COMPLEX SYSTEM MODELING AND SIMULATION
ISSN 2096-9929 05/06 pp 184–209
Volume 4, Number 2, June 2024
DOI: 10.23919/CSMS.2024.0007

A Q-Learning Based Hybrid Meta-Heuristic for Integrated Scheduling of Disassembly and Reprocessing Processes Considering Product Structures and Stochasticity

Fuquan Wang, Yaping Fu*, Kaizhou Gao, Yaoxin Wu, and Song Gao

Abstract: Remanufacturing is regarded as a sustainable manufacturing paradigm of energy conservation and environment protection. To improve the efficiency of the remanufacturing process, this work investigates an integrated scheduling problem for disassembly and reprocessing in a remanufacturing process, where product structures and uncertainty are taken into account. First, a stochastic programming model is developed to minimize the maximum completion time (makespan). Second, a Q-learning based hybrid meta-heuristic (Q-HMH) is specially devised. In each iteration, a Q-learning method is employed to adaptively choose a premium algorithm from four candidate ones, including genetic algorithm (GA), artificial bee colony (ABC), shuffled frog-leaping algorithm (SFLA), and simulated annealing (SA) methods. At last, simulation experiments are carried out by using sixteen instances with different scales, and three state-of-the-art algorithms in literature and an exact solver CPLEX are chosen for comparisons. By analyzing the results with the average relative percentage deviation (RPD) metric, we find that Q-HMH outperforms its rivals by 9.79%–26.76%. The results and comparisons verify the excellent competitiveness of Q-HMH for solving the concerned problems.

Key words: remanufacturing scheduling; disassembly; reprocessing; meta-heuristic; Q-learning

1 Introduction

As the fast growth of economy and the quick advance of technology, a great many of products, e.g., electronics and automobiles, are rapidly updated, and their lifecycle becomes shorter[1]. Consequently, a vast quantity of end-of-life (EOL) products are flooding into our environments at an unprecedented rate. In the case that these products are directly cast away, they will cause severe environment contamination and massive resource waste[2]. Remanufacturing is treated as a promising way to achieve recycling usage of EOL products[3]. It can enable products to achieve the best performance and life requirements with less environment pollution, higher resource utilization, and lower production cost[4, 5].

Remanufacturing concentrates on transforming EOL products to like-new circumstances by using disassembly, reprocessing, and reassembly operations[6]. In a remanufactured system, EOL products are dismantled into multiple components along with inspection operations at the disassembly shop. Subsequently, the dismantled components are recovered via reconditioning operations at the reprocessing shop. At last, the repaired components, if

• Fuquan Wang and Yaping Fu are with the School of Business, Qingdao University, Qingdao 266071, China. E-mail: wangfuquan1117@163.com; fuyaping0432@163.com.
• Kaizhou Gao is with the Macau Institute of Systems Engineering, Macau University of Science and Technology, Macao 999078, China. E-mail: gaokaizh@aliyun.com.
• Yaoxin Wu is with the Department of Industrial Engineering & Innovation Sciences, Eindhoven University of Technology, Eindhoven, 5600 MB, the Netherlands. E-mail: wyxacc@hotmail.com.
• Song Gao is with the College of Information Science and Engineering, Northeastern University, Shenyang 110819, China. E-mail: gaosong_2022@163.com.
∗ To whom correspondence should be addressed.
※ This article was recommended by Associate Editor Xinyu Li.
Manuscript received: 2024-02-25; revised: 2024-04-10; accepted: 2024-04-29

required, are further reassembled into remanufactured products at the reassembly shop. Production planning and scheduling are very important to improve operation efficiency of manufacturing and remanufacturing processes[7–9]. Over the last years, many researchers are dedicated to independently scheduling the disassembly[10], reprocessing[11], and reassembly shops[12]. Noteworthily, to find a universal optimization of the three shops, a few studies propose an integrated method to schedule the three shops together[13–15].

In practical remanufacturing processes, we must obey product structures when performing disassembly, reprocessing, and reassembly operations. By making full analysis on existing work regarding remanufacturing scheduling problems, we discover that product structures are fully considered in the disassembly process. Yet, rare attention is devoted to taking such inherent characteristics into account in solving integrated remanufacturing scheduling problems. Besides, although meta-heuristics are widely employed to address such difficult scheduling problems, a hybridization of meta-heuristic and reinforcement learning methods does not draw concern.

As an assembly operation of products in manufacturing processes which must follow the product structures[12], obeying EOL product structures in a remanufacturing process is very essential to find feasible and performable decisions. Thus, the product structures must be followed when we address integrated remanufacturing scheduling problems. Usually, it ought to remanufacture multiple EOL products during a planning period, and thus we have to disassemble them obeying their product structures simultaneously in a remanufacturing process. Generally, we are knotty to gain precise conditions of EOL products, such as depreciation and wear, since their diverse usage circumstances, causing that the remanufacturing process cannot be implemented as planned. Thus, we must consider such uncertainties in a remanufacturing process[16–19]. This work focuses on the disassembly and reprocessing operations under the circumstances that the remanufactured components are sold to downstream customers rather than are reassembled into new products immediately. In fact, the investigated problem widely exists in practical remanufacturing systems, e.g., a vehicle engine remanufacturing process. In practice, many engines from EOL vehicles are recycled, and then are delivered to a remanufacturing system. They are disassembled into multiple components, e.g., crankshafts, camshafts, cylinder blocks, and cylinder heads, in a disassembly shop with identical workstations. Subsequently, these components are reprocessed by a series of operations, such as cleaning, inspecting, and reconditioning, in a reprocessing shop including parallel flow-shop-type reprocessing lines. At last, the repaired components are sold separately or reassembled. A manager needs to make disassembly and reprocessing scheduling schemes in disassembly and reprocessing shops together to achieve an overall optimization. Thereby, this research addresses an integrated scheduling problem of disassembly and reprocessing processes to achieve minimal makespan, where product structures and uncertain operation time are fully considered. A Q-learning based hybrid meta-heuristic (Q-HMH) is specially developed by combining effective search algorithms to find satisfaction solutions. Through performing comparisons between this research and previous work, we make the following contributions:

(1) A stochastic programming model is formulated to achieve minimal expected maximum completion time (makespan) by considering product structures and random processing time.

(2) In the proposed algorithm, a Q-learning method is employed to adaptively select the premium search algorithm in each iteration.

(3) A stochastic simulation method is incorporated into the search process to assess the performance and feasibility of acquired solutions.

The remainder of this article is structured as: Section 2 briefly sums up relevant research. Section 3 states the considered issue along with formulizing it. Section 4 presents the optimization approach. Section 5 performs comparative experiments and analyzes attained findings. Lastly, Section 6 epitomizes this article and explores the subsequent directions.

2 Literature Review

2.1 Related work

Over the last years, many researchers are dedicated to modeling and optimization of remanufacturing scheduling problems[20]. They concentrate on enhancing operational efficiency of disassembly, reprocessing, and reassembly processes from the perspective of independence and integration.

2.1.1 Scheduling only one shop

Recently, many studies are proposed to make optimal disassembly decisions by employing limited resources to optimize given criteria. Two important optimization problems, i.e., disassembly sequence planning problems (DSPPs) and disassembly line balancing problems (DLBPs), are addressed to find disassembly sequence following product structure constraints. Yet, nearly all existing studies concentrate on disassembling only one product. Liang et al.[21] developed a marine predators method combining a stochastic simulation to solve a DSPP with achieving disassembly profit maximization in consideration of noise contamination and energy consumption. Fu et al.[22] studied a DSPP with operation failures to realize maximal disassembly revenue and minimal energy consumption by employing a multiverse optimization method. Yu et al.[23] addressed a DSPP to realize minimal energy consumption via improving a whale optimization approach. Gao et al.[24] designed a data-driven method for dismantling products with uncertainties to find an optimal selective disassembly sequence. Via taking resource constraints into account in addressing DSPPs, Guo et al.[25] aimed to reach maximal disassembly revenue and minimal energy consumption by improving a genetic algorithm (GA). Zhang et al.[26] studied a DSPP to minimize disassembly costs via improving a social engineering method. Lee et al.[27] considered solving a DSPP with limited resources and work safety. A comprehensive sequence planning method is developed to reach a minimization of disassembly time. Guo et al.[28] investigated a partial destructive DSPP with consideration of the collaboration between workers and robots. They improved a GA to minimize disassembly time and cost.

In practice, it is necessary to dismantle multiple products in a planning period. Thus, finding optimal disassembly decisions in such circumstances is very essential. In recent years, some researchers start to study the modeling and optimization of multi-product disassembly optimization problems. Liang et al.[29] focused on maximizing disassembly profit for dismantling multiple products in DSPPs with random disassembly time. They designed an equilibrium method incorporating a stochastic simulation to tackle it. Liang et al.[30] formulated multi-product DLBPs focusing on gaining maximal disassembly profit under the constraints of disassembly time. A group teaching optimization method is improved to deal with it. Guo et al.[31] addressed multi-product DSPP and DLBP to reach maximal disassembly profit, minimal energy consumption, and carbon footprint by using simulated annealing (SA) methods and grey wolf optimization methods. Liu et al.[32] devoted to minimizing disassembly cost in multi-product DLBPs with workforce assignment. They used a cutting-plane algorithm and an approximated approach to address it. Hu et al.[33] proposed DLBPs considering multiple product disassembly having stochastic processing time to minimize disassembly cost. An integrated method of chance-constrained programming and distribution-free model was developed to settle it. Meanwhile, Yin et al.[34] addressed such problems with multi-robot workstations to realize minimal cycle time, minimal energy consumption, and minimal hazardous indices. They devised a mixed driving method to optimally solve it.

The existing studies on scheduling reprocessing operations mainly consider two classifications: flow-shop-type and job-shop-type reprocessing lines. Yu et al.[11] modeled a job-shop-type reprocessing scheduling problem having job families. They introduced two heuristics to reach minimal flow time of job families. Afterwards, Kim et al.[35] developed two iterated greedy methods to handle such a scheduling issue considering sequence-dependent setup operations with the applications in a reprocessing shop. Gao et al.[36] aimed to schedule and reschedule a flow-shop-type reprocessing process with new job insertion and variable processing time to minimize mean of earliness and tardiness and maximum completion time. A harmony search method was developed to handle it. Shi et al.[37] studied a reprocessing scheduling issue having concurrent flow-shop-type lines considering uncertain processing time and reliability. They improved a particle swarm optimization (PSO) method to gain minimal maximum completion time as well as maximal reliability. Liu et al.[38] considered a job-shop-type reprocessing scheduling issue having job families to find a tradeoff between replacement and repair modes. It was solved by improving an artificial bee colony method (ABC).

In addition to the above studies on scheduling disassembly and reprocessing operations, optimizing assembly processes also receive concern. Xiao et al.[12] studied an assembly sequence problem of remanufactured parts with diverse accuracy levels. In consideration of selection matching requirements, they hybridized GA and PSO to find an optimal assembly sequence. Li et al.[39] focused on a quality control issue of assembly operations in remanufacturing. They transformed it into a convex quadratic programming problem. Liu et al.[40] formulated an integrated optimization method according to dynamic programming for assembly operations with dynamicity and uncertainties.

2.1.2 Integrated scheduling

Different from the above studies focusing on scheduling only one shop, a few studies consider the integrated scheduling models to achieve an overall optimization. Doh and Lee[13] addressed an integrated disassembly and reprocessing lot-sizing issue to reach minimal total cost via adopting heuristics. Hojati[41] solved a disassembly flow-shop scheduling issue with reaching minimal makespan, and three heuristics were designed according to the problem properties. Kim et al.[42] investigated an integrated scheduling issue having flow-shop-type reprocessing lines to minimize total flow time. A heuristic-based local search method was devised to deal with it. To minimize total tardiness, Kim et al.[43] devised a priority method to address an integrated scheduling of disassembly, reprocessing and assembly operations. To reach minimal energy consumption, Wang et al.[44] improved a GA to tackle an integrated scheduling problem. Yu and Lee[45] considered scheduling three shops together with component matching requirements. They designed two solution methods that consider the three shops separately and integrally, respectively. Gong et al.[46] improved an evolutionary algorithm to address an integrated scheduling with minimizing makespan, flow time, and maximum machine load. Wen et al.[47] devised an optimization approach with two stages for solving an integrated scheduling problem to minimize makespan, carbon footprint, and tardiness. Zhang et al.[48] put forward an extended network graph to define the integrated scheduling problem and improved an ABC method to minimize makespan. Shi et al.[49] dealt with a remanufacturing scheduling to minimize completion time and carbon footprint by developing a flower pollination approach.

2.1.3 Optimization algorithms

By analyzing the above studies, we find that most of them use meta-heuristics to solve remanufacturing scheduling problems. The meta-heuristic methods win wide acceptance from academia because of their high efficiency and accuracy in response time and quality[50]. In recent years, a hybridization of artificial intelligence and meta-heuristics has been regarded as a favorable way in solving complex optimization issues. Ala et al.[51] studied patient information performance optimization in smart healthcare centers. The improved PSO-long short-term memory algorithm was designed to deal with it. Wang et al.[52] developed an ABC with reinforcement learning methods to address a distributed three-stage assembly scheduling with maintenance activities. Qi et al.[53] solved a time-dependent green vehicle routing problems with time windows by using a Q-learning based evolutionary algorithm. Ji et al.[54] devised a Q-learning based hyper-heuristic algorithm to handle dynamic task allocation of crowdsensing problems. Cai et al.[55] solved distributed hybrid flow shop scheduling problems with assembly operations by designing a hybridization of shuffled frog-leaping algorithm (SFLA) and reinforcement learning algorithms. Similarly, the hybridization framework was also applied to remanufacturing scheduling problems. Ren et al.[56] investigated a DLBP to minimize the smoothing index via a Q-learning based variable neighborhood iterative search algorithm. Chu and Chen[57] designed a PSO with a Q-learning method for human-robot collaboration disassembly planning problems.

2.2 Discussion

Via summing up previous work on remanufacturing scheduling problems, we find that they feature with the subsequent characteristics:

(1) Most of them make disassembly sequence decisions of disassembly shops following product structures in addressing DSPPs and DLBPs. Also, a few studies devote to solving multi-product disassembly optimization problems.

(2) All of the studies on integrated remanufacturing scheduling do not consider product structures. In order to obtain a performable schedule for a practical remanufacturing process, we ought to obey product structures in solving such integrated remanufacturing scheduling problems. Also, the uncertainties have to be taken into account since we are difficult to exactly know the conditions of EOL products.

(3) Meta-heuristic approaches have been widely and victoriously employed to deal with disassembly optimization and integrated remanufacturing scheduling problems. However, rare attention is dedicated to studying a hybridization of meta-heuristic and reinforcement learning methods to deal with such difficult problems.

A tabular review and comparison are given in Table 1. It is seen that integrated scheduling problems of disassembly and reprocessing considering product structures in uncertain environments are missing. Thereby, this work studies the modeling and

Table 1 Comparisons of existing studies and our work in terms of shop types, characteristics, environments, and methodologies.

Reference	Shop type			Characteristic		Environment		Solution method
	DS	RS	AS	PS	MP	DP	UP	
[30]	√	−	−	√	√	−	√	Meta-heuristic (EGTOA)
[31]	√	−	−	√	√	−	√	Meta-heuristic (SMDG)
[32]	√	−	−	√	√	−	√	Matheuristic
[33]	√	−	−	√	√	−	√	Heuristic; matheuristic
[34]	√	−	−	√	√	√	−	Meta-heuristic (HDA)
[35]	−	√	−	−	√	√	−	Meta-heuristic (IGA)
[36]	−	√	−	−	√	−	√	Meta-heuristic (DHS)
[37]	−	√	−	−	√	−	√	Meta-heuristic (EDPSO)
[38]	−	√	−	−	√	√	−	Meta-heuristic (EABC)
[39]	−	−	√	−	−	−	√	Matheuristic
[40]	−	−	√	−	−	−	√	Matheuristic
[41]	√	√	−	−	√	√	−	Heuristic
[42]	√	√	√	−	√	√	−	Heuristic
[43]	√	√	√	−	√	√	−	Heuristic
[44]	√	√	√	−	√	√	−	Meta-heuristic (GAVNS)
[45]	√	√	√	−	√	√	−	Heuristic
[46]	√	√	√	−	√	√	−	Meta-heuristic (HMEA)
[47]	√	√	√	−	√	√	−	Meta-heuristic (INSGA-II)
[48]	√	√	√	−	√	√	−	Meta-heuristic (IABC)
[49]	√	√	√	−	√	√	−	Meta-heuristic (IFPA)
Our work	√	√	−	√	√	−	√	Q-HMH

Note: DS: disassembly shop; RS: reprocessing shop; AS: assembly shop; PS: product structure; MP: multi-product; DP: deterministic problem; UP: uncertain problem; EGTOA: enhanced group teaching optimization algorithm; SMDG: SA-based multi-objective discrete grey wolf optimization; HDA: hybrid driving algorithm; IGA: iterated GA; DHS: discrete harmony search; EDPSO: extended discrete particle swarm optimization; EABC: extended artificial bee colony; GAVNS: hybrid genetic algorithm based on variable neighborhood search; HMEA: hybrid multi-objective evolutionary algorithm; INSGA-II: improved NSGA-II; IABC: improved ABC; and IFPA: improved flower pollination algorithm.

optimization of such problems. The details are given below.

3 Problem Definition

3.1 Problem statement

The considered integrated remanufacturing scheduling problem of disassembly and reprocessing processes is defined as follows. At a disassembly shop, all EOL products are completely dismantled into components on multiple identical workstations by performing disassembly operations and following their product structures. Once a component is disassembled from a product, it enters the reprocessing shop to be fabricated on its dedicated flow-shop-type reprocessing line. As a matter of fact, there exist multiple reprocessing lines in the reprocessing shop, and they have diverse functions. A disassembled component must be reprocessed on the specific flow-shop-type reprocessing line. Namely, a reprocessing line has a specified function to handle given components, and therefore it is defined as a dedicated flow-shop-type reprocessing line. Each line contains multiple machines and the components assigned to it must pass all the involved machines as the same route. The disassembly time of products and processing time of components obey known random distributions.

An executable schedule should conform to the subsequent restrictions: (1) At any time, each workstation can dismantle no more than one product; (2) Each product can be disassembled at most on one workstation at any time; (3) Each machine can fabricate no more than one component at any time; (4) Each component can be fabricated by one machine at the utmost at any time; (5) Preemption of workstations and machines is forbidden at the disassembly and reprocessing shops. The target of addressing the considered problem is to achieve minimal makespan by determining the following decisions: (1) disassembly workstation assignment of products; (2) disassembly sequence of products on workstations; (3) performed disassembly operations of products; (4) performed sequence of disassembly operations; (5) reprocessing line assignment of components; (6) processing sequence of components on reprocessing lines.

3.2 Model formulation

To define the products' structure mathematically, this work uses AND/OR graphs[22] to formulize relationships of subassemblies and disassembly operations. Notice that a product includes multiple subassemblies, and a subassembly can be further dismantled into other subassemblies. If a subassembly cannot be further disassembled, it is called a component in this research. Let i be a subassembly index, while k and k' are disassembly operation indexes. An AND/OR graph is depicted by the subsequent matrices:

(1) A precedence matrice $\mathcal{T}=[\tau_{kk'}]$ is adopted to define conflict and precedence relations of disassembly operations. It is given as

$$\tau_{kk'} = \begin{cases} 1, & \text{if } k' \text{ can be implemented following } k; \\ -1, & \text{if } k \text{ and } k' \text{ conflict with each other;} \\ 2, & \text{if } k \text{ and } k' \text{ do not have sequential relationships;} \\ 0, & \text{otherwise.} \end{cases}$$

(2) A succession matrix $\mathcal{S}=[\xi_{kk'}]$ is adopted to formulize succession relationships of disassembly operations. It is formulated as

$$\xi_{kk'} = \begin{cases} 1, & \text{if } k' \text{ can be implemented next to } k; \\ 0, & \text{otherwise.} \end{cases}$$

(3) An incidence matrix $\mathcal{I}=[\omega_{ik}]$ is established to depict relationships of subassemblies and operations. It is formulated as follows:

$$\omega_{ik} = \begin{cases} 1, & \text{if } i \text{ is acquired by performing } k; \\ -1, & \text{if } i \text{ is dismantled by } k; \\ 0, & \text{otherwise.} \end{cases}$$

To model the problem under investigation, we give the following symbols:

Indexes:

p: Product index, $p \in \{0,1,2,\ldots,P\}$, in which P denotes the number of products, and 0 represents a dummy product.

i: Subassembly index, $i \in \{1,2,\ldots,V_p,V_p+1,\ldots,I_p\}$, where V_p denotes the quantity of subassemblies of product p, and I_p is the quantity of subassemblies and components of product p.

j: Component index, $j \in \{0,1,2,\ldots,N\}$, in which N signifies the quantity of components disassembled from products, and 0 represents a dummy component.

k: Disassembly operation index, $k \in \{0,1,2,\ldots,J_p\}$, in which J_p represents the quantity of operations of product p, and 0 denotes a dummy operation without any disassembly time.

l: Disassembly workstation index, $l \in \{1,2,\ldots,L\}$, in which L is the quantity of disassembly workstations at the disassembly shop.

r: Reprocessing line index, $r \in \{1,2,\ldots,R\}$, in which R represents the quantity of reprocessing lines at the reprocessing shop.

s: Stage index of reprocessing lines, $s \in \{1,2,\ldots,S\}$, in which S denotes the quantity of stages in a reprocessing line.

Parameters:

t^d_{pk}: Disassembly time of operation k regarding product p.

$t^s_{pkk'}$: Setup time in the cast that operation k' is executed following k regarding product p.

t_{js}: Processing time of component j at stage s at the reprocessing shop.

$\mathcal{I}_p$: Incidence matrix of product p.

$\mathcal{S}_p$: Succession matrix of product p.

ω_{pik}: An element in the i-th row and the k-th column of $\mathcal{I}_p$ as regards p.

$\xi_{pkk'}$: An element in the k-th row and the k'-th column of $\mathcal{S}_p$ as regards p.

h_{jr}: Equaling to 1, if component j can be processed on the reprocessing line r, and 0, otherwise.

G: A very large number.

Notice that t^d_{pk}, $t^s_{pkk'}$, and t_{js} are random.

Decision variables:

x_{pl}: If product p is assigned to workstation l for disassembling, $x_{pl}=1$; otherwise, $x_{pl}=0$.

$y_{pp'l}$: If product p' is disassembled after p on workstation l, $y_{pp'l}=1$; otherwise, $y_{pp'l}=0$.

z_{pk}: If operation k of product p is executed, $z_{pk}=1$; otherwise, $z_{pk}=0$.

$u_{pkk'}$: If operation k' of product p is executed following k, $u_{pkk'}=1$; otherwise, $u_{pkk'}=0$.

v_{jr}: If component j is assigned to the reprocessing line r for processing, $v_{jr}=1$; otherwise, $v_{jr}=0$.

$w_{jj'r}$: If component j' is processed after j on the reprocessing line r, $w_{jj'r}=1$; otherwise, $w_{jj'r}=0$.

a_p: Disassembly completion time of product p.

b_{pk} : Disassembly completion time of operation k regarding product p.

c_{pi} : Disassembly completion time of subassembly i of product p.

d_j : Disassembly completion time of component j.

e_{rjs} : Reprocessing completion time of component j at stage s on the reprocessing line r.

C_M : Makespan.

Notice that $a_p, b_{pk}, c_{pi}, d_j, e_{rjs}$, and C_M are stochastic variables.

By employing the aforesaid symbols, this research formulizes a stochastic programming model to minimize expected makepan below.

$$\min E(C_M) \tag{1}$$

subject to:

$$z_{pk'} = \sum_{k=0}^{J_p} u_{pkk'}, k' = 1,2,\ldots,J_p; p = 1,2,\ldots,P \tag{2}$$

$$z_{pk} + z_{pk'} \leqslant 1, \forall \tau_{pkk'} = -1; k,k' = 0,1,2,\ldots,J_p; p = 1,2,\ldots,P \tag{3}$$

$$\xi_{pkk'} - u_{pkk'} \geqslant 0, \forall \tau_{pkk'} = 1 \text{ or } 0; k,k' = 0,1,2,\ldots,J_p; p = 1,2,\ldots,P \tag{4}$$

$$\sum_{k=0}^{J_p} \xi_{pkk'} \cdot u_{pkk'} \geqslant u_{pk'k''}, \forall \tau_{pk'k''} = 2; k',k'' = 0,1,2,\ldots,J_p; p = 1,2,\ldots,P \tag{5}$$

$$0 \leqslant \sum_{k=0}^{J_p} \omega_{pik} \cdot z_{pk} \leqslant 1, i = 1,2,\ldots,I_p; p = 1,2,\ldots,P \tag{6}$$

$$\sum_{k'=1}^{k} u_{pk'k} \geqslant \sum_{k''=1}^{J_p} u_{pkk''}, k = 1,2,\ldots,J_p; p = 1,2,\ldots,P \tag{7}$$

$$\sum_{k=0}^{J_p} \omega_{pik} \cdot z_{pk} = 1, i = V_p+1, V_p+2,\ldots,I_p; p = 1,2,\ldots,P \tag{8}$$

$$\sum_{l=1}^{L} x_{pl} = 1, p = 1,2,\ldots,P \tag{9}$$

$$\sum_{l=1}^{L}\sum_{p=0}^{P} y_{pp'l} = 1, p' = 0,1,2,\ldots,P \tag{10}$$

$$\sum_{l=1}^{L}\sum_{p'=0}^{P} y_{pp'l} = 1, p = 0,1,2,\ldots,P \tag{11}$$

$$E\left(b_{p'0} + \left(3 - x_{pl} - x_{p'l} - y_{pp'l}\right) \cdot G - a_p\right) \geqslant 0, p,p' = 1,2,\ldots,P; l = 1,2,\ldots,L \tag{12}$$

$$E\left(a_p - b_{p0} - \sum_{k=1}^{J_p} t_{pk}^{d} \cdot z_{pk} - \sum_{k=1}^{J_p}\sum_{k'=1}^{J_p} t_{pkk'}^{s} \cdot u_{pkk'}\right) \geqslant 0, p = 1,2,\ldots,P \tag{13}$$

$$E\left(b_{pk'} - b_{pk} - t_{pk}^{d} - t_{pkk'}^{s}\right) \geqslant 0, \forall z_{pk'} \cdot u_{pkk'} = 1, p = 1,2,\ldots,P; k,k' = 0,1,2,\ldots,J_p \tag{14}$$

$$E\left(c_{pi} - b_{pk}\right) \geqslant 0, \forall z_{pk} \cdot \omega_{pik} = 1; p = 1,2,\ldots,P; k = 0,1,2,\ldots,J_p; i = 1,2,\ldots,I_p \tag{15}$$

$$E\left(d_{(p-1)\cdot(I_{p-1}-V_{p-1})+i-V_p} - c_{pi}\right) \geqslant 0, p = 1,2,\ldots,P; i = V_p+1, V_p+2,\ldots,I_p \tag{16}$$

$$\sum_{r=1}^{R} v_{jr} = 1, j = 1,2,\ldots,N \tag{17}$$

$$v_{jr} \leqslant h_{jr}, j = 1,2,\ldots,N; r = 1,2,\ldots,R \tag{18}$$

$$E\left(e_{rj1} + \left(1 - v_{jr}\right) \cdot G - d_j - t_{j1}\right) \geqslant 0, j = 1,2,\ldots,N; r = 1,2,\ldots,R \tag{19}$$

$$E\left(e_{rj(s+1)} + \left(1 - v_{jr}\right) \cdot G - e_{rjs} - t_{j(s+1)}\right) \geqslant 0, j = 1,2,\ldots,N; r = 1,2,\ldots,R; s = 1,2,\ldots,S-1 \tag{20}$$

$$\sum_{j=1}^{N} w_{jj'r} = \sum_{j=1}^{N} w_{j'jr}, \forall v_{j'r} = 1; r = 1,2,\ldots,R; j' = 0,1,2,\ldots,N \tag{21}$$

$$\sum_{j=1}^{N} w_{jj'r} = 1, \forall v_{j'r} = 1; r = 1,2,\ldots,R; j' = 0,1,2,\ldots,N \tag{22}$$

$$\sum_{j'=1}^{N} w_{jj'r} = 1, \forall v_{jr} = 1; r = 1,2,\ldots,R; j = 0,1,2,\ldots,N \tag{23}$$

$$E\left(e_{rj's} + \left(3 - v_{j'r} - v_{jr} - w_{jj'r}\right) \cdot G - e_{rjs} - t_{js}\right) \geqslant 0, r = 1,2,\ldots,R; s = 1,2,\ldots,S; j,j' = 1,2,\ldots,N \tag{24}$$

$$E\left(C_M - e_{rjs}\right) \geqslant 0, r = 1,2,\ldots,R; j = 1,2,\ldots,N \tag{25}$$

$$x_{pl} \in \{0,1\}, y_{pp'l} \in \{0,1\}, z_{pk} \in \{0,1\}, u_{pkk'} \in \{0,1\}, v_{jr} \in \{0,1\}, w_{jj'r} \in \{0,1\}, p,p' = 1,2,\ldots,P; l = 1,2,\ldots,L; k,k' = 0,1,2,\ldots,J_p; j,j' = 1,2,\ldots,N; r = 1,2,\ldots,R \tag{26}$$

$$a_p \geqslant 0, b_{pk} \geqslant 0, c_{pi} \geqslant 0, d_j \geqslant 0, e_{rjs} \geqslant 0, C_M \geqslant 0, p = 1,2,\ldots,P; k = 0,1,2,\ldots,J_p; i = 1,2,\ldots,I_p; j = 1,2,\ldots,N; r = 1,2,\ldots,R; s = 1,2,\ldots,S \tag{27}$$

where Formula (1) aims to achieve minimal expected makespan. Equation (2) requires that each disassembly operation can be implemented at most once. Formulas (3)−(5) ensure that precedence and conflict relationships of operations must be met. Formula (6) guarantees that each subassembly ought to be dismantled by implementing a corresponding operation at most once. Formula (7) assures flow balances between in-degree and out-degree of operations. Equation (8) requires that all products must be completely disassembled. Equation (9) ensures that each product must be disassembled on only one workstation. Formulas (10)−(12) give sequence relationships of two adjacent products on workstations. Formula (13) defines the disassembly completion time of products on workstations. Formulas (14)−(16) formulate the disassembly completion time of components. Formulas (17) and (18) signify that each component must be processed on only one reprocessing line. Formula (19) denotes that the start time of components at the reprocessing shop ought to be equal to or larger than their disassembly completion time. Formula (20) ensures that a component can start to be fabricated at a stage in the case that it finishes being processed at the precedence stage. Equations (21)−(23) indicate that each component must be fabricated just once at a stage on a reprocessing line. Formula (24) gives sequence relationships of two adjacent components on a reprocessing line. Formula (25) formulates the makespan of the entire process. The variable ranges are limited by Formulas (26) and (27).

4 Proposed Q-HMH

In recent years, a hybridization of meta-heuristics and Q-learning methods attracts widespread attention. Such hybridization frameworks have gained great success in solving various optimization problems. Inspired by their excellent performance, this work develops a Q-learning based hybrid meta-heuristic in consideration of problem-specific characteristics. It effectively utilizes the feedback information acquired from the previous search process, and further chooses the most promising meta-heuristic to search satisfaction solutions.

4.1 Solution encoding and population initialization

This research employs a combination of an integer string and multiple double-link integer strings to represent a solution (called an individual as well) of the considered problem, i.e., $\pi = \left(\pi_0, \left(\pi'_1, \pi''_1\right), \left(\pi'_2, \pi''_2\right), \ldots, \left(\pi'_P, \pi''_P\right)\right)$. $\pi_0 = (d_1, d_2, \ldots, d_P)$ in π denotes a product sequence string, where each integer d_p on the p-th position indicates a product index, $p \in \{1, 2, \ldots, P\}$. A double-link integer string $\left(\pi'_p, \pi''_p\right)$ in π represents a disassembly decision of product p. $\pi'_p = \left(o_{p1}, o_{p2}, \ldots, o_{pJ_p}\right)$ denotes an operation sequence substring for product p, where each integer o_{pk} on the k-th position indicates a disassembly operation index, $k \in \left\{1, 2, \ldots, J_p\right\}$; $\pi''_p = \left(x_{p1}, x_{p2}, \ldots, x_{pJ_p}\right)$ including binary numbers is an operation execution substring for product p. If an element in π''_p equals to 1, we implement the operation on the associated position in π'_p, and otherwise, it is not done. Figure 1c illustrates a solution of an instance with two products as shown in Figs. 1a and 1b. It is seen that Products 2 and 1 are disassembled following their relative sequence at the disassembly shop. Disassembly operations 1, 3, and 5

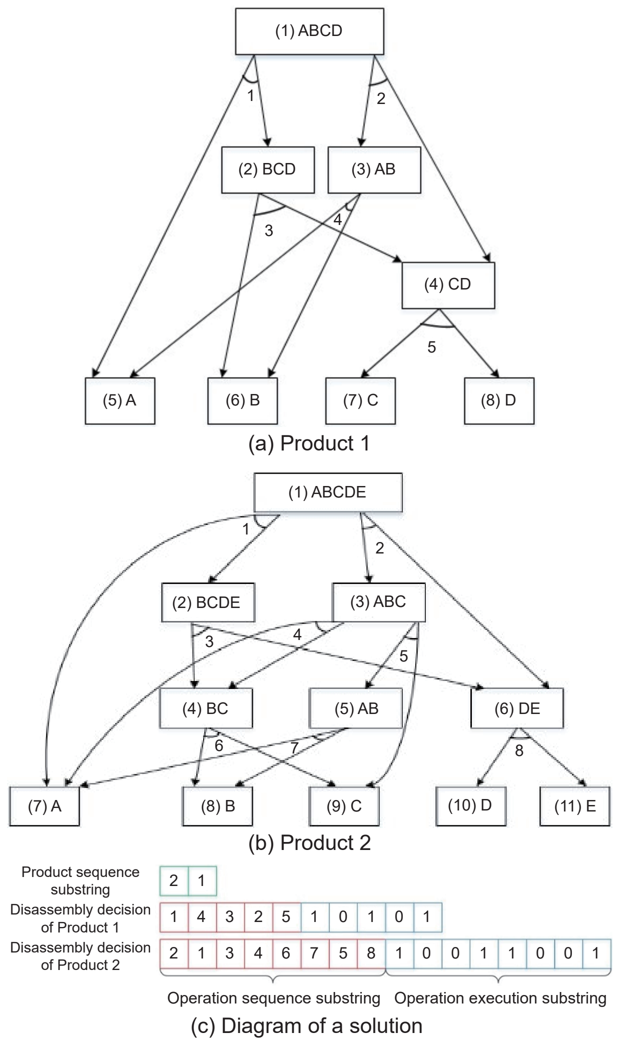


Fig. 1 Illustration of solution representation methods.

of Product 1 and 2, 4, 6, and 8 of Product 2 are executed according to the given sequence, respectively.

The above methods just give the disassembly decisions at the disassembly shop, and we cannot obtain the scheduling decisions at the reprocessing shop directly. Thereby, we design heuristics to decode an individual π to an executable schedule as: (1) At the disassembly shop, we schedule all products according to the product sequence string, and assign a product to the earliest available disassembly workstation. Once a component is dismantled from a product, it enters a reprocessing shop for reprocessing immediately; (2) At the reprocessing shop, the arrival components from the disassembly shop are sequenced in an ascending order according to the disassembly completion time, and they are sequentially assigned to their accordingly dedicated reprocessing lines with the least workload. By employing the above approach, we can decode a solution to an executable schedule.

Noteworthily, a solution might be unfeasible on condition that it violates the precedence and conflict constraints of disassembly operations. In such situation, we use the following methods to repair the disassembly decisions of a product p:

First, we adjust the operation execution substring according to the precedence matrix of product p as: (1) The first executed disassembly operation is identified, and its conflict operations are found; If the corresponding elements of conflict operations in π''_p are 1, they are adjusted to 0, and the other elements in π''_p are adjusted to 1; (2) The first operation corresponding to 1 in π''_p is identified again, and we adjust the following elements of its conflict operations in π''_p to 0. By repeating (2), we check the disassembly operations sequentially until all executed disassembly operations do not conflict with each other.

Second, we change the operation sequence substring according to the succession matrix as follows: (1) All the executed operations are deleted from π'_p; (2) For each executed operation, we find out its executed immediate precedence operations. Then, we can construct a precedence operation list for each executed operation; (3) The operation without immediate precedence operations is inserted into the first vacant position in π'_p; (4) We delete this operation from the lists of the other executed operations. By repeating (3) and (4), we can form a disassembly operation sequence obeying precedence relationships.

By adopting the aforesaid methods, we are able to generate F feasible individuals as a population.

4.2 Assessment of solutions’ objective values

The objective function in the formulated model concentrates on minimizing expected makespan because the disassembly time of products and processing time of components are stochastic. Q-HMH incorporates a stochastic simulation[29] to appraisal the objective value. It averages a vast quantity of samples as an estimation to the true value. A sample contains all the disassembly time of products and processing time of components which are created according to their associated stochastic distributions. On condition that the quantity of samples go to infinite, this approach is able to gain the true value. However, we need to use finite computation resources for simulation in practice. Let π be an assessed solution and ϕ be the quantity of samples. The main steps are provided as follows:

Step 1: Set $C_M(\pi) := 0$ and $\phi := 10$.

Step 2: Produce ϕ samples, and each of which is denoted as τ_λ, $\lambda = 1,2,\ldots,\phi$.

Step 3: Evaluate the makespan of τ_λ, $C'_M(\tau_\lambda)$, $C'_M(\tau_\lambda)$ denotes the makespan of π associated with the sample τ_λ, $\lambda = 1,2,\ldots,\phi$.

Step 4: Calculate the average value $C_M(\pi)$ of all samples as

$$C_M(\pi) = \frac{\sum_{\lambda=1}^{\phi} C'_M(\tau_\lambda)}{\phi}.$$

Step 5: Return $C_M(\pi)$ as an approximation to the expected makespan of π.

4.3 Design of search methods for solving the studied problem

This article improves the GA, ABC method, SFLA, and SA method for solving the problem under consideration. Their detailed designs are given below.

4.3.1 Design of GA

GA, motivated by the evolution theory of natural selection, is a population-based meta-heuristic. It uses three basic operations, i.e., selection, crossover, and mutation, to perform an evolution process[44]. In Q-HMH, they are devised as follows:

The selection operation adopts a tournament selection approach[58] to choose parent individuals for implementing a crossover approach. Subsequently, the crossover approach is implemented on two parent individuals as

(1) For the product sequence string, an order-based crossover (OX) method[59] is adopted to recombine two

product sequence strings. An example with eight products is shown in Fig. 2. Two cut points are chosen at random, and the product indexes between them in Parent 1 are copied to an offspring at the same positions directly. The rest products in the offspring are filled with the products from Parent 2 according to their related sequence; (2) For the operation sequence substring and operation execution substring of products, a precedence preservative crossover (PPX) method[60] is adopted to produce new substrings. To display the PPX, an example of a product p owning four disassembly operations is given in Fig. 3. A binary string possessing the same length with the quantity of disassembly operations is produced at random. If a number in this string equals to 0, the element in the operation sequence substring π'_p at the corresponding position is from one parent individual, and otherwise, it is from the other parent individual. Then, this element is deleted from two parent individuals; (3) For the operation execution substring, the elements corresponding to 1 in the binary string are sequentially chosen from a parent individual, and the rest are successively from the other parent individual.

After the crossover operation, a mutation operation is applied with probability κ_m on the newly generated individuals as: (1) For the product sequence string, we

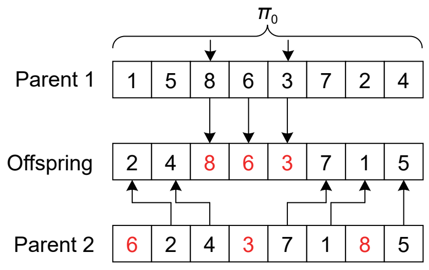


Fig. 2 Illustration of the OX method.

randomly choose two products and swap them; (2) For the disassembly decisions of products, we randomly choose a product and swap any two operations in its associated operation sequence string. Also, the corresponding elements in the operation execution substring are swapped; (3) In the case that the new individual is infeasible, this research repairs it to obtain a feasible individual employing the method in Section 4.1.

It is noted that the best individual in population is inherited into the new population directly, and the rest are created by using the crossover and mutation methods.

4.3.2 Design of ABC approach

ABC is a population-based meta-heuristic method in accordance with foraging activities of honey bee swarm. In the ABC method, the position of a food source denotes a solution of a solved issue, and its nectar quantity is related to the performance of its corresponding solution[61]. In general, the colony includes three sections: employed, onlooker, and scout bees. Accordingly, its search process contains three stages, i.e., employed bee phase, onlooker bee phase, and scout bee phase. In Q-HMH, they are devised as:

In the employed bee phase, we use crossover operations to create new individuals, i.e., solutions or food sources. For an individual in population, the other individual is correspondingly selected from population using a tournament selection method[59], and the two individuals are seen as parent individuals. Then, a new individual is produced by employing OX and PPX methods. Afterwards, an SA method introduced in the following is employed to further refine the new individuals.

In the onlooker bee phase, this research employs a tournament selection approach[59] to select F

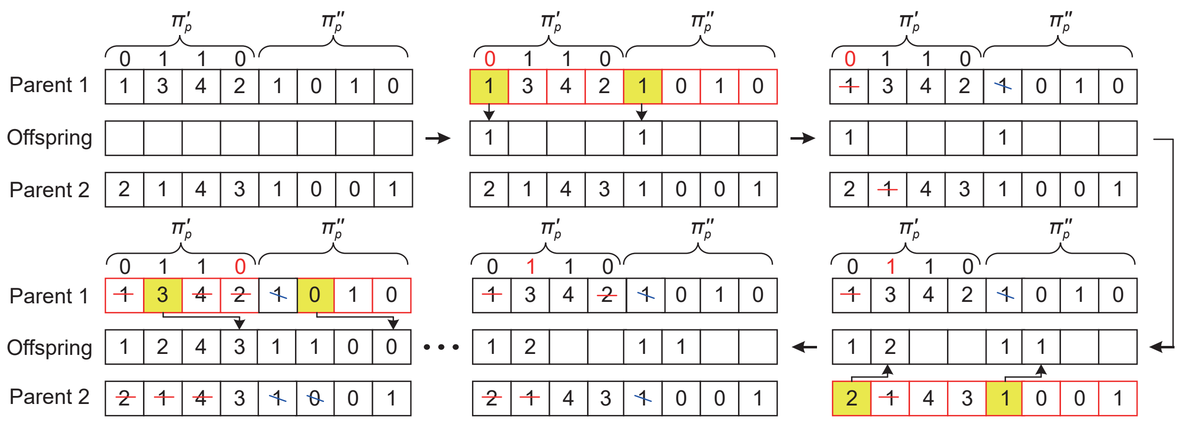


Fig. 3 Illustration of the PPX approach.

individuals from the population and new generated individuals as a new population. It is noted that the best individual found in the previous search process is straightforwardly stored into the new population.

In the scout bee phase, on condition that an individual is not improved during a given κ_l' iterations, we randomly generate two individuals, and the better one is employed to replace it directly. κ_l' is a specified parameter that is named as the number of food sources limit.

4.3.3 Design of SFLA

SFLA is a population-based meta-heuristic in line with observation and imitation of frog group behavior. It has been widely used in coping with various optimization problems[62]. In SFLA, a solution of the solved problem is denoted as a frog's position (i.e., an individual), and a population includes a group of individuals. It consists of three main procedures, i.e., memeplex construction, solution searching, and population recombination[62]. Their designs in Q-HMH are given as:

The memeplex construction aims to partition the population into η memeplexes. The individuals are sequenced in an ascending order in accordance with their quality. Then, the first is allotted to the memeplex $\mathcal{M}_1$, the second goes to $\mathcal{M}_2$, the η-th is assigned to $\mathcal{M}_\eta$, and the $(\eta+1)$-th enters $\mathcal{M}_1$. In this way, we construct η memeplexes.

The search process aims to evolve all the memeplexes with μ iterations. At each iteration, β individuals are selected at random from the present memeplex to form a submemeplex, and we identify the best and worst individuals in this submemeplex which are recorded as π_b and π_w, respectively. Then, they are employed to produce a new individual π_o with OX and PPX methods. If π_o is better than π_w, π_o replaces π_w, and otherwise, the global best individual π_g in population and π_w are adopted to create another individual that is allowed to replace π_w if it is better than π_w. If both the two methods do not produce a better individual compared with π_w, an individual is generated at random and replaces π_w directly. The above steps are repeated until all memeplexes are updated.

The population recombination combines all the memeplexes into a new population. By using the above procedures, SFLA can perform an evolution process.

4.3.4 Design of SA method

SA is a meta-heuristic method according to the probabilistic technique for searching a global solution of an optimization problem. It imitates an annealing process in metallurgy, a technique including heating and controlled cooling to vary its physical properties. Different from a basic local search method that prefers a better neighborhood solution in a greedy way, SA allows to accept a worst solution with a probability. Thus, it has better abilities of jumping from local optima. In Q-HMH, according to the work[63], SA is designed to refine the best individual in population. Let π be a refined solution, the main steps of the SA method in Q-HMH are given as:

Step 1: Set an initial temperature $t_o := C_M(\pi_w) - C_M(\pi_g)$, where π_w and π_g denote the worst and best solutions in population, respectively, $C_M(\pi_w)$ and $C_M(\pi_g)$ are severally the objective values of π_w and π_g, respectively. Let the current temperature $t := t_0$.

Step 2: Create π_1 based on π by choosing one product at random and swapping any two disassembly operations without sequential relationships.

Step 3: If π_1 bests π, $\pi := \pi_1$, switch to Step 5; if not, switch to Step 4.

Step 4: If $\mathrm{rand}(0,1) \leqslant \exp(-\Delta/t)$, $\pi := \pi_1$, go to Step 5. Notice that $\mathrm{rand}(0,1)$ creates a number at random between 0 and 1, and $\Delta := C_M(\pi_1) - C_M(\pi)$, where $C_M(\pi_1)$ and $C_M(\pi)$ are the objective values of π_1 and π, respectively.

Step 5: Let $t := t \cdot \rho$, where ρ is an annealing coefficient.

Step 6: Repeat Steps 2−5 until $t < t_0 \cdot (1-\rho)$.

Step 7: Update the global best solution π_g by using the found best solution.

4.4 Q-learning based selection method

There exist five essential items in the Q-learning approaches: states $\mathscr{s}$, actions $\mathscr{a}$, rewards $\mathscr{r}$, action selection strategies, and Q-table. In this research, the states are defined as a combination of the search stage and population evaluation, the actions are to select one search method from GA, ABC, SFLA, and SA for performing, the rewards are relevant with the state movement, and the Q-table is employed to record actions' feedback results.

Q-HMH concentrates on finding optimal solutions for solving the considered problem, and thus we mainly concern the obtained best solution in a search process. Thereby, the variation of the best solution π_g in population is regarded as an evaluation criterion.

Additionally, the employed methods have essential roles at different search stages in a search process, and thus it is better to choose befitting search methods for performing at the stages. Thereby, the whole search process is partitioned into a series of stages. Consequently, the variation of obtained best solutions and the search stage are combined to define eight states as shown in Table 2. b_l is employed to formulate the improvement of π_g. In the case that π_g is improved in the present iteration, $b_l = 1$; otherwise, $b_l = 0$. e_s denotes a search stage. This work evenly partitions the whole search process into four stages.

According to the states, we define four actions: a_1, a_2, a_3, and a_4, signifying that Q-HMH chooses GA, ABC, SFLA, and SA for implementing at an iteration, respectively. In Q-HMH, the reward function is defined as

$$r_{t+1} = \begin{cases} s_t - s_{t+1}, s_t \neq s_{t+1}; \\ s_t - s_{t+1}, s_t = s_{t+1}, s_t > 4, s_{t+1} > 4; \\ s_{\max} - s_{\min}, s_t = s_{t+1}, s_t \leqslant 4, s_{t+1} \leqslant 4 \end{cases} \quad (28)$$

where s_t denotes the state at the t-th iteration, $s_t \in \{1,2,\ldots,8\}$. $s_{\max}$ and $s_{\min}$ severally mean the maximal and minimal indexes of states, respectively, i.e., $s_{\max} = 8$ and $s_{\min} = 1$. r_{t+1} represents the reward value at the $(t+1)$-th iteration. By dissecting all the states, it is seen that the best solution is improved at States 1−4, and thus a positive reward value is given. For the rest states, we give a negative value. The Q-table is updated as Qi et al.[53]

$$\begin{aligned} & Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \\ & \alpha \cdot \left(r_{t+1} + \gamma \cdot \max_{a} \{Q(s_{t+1}, a)\} - Q(s_t, a_t) \right) \end{aligned} \quad (29)$$

where $Q(s_t, a_t)$ denotes a Q value that chooses a_t at s_t, α is a learning rate, and γ indicates a discount factor. α and γ are set to 0.1 and 0.9, respectively. r_{t+1} is a reward value by taking a_t at s_t, $s_t \in \{1,2,\ldots,8\}$ and $a_t \in \{a_1, a_2, a_3, a_4\}$. $\max_{a} \{Q(s_{t+1}, a)\}$ indicates the maximal Q value in Q-table at s_{t+1}, $a \in \{a_1, a_2, a_3, a_4\}$.

Table 2 State definition.

State No.	Definition
1	$b_l = 1$ and $e_s = 1$
2	$b_l = 1$ and $e_s = 2$
3	$b_l = 1$ and $e_s = 3$
4	$b_l = 1$ and $e_s = 4$
5	$b_l = 0$ and $e_s = 1$
6	$b_l = 0$ and $e_s = 2$
7	$b_l = 0$ and $e_s = 3$
8	$b_l = 0$ and $e_s = 4$

This article uses an ϵ-greedy method[54, 64] to choose an action for execution. In the case that a created number between 0 and 1 is smaller than a given parameter ϵ, we randomly choose an action; otherwise, an action with the maximal Q value is chosen.

4.5 Framework of Q-HMH

In order to exhibit the procedures of Q-HMH intuitively, we illustrate its flowchart as provided in Fig. 4. First, we initialize all parameters and create a group of individuals as a population. Second, the method continuously iterates as: (1) The Q-learning approach is employed to choose a search method; (2) The selected search method is performed to produce a new population, of which valuable information is used to update the Q-table. At last, the obtained best solution is exported if a given stopping criterion is met.

5 Experiment Result and Dissection

To validate the capacities of Q-HMH in working out the problem under consideration, we perform experiments by using a group of test instances. Furthermore, we choose three popular and classical meta-heuristics for solving disassembly optimization problems in literature, i.e., GA[65], ABC[58], enhanced equilibrium optimizer (EEO)[29], and a mathematical programming solver CPLEX, for comparisons. All the approaches are programmed with C++ language on the VC++ 2017 and implemented on a computer owning an Intel Core i5-8265U CPU @ 1.60 GHz with 8 GB RAM.

5.1 Test instance construction

This study chooses a product from the existing literature[58] and three randomly generated products as test cases. Then, four test instances including P products are produced via choosing the four products at random, $P \in \{2,4,6,8\}$. For each instance, the quantity of workstations at the disassembly shop is selected at random from $\{1,2,3,4\}$, the number of reprocessing lines at the reprocessing shop is produced from $\{2,3,4\}$, and the quantity of stages in a reprocessing line is chosen from $\{3,4\}$. The dedicated reprocessing lines of components are created at random in accordance with the quantity of reprocessing lines. It is noted the disassembly time and setup time of operations and processing time of components are stochastic numbers obeying truncated normal distributions. The mean

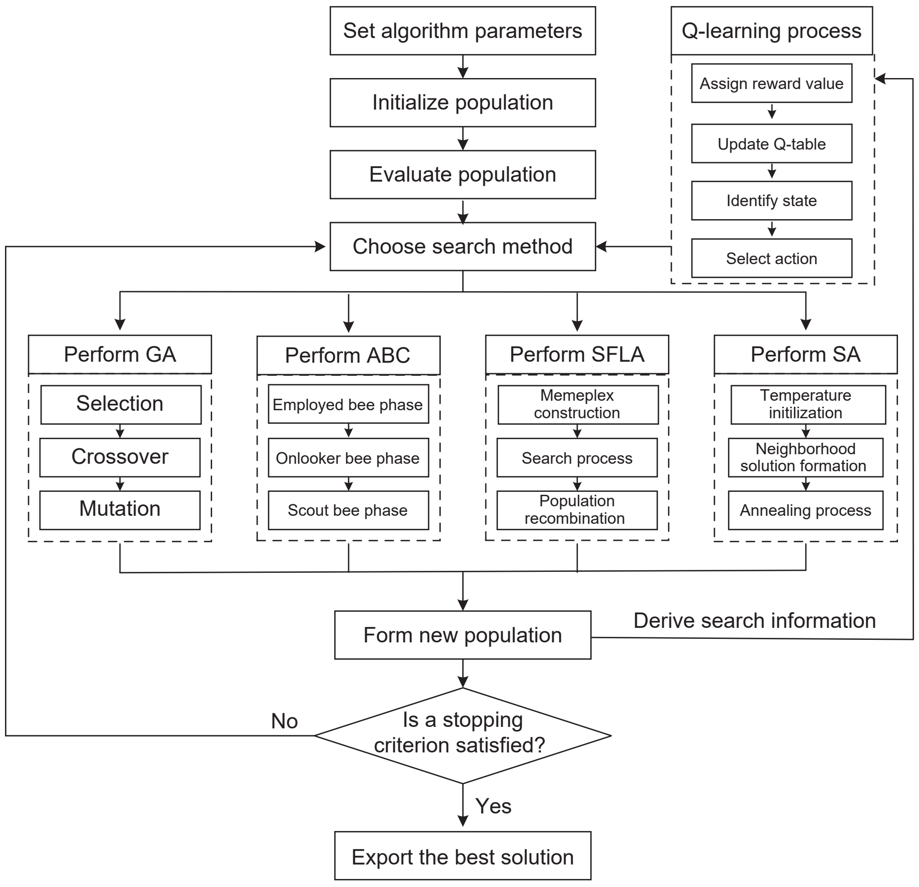


Fig. 4 Flow chart of Q-HMH.

disassembly time of operations is generated at random from an interval $[100, 400]$, the mean setup time of operations is created from $[100, 200]$, and the mean processing time of components at stages is from $[100, 200]$. The standard deviation values of disassembly time, setup time, and processing time are set to their mean time multiplied by 0.0001.

5.2 Parameter tuning

To analyze the influence of user parameters on the Q-HMH, this research does Taguchi experiments[66, 67]. An instance with four products, two disassembly workstations, two reprocessing lines, and three stages in a reprocessing line is employed. All the methods use the quantity of fitness evaluations as stopping criteria equaling to $30 \cdot P \cdot H$, in which P denotes the quantity of products, and H indicates the maximum quantity of operations of all products.

Q-HMH includes five important parameters, i.e., population size F, memeplex quantity η, annealing coefficient ρ, mutation probability κ_m, and number of food sources limit κ_l'. Each of them has four levels: $F \in \{20, 40, 60, 80\}$, $\eta \in \{2, 3, 4, 5\}$, $\rho \in \{0.75, 0.80, 0.85, 0.90\}$, $\kappa_m \in \{0.05, 0.10, 0.15, 0.20\}$, and $\kappa_l' \in \{20, 40, 60, 80\}$. Hence, an orthogonal array $L_{16}\left(4^5\right)$ comprising of 16 parameter combinations is employed as provided in Table 3. Q-HMH with each combination performs 20 runs, and the average objective value is computed as an average response variable (ARV) value. In the Q-learning approach, all elements in the Q-table are initially set to 0, and ϵ is set to 0.2. In addition, μ and β in the SFLA are severally set to 3 and 4. Table 3 provides the final outcome, and the significance rank of parameters and corresponding influence tendency are provided in Table 4 and Fig. 5, respectively. It is found that η and κ_m have the most significant and the second roles since they largely affect Q-HMH's search abilities. Besides, κ_l', F, and ρ severally rank the third, fourth, and fifth. It is seen that Q-HMH with $F = 60$, $\eta = 4$, $\rho = 0.85$, $\kappa_m = 0.20$, and $\kappa_l' = 40$ can achieve better results. Thus, Q-HMH adopts them in the following.

To perform comparisons between Q-HMH and its rivals impartially, the Taguchi experiments are also conducted on the peer methods by using the same instance with Q-HMH.

Table 3 Orthogonal table and ARV values of parameters in Q-HMH.

No.	Factor level					ARV
	F	η	ρ	κ_m	κ_l'	
1	20	2	0.75	0.05	20	5768.69
2	20	3	0.80	0.10	40	5739.61
3	20	4	0.85	0.15	60	5706.93
4	20	5	0.90	0.20	80	5705.35
5	40	2	0.80	0.15	80	5739.15
6	40	3	0.75	0.20	60	5703.26
7	40	4	0.90	0.05	40	5684.30
8	40	5	0.85	0.10	20	5696.18
9	60	2	0.85	0.20	40	5676.94
10	60	3	0.90	0.15	20	5689.52
11	60	4	0.75	0.10	80	5700.97
12	60	5	0.80	0.05	60	5736.55
13	80	2	0.90	0.10	60	5738.03
14	80	3	0.85	0.05	80	5719.99
15	80	4	0.80	0.20	20	5692.73
16	80	5	0.75	0.15	40	5659.94

Table 4 Response and rank of parameters in Q-HMH.

Level	F	η	ρ	κ_m	κ_l'
1	5730	5731	5708	5727	5712
2	5706	5713	5727	5719	5690
3	5701	5696	5700	5699	5721
4	5703	5700	5704	5695	5716
Delta	29	34	27	33	31
Rank	4	1	5	2	3

GA possesses three parameters: population size ζ_N, crossover probability ι_c, and mutation probability κ_m'. Each parameter has four levels: $\zeta_N \in \{20,40,60,80\}$, $\iota_c \in \{0.6,0.7,0.8,0.9\}$, and $\kappa_m' \in \{0.05,0.1,0.15,0.2\}$. All the combinations and corresponding outcome are provided in Table 5. Furthermore, Table 6 and Fig. 6 reveal the significance rank and influence trend of parameters, respectively. By analyzing the results, we discover that GA is able to gain more excellent outcome when $\zeta_N = 40$, $\iota_c = 0.8$, and $\kappa_m' = 0.20$. They will be employed in the following.

ABC contains two key parameters, i.e., population size ζ_N' and number of food sources limit κ_l'. They have four levels: $\zeta_N' \in \{20,40,60,80\}$ and $\kappa_l' \in \{20,40,60,80\}$. All the parameter combinations and corresponding ARV values are provided in Table 7. Through dissecting the outcome, we detect that ABC achieves a more superior result when ζ_N' and κ_l' are set to 40 and 80, respectively. They will be employed in the following experiments.

In EEO, there are four parameters: population size ζ_q'', the quantity of particles in equilibrium pool ι_u'', local-best search probability κ_m'', and number of local-best search iterations ϑ. Each of them has four levels: $\zeta_q'' \in \{20,40,60,80\}$, $\iota_u'' \in \{3,5,7,9\}$, $\kappa_m'' \in \{0.05,0.1,0.15,0.2\}$, and $\vartheta \in \{20,40,60,80\}$. Accordingly, an orthogonal experiment $L_{16}\left(4^4\right)$ is applied, and Table 8 displays the ARV values of parameter combinations. The significance rank of parameters and associated influence trend are furnished in Table 9 and Fig. 7,

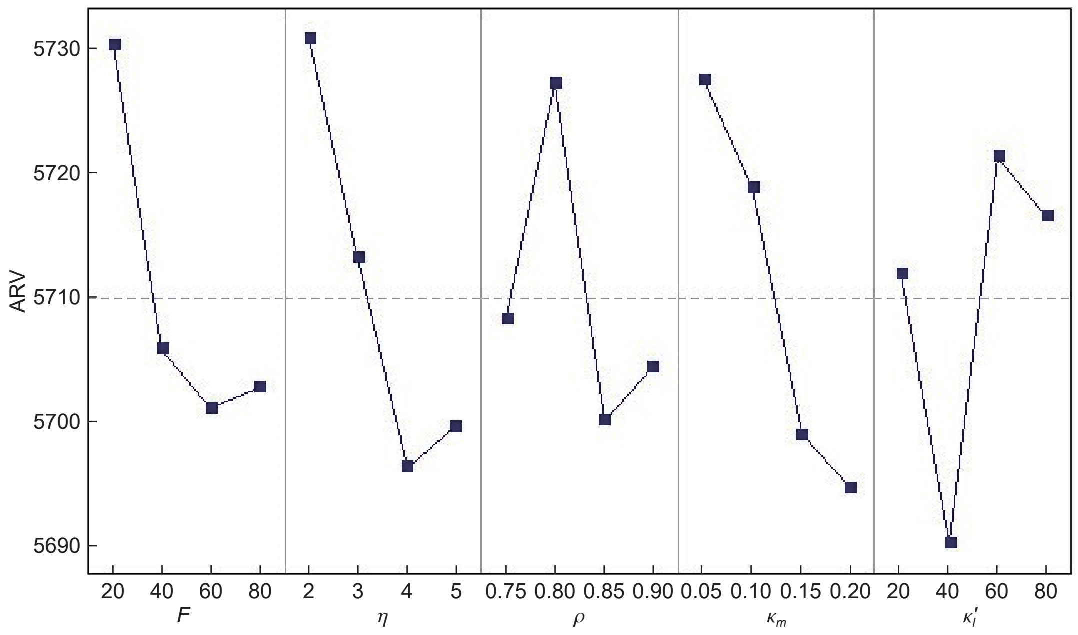


Fig. 5 Influence trends of parameters in Q-HMH.

Table 5 Orthogonal table and ARV values of parameters regarding GA.

No.	Factor level			ARV
	ζ_N	ι_c	κ'_m	
1	20	0.6	0.05	6038.65
2	20	0.7	0.10	5942.80
3	20	0.8	0.15	5907.02
4	20	0.9	0.20	5918.87
5	40	0.6	0.10	5916.69
6	40	0.7	0.05	5977.51
7	40	0.8	0.20	5927.34
8	40	0.9	0.15	5946.53
9	60	0.6	0.15	5929.69
10	60	0.7	0.20	5963.60
11	60	0.8	0.05	5940.92
12	60	0.9	0.10	5978.65
13	80	0.6	0.20	5949.35
14	80	0.7	0.15	5980.21
15	80	0.8	0.10	5954.99
16	80	0.9	0.05	5989.01

Table 6 Response and rank of parameters regarding GA.

Level	ζ_N	ι_c	κ'_m
1	5952	5959	5987
2	5942	5966	5948
3	5953	5933	5941
4	5968	5958	5940
Delta	26	33	47
Rank	3	2	1

respectively. On the basis of the afore-mentioned outcome, a favorable parameter setting for EEO is suggested as: $\zeta''_q = 60$, $\iota''_u = 5$, $\kappa''_m = 0.05$, $\vartheta = 80$. EEO will use it in the following experiments.

5.3 Effectiveness of Q-learning method

To validate the effects of the Q-learning approach in Q-HMH, this work develops a variation of Q-HMH which chooses actions at random (recorded as R-HMH) instead of using the Q-learning method. We conduct comparison experiments between Q-HMH and R-HMH on the employed instances, and each one is solved 20 times by using Q-HMH and its rival. In the following, we analyze the comparison results of Q-HMH and the rival by adopting a relative percentage deviation (RPD). It is calculated as

$$\text{RPD} = \frac{\theta_v - \theta^*}{\theta^*} \tag{30}$$

where θ^* denotes the best outcome acquired by Q-HMH and its rival, and θ_v is the result obtained by an approach in the v-th run. The symbols aRPD, bRPD and sRPD denote the average RPD, the best RPD, and the standard deviation of RPD over 20 runs, respectively. It is noted that an algorithm with the smaller aRPD, bRPD, and sRPD values means the better performance.

Q-HMH and R-HMH take $30 \cdot P \cdot H$ fitness evaluations as stopping conditions. The comparison results of Q-HMH and R-HMH are reported in Table 10. It is found that Q-HMH exhibits better performance than R-HMH in 14 out of 16 instances with respect to aRPD and sRPD metrics. Besides, the average values of aRPD, bRPD, and sRPD of Q-HMH on all the instances are 0.0152, 0.0010, and 0.0088, respectively, while those obtained by R-HMH are 0.0217, 0.0018, and 0.0115, respectively. This indicates that Q-HMH wins R-HMH in solving the instances. The t-test at 0.05 level of significance is applied to further examine

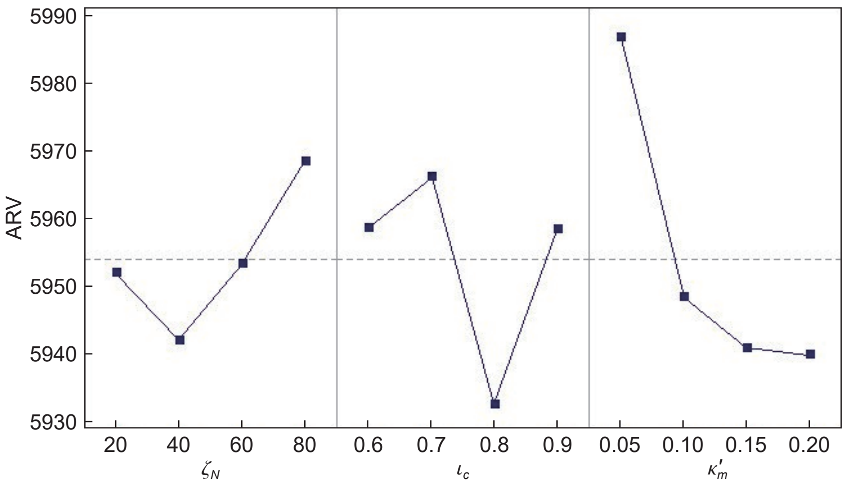


Fig. 6 Influence trends of parameters in GA.

Table 7 Parameter combinations and ARV values of parameters regarding ABC.

No.	Factor level		ARV
	ζ'_N	κ'_l	
1	20	20	5816.42
2	20	40	5776.02
3	20	60	5827.23
4	20	80	5773.88
5	40	20	5732.11
6	40	40	5757.10
7	40	60	5746.63
8	40	80	5755.47
9	60	20	5763.41
10	60	40	5758.01
11	60	60	5757.56
12	60	80	5751.51
13	80	20	5774.89
14	80	40	5770.89
15	80	60	5776.07
16	80	80	5771.22

Table 8 Orthogonal table and ARV values of parameters regarding EEO.

No.	Factor level				ARV
	ζ''_q	ι''_u	κ''_m	ϑ	
1	20	3	0.05	20	5926.38
2	20	5	0.10	40	5931.32
3	20	7	0.15	60	5971.91
4	20	9	0.20	80	5954.06
5	40	3	0.10	60	5943.74
6	40	5	0.05	80	5917.38
7	40	7	0.20	20	5933.72
8	40	9	0.15	40	5931.65
9	60	3	0.15	80	5908.90
10	60	5	0.20	60	5912.33
11	60	7	0.05	40	5926.09
12	60	9	0.10	20	5929.07
13	80	3	0.20	40	5933.93
14	80	5	0.15	20	5941.32
15	80	7	0.10	80	5934.65
16	80	9	0.05	60	5923.85

Q-HMH and R-HMH. The outcome is marked as "+", "–", and "~" when Q-HMH is significantly superior to, significantly inferior to, and statistically equivalent to R-HMH. By detecting the t-test results, we discover that Q-HMH significantly exceeds R-HMH in 12 out of 16 instances, R-HMH is significantly better than Q-HMH in only one instance, and the rest are statistically equivalent. Through dissecting the comparison outcome, we find out that Q-HMH reveals more superior results than R-HMH. Namely, the Q-learning method plays essential roles in reinforcing the performance of Q-HMH for solving the problem under consideration.

Table 9 Response and rank of parameters regarding EEO.

Level	ζ''_q	ι''_u	κ''_m	ϑ
1	5946	5928	5923	5933
2	5932	5926	5935	5931
3	5919	5942	5938	5938
4	5933	5935	5934	5929
Delta	27	16	15	9
Rank	1	2	3	4

To reveal the search process of Q-HMH and R-HMH visually, we equally divide the entire search process into 20 phases according to the quantity of fitness evaluations, and then the ratio of using GA, ABC, SFLA, and SA at each phase across 20 times is illustrated in Fig. 8. It is seen that R-HMH chooses the four methods in a random way and their selected percentages are very similar since it does not employ any valuable information. Looking at the results of Q-HMH, we can find that the selected percentages of four methods are diverse since the Q-learning method fully uses the search information to choose a befitting method. In the search process, GA and SA are frequently selected to update the population, followed by ABC and SFLA. In accordance with the above outcome and dissection, we verdict that the Q-learning methods have essential roles in improving the performance of Q-HMH.

5.4 Comparison of Q-HMH and CPLEX

In order to verify the abilities of Q-HMH in searching optimal solutions, this study makes comparisons between Q-HMH and CPLEX in addressing the studied problem. All the employed instances are transformed into deterministic circumstances by taking the mean values as regards disassembly and setup time of operations and processing time of components. CPLEX takes the maximum running time equaling to 3 h as a suspensive condition. Namely, CPLEX is forced to stop when the given computation time exhausts. Thus, it gains an approximately optimal value (AOV) like Q-HMH. Q-HMH and CPLEX solve the instances with P products, $P \in \{2,4,6,8\}$. Q-HMH solves each instance 20 runs, and the average is computed for comparisons. The experimental results obtained by Q-HMH and

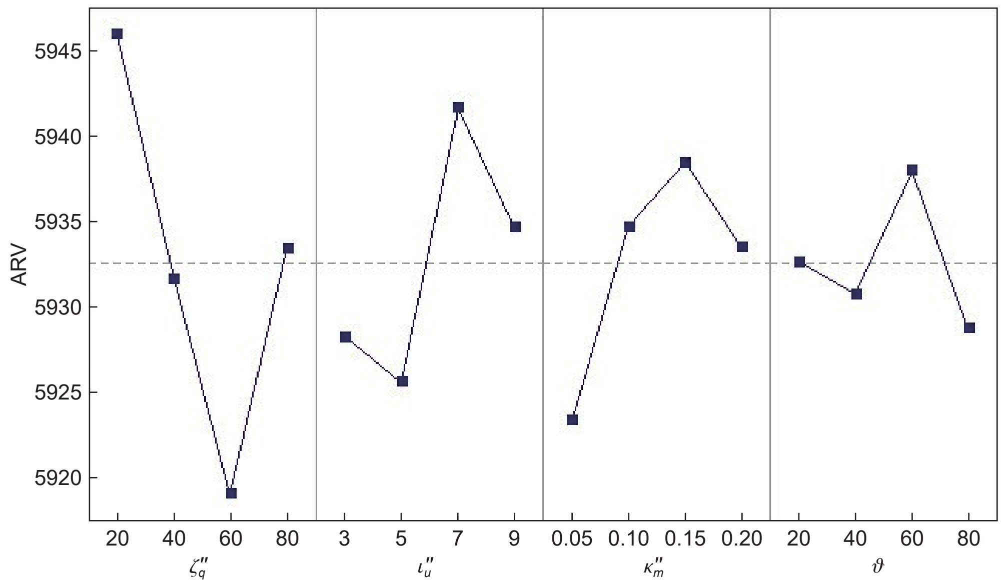


Fig. 7 Influence trends of parameters in EEO.

Table 10 Comparison results of Q-HMH and R-HMH.

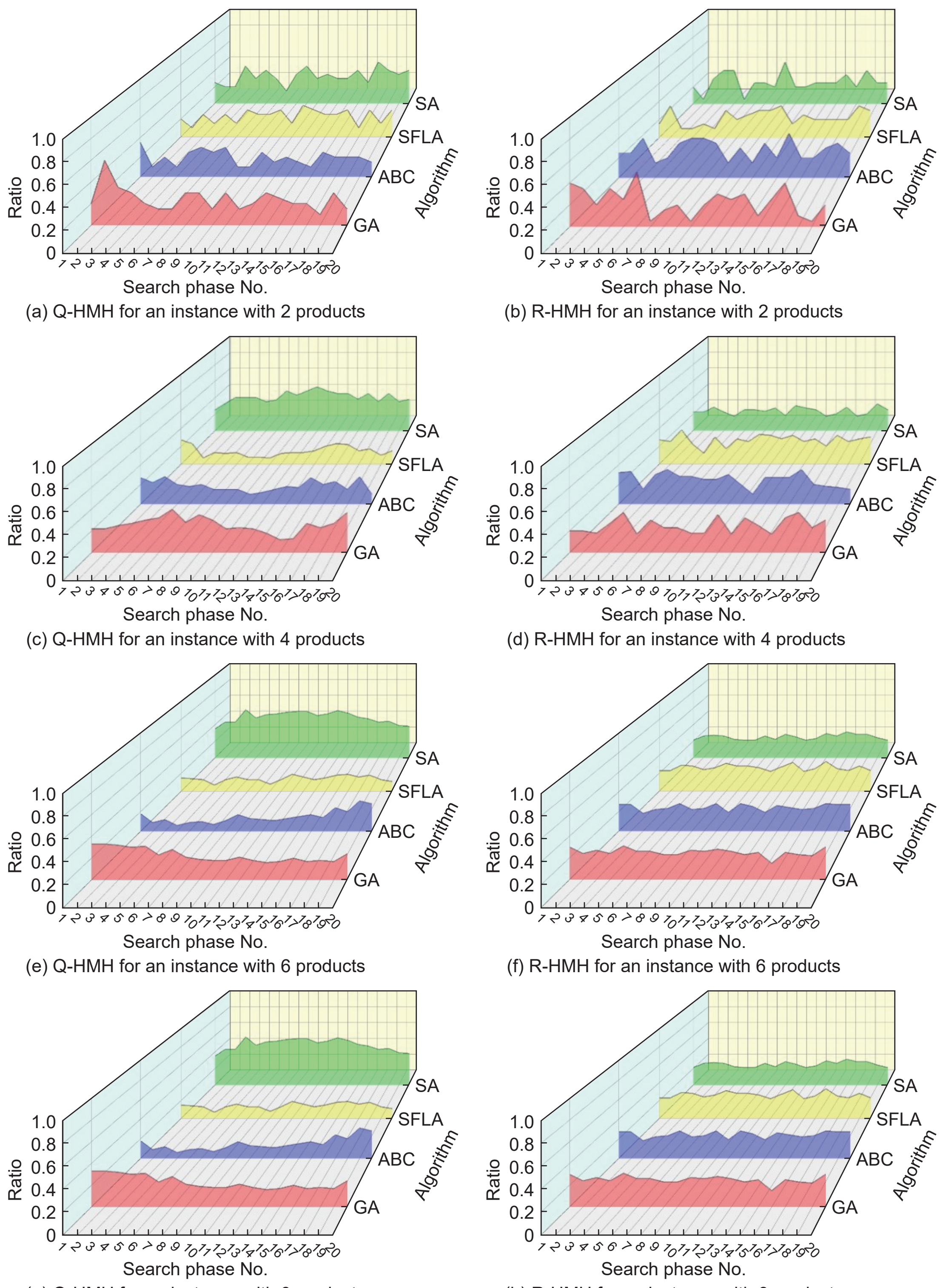
Number of products	Instance No.	Q-HMH			R-HMH			t-test
		aRPD	bRPD	sRPD	aRPD	bRPD	sRPD	
2	1	**0.0094**	**0.0000**	**0.0068**	0.0172	0.0059	0.0093	+
	2	**0.0078**	**0.0000**	**0.0061**	0.0139	0.0012	0.0097	+
	3	**0.0189**	0.0000	0.0156	0.0341	0.0000	**0.0151**	+
	4	0.0022	0.0000	**0.0019**	**0.0017**	0.0000	0.0020	~
4	1	**0.0090**	0.0000	**0.0084**	0.0174	0.0000	0.0091	+
	2	**0.0245**	0.0010	**0.0134**	0.0349	**0.0000**	0.0156	+
	3	**0.0251**	**0.0000**	0.0171	0.0331	0.0036	**0.0166**	~
	4	**0.0057**	0.0000	**0.0061**	0.0132	0.0000	0.0119	+
6	1	**0.0198**	**0.0000**	**0.0104**	0.0303	0.0018	0.0149	+
	2	**0.0163**	**0.0000**	**0.0088**	0.0248	0.0025	0.0108	+
	3	0.0379	0.0136	**0.0121**	**0.0266**	**0.0000**	0.0140	−
	4	**0.0187**	**0.0000**	**0.0094**	0.0286	0.0102	0.0114	+
8	1	**0.0241**	0.0015	**0.0100**	0.0359	**0.0000**	0.0190	+
	2	**0.0025**	**0.0000**	**0.0018**	0.0055	0.0003	0.0057	+
	3	**0.0124**	0.0001	**0.0071**	0.0125	**0.0000**	0.0096	~
	4	**0.0083**	**0.0000**	**0.0060**	0.0173	0.0033	0.0094	+
Average		**0.0152**	**0.0010**	**0.0088**	0.0217	0.0018	0.0115	

CPLEX are reported in Table 11. This research calculates the gap of Q-HMH and CPLEX to assess their difference degree. It is calculated as Eq. (31).

$$\text{gap} = \frac{\psi - \psi_Q}{\psi} \times 100\% \tag{31}$$

where ψ_Q and ψ denote the results found by Q-HMH and CPLEX, respectively. A larger gap means that the solution obtained by Q-HMH is much better.

It is seen that Q-HMH and CPLEX can reach the optimal solution for the instance with two products, while CPLEX achieves approximately optimal solutions for the instances with more than two products. In addition, compared with CPLEX, Q-HMH can get better results for the instances with more than two products by spending less time. It is noted that their gap becomes larger as the problem size increases. Via dissecting the comparison outcome, we declare that Q-HMH exhibits evident advantage over CPLEX in addressing the studied problem, especially for the large-size problems.



(a) Q-HMH for an instance with 2 products
(b) R-HMH for an instance with 2 products
(c) Q-HMH for an instance with 4 products
(d) R-HMH for an instance with 4 products
(e) Q-HMH for an instance with 6 products
(f) R-HMH for an instance with 6 products
(g) Q-HMH for an instance with 8 products
(h) R-HMH for an instance with 8 products

Fig. 8 Selected percentage of search methods for Q-HMH and R-HMH.

5.5 Comparison of Q-HMH and peer methods

This work carries out experiments to verify the capacities of Q-HMH and its competitive methods, i.e., GA[65], ABC[58], and EEO[29]. GA and ABC are two well-known meta-heuristics, and a great many of studies give them positive evaluations in solving disassembly optimization problems[58, 65]. EEO is a recently-published work for solving multi-product DSPPs, and extensive comparison experiments verify

Table 11 Comparison results of Q-HMH and CPLEX.

Number of products	AOV		Time of Q-HMH (s)	Gap (%)
	CPLEX	Q-HMH		
2	1023◎	1023◎	0.9901	0.0000
4	1352	1283	3.4892	5.1036
6	1801	1659	8.2020	7.8845
8	2351	2100	14.0501	10.6763

Note: The symbol "◎" signifies that the corresponding result is optimal.

its excellent performance in addressing such problems. Besides, ABC and EEO use the same solution representation methods with Q-HMH like their original studies[29, 58], which can be straightforwardly extended to solve the considered problem. They are extended to solve the studied problem by selecting the earliest available reprocessing line for arrival components as Q-HMH. Therefore, this work selects the three meta-heuristics for comparisons. The details of extending GA, ABC, and EEO for solving the considered problem are given as follows:

(1) GA uses the same solution representation as Q-HMH. It contains three stages, i.e., selection, crossover, and mutation. The crossover and mutation operations are the same with Q-HMH, and the selection and population update are identical to GA in the original literature[65].

(2) ABC also employs the same solution representation approach like Q-HMH. At the employed bee phase, the crossover operation of employed bees is the same with Q-HMH. At the onlooker bee phase, a tournament selection method[58] is applied to choose initial onlooker bees, and then a neighborhood method, which is the same with ABC[58], is used. At the scout bee phase, new individuals are generated by the tournament selection method, and the predetermined limit is equal to 0.

(3) EEO adopts the same solution representation approach like Q-HMH as well. Its equilibrium pool construction, concentrations of particles update, and particle's memory saving are the same with EEO in the original literature[29]. Additionally, it uses the same crossover operations like Q-HMH and the local-best search approach in EEO[29].

Furthermore, the Taguchi experiments[66, 67] are also adopted to determine the user parameters of three competitive methods. Q-HMH and the rivals use the same number of fitness evaluations equaling to $30 \cdot P \cdot H$ as stopping conditions. Q-HMH and the rivals address each instance 20 times, and the aRPD, bRPD, and sRPD values across 20 times are calculated to dissect their outcome.

Table 12 reports the optimization results of Q-HMH and its peers respecting aRPD, bRPD, and sRPD. The outcome regarding aRPD reveals that Q-HMH surpasses ABC in 13 out of 16 instances. It is remarkable that Q-HMH has smaller aRPD values than GA and EEO in handling all instances. Besides, the

Table 12 Comparison outcomes of Q-HMH and the peer algorithms.

Number of products	Instance No.	Q-HMH			GA			ABC			EEO		
		aRPD	bRPD	sRPD	aRPD	bRPD	sRPD	aRPD	bRPD	sRPD	aRPD	bRPD	sRPD
2	1	**0.0094**	**0.0000**	**0.0068**	0.0594	0.0433	0.0140	0.0417	0.0244	0.0092	0.0463	0.0300	0.0084
	2	**0.0078**	**0.0000**	**0.0061**	0.0421	0.0226	0.0068	0.0321	0.0203	0.0076	0.0315	0.0205	0.0076
	3	**0.0192**	0.0003	0.0156	0.0667	0.0518	**0.0070**	0.0490	**0.0000**	0.0130	0.0517	0.0326	0.0123
	4	0.0022	0.0000	**0.0019**	0.0023	0.0000	0.0024	**0.0016**	0.0000	0.0020	0.0036	0.0004	0.0020
4	1	**0.0090**	**0.0000**	0.0084	0.0367	0.0080	0.0186	0.0188	0.0036	0.0087	0.0287	0.0135	**0.0072**
	2	**0.0235**	**0.0000**	**0.0134**	0.0748	0.0532	0.0173	0.0514	0.0260	0.0136	0.0791	0.0380	0.0138
	3	0.0251	**0.0000**	0.0171	0.0467	0.0179	0.0139	**0.0234**	0.0142	0.0106	0.0359	0.0178	**0.0104**
	4	**0.0057**	**0.0000**	**0.0061**	0.0398	0.0163	0.0164	0.0187	0.0027	0.0121	0.0399	0.0269	0.0092
6	1	**0.0198**	**0.0000**	**0.0104**	0.0624	0.0280	0.0143	0.0305	0.0159	0.0111	0.0693	0.0313	0.0124
	2	**0.0163**	**0.0000**	**0.0088**	0.0708	0.0430	0.0151	0.0344	0.0162	0.0124	0.0712	0.0430	0.0097
	3	0.0240	**0.0000**	0.0119	0.0736	0.0419	0.0137	**0.0231**	0.0076	0.0114	0.0850	0.0653	**0.0085**
	4	**0.0187**	**0.0000**	**0.0094**	0.0536	0.0236	0.0182	0.0317	0.0051	0.0174	0.0584	0.0413	0.0099
8	1	**0.0226**	**0.0000**	**0.0099**	0.1061	0.0586	0.0181	0.0458	0.0173	0.0154	0.1069	0.0234	0.0213
	2	**0.0025**	**0.0000**	**0.0018**	0.0296	0.0118	0.0144	0.0093	0.0030	0.0081	0.0229	0.0127	0.0076
	3	**0.0123**	**0.0000**	0.0071	0.0320	0.0103	0.0095	0.0162	0.0015	0.0083	0.0302	0.0221	**0.0040**
	4	**0.0083**	**0.0000**	0.0060	0.0391	0.0209	0.0088	0.0217	0.0101	0.0080	0.0311	0.0195	**0.0042**
Average		**0.0142**	**0.0000**	**0.0088**	0.0522	0.0282	0.0130	0.0281	0.0105	0.0106	0.0495	0.0274	0.0093

average aRPD values of Q-HMH, GA, ABC, and EEO in all the instances are 0.0142, 0.0522, 0.0281, and 0.0495, respectively. A relative percentage increase ($\mathcal{R}$), defined in Eq. (32), is adopted to further analyze the experiment results. It is found that Q-HMH exceeds GA, ABC, and EEO by 26.76%, 9.79%, and 24.56% on average, respectively. Accordingly, it is observed that Q-HMH produces much smaller results than ABC in 14 out of 16 instances with respect to bRPD, and it performs better than GA and EEO in all the instances. The average bRPD values of Q-HMH, GA, ABC, and EEO are severally equal to 0.0000, 0.0282, 0.0105, and 0.0274, respectively. Moreover, by computing the sRPD values on all instances, we find that Q-HMH severally exceeds GA, ABC, and EEO in 14, 13, and 10 instances. Furthermore, the average sRPD values of Q-HMH, GA, ABC, and EEO are 0.0088, 0.0130, 0.0106, and 0.0093, respectively. Concerning the relative percentage increase on average, Q-HMH outperforms GA, ABC, and EEO by 4.77%, 2.05%, and 0.57%, respectively, implying that Q-HMH is much steadier in working out this problem. By analyzing the comparison outcome, we can declare that Q-HMH defeats its rivals in figuring out the investigated problem.

$$\mathcal{R} = \frac{\varphi_A - \varphi_A^*}{\varphi_A^* \times 10} \times 100\% \tag{32}$$

where φ_A^* and φ_A denote the average values of Q-HMH and its rivals on all the instances regarding RPD metrics, respectively.

Moreover, to reveal the comparison outcome intuitively, Fig. 9 diagrams the boxplots of Q-HMH and its peers in dealing with the instances with diverse number of products. It is observed that Q-HMH is more concentrated and more stable than its competitors. Figure 10 illustrates the results of instances with different number of products in terms of the average aRPD. It is found that Q-HMH achieves the smaller results than the rivals on the instances with different number of products. Furthermore, we calculate the confidence interval with a 95% confidence interval in terms of aRPD values, and Fig. 11 illustrates the confidence intervals of Q-HMH and the peers in addressing some instances with different products. The confidence intervals of Q-HMH do not overlap with those of GA, ABC, and EEO in most of the instances, implying that there are significant differences between Q-HMH and the rivals. Meanwhile, Q-HMH has narrower confidence interval in most of the instances than the peers, indicating that the estimates are more

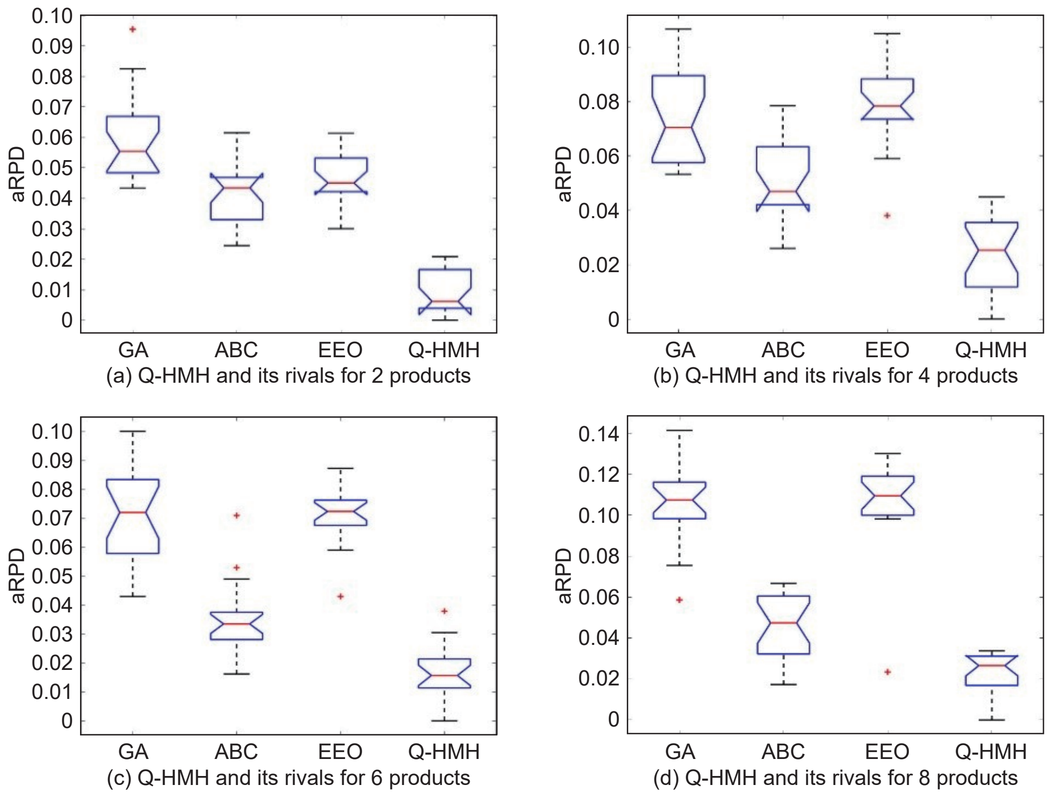


Fig. 9 Boxplot graphs of the aRPD values by four methods in solving different instances.

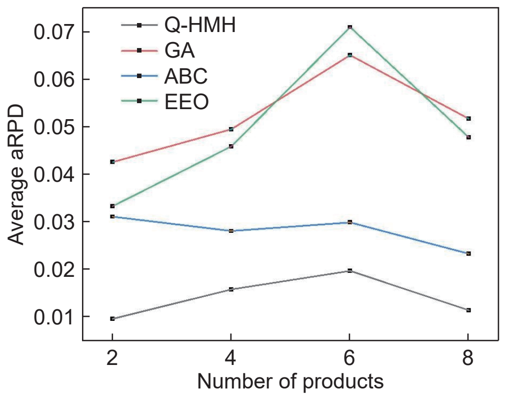


Fig. 10 Illustration of the results of instances with different number of products in terms of the average aRPD.

precise. In consequence, we are able to declare that Q-HMH has excellent performance in solving the considered problem.

To clearly dissect the experimental results, we further use the Friedman test[68], Nemenyi post-hoc test[69], and Wilcoxon signed-rank test[70] to sort Q-HMH and its peers. The concrete dissection of statistical results is furnished as:

(1) The Friedman test is performed at 0.05 level of significance. All the methods are assigned to order values 1, 2, 3, and 4 in accordance with aRPD values. Afterwards, the average order values across the used instances are calculated as displayed in Table 13. Looking at the obtained optimization results, we can conclude that Q-HMH and the rivals are statistically discrepant since they have different average order values.

(2) The Nemenyi post-hoc test at 0.05 level of significance is adopted to dissect the discrepancy of average order values. The differences of Q-HMH, GA, and EEO as regards average order values are 2.3125 and 2.2500. By comparing with a critical range value 1.1726, the results imply that Q-HMH is significantly better than GA and EEO. Besides, the difference of Q-HMH and ABC is 0.6875 (smaller than 1.1726), signifying that their capacities are statistically equivalent.

(3) The t-test at 0.05 level of significance is used to further examine Q-HMH and its competitors in figuring out the employed instances. In Table 13, the outcomes are marked as "+", "–", and "~" in the bracket when Q-HMH is significantly superior to, significantly inferior to, and statistically equivalent to its peers, respectively. It is found that Q-HMH exhibits significantly better performance than GA, ABC, and EEO on 15, 12, and 16 instances, respectively, and they are statistically equivalent for the rest.

(4) The results of the Wilcoxon signed-rank test are shown in Table 14. The symbols "w^+", "w^-", and "$w^\sim$" denote that Q-HMH is superior to, inferior to, and equivalent to its competitors, respectively. "R_s^+" indicates the sum of ranks that the first algorithm outperforms the second, and "R_s^-" is calculated for the opposite. It is observed that Q-HMH exhibits better performance than its rivals respecting aRPD, bRPD, and sRPD. The statistical results verify that Q-HMH is a better method, and it can gain positive acceptance in addressing the considered issue.

Furthermore, we collect the computation time of Q-HMH and the competitors in solving the instances, and then we calculate the average computation time of the instances having the identical quantity of products. Table 15 reports the experimental outcome. By observing these results, we can see that Q-HMH spends less computation time to reach the above superior performance. Thus, Q-HMH has higher computation efficiency.

6 Conclusion and Future Work

This study addresses an integrated scheduling problem of disassembly and reprocessing processes considering product structures with random disassembly and processing time. A stochastic programming model is established to realize minimal expected makespan. A Q-learning based hybrid meta-heuristic method is accordingly devised with consideration of the problem's characteristics. The formulated model is validated, and the strong competitiveness of the proposed algorithm is verified. This research provides a high-efficiency approach to solve integrated disassembly and reprocessing scheduling problems in uncertain circumstances when the product structures need to be considered. The achievements can help managers and practicers to make much better and credible decisions for reference.

In the future, we will consider the subsequent directions: (1) formulize the integrated scheduling models of disassembly, reprocessing, and reassembly with consideration of product structures and uncertainties; (2) design effective meta-heuristics incorporating multi-task optimization and digital twin methods to handle the formulated problems[71, 72].

Acknowledgment

This work was in part supported by the Science and

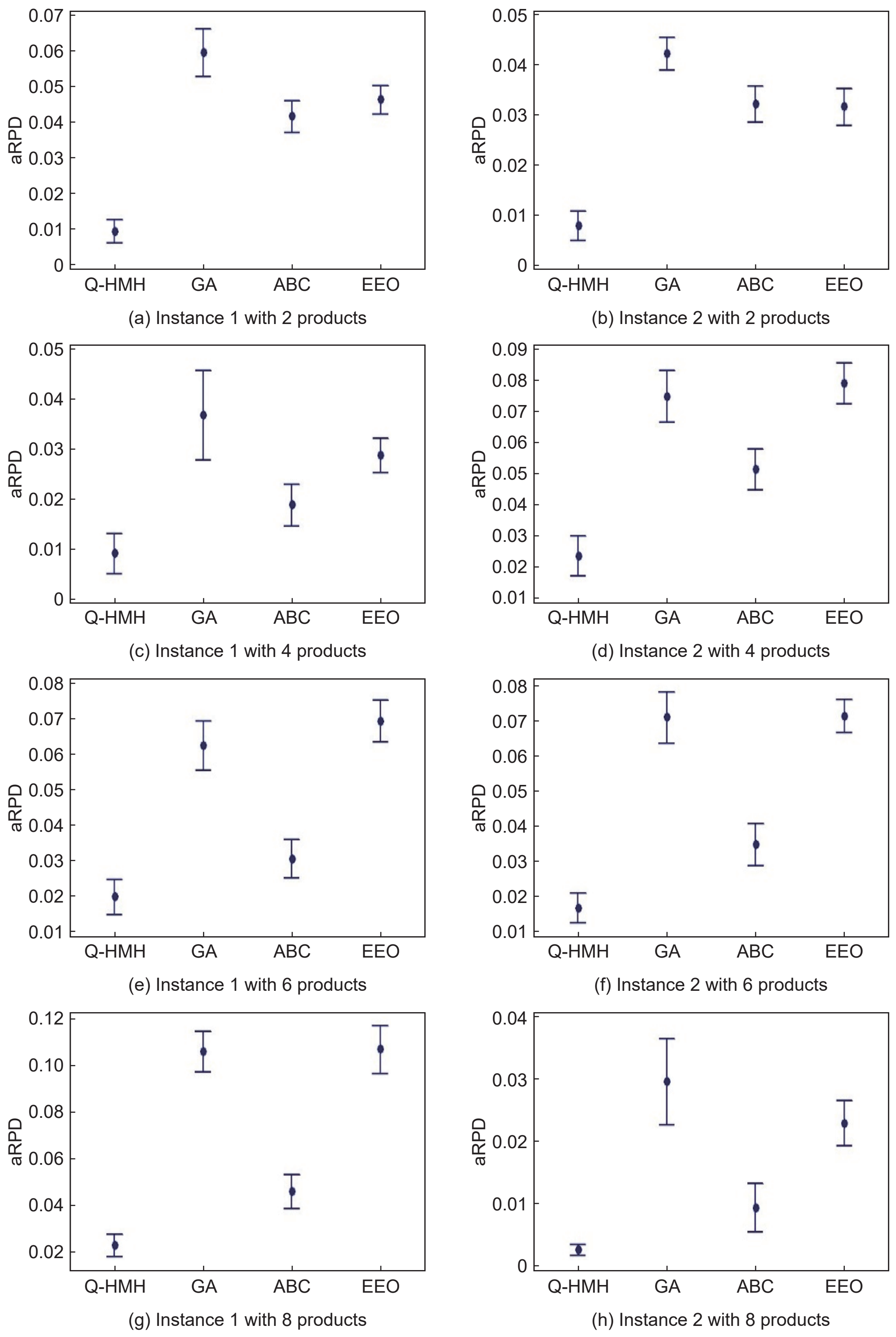


Fig. 11 Confidence interval graphs of aRPD values found by four algorithms in solving different instances.

Technology Development Fund (FDCT), Macau SAR, (No. 0019/2021/A), Shandong Province Outstanding Youth Innovation Team Project of Colleges and Universities (No. 2020RWG011), National Natural Science Foundation of China (Nos. 62173356 and 61703320), Natural Science Foundation of Shandong Province (No. ZR202111110025), Guangdong Basic and Applied Basic Research Foundation (No.

Table 13 Statistic outcomes of Q-HMH and the rivals.

Number of products	Instance No.	Rank (*t*-test)			
		GA	ABC	EEO	Q-HMH
2	1	4 (+)	2 (+)	3 (+)	1
	2	4 (+)	3 (+)	2 (+)	1
	3	4 (+)	2 (+)	3 (+)	1
	4	3 (~)	1 (~)	4 (+)	2
4	1	4 (+)	2 (+)	3 (+)	1
	2	3 (+)	2 (+)	4 (+)	1
	3	4 (+)	1 (~)	3 (+)	2
	4	3 (+)	2 (+)	4 (+)	1
6	1	3 (+)	2 (+)	4 (+)	1
	2	3 (+)	2 (+)	4 (+)	1
	3	3 (+)	1 (~)	4 (+)	2
	4	3 (+)	2 (+)	4 (+)	1
8	1	3 (+)	2 (+)	4 (+)	1
	2	4 (+)	2 (+)	3 (+)	1
	3	4 (+)	2 (~)	3 (+)	1
	4	4 (+)	2(+)	3 (+)	1
Average		3.5000	1.8750	3.4375	1.1875

Note: The outcomes are marked as "+", "−", and "~" in the bracket when Q-HMH is significantly superior to, significantly inferior to, and statistically equivalent to its peers, respectively.

2023A1515011531), and Zhuhai Industry-University-Research Project with Hongkong and Macao (No. ZH22017002210014PWC).

References

[1] J. Chen, H. Wang, and Y. Fu, A multi-stage supply chain disruption mitigation strategy considering product life cycle during COVID-19, *Environ. Sci. Pollut. Res.*, doi: 10.1007/s11356-022-18931-7.

[2] K. Li, L. Zhang, H. Fu, and B. Liu, The effect of intelligent manufacturing on remanufacturing decisions, *Comput. Ind. Eng.*, vol. 178, p. 109114, 2023.

[3] P. Liang, Y. Fu, and K. Gao, Multi-product disassembly line balancing optimization method for high disassembly profit and low energy consumption with noise pollution constraints, *Eng. Appl. Artif. Intell.*, vol. 130, p. 107721, 2024.

[4] C. Liu, X. Meng, C. Liu, and Z. Liu, Carbon footprint-based optimization method for remanufacturing machining paths, *Int. J. Adv. Manuf. Technol.*, vol. 124, no. 10, pp. 3391–3406, 2023.

[5] W. Wang, G. Tian, H. Zhang, K. Xu, and Z. Miao, Modeling and scheduling for remanufacturing systems with disassembly, reprocessing, and reassembly considering total energy consumption, *Environ. Sci. Pollut. Res. Int.*, doi: 10.1007/s11356-021-17292-x.

[6] W. Zhang, J. Shi, S. Zhang, and M. Chen, Scenario-based robust remanufacturing scheduling problem using improved biogeography-based optimization algorithm, *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 53, no. 6, pp. 3414–3427, 2023.

[7] Y. Fu, G. Tian, A. M. Fathollahi-Fard, A. Ahmadi, and C. Zhang, Stochastic multi-objective modelling and optimization of an energy-conscious distributed permutation flow shop scheduling problem with the total tardiness constraint, *J. Clean. Prod.*, vol. 226, pp. 515–525, 2019.

[8] J. Li, X. Li, and L. Gao, An operator-inspired framework for metaheuristics and its applications on job-shop scheduling problems, *Appl. Soft Comput.*, vol. 157, p.

Table 14 Results of Wilcoxon signed-rank test regarding the four algorithms.

Algorithm	Metric	$w^+/w^-/w^\sim$	R_s^+	R_s^-	p-value	$\alpha = 0.05$
Q-HMH vs. GA	aRPD	16/0/0	136.0	0.0	0.0000	Yes
	bRPD	15/0/1	120.0	0.0	0.0006	Yes
	sRPD	14/2/0	118.0	18.0	0.0097	Yes
Q-HMH vs. ABC	aRPD	13/3/0	130.0	6.0	0.0013	Yes
	bRPD	14/1/1	119.0	1.0	0.0008	Yes
	sRPD	13/3/0	107.0	29.0	0.0437	Yes
Q-HMH vs. EEO	aRPD	16/0/0	136.0	0.0	0.0000	Yes
	bRPD	16/0/0	136.0	0.0	0.0000	Yes
	sRPD	10/6/0	72.5	63.5	0.8159	No

Table 15 Computation time of Q-HMH and the rivals.

Number of products	Average computational time (s)			
	Q-HMH	GA	ABC	EEO
2	9210	9702	9908	10116
4	37580	39441	40248	41272
6	101319	104946	106582	109427
8	168359	175770	178761	182793
Average	79117	82464.75	83874.75	85902

111522, 2024.

[9] Y. P. Fu, G. D. Tian, Z. W. Li, and Z. L. Wang, Parallel machine scheduling with dynamic resource allocation via a master–slave genetic algorithm, *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 13, no. 5, pp. 748–756, 2018.

[10] G. Tian, X. Zhang, A. M. Fathollahi-Fard, Z. Jiang, C. Zhang, G. Yuan, and D. T. Pham, Hybrid evolutionary algorithm for stochastic multiobjective disassembly line balancing problem in remanufacturing, *Environ. Sci. Pollut. Res. Int.*, doi: 10.1007/s11356-023-27081-3.

[11] J. M. Yu, J. S. Kim, and D. H. Lee, Scheduling algorithms to minimise the total family flow time for job shops with job families, *Int. J. Prod. Res.*, vol. 49, no. 22, pp. 6885–6903, 2011.

[12] Y. Xiao, J. Zhou, S. Xing, and X. Zhu, Research on assembly sequence optimization classification method of remanufacturing parts based on different precision levels, *Processes*, vol. 11, no. 2, p. 383, 2023.

[13] H. H. Doh and D. H. Lee, Integrated disassembly and reprocessing lot-sizing for multi-level structured products in remanufacturing systems, *Eng. Optim.*, vol. 54, no. 9, pp. 1476–1493, 2022.

[14] Y. Fu, M. Zhou, X. Guo, and L. Qi, Stochastic multi-objective integrated disassembly-reprocessing-reassembly scheduling via fruit fly optimization algorithm, *J. Clean. Prod.*, vol. 278, p. 123364, 2021.

[15] W. Zhang, Y. Zheng, and R. Ahmad, An energy-efficient multi-objective integrated process planning and scheduling for a flexible job-shop-type remanufacturing system, *Adv. Eng. Inform.*, vol. 56, p. 102010, 2023.

[16] X. Li, N. Li, I. Kolmanovsky, and B. I. Epureanu, Stochastic model predictive control for remanufacturing system management, *J. Manuf. Syst.*, vol. 59, pp. 355–366, 2021.

[17] G. Yuan, Y. Yang, G. Tian, and A. M. Fathollahi-Fard, Capacitated multi-objective disassembly scheduling with fuzzy processing time via a fruit fly optimization algorithm, *Environ. Sci. Pollut. Res.*, doi: 10.1007/s11356-022-18883-y.

[18] Y. Fu, M. Zhou, X. Guo, L. Qi, K. Gao, and A. Albeshri, Multiobjective scheduling of energy-efficient stochastic hybrid open shop with brain storm optimization and simulation evaluation, *IEEE Trans. Syst. Man Cybern. Syst.*, doi: 10.1109/TSMC.2024.3376292.

[19] Y. Lv, C. Li, X. Zhao, L. Li, and J. Li, A novel approach for remanufacturing process planning considering uncertain and fuzzy information, *Front. Mech. Eng.*, vol. 16, no. 3, pp. 546–558, 2021.

[20] W. K. Zhang, Y. F. Zheng, and R. Ahmad, An energy-efficient multi-objective scheduling for flexible job-shop-type remanufacturing system, *J. Manuf. Syst.*, vol. 66, pp. 211–232, 2023.

[21] P. Liang, Y. Fu, S. Ni, and B. Zheng, Modeling and optimization for noise-aversion and energy-awareness disassembly sequence planning problems in reverse supply chain, *Environ. Sci. Pollut. Res. Int.*, doi: 10.1007/s11356-021-14124-w.

[22] Y. Fu, M. Zhou, X. Guo, L. Qi, and K. Sedraoui, Multiverse optimization algorithm for stochastic biobjective disassembly sequence planning subject to operation failures, *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 52, no. 2, pp. 1041–1051, 2022.

[23] D. Yu, X. Zhang, G. Tian, Z. Jiang, Z. Liu, T. Qiang, and C. Zhan, Disassembly sequence planning for green remanufacturing using an improved whale optimisation algorithm, *Processes*, vol. 10, no. 10, p. 1998, 2022.

[24] Y. Gao, S. Lou, H. Zheng, and J. Tan, A data-driven method of selective disassembly planning at end-of-life under uncertainty, *J. Intell. Manuf.*, vol. 34, no. 2, pp. 565–585, 2023.

[25] X. Guo, M. Zhou, S. Liu, and L. Qi, Multiresource-constrained selective disassembly with maximal profit and minimal energy consumption, *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 2, pp. 804–816, 2021.

[26] C. Zhang, A. M. Fathollahi-Fard, J. Li, G. Tian, and T. Zhang, Disassembly sequence planning for intelligent manufacturing using social engineering optimizer, *Symmetry*, vol. 13, no. 4, p. 663, 2021.

[27] M. L. Lee, S. Behdad, X. Liang, and M. Zheng, Task allocation and planning for product disassembly with human–robot collaboration, *Robot. Comput. Integr. Manuf.*, vol. 76, p. 102306, 2022.

[28] L. Guo, Z. Zhang, and X. Zhang, Human–robot collaborative partial destruction disassembly sequence planning method for end-of-life product driven by multi-failures, *Adv. Eng. Inform.*, vol. 55, p. 101821, 2023.

[29] P. Liang, Y. Fu, X. Guo, and L. Qi, Stochastic multi-product disassembly sequence planning, in *Proc. IEEE 17th Int. Conf. Automation Science and Engineering* (*CASE*), Lyon, France, 2021, pp. 1201–1206.

[30] P. Liang, Y. Fu, K. Gao, and H. Sun, An enhanced group teaching optimization algorithm for multi-product disassembly line balancing problems, *Complex Intell. Syst.*, vol. 8, no. 6, pp. 4497–4512, 2022.

[31] X. Guo, Z. Zhang, L. Qi, S. Liu, Y. Tang, and Z. Zhao, Stochastic hybrid discrete grey wolf optimizer for multi-objective disassembly sequencing and line balancing planning in disassembling multiple products, *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 3, pp. 1744–1756, 2022.

[32] X. Liu, F. Chu, F. Zheng, C. Chu, and M. Liu, Distributionally robust and risk-averse optimisation for the stochastic multi-product disassembly line balancing problem with workforce assignment, *Int. J. Prod. Res.*, vol. 60, no. 6, pp. 1973–1991, 2022.

[33] P. Hu, F. Chu, M. Liu, S. J. Wang, and P. Wu, An integrated approach for a new flexible multi-product disassembly line balancing problem, *Comput. Oper. Res.*, vol. 148, p. 105932, 2022.

[34] T. Yin, Z. Zhang, Y. Zhang, T. Wu, and W. Liang, Mixed-integer programming model and hybrid driving algorithm for multi-product partial disassembly line balancing problem with multi-robot workstations, *Robot. Comput. Integr. Manuf.*, vol. 73, p. 102251, 2022.

[35] J. S. Kim, J. H. Park, and D. H. Lee, Iterated greedy algorithms to minimize the total family flow time for job-shop scheduling with job families and sequence-dependent

set-ups, *Eng. Optim.*, vol. 49, no. 10, pp. 1719–1732, 2017.

[36] K. Gao, L. Wang, J. Luo, H. Jiang, A. Sadollah, and Q. Pan, Discrete harmony search algorithm for scheduling and rescheduling the reprocessing problems in remanufacturing: A case study, *Eng. Optim.*, vol. 50, no. 6, pp. 965–981, 2018.

[37] J. Shi, W. Zhang, S. Zhang, and J. Chen, A new bifuzzy optimization method for remanufacturing scheduling using extended discrete particle swarm optimization algorithm, *Comput. Ind. Eng.*, vol. 156, p. 107219, 2021.

[38] X. Liu, J. Chen, X. Huang, S. Guo, S. Zhang, and M. Chen, A new job shop scheduling method for remanufacturing systems using extended artificial bee colony algorithm, *IEEE Access*, vol. 9, pp. 132429–132441, 2021.

[39] W. Li, C. Zhang, C. Liu, and X. Liu, Error propagation model and optimal control method for the quality of remanufacturing assembly, *J. Intell. Fuzzy Syst.*, vol. 42, no. 3, pp. 2533–2547, 2022.

[40] C. Liu, Q. Zhu, F. Wei, W. Rao, J. Liu, J. Hu, and W. Cai, An integrated optimization control method for remanufacturing assembly system, *J. Clean. Prod.*, vol. 248, p. 119261, 2020.

[41] M. Hojati, Minimizing make-span in 2-stage disassembly flow-shop scheduling problem, *Comput. Ind. Eng.*, vol. 94, pp. 1–5, 2016.

[42] M. G. Kim, J. M. Yu, and D. H. Lee, Scheduling algorithms for remanufacturing systems with parallel flow-shop-type reprocessing lines, *Int. J. Prod. Res.*, vol. 53, no. 6, pp. 1819–1831, 2015.

[43] J. M. Kim, Y. D. Zhou, and D. H. Lee, Priority scheduling to minimize the total tardiness for remanufacturing systems with flow-shop-type reprocessing lines, *Int. J. Adv. Manuf. Technol.*, vol. 91, no. 9, pp. 3697–3708, 2017.

[44] W. Wang, G. Tian, H. Zhang, Z. Li, and L. Zhang, A hybrid genetic algorithm with multiple decoding methods for energy-aware remanufacturing system scheduling problem, *Robot. Comput. Integr. Manuf.*, vol. 81, p. 102509, 2023.

[45] J. M. Yu and D. H. Lee, Scheduling algorithms for job-shop-type remanufacturing systems with component matching requirement, *Comput. Ind. Eng.*, vol. 120, pp. 266–278, 2018.

[46] G. Gong, Q. Deng, R. Chiong, X. Gong, H. Huang, and W. Han, Remanufacturing-oriented process planning and scheduling: Mathematical modelling and evolutionary optimisation, *Int. J. Prod. Res.*, vol. 58, no. 12, pp. 3781–3799, 2020.

[47] X. Wen, K. Wang, H. Li, H. Sun, H. Wang, and L. Jin, A two-stage solution method based on NSGA-II for green multi-objective integrated process planning and scheduling in a battery packaging machinery workshop, *Swarm Evol. Comput.*, vol. 61, p. 100820, 2021.

[48] W. Zhang, Y. Zheng, and R. Ahmad, The integrated process planning and scheduling of flexible job-shop-type remanufacturing systems using improved artificial bee colony algorithm, *J. Intell. Manuf.*, vol. 34, no. 7, pp. 2963–2988, 2023.

[49] J. Shi, W. Zhang, S. Zhang, W. Wang, J. Lin, and R. Feng, A new environment-aware scheduling method for remanufacturing system with non-dedicated reprocessing lines using improved flower pollination algorithm, *J. Manuf. Syst.*, vol. 57, pp. 94–108, 2020.

[50] A. Ala, A. Goli, S. Mirjalili, and V. Simic, A fuzzy multi-objective optimization model for sustainable healthcare supply chain network design, *Appl Soft Comput*, vol. 150, p. 111012, 2024.

[51] A. Ala, V. Simic, D. Pamucar, and N. Bacanin, Enhancing patient information performance in Internet of Things-based smart healthcare system: Hybrid artificial intelligence and optimization approaches, *Eng. Appl. Artif. Intell.*, vol. 131, p. 107889, 2024.

[52] J. Wang, D. Lei, and J. Cai, An adaptive artificial bee colony with reinforcement learning for distributed three-stage assembly scheduling with maintenance, *Appl. Soft Comput.*, vol. 117, p. 108371, 2022.

[53] R. Qi, J. Q. Li, J. Wang, H. Jin, and Y. Y. Han, QMOEA: A Q-learning-based multiobjective evolutionary algorithm for solving time-dependent green vehicle routing problems with time windows, *Inf. Sci. Int. J.*, vol. 608, pp. 178–201, 2022.

[54] J. J. Ji, Y. N. Guo, X. Z. Gao, D. W. Gong, and Y. P. Wang, Q-learning-based hyperheuristic evolutionary algorithm for dynamic task allocation of crowdsensing, *IEEE Trans. Cybern.*, vol. 53, no. 4, pp. 2211–2224, 2023.

[55] J. Cai, D. Lei, J. Wang, and L. Wang, A novel shuffled frog-leaping algorithm with reinforcement learning for distributed assembly hybrid flow shop scheduling, *Int. J. Prod. Res.*, vol. 61, no. 4, pp. 1233–1251, 2023.

[56] Y. Ren, K. Gao, Y. Fu, H. Sang, D. Li, and Z. Luo, A novel Q-learning based variable neighborhood iterative search algorithm for solving disassembly line scheduling problems, *Swarm Evol. Comput.*, vol. 80, p. 101338, 2023.

[57] M. Chu and W. Chen, Human-robot collaboration disassembly planning for end-of-life power batteries, *J. Manuf. Syst.*, vol. 69, pp. 271–291, 2023.

[58] G. Tian, M. Zhou, and P. Li, Disassembly sequence planning considering fuzzy component quality and varying operational cost, *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 748–760, 2018.

[59] Y. Hou, Y. Fu, K. Gao, H. Zhang, and A. Sadollah, Modelling and optimization of integrated distributed flow shop scheduling and distribution problems with time windows, *Expert Syst. Appl.*, vol. 187, p. 115827, 2022.

[60] C. Bierwirth and D. C. Mattfeld, Production scheduling and rescheduling with genetic algorithms, *Evol. Comput.*, vol. 7, no. 1, pp. 1–17, 1999.

[61] Y. Fu, X. Ma, K. Gao, Z. Li, and H. Dong, Multi-objective home health care routing and scheduling with sharing service via a problem-specific knowledge-based artificial bee colony algorithm, *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 2, pp. 1706–1719, 2024.

[62] Y. Yang, Y. Song, W. Guo, Q. Lei, A. Sun, and L. Fan, Guided shuffled frog-leaping algorithm for flexible job shop scheduling problem with variable sublots and

overlapping in operations, *Comput. Ind. Eng.*, vol. 180, p. 109209, 2023.

[63] D. Z. Zheng and L. Wang, An effective hybrid heuristic for flow shop scheduling, *Int. J. Adv. Manuf. Technol.*, vol. 21, no. 1, pp. 38–44, 2003.

[64] L. Wang, Z. Pan, and J. Wang, A review of reinforcement learning based intelligent optimization for manufacturing scheduling, *Complex System Modeling and Simulation*, vol. 1, no. 4, pp. 257–270, 2021.

[65] G. Tian, M. Zhou, and J. Chu, A chance constrained programming approach to determine the optimal disassembly sequence, *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 4, pp. 1004–1013, 2013.

[66] D. C. Montgomery, *Design and Analysis of Experiments*. Hoboken, NJ, USA: John Wiley & Sons, 2017.

[67] X. Ma, Y. Fu, K. Gao, L. Zhu, and A. Sadollah, A multi-objective scheduling and routing problem for home health care services via brain storm optimization, *Complex System Modeling and Simulation*, vol. 3, no. 1, pp. 32–46, 2023.

[68] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, *J. Am. Stat. Assoc.*, vol. 32, no. 200, pp. 675–701, 1937.

[69] D. G. Pereira, A. Afonso, and F. M. Medeiros, Overview of Friedman's test and post-hoc analysis, *Commun. Stat. Simul. Comput.*, vol. 44, no. 10, pp. 2636–2653, 2015.

[70] Z. Zhang, Y. Fu, K. Gao, H. Zhang, and L. Wang, A cooperative evolutionary algorithm with simulated annealing for integrated scheduling of distributed flexible job shops and distribution, *Swarm Evol. Comput.*, vol. 85, p. 101467, 2024.

[71] Z. Wu, R. Zhou, M. Goh, Y. Wang, Z. Xu, and W. Song, (DT4Smart) a digital twin-based modularized design approach for smart warehouses, *Int. J. Comput. Integr. Manuf.*, doi: 10.1080/0951192X.2023.2278100.

[72] K. Z. Gao, Z. M. He, Y. Huang, P. Y. Duan, and P. N. Suganthan, A survey on meta-heuristics for solving disassembly line balancing, planning and scheduling problems in remanufacturing, *Swarm Evol. Comput.*, vol. 57, p. 100719, 2020.

Fuquan Wang received the BS degree in engineering cost from Shandong Jianzhu University, Jinan, China in 2021. He is currently pursuing the MS degree in management science and engineering at Qingdao University, Qingdao, China. His current research interests include system modeling and optimization, remanufacturing scheduling, and Q-learning algorithms.

Yaping Fu received the BS degree in commodity science from Harbin University of Commerce, Harbin, China in 2008, the MS degree in economics and management from Northeast Electric Power University, Jilin, China in 2011, and the PhD degree in systems engineering from Northeastern University, Shenyang, China in 2015. From 2018 to 2019, he was a research fellow at the Department of Systems Engineering and Management, National University of Singapore (NUS), Singapore. He is currently a professor with management and science engineering, Qingdao University. His research focuses on multi-objective production planning and scheduling, routing optimization, and evolutionary multi-objective optimization.

Song Gao received the BS degree in international trade from Northeastern University, Shenyang, China in 2001, and the MS degree in computer application from Northeastern University, Shenyang, China in 2004. He is currently pursuing the PhD degree in electronics and information at Northeastern University, Shenyang, China. His research interests include production scheduling and meta-heuristic optimization.

Kaizhou Gao received the BSc degree in computer science and technology from Liaocheng University, Liaocheng, China in 2005, the master degree in computer science and application from Yangzhou University, Yangzhou, China in 2008, and the PhD degree in artificial intelligence and system engineering from Nanyang Technological University (NTU), Singapore in 2016. From 2008 to 2012, he was at the School of Computer Science, Liaocheng University, China. From 2012 to 2013, he was a research associate at the School of Electronic and Electrical Engineering, NTU, Singapore, where he has been a research fellow from 2015 to 2018. He is currently an assistant professor at the Macau Institute of Systems Engineering, Macau University of Science and Technology, Macau, China. His research interests include intelligent computation, optimization, scheduling, and intelligent transportation. He has published over 100 refereed papers. He is an associate editor of *Swarm and Evolutionary Computation*, *IET Collaborative Intelligent Manufacturing*, and *The Chinese Journal of Artificial Intelligence*.

Yaoxin Wu received the BS degree in traffic engineering from Wuyi University, Jiangmen, China in 2015, the MS degree in control engineering from Guangdong University of Technology, Guangzhou, China in 2018, and the PhD degree in computer science from Nanyang Technological University, Singapore in 2023. He was a research associate at the Singtel Cognitive and Artificial Intelligence Lab for Enterprises (SCALE@NTU). He is an assistant professor at the Department of Industrial Engineering & Innovation Sciences, Eindhoven University of Technology, the Netherlands. His research interests include combinatorial optimization, integer programming, and deep learning.