases, the GD values

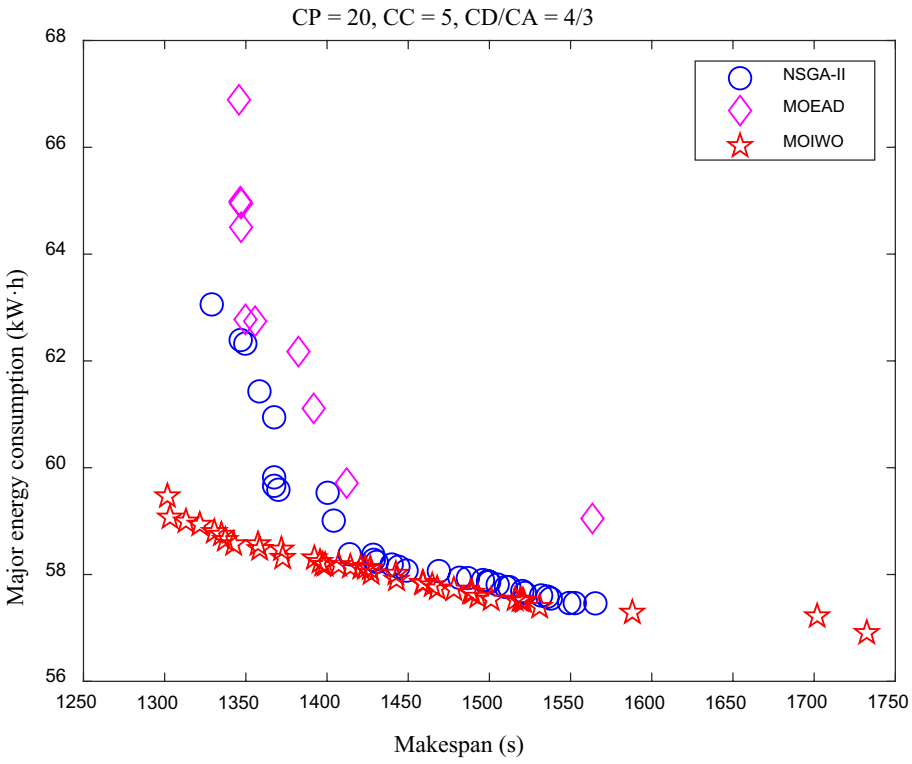

Fig. 8 PF comparison of MOEA/D, MOIWO, and NSGA-II

and IGD values show a decreasing trend, which means our proposed algorithm is more suitable for addressing large-scale problem. Regarding the Δ values, MOIWO wins in the majority of these experiments (11 out of 20). However, MOEA/D be the winner instead of MOIWO in the rest experiments (except the fourth experiment). This may be owing to that MOEA/D naturally does well in producing an even distribution of solutions along the PF (Zhang & Li, 2007). In summary, the effectiveness of MOIWO algorithm for addressing multi-objective scheduling problem in studied remanufacturing systems is verified based on the obtained results.

Case study

The proposed MOIWO algorithm is also applied to address a case from practical EOL products, i.e., transmission devices (Tian et al., 2012). The structure of transmission devices is illustrated in Fig. 9 and its detailed list of components in shown in Table 2. Main failures of transmission devices mainly occur on parts such as box, bearings, shafts, and gears (Wang et al., 2020a). It is of great benefits to remanufacture those parts and reuse them, and their remanufacturing processes are represented in Fig. 10.

As can be seen from Fig. 10, the box remanufacturing process is as: clean → pol4ish → cold welding → brush plating → polish; the gear remanufacturing process is as: clean → polish → laser cladding → shot peening → honing; the shaft remanufacturing process is as: clean → polish → brush plating → grinding; the bearing remanufacturing process is as: clean → repair welding → raceway grinding → rollers replacement (Zaretsky & Branzai, 2005).

Table 2 Detailed list of the components in the transmission device

Code	Name	Figure Code	Material	Quantity
1	Left cover		HT150	1
2	Screw I	GB70	35	4
3	Washer I		08F	1
4	Box		HT200	1
5	Bearing I	GB/T276	GCr15	1
6	Step shaft		45	1
7	Gear	GB/T10095	40Cr	1
8	Flat key	GB/T1096	45	1
9	Axle bush		Q235	1
10	Bearing II	GB/T276	GCr15	1
11	Washer II		08F	1
12	Screw II	GB70	35	4
13	Right cover		HT150	1

In this case study, a set of 5 EOL transmission devices are waiting to be remanufactured in the remanufacturing system where three DWs, two AWs, four RLs are set. Their detailed parameters are shown in Tables 5, 6, 7, 8 and 9 of Appendix. The proposed MOIWO algorithm and its comparison algorithms NSGA-II and MOEA/D are utilized again to address this case, whereas each algorithm are executed ten times independently. The PFs of these three algorithms are illustrated in Fig. 11. Besides, the mean and standard of each metrics in the ten execution times are analyzed in Table 3.

From Fig. 11, each of NSGA-II and MOIWO finally receives six Pareto solutions and PFs gotten by NAGA-II and MOIWO are similar. Though the PF obtained by MOEA/D is similar to PFs by NSGA-II and MOIWO, MOEA/D only obtains five Pareto solutions and two of them lie in the true

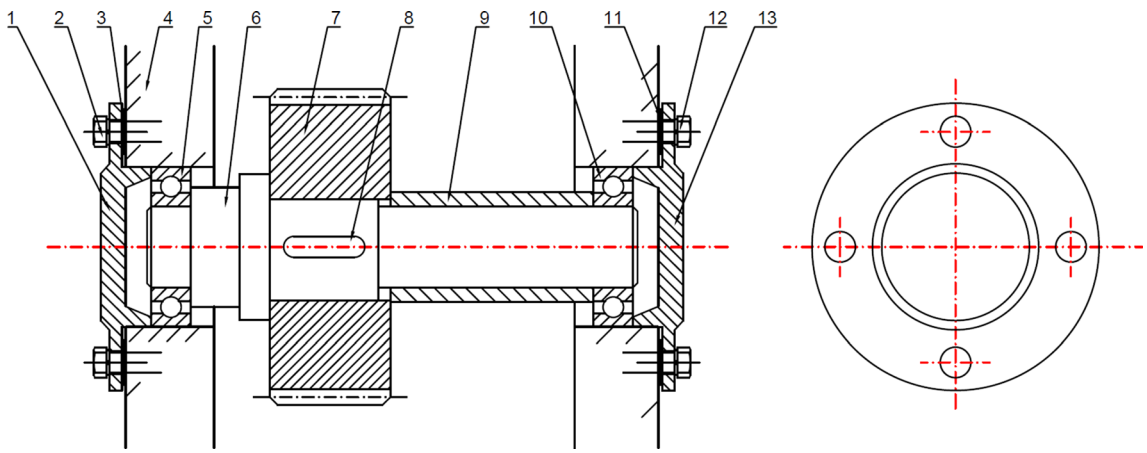

Fig. 9 Structure of the transmission device

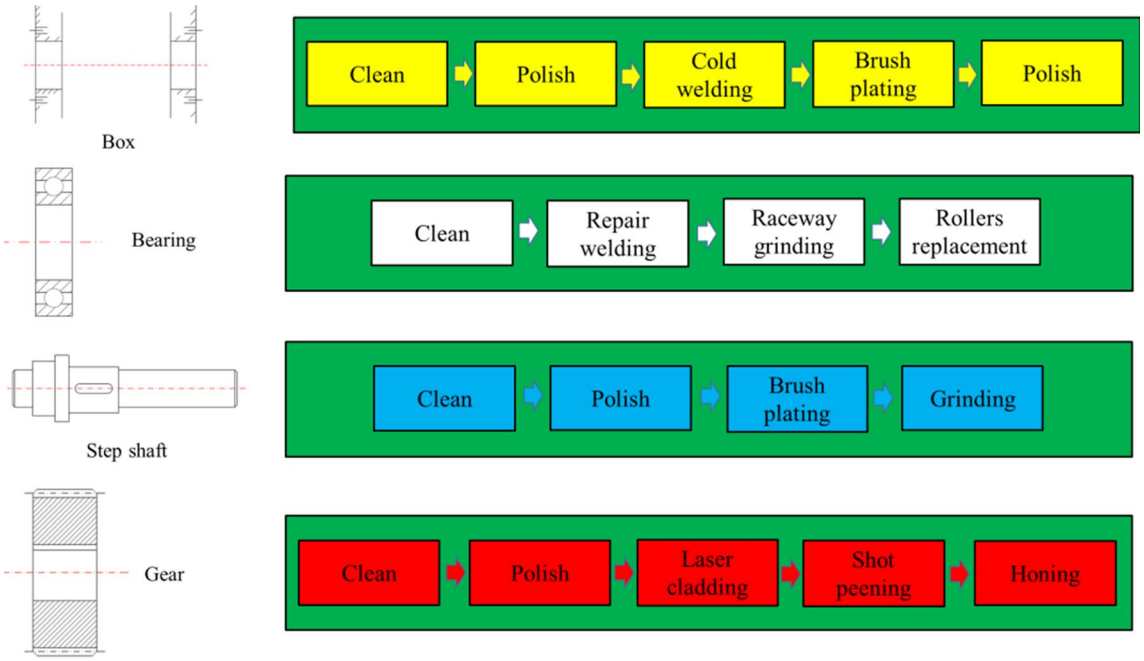

Fig. 10 Major components remanufacturing process/layout

Table 3 The comparison indexes values of each algorithm for case study

Indexes	NSGA-II	MOEA/D	MOIWO
GD	2.01E−02 (0.0092)	1.13E−01 (0.0648)	**1.98E−04** (**2.88E−05**)
Δ	5.36E−01 (0.1827)	3.94E−01 (0.1850)	**3.85E−01** (**0.1137**)
IGD	1.22E−02 (**0.0032**)	3.18E−02 (0.0069)	**5.50E−03** (0.0064)

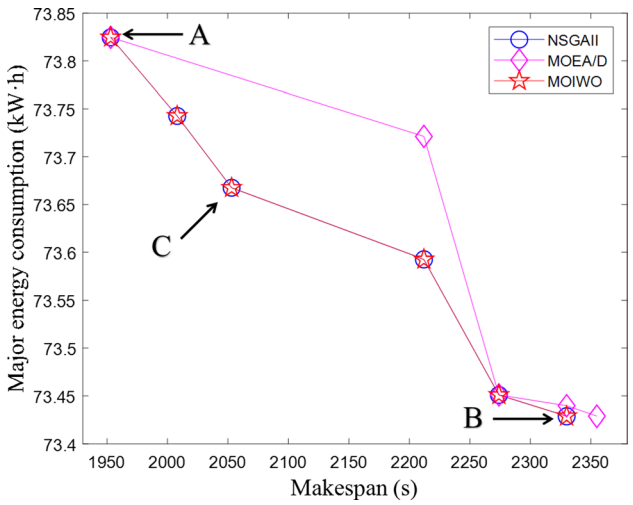

Fig. 11 PF comparison of MOEA/D, MOIWO, and NSGA-II in the case study

PF. It can be gotten from Fig. 11 and Table 3 that the proposed MOIWO algorithm has the strongest domination ability. In Table 3, numbers in brackets indicate the standard of each algorithm in ten runs and best results will be shown in bold. It can be seen that MOIWO is feasible and effective for addressing the remanufacturing system scheduling problem based on the results of comparison with NSGA-II and MOEA/D. In Fig. 11, points A and B are the solution with optimal makespan and optimal MEC in the PFs, respectively. Point C is an example of solutions with comprehensive consideration of objectives. Points A, B, and C are with (1953, 73.8247), (2330, 73.4287), and (2053, 73.6675), respectively. Besides, the solution of point A is [1 2 3 2 1 2 1 1 2 2 | 2 4 5 3 1 1 3 2 5 4], the solution of point B is [1 2 3 2 2 1 1 1 2 1 | 1 3 2 5 4 2 1 5 3 4], and the solution of point C is [1 2 3 2 1 1 2 2 1 2 | 3 2 5 4 1 3 2 5 4 1].

For solution B, it has the highest makespan and the lowest MEC. This is due to the fact that an appropriate scheduling in the disassembly shop allows workstations at parallel reprocessing lines to avoid long-term idle when processing components. Besides, to gain an ideal energy consumption, components tend to be reassembled on a small number of AWs to avoid idle times, which obviously reduces the production efficiency and leads to a higher makespan. For solution A, it has the lowest makespan and the highest MEC. This is mainly due to the fact that if a lower makespan is

preferable, components will be reassembled immediately once there exists a ready machine and the difference in processing times required to reassemble components may cause some workstations to be on idle for a long time, which will lead to an additional energy consumption.

To further illustrate the relationship between idle powers of workstations and MEC, a sensitivity analysis is also conducted. We take the case study as an experimental object and its data are set as the default group. Next, we just increase the idle powers of workstations by 10%, 20%, 50% and 75% in turn based on the default group, and then MOIWO algorithm is utilized again to solve the four newly generated cases. For each case, MOIWO will be executed ten times independently and finally Pareto solutions are obtained. The maximum MEC and minimum MEC are stored, as reported in Table 4. To be clearly show the experimental results, we calculate the proportion of idle power consumption in MEC and the outcome is visualized in Fig. 12. It can be seen from Table 4 and Fig. 12 that the idle powers of workstations are relevant to MEC and they are approximately positively correlated. Though idle energy consumption takes a small part of MEC, the maximum MEC and minimum MEC will both increase when the idle powers of workstations become bigger.

Conclusions

In this article, we have studied an energy- and time-oriented scheduling problem of remanufacturing systems with parallel disassembly workstations, parallel flow-shop-type reprocessing lines and parallel reassembly workstations. First, a multi-objective mathematical model with consideration of minimizing both energy consumption and makespan is formulated. Second, an effective MOIWO algorithm including the notions of the basic IWO, Pareto-optimal, Sigma method, and distance-based spatial dispersal, external Pareto archive is designed. Thirdly, the effectiveness of the MOIWO algorithm is examined with two well-accepted multi-objective algorithms: NSGA-II and MOEA/D against a numerical experiment. Finally, the established model and proposed algorithm are utilized again to address a case of EOL transmission devices, in which MOIWO performs better than NSGA-II and MOEA/D. Future work directions can concentrate on: (1) considering the scheduling problems of remanufacturing systems with more detailed optimization objectives; (2) introducing some energy control strategies into this field for energy saving (Frigerio & Matta, 2015).

Table 4 Experiments for sensitivity analysis

Idle powers (kW)	Default	Default + 10%	Default + 20%	Default + 50%	Default + 75%
Max MEC (kW·h)	73.8275	74.1518	74.4761	75.4490	76.2598
Min MEC (kW·h)	73.4316	73.7163	74.0009	74.8550	75.5668

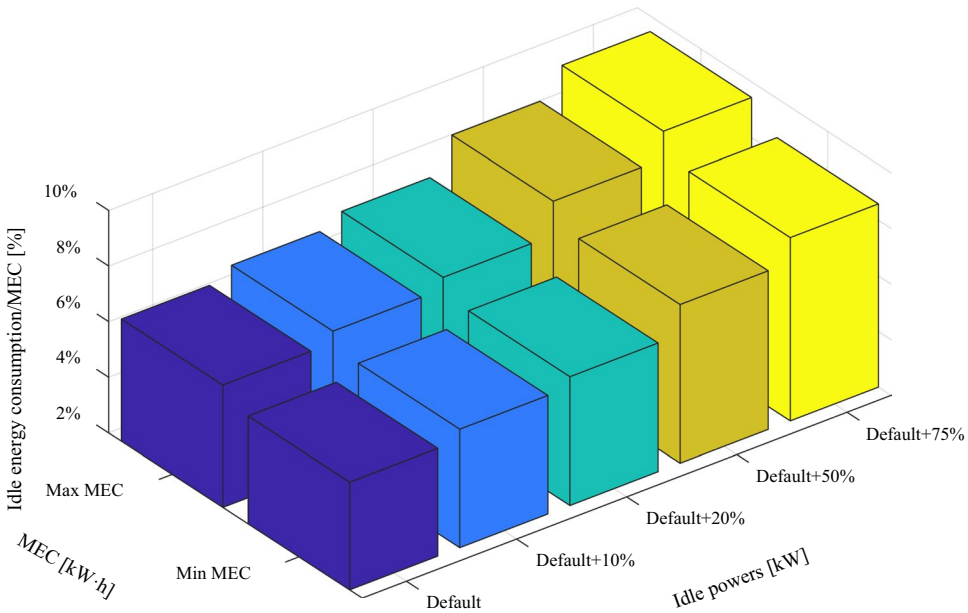

Fig. 12 Proportion results of sensitivity analysis for 5 experiments

Appendix

See Tables 5, 6, 7, 8 and 9.

Table 5 Working times of products and working/idle powers of workstations in the disassembly/reassembly shop

No	Disassembly shop		Reassembly shop		
	t_i^D (s)	p_i^D (kW)	t_i^A (s)	p_i^A (kW)	p_o^A (kW)
1	300	30	320	50	10
2	240		280		
3	420		450		
4	100		520		
5	150		360		

Table 6 Processing times of components on workstations in RL_1

No	RW_1	RW_2	RW_3	RW_4	RW_5
1	60	220	110	90	30
2	90	250	130	105	60
3	70	263	124	120	55
4	85	229	117	100	45
5	65	256	136	105	60
	22.0[a]/5.2[b]	7.7/1.3	15.9/5.2	14.8/4.3	7.7/1.3

a. reprocessing power (kW) of the workstation

b. idle power (kW) of the workstation

Table 7 Processing times of components on workstations in RL_2

No	RW_1	RW_2	RW_3	RW_4
1	50	175	30	25
2	63	186	35	25
3	75	175	42	25
4	60	153	40	25
5	58	190	36	25
	5.3/1.7	13.2/2.9	12.3/2.3	10.8/2.0

Table 8 Processing times of components on workstations in RL_3

No	RW_1	RW_2	RW_3	RW_4
1	90	125	200	142
2	86	146	205	160
3	92	125	212	155
4	100	145	220	186
5	95	150	207	173
	22.0/5.2	10.3/2.6	14.8/4.3	13.2/2.2

Table 9 Processing times of components on workstations in RL_4

No	RW_1	RW_2	RW_3	RW_4	RW_5
1	45	100	235	130	122
2	50	105	220	115	140
3	59	95	214	102	135
4	50	105	180	126	145
5	48	110	200	136	150
	5.3/1.7	8.8/2.3	9.0/1.9	10.3/3.4	3.5/1.0

Acknowledgements This work is supported in part by National Natural Science Foundation of China (Grant Nos 52075303 and 51775238), and in part by the Open Project of the State Key Laboratory of Fluid Power and Mechatronic Systems (Grant No GZKF-202012), and in part by the Fundamental Research Funds for the Central Universities (Grant No. 2019GN048).

References

Albertelli, P., Keshari, A., & Matta, A. (2016). Energy oriented multi cutting parameter optimization in face milling. *Journal of Cleaner Production, 137*, 1602–1618.

An, Y. J., Chen, X. H., Zhang, J., & Li, Y. H. (2020). A hybrid multi-objective evolutionary algorithm to integrate optimization of the production scheduling and imperfect cutting tool maintenance considering total energy consumption. *Journal of Cleaner Production, 268*, 121540.

Cai, C., Lei, D. M., & Li, M. (2020). A shuffled frog-leaping algorithm with memeplex quality for bi-objective distributed scheduling in hybrid flow shop. *International Journal of Production Research*. https://doi.org/10.1010/00207543.2020.1780333

Chen, H., Zhou, Y. Q., He, S. C., Quyang, X. X., & Guo, P. G. (2013). Invasive weed optimization algorithm for solving permutation flow-shop scheduling problem. *Journal of Computational and Theoretical Nanoscience, 10*(3), 708–713.

Chen, T. L., Cheng, C. Y., & Chou, Y. H. (2020). Multi-objective genetic algorithm for energy-efficient hybrid flow shop scheduling with lot streaming. *Annals of Operations Research, 290*(1–2), 813–836.

Chikhi, N., Abbas, M., Bekrar, A., Benmansour, R., & Hanafi, S. (2014). On the complexity of robotic flow shop with transportation constraints. In *ROADEF-15ème congrès annuel de la Société française de recherche opérationnelle et d'aide à la décision, Société française de recherche opérationnelle et d'aide à la decision*.

Daniel, V., & Guide, R. (1997). Scheduling with priority dispatching rules and drum-buffer-rope in a recoverable manufacturing system. *International Journal of Production Economics, 53*(1), 101–116.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182–197.

Enayatifar, R., Yousefi, M., Adbullah, A. H., & Darus, A. N. (2013). MOICA: A novel multi-objective approach based on imperialist competitive algorithm. *Applied Mathematics and Computation, 219*(17), 8829–8841.

Fang, K., Uhan, N., Zhao, F., & Sutherland, J. W. (2011). A new approach to scheduling in manufacturing for power consumption and carbon consumption and carbon footprint reduction. *Journal of Manufacturing Systems, 30*(4), 234–240.

Fattahi, P., Hosseini, S. M. H., Jolai, F., & Tavakkoli-Moghaddam, R. (2014). A branch and bound algorithm for hybrid flow shop scheduling problem with setup time and assembly operations. *Applied Mathematical Modelling, 38*(1), 119–134.

Feng, Y. X., Zhou, M. C., Tian, G. D., Li, Z. W., Zhang, Z. F., Zhang, Q., et al. (2019). Target disassembly sequencing and scheme evaluation for CNC machine tools using improved multiobjective ant colony algorithm and fuzzy integral. *IEEE Transactions on Systems Man Cybernetics-Systems*, *49*(12), 2438–2451.

Foumani, M., & Jenab, K. (2013). Analysis of flexible robotic cells with improved pure cycle. *International Journal of Computer Integrated Manufacturing, 26*(3), 201–215.

Foumani, M., & Smith-Miles, K. (2019). The impact of various carbon reduction policies on green flowshop scheduling. *Applied Energy, 249*, 300–315.

Frigerio, N., & Matta, A. (2015). Energy-efficient control strategies for machine tools with stochastic arrivals. *IEEE Transactions on Automation Science and Engineering, 12*(1), 50–61.

Fu, Y. P., Tian, G. D., Fathollahi-Fard, A. M., Ahmadi, A., & Zhang, C. Y. (2019). Stochastic multi-objective modelling and optimization of an energy-conscious distributed permutation flow shop scheduling problem with the total tardiness constraint. *Journal of Cleaner Production*, *226*, 515–525.

Fu, Y. P., Zhou, M. C., Guo, X. W., & Qi, L. (2021). Stochastic multi-objective integrated disassembly-reprocessing-reassembly scheduling via fruit fly optimization algorithm. *Journal of Cleaner Production, 278*, 123364.

Guide, V. D. R. (1995). A simulation-model of drum-buffer-rope for production planning and control at a Naval Aviation Depot. *SIMULATION, 35*(3), 157–168.

Guide, V. D. R. (2000). Production planning and control for remanufacturing: Industry practice and research needs. *Journal of Operations Management, 18*(4), 467–483.

Guide, V. D. R., Kraus, M. E., & Srivastava, R. (1997). Scheduling policies for remanufacturing. *International Journal of Production Economics, 48*(2), 187–204.

Guide, V. D. R., & Srivastava, R. (1997). An evaluation of order release strategies in a remanufacturing environment. *Computers & Operations Research, 24*(1), 37–47.

Gungor, A., & Gupta, S. M. (2001). Disassembly sequence plan generation using a branch-and-bound algorithm. *International Journal of Production Research, 39*(3), 481–509.

Heese, S., Cattani, K., & Ferrer, G. (2005). Competitive advantage through take-back of used products. *European Journal of Operational Research, 164*(1), 143–157.

Jahangir, H., Mohammadi, M., Pasandideh, S. H. R., & Nobari, N. Z. (2019). Comparing performance of genetic and discrete invasive weed optimization algorithms for solving the inventory routing problem with an incremental delivery. *Journal of Intelligent Manufacturing, 30*(6), 2327–2353.

Jiang, Z. G., Jiang, Y., Wang, Y., Zhang, H., Cao, H. J., & Tian, G. D. (2019). A hybrid approach of rough set and case-based reasoning to remanufacturing process planning. *Journal of Intelligent Manufacturing, 30*(1), 19–32.

Jolai, F., Foumani, M., Tavakoli-Moghadam, R., & Fattahi, P. (2012). Cyclic scheduling of a robotic flexible cell with load lock and swap. *Journal of Intelligent Manufacturing, 23*(5), 1885–1891.

Joshi, D., & Gupta, S. M. (2019). Evaluation of design alternatives of End-Of-Life products using internet of things. *International Journal of Production Economics, 208*, 281–293.

Kalayi, B., & Gupta, S. M. (2013). Ant colony optimization for sequence-dependent disassembly line balancing problem. *Journal of Manufacturing Technology Management, 24*(3), 413–427.

Kerin, M., & Pham, D. T. (2019). A review of emerging industry 4.0 technologies in remanufacturing. *Journal of Cleaner Production, 237*, 117805.

Kim, G., Yu, J. M., & Lee, D. H. (2015). Scheduling algorithms for remanufacturing systems with parallel flow-shop-type reprocessing lines. *International Journal of Production Research, 53*(6), 1819–1831.

Kim, J. M., Zhou, Y. D., & Lee, D. H. (2017). Priority scheduling to minimize the total tardiness for remanufacturing systems with flow-shop-type reprocessing lines. *International Journal of Advanced Manufacturing Technology, 91*(9–12), 3697–3708.

Li, D. S., Zhang, C. Y., Tian, G. D., Shao, X. Y., & Li, Z. W. (2018a). Multiobjective program and hybrid imperialist competitive algorithm for the mixed-model two-sided assembly lines subject to multiple constraints. *IEEE Transactions on Systems, Man, and Cybernetics: System, 48*(1), 119–129.

Li, X. Y., Lu, C., Gao, L., Xiao, S. Q., & Wen, L. (2018b). An effective multiobjective algorithm for energy-efficient scheduling in a real-life welding shop. *IEEE Transactions on Industrial Informatics, 14*(12), 5400–5409.

Liu, Y., Zhou, Z. D., Pham, D. T., Xu, W. J., Cui, J., & Yang, C. (2020). Service platform for robotic disassembly planning in remanufacturing. *Journal of Manufacturing Systems, 57*, 338–356.

Lu, C., Gao, L., Li, X. Y., Pan, Q. K., & Wang, Q. (2017). Energy-efficient permutation flow shop scheduling problem using a hybrid multi-objective backtracking search algorithm. *Journal of Cleaner Production, 144*, 228–238.

Lugaresi, G., Alba, V. V., & Matta, A. (2021). Lab-scale models of manufacturing systems for testing real-time simulation and production control technologies. *Journal of Manufacturing Systems, 58*, 93–108.

Lund, R. T. (1984). Remanufacturing. *Technology Review, 87*(2), 19–29.

Mehrabian, A. R., & Lucas, C. (2006). A novel numerical optimization algorithm inspired from weed colonization. *Ecological Informatics, 1*(4), 355–366.

Misaghi, M., & Yaghoobi, M. (2019). Improved invasive weed optimization algorithm (IWO) based on chaos theory for optimal design of PID controller. *Journal of Computational Design and Engineering, 6*(3), 284–295.

Mousavi, S. M., Alikar, N., Tavana, M., & Di-Caprio, D. (2019). An improved particle swarm optimization model for solving homogeneous discounted series-parallel redundancy allocation problems. *Journal of Intelligent Manufacturing, 30*(3), 1175–1194.

Natarajan, E., Kaviarasan, V., Lim, W. H., Tiang, S. S., Parasuraman, S., & Elango, S. (2020). Non-dominated sorting modified teaching-learning-based optimization for multi-objective machining of polytetrafluoroethylene (PTFE). *Journal of Intelligent Manufacturing, 31*(4), 911–935.

Ozceylan, E., Kalayci, C. B., Gungor, A., & Gupta, S. M. (2019). Disassembly line balancing problem: A review of the state of the art and future directions. *International Journal of Production Research, 57*(15–16), 4805–4827.

Parkinson, J., & Thompson, G. (2003). Analysis and taxonomy of remanufacturing industry practice. *Proceedings of the Institution of Mechanical Engineers Part E-Journal of Process Mechanical Engineering, 217*(E3), 243–256.

Pedrielli, G., Matta, A., Alfieri, A., & Zhang, M. Y. (2018). Design and control of manufacturing systems: A discrete event optimization methodology. *International Journal of Production Research, 56*(1–2), 543–564.

Sang, Y., Pan, Q. K., Duan, P. Y., & Li, J. Q. (2018). An effective discrete invasive weed optimization algorithm for lot-streaming flowshop scheduling problems. *Journal of Intelligent Manufacturing, 29*(6), 1337–1349.

Stanfield, P. M., King, R. E., & Hodgson, T. J. (2006). Determining sequence and ready times in a remanufacturing system. *IIE Transactions, 38*(7), 597–607.

Tang, H. T., Chen, R., Li, Y. B., Peng, Z., Guo, S. S., & Du, Y. Z. (2019). Flexible job-shop scheduling with tolerated time interval and limited starting time interval based on hybrid discrete PSO-SA: An application from a casting workshop. *Applied Soft Computing, 78*, 176–194.

Thornton, W., & Hunsucker, J. L. (2004). A new heuristic for minimal makespan in flow shops with multiple processors and no intermediate storage. *European Journal of Operational Research, 152*(1), 96–114.

Tian, G. D., Liu, Y. M., Ke, H., & Chu, J. W. (2012). Energy evaluation method and its optimization models for process planning with stochastic characteristics: A case study in disassembly decision-making. *Computers & Industrial Engineering, 63*(3), 553–563.

Tian, G. D., Ren, Y. P., Feng, Y. X., Zhou, M. C., Zhang, H. H., & Tan, J. R. (2019). Modeling and planning for dual-objective selective disassembly using and/or graph and discrete artificial bee colony. *IEEE Transactions on Industrial Informatics, 15*(4), 2456–2468.

Tian, G. D., Ren, Y. P., & Zhou, M. C. (2016). Dual-objective scheduling of rescue vehicles to distinguish forest fires via differential evolution and particle swarm optimization combined algorithm. *IEEE Transactions on Intelligent Transportation Systems, 17*(11), 3009–3021.

Tian, G. D., Zhang, H. H., Feng, Y. X., Jia, H. F., Zhang, C. Y., Jiang, Z. G., Li, Z. W., & Li, P. G. (2017). Operation patterns analysis of automotive components remanufacturing industry development in China. *Journal of Cleaner Production, 164*, 1363–1375.

Tian, G. D., Zhou, M. C., & Li, P. G. (2018). Disassembly sequence planning considering fuzzy component quality and varying operational cost. *IEEE Transactions on Automation Science and Engineering, 15*(2), 748–760.

Wang, L., Guo, Y. Y., Zhang, Z. L., Xia, X. H., & Cao, J. H. (2020a). Generalized growth decision based on cascaded failure information: Maximizing the value of retired mechanical products. *Journal of Cleaner Production, 269*, 122176.

Wang, W. Y., Mo, D. Y., Wang, Y., & Tseng, M. M. (2019). Assessing the cost structure of component reuse in a product family for remanufacturing. *Journal of Intelligent Manufacturing, 30*(2), 575–587.

Wang, W. J., Tian, G. D., Chen, M. N., Tao, F., Zhang, C. Y., AI-Ahmari, A., Li, Z. W., & Jiang, Z. G. (2020b). Dual-objective program and improved artificial bee colony for the optimization of energy-conscious milling parameters subject to multiple constraints. *Journal of Cleaner Production, 245*, 118714.

Wang, W. J., Tian, G. D., Zhang, T. Z., Jabarullah, N. H., Li, F. Y., Fathollahi-Fard, A. M., Wang, D. Q., & Li, Z. W. (2021). Scheme selection of design for disassembly (DFD) based on sustainability: A novel hybrid of interval 2-tuple linguistic intuitionistic fuzzy numbers and regret theory. *Journal of Cleaner Production, 281*, 124724.

Yang, Y. S., Yang, G., Tian, G. D., & Zhuang, Q. W. (2020). Comprehensive evaluation of disassembly performance based on the ultimate cross-efficiency and extension-gray correlation degree. *Journal of Cleaner Production, 245*, 118800.

Yu, J. M., & Lee, D. H. (2018). Scheduling algorithms for job-shop-type remanufacturing systems with component matching requirement. *Computers & Industrial Engineering, 120*, 266–279.

Yuan, G., Yang, Y. S., & Pham, D. T. (2020). Multiobjective ecological strategy optimization for two-stage disassembly line balancing with constrained-resource. *IEEE Access, 8*(88745–88758), 2020.

Zaretsky, V., & Branzai, E. V. (2005). Effect of rolling bearing refurbishment and restoration on bearing life and reliability. *Tribology Transactions, 48*(1), 32–44.

Zhang, H. H., Peng, Y., Hou, L., Tian, G. D., & Li, Z. W. (2019). A hybrid multi-objective optimization approach for energy-absorbing structures in train collisions. *Information Sciences, 481*, 491–506.

Zhang, H. H., Peng, Y., Hou, L., Wang, D. Q., Tian, G. D., & Li, Z. W. (2020). Multistage impact energy distribution for whole vehicles in high-speed train collisions: Modeling and solution methodology. *IEEE Transactions on Industrial Informatics, 16*(4), 2486–2499.

Zhang, Q. F., & Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation, 11*(6), 712–731.

Zhou, Y. Q., Chen, H., & Zhou, G. (2014). Invasive weed optimization algorithm for optimization no-idle flow shop scheduling problem. *Neurocomputing, 137*, 285–292.

Zhu, L. X., Zhang, Z. Q., Wang, Y., & Cai, N. (2020). On the end-of-life state oriented multi-objective disassembly line balancing problem. *Journal of Intelligent Manufacturing, 31*(6), 1403–1428.