Int J Adv Manuf Technol (2017) 91:3697–3708
DOI 10.1007/s00170-017-0057-z

ORIGINAL ARTICLE

Priority scheduling to minimize the total tardiness for remanufacturing systems with flow-shop-type reprocessing lines

Jong-Min Kim[1] · **Yi-Dong Zhou**[1] · **Dong-Ho Lee**[1]

Received: 9 October 2016 / Accepted: 16 January 2017 / Published online: 27 January 2017

Abstract This study considers the scheduling problem in remanufacturing systems in which end-of-use/life products are separated into their components at a single disassembly workstation, then each component is reprocessed at one of parallel flow-shop-type reprocessing lines, and finally the reprocessed components are reassembled into remanufactured products at parallel reassembly workstations. The problem is to determine the sequence of products to be disassembled at the disassembly workstation, the sequence of components to be reprocessed at each workstation of the reprocessing lines, and the allocation and sequence of the products to be reassembled at each reassembly workstation for a due date-based objective of minimizing the total tardiness. A mathematical programming model is developed to represent the problem, and a priority scheduling approach is proposed for practical applications. To test performances of priority rules, simulation experiments were done on various test instances, and the results are reported. In particular, we show from additional tests that the approach proposed in this study outperforms the previous one that determines reprocessing and reassembly schedules according to the sequence of disassembling products significantly, and also, the rule combination approach that uses different priority rules on disassembly, reprocessing, and reassembly shops outperforms the single rule approach that uses the same rule over the three subsystems.

✉ Dong-Ho Lee
leman@hanyang.ac.kr

[1] Department of Industrial Engineering, Hanyang University, Seoul, Republic of Korea

Keywords Remanufacturing · Flow-shop-type reprocessing lines · Scheduling · Tardiness · Priority rules

1 Introduction

When a product reaches its end-of-life state, it can be treated further in various ways such as reuse, remanufacturing, recycling, and disposal. Among them, remanufacturing is a product recovery option that reprocesses end-of-use/life products in such a way that their qualities, performance, reliability, and appearance are as good as new. According to Steinhilper [1], remanufacturing is one of the advanced end-of-life options since the geometric form of the original product is retained while restoring it to like-new conditions. Compared with the material recycling option in which products are destroyed and useful materials are extracted, the remanufacturing option has both economic and environmental benefits. For example, remanufacturing is an emerging industry with high profit margins, and also, the energy embodied in producing a new product is estimated as four to five times as high as the energy embodied in the remanufactured product [2].

A typical remanufacturing system consists of three subsystems: disassembly, reprocessing, and reassembly shops [3]. An end-of-use/life product is disassembled into its constituent components at the disassembly shop. Then, the components are reconditioned into like-new ones at the reprocessing shop. Finally, the reprocessed components are reassembled into remanufactured products at the reassembly shop. See Steinhilper [1], Lund [2], and Sundin and Bras [4] for details on remanufacturing processes. According to the physical structure of the three dependent subsystems, there may be various system configurations. Among them, this study considers the one with a single disassembly workstation, parallel

flow-shop-type reprocessing lines, and parallel reassembly workstations. Note that the typical remanufacturing configuration considered in this study can be found in various remanufacturing systems, especially in automotive part remanufacturing systems.

For the remanufacturing configuration, we address the scheduling problem that determines the sequence of products to be disassembled at the disassembly workstation, the sequence of reprocessing components at each workstation of flow-shop-type reprocessing lines, and the allocation and sequence of reassembling products at parallel reassembly workstations. In the theoretical aspect, the problem is different from the ordinary two-stage assembly flow shop scheduling problem because the system considered in this study has the first-stage disassembly workstation that separates products into their components. Note that the two-stage assembly flow shop processes the components simultaneously at the first stage, and then, they are assembled at the second stage after all components are processed. For the characteristics of the two-stage assembly flow shop, refer to Lee et al. [5], Al-Anzi and Allahverdi [6, 7], Allahverdi and Al-Anzi [8], and Shokrollahpour et al. [9]. Also, the remanufacturing system considered in this study is different from hybrid flow shops in that the second-stage reprocessing shop consists of parallel flow-shop-type lines. See Choi et al. [10], Chamnanlor et al. [11], and Yang [12] for various configurations for the hybrid flow shop and Linn and Zhang [13] for a literature review on the associated scheduling problems.

Most previous studies on remanufacturing scheduling are done by Guide and his collaborators. Guide [14, 15] reports an application of the synchronous scheduling approach, called the drum-buffer-rope, to a military depot that remanufactures aircraft engines and shows from simulation experiments that the proposed method outperforms the scheduling method used at the depot. Later, more studies are performed on priority rules, and the simulation results show that simple rules work well to support the drum-buffer-rope method [16–18]. Kim et al. [19] consider a remanufacturing system with a single disassembly workstation, parallel flow-shop-type reprocessing lines, and a single reassembly workstation and propose various scheduling algorithms that minimize the total flow time. Also, some other articles consider the scheduling problem for each of the three subsystems. For the disassembly shop, Kizilkaya and Gupta [20] propose a flexible kanban system that lowers shortage levels, and later, Udomsawat and Gupta [21] extend it to a multi-kanban system. Yu et al. [22] define the reprocessing shop scheduling problem as job shop scheduling with job families in which reprocessing jobs are grouped into job families each of which corresponds to a remanufactured product type. Later, Kim et al. [23] extend it to the problem with sequence-dependent setups.

As explained earlier, this study considers a scheduling problem in remanufacturing systems with a single disassembly workstation, parallel flow-shop-type reprocessing lines, and parallel reassembly workstations. The problem is to determine the sequence of products to be disassembled at the disassembly workstation, the sequence of components to be reprocessed at each workstation of the reprocessing lines, and the allocation and sequence of the remanufactured products to be reassembled at parallel reassembly workstations. In fact, the problem considered in this study extends Kim et al. [19] by considering the due date-based objective and parallel reassembly workstations. As in other scheduling problems, the tardiness measure is particularly important in the systems that there is a penalty to complete a job beyond its due date and the penalty increases as the gap between the job's due date and its completion time increases. Also, as hybrid flow shops, the parallel workstations are installed at the reassembly shop for the purpose of increasing both productivity and flexibility of remanufacturing systems. To the best of the authors' knowledge, there is no previous study on due date-based scheduling for remanufacturing systems with flow-shop-type reprocessing lines and parallel reassembly workstations.

To represent the problem mathematically, a mixed integer programming model is developed for the objective of minimizing the total tardiness. Then, due to the problem complexity, we propose the priority scheduling approach that employs a priority rule to select an operation among those waiting in a queue of a workstation for practical applications. To test the performances of various priority rules, simulation experiments were done on a number of test instances and the results are reported. In particular, the scheduling approach proposed in this study is compared with the previous one that determines the reprocessing and reassembly schedules according to the disassembly sequence. Also, we report the results on the comparison between the rule combination approach that uses different rules on disassembly, reprocessing and reassembly shops, and the single rule approach that uses the same rule over the three subsystems.

The rest of this paper is organized as follows. The next section describes the problem in more detail with a mixed integer programming model. Sections 3 and 4 present the priority scheduling approach and the simulation results, respectively. Finally, Sect. 5 concludes the paper with further research areas.

2 System and problem descriptions

This section explains the remanufacturing systems considered in this study, and then describes the problem in more detail with a mixed integer programming model.

As explained earlier, the remanufacturing system consists of three serial stages: single disassembly workstation, parallel flow-shop-type reprocessing lines, and parallel reassembly

workstations. In the aspect of scheduling theory, the system can be regarded as a three-stage hybrid flow shop, but it is different from the ordinary one because it has a disassembly workstation and parallel flow-shop-type reprocessing lines with serial workstations. Figure 1 shows a remanufacturing system with three flow-shop-type reprocessing lines and two parallel reassembly workstations. As can be seen in the figure, each of the two end-of-use/life product types is disassembled into its components at the disassembly workstation, i.e., product P_A into components C_{A1}, C_{A2}, and C_{A3} and product P_B into components C_{B1} and C_{B2}. Then, each of the components is reprocessed at its dedicated flow-shop-type reprocessing line, where the required reprocessing operations are done through its serial workstations. Finally, the reprocessed components for each product are reassembled into the corresponding remanufactured product at one of the parallel reassembly workstations. Note that the reassembly operation for a remanufactured product can be started only when all of its components are reprocessed on the reprocessing lines. It is assumed that the parallel reassembly workstations are identical, and hence the reassembly operations of a product can be done at any reassembly workstation with equal time.

Now, the problem considered here can be briefly described as follows. For a given set of end-of-use/life products, the problem is to determine the sequence of products to be disassembled at the disassembly workstation, the sequence of reprocessing components at each workstation of the flow-shop-type reprocessing lines, and the allocation and sequence of reassembling products at the parallel reassembly workstations. Figure 2 shows an example of remanufacturing schedule for the example system given in Fig. 1. As can be seen in the figure, the disassembly sequence is $P_A \rightarrow P_B$, and the reprocessing sequences are $C_{A1} \rightarrow C_{B1}$, $C_{A2} \rightarrow C_{B2}$, and $C_{A3} \rightarrow C_{B3}$ at reprocessing lines 1, 2, and 3, respectively. Finally, remanufactured products P_A and P_B are reassembled at reassembly workstations 1 and 2, respectively. The objective is to minimize the total tardiness, where the tardiness of a remanufactured product is defined as the positive deviation of the completion time and the due date of the product, i.e.,

$$\sum_i \max\left(C_i^A - d_i, 0\right),$$

where C_i^A and d_i denote the completion time of reassembling product i at a reassembly workstation and the due date of remanufactured product i, respectively.

This study considers a static and deterministic version of the problem, i.e., all jobs are ready for processing at time 0, and the problem data, such as processing times, due dates, and so on, are deterministic and given in advance. Other assumptions made are as follows: (a) each workstation can process only one operation at a time; (b) setup times are sequence-independent, and hence can be included in the corresponding processing times; (c) transportation times between workstations are ignorable; (d) preemption is not allowed; and (e) component defectives are not considered.

To describe the problem more clearly, a mixed integer programming model is proposed. The following notations are used in the formulation.

Indices

i products, $i = 1, 2, \ldots n$

j components (reprocessing lines), $j = 1, 2, \ldots m$

k workstations in a reprocessing line, $k = 1, 2, \ldots l_j$

l reassembly workstations, $l = 1, 2, \ldots r$

Parameters

t_i^D processing time required to disassemble product i

t_{ijk}^R processing time required to reprocess component j at workstation k of reprocessing line j

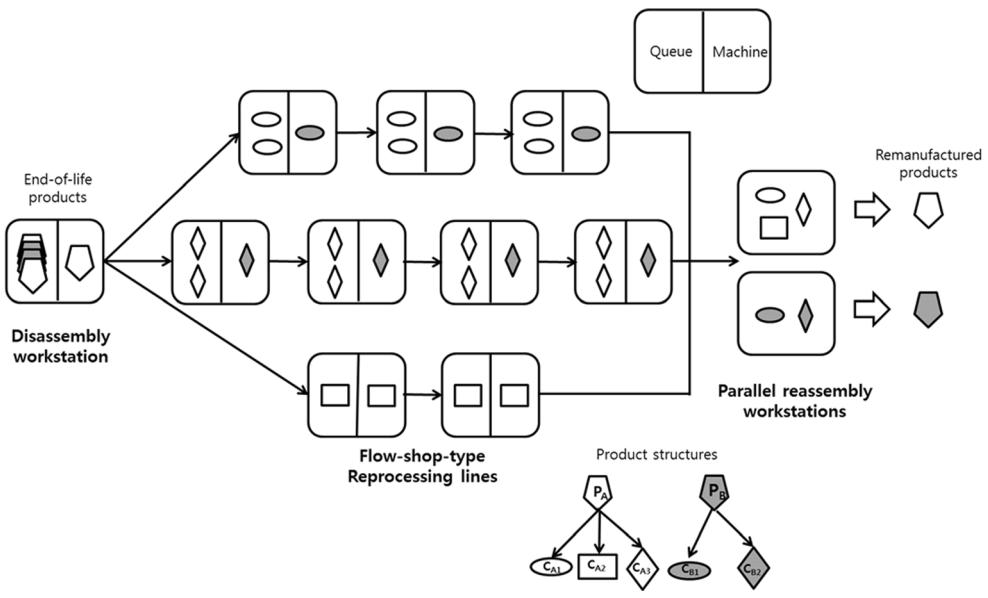

Fig. 1 Remanufacturing system configuration: example

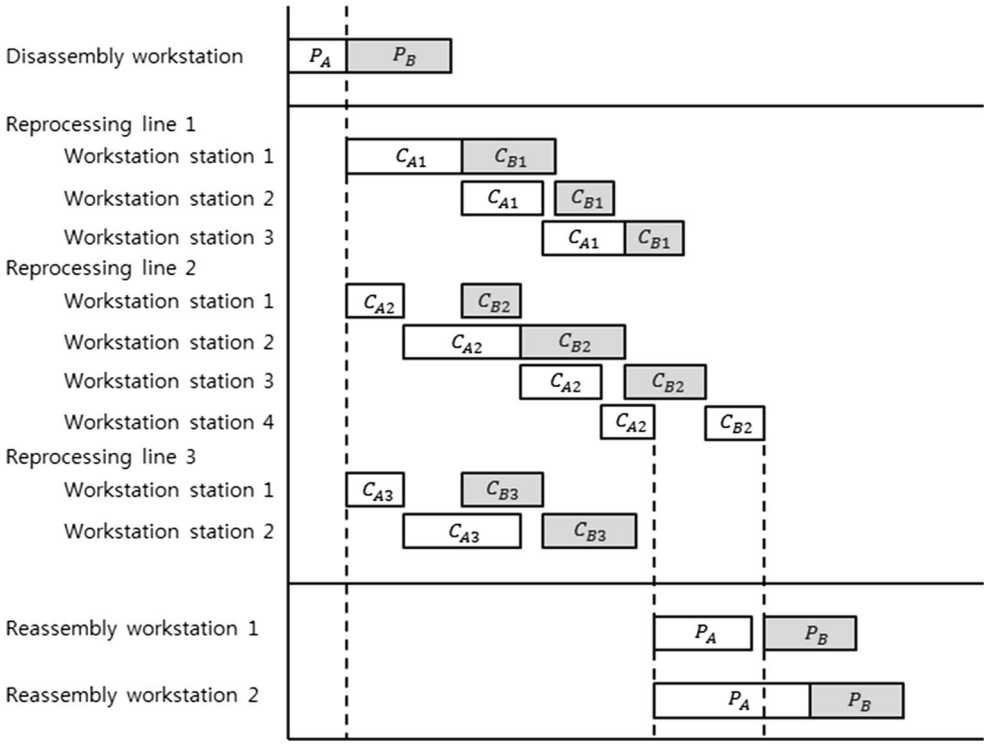

Fig. 2 Remanufacturing schedule: example

t_i^A processing time required to reassemble product i at reassembly workstations

d_i due date of remanufactured product i

M large number

Decision variables

C_i^D completion time of disassembling product i at disassembly workstation

C_{ijk}^R completion time of reprocessing component j of product i at workstation k of the corresponding reprocessing line

C_i^M largest completion time among those of reprocessing components of product i at the last workstations of reprocessing lines

C_i^A completion time of reassembling product i at a reassembly workstation

$x_{ii'}$ = 1 if product i is disassembled directly before product i', and 0 otherwise

$y_{ii'jk}$ = 1 if component j of product i is reprocessed directly before the component j' of product i' at workstation k of reprocessing line j, and 0 otherwise

u_i^A = 1 if product i is reassembled firstly at one of the reassembly workstation, and 0 otherwise

$z_{ii'}$ = 1 if product i is reassembled directly before product i', and 0 otherwise

Now, the mixed integer programming model is given below.

[P] Minimize $\sum_{i=1}^{n}\max\left(C_i^A-d_i,0\right)$

subject to

$$C_i^D \geq t_i^D \quad \text{for all } i \tag{1}$$

$$C_{i'}^D - C_i^D + M\cdot\left(1-x_{ii'}\right) \geq t_{i'}^D \quad \text{for all } i \text{ and } i'\,(i\neq i') \tag{2}$$

$$C_i^D - C_{i'}^D + M\cdot x_{ii'} \geq t_i^D \quad \text{for all } i \text{ and } i'\,(i\neq i') \tag{3}$$

$$C_{ij1}^R \geq C_i^D + t_{ij1}^R \quad \text{for all } i \text{ and } j \tag{4}$$

$$C_{ij,k+1}^R \geq C_{ijk}^R + t_{ij,k+1}^R \quad \text{for all } i, j \text{ and } k \tag{5}$$

$$C_{i'jk}^R - C_{ijk}^R + M\cdot\left(1-y_{ii'jk}\right) \geq t_{i'jk}^R \quad \text{for all } i, i'\,(i\neq i'), j \text{ and } k \tag{6}$$

$$y_{ii'jk} + y_{i'ijk} \leq 1 \quad \text{for all } i, i'\,(i\neq i'), j \text{ and } k \tag{7}$$

$$C_i^M \geq C_{ijl_j}^R \quad \text{for all } i \text{ and } j \tag{8}$$

$$C_i^A \geq C_i^M + t_i^A \quad \text{for all } i \tag{9}$$

$$C_{i'}^A - C_i^A + M\cdot\left(1-z_{ii'}\right) \geq t_{i'}^A \quad \text{for all } i \text{ and } i'\,(i\neq i') \tag{10}$$

$$z_{ii'} + z_{i'i} \leq 1 \quad \text{for all } i \text{ and } i'\,(i\neq i') \tag{11}$$

$$\sum_{i=1}^{n} u_i^A \leq r \tag{12}$$

$$\sum_{i=1}^{n} z_{i,n+1} \leq r \tag{13}$$

$$u_i^A + \sum_{i'=1}^{n} z_{i'i} = 1 \quad \text{for all } i\,(i\neq i') \tag{14}$$

$$u_i^A + \sum_{i'=1}^{n+1} z_{ii'} \leq 2 \quad \text{for all } i\,(i\neq i') \tag{15}$$

$$\sum_{i'=1}^{n+1} z_{ii'} \leq 1 \quad \text{for all } i\,(i\neq i') \tag{16}$$

$$x_{ii'}, y_{ii'jk}, u_i^A, z_{ii'} \in \{0,1\} \quad \text{for all } i, i'\,(i\neq i'), j \text{ and } k \tag{17}$$

The objective function represents minimizing the total tardiness. Constraint set (1) specifies the completion times of disassembling products. Constraint sets (2) and (3) ensure that no two products can be disassembled simultaneously at the disassembly workstation. Constraint set (4) specifies the completion time of the first reprocessing operation of each component, which also ensures that the first reprocessing operation must be done after the corresponding disassembly operation is completed. Constraint set (5) represents the precedence relation between two successive reprocessing operations of a component. Similar to (2) and (3), constraint sets (6) and (7) ensure that no two operations can be done simultaneously at each workstation of reprocessing lines. Constraint set (8) specifies the largest completion time among those of reprocessing components for a product. Similar to (4), constraint set (9) specifies the completion time of reassembling operation of a product, which also ensures that a product reassembly must be done after the corresponding reprocessing operations are completed. Similar to (2) and (3), constraint sets (10) and (11) ensure that no two products can be reassembled simultaneously at each reassembly workstation. Constraint sets (12), (13), (14), (15), and (16) specify the allocation and sequence of reassembling products, which can be obtained by allocating reassembly jobs to the earliest available workstation according to the sequence. Specifically, constraint sets (12) and (13) ensure that at most, one product can be the first and the last ones of the sequence, respectively. Constraint sets (14) and (15), together with (12) and (13), specify the sequence of products to be allocated and sequenced at reassembly workstations. Also, constraint set (16) represents that each reassembly job must have at most one direct successor on the reassembly sequence. Finally, constraint set (17) represents the conditions of decision variables.

We can easily see that the problem [P] considered in this study is NP-hard since the single machine tardiness scheduling problem is proven to be NP-hard. See Du and Leung [24] for its proof. Therefore, instead of the optimal approach with very limited applications, we propose a practical priority rule-based approach. Recall that the remanufacturing scheduling problem is much more complicated than the ordinary hybrid flow shop scheduling problem since it contains parallel flow-shop-type reprocessing lines.

3 Priority scheduling approach

This section explains the priority scheduling approach that determines disassembly, reprocessing, and reassembly schedules by employing one or more priority rules. In this study, we adopt the priority scheduling approach since it is simple and easy to implement while yielding reasonable solutions with very short computation times.

As in other scheduling problems, the priority rules are used according to the non-delay schedule generation method in which a schedule is generated in such a way that no workstation is kept idle at a time when it can begin processing some operations. In other words, the next job to be processed on a workstation is selected when a workstation becomes idle after the current job is completed. Here, a job implies a product at disassembly and reassembly workstations and a component at reprocessing workstations, respectively. See Baker [25] for more detail on non-delay schedules. Recall that the reassembly schedule is obtained by selecting an available product to be reassembled using a priority rule, and then, it is allocated to the earliest available reassembly workstation.

In this study, the following priority rules were tested because they perform better than others for due date-based scheduling problems. Note that some priority rules are modified according to the characteristics of the remanufacturing systems. Before describing the priority rules, the additional notations are summarized below.

- t time at which a priority rule is applied, i.e., when a workstation becomes idle after completing the current job
- N_{wt} set of jobs waiting in front of workstation w at time t

Now, the priority rules are explained below.

- FCFS select a job that arrived the earliest at the queue of a workstation, i.e., first come first served
- SPT select a job with the shortest processing time, i.e., select a job i^* such that
  $i^* = \text{argmin}_{i \in N_{wt}} \{t_i^D\}$ for disassembly workstation;
  $i^* = \text{argmin}_{i \in N_{wt}} \left\{t_{ijw}^R\right\}$ for reprocessing workstations; and
  $i^* = \text{argmin}_{i \in N_{wt}} \{t_i^A\}$ for reassembly workstations
- LPT select a job with the longest processing time, i.e., select a job i^* such that
  $i^* = \text{argmax}_{i \in N_{wt}} \{t_i^D\}$ for disassembly workstation;
  $i^* = \text{argmax}_{i \in N_{wt}} \left\{t_{ijw}^R\right\}$ for reprocessing workstations; and
  $i^* = \text{argmax}_{i \in N_{wt}} \{t_i^A\}$ for reassembly workstations
- EDD select a job with the earliest due date, i.e., select a job i^* such that $i^* = \text{argmin}_{i \in N_{wt}} \{d_i\}$
- SLACK select a job with the minimum slack value, i.e., select a job i^* such that $i^* = \text{argmin}_{i \in N_{wt}} \{d_i - t - r_{iw}\}$,

where r_{iw} denotes the remaining work of job i on workstation w, i.e., sum of processing times of successor operations including itself

MDD select a job with the minimum modified due date, i.e., select a job i^* such that $i^* = \text{argmin}_{i \in N_{wt}} \{d_i, t + r_{iw}\}$

CR select a job with the minimum critical ratio value, i.e., select a job i^* such that $i^* = \text{argmin}_{i \in N_{wt}} \{(d_i - t)/r_{iw}\}$

COVERT select a job with the maximum cost over time value, i.e., select a job i^* such that

$i^* = \text{argmax}_{i \in N_{wt}} \left\{ \left(1/t_i^D\right) \cdot (1-(d_i - t - r_{iw})/k_b \cdot \left(r_{iw} - t_i^D\right) \right\}$ for disassembly workstation;

$i^* = \text{argmax}_{i \in N_{wt}} \left\{ \left(1/t_{iiw}^R\right) \cdot (1-(d_i - t - r_{iw})/k_b \cdot \left(r_{iw} - t_{ijw}^D\right) \right\}$ for reprocessing workstations; and

$i^* = \text{argmax}_{i \in N_{wt}} \left\{ 1/t_i^A \right\}$ for reassembly workstations,

where k_b denotes the parameter used to estimate the completion time of a job (from a preliminary test, k_b was set to 0.7 in the test).

RMDD select a job with the minimum reverse modified due date, i.e., select a job i^* such that

$i^* = \text{argmin}_{i \in N_{wt}} \left\{ t_i^D + \min(d_i - t - r_{iw}, 0) \right\}$ for disassembly workstation;

$i^* = \text{argmin}_{i \in N_{wt}} \left\{ t_{ijw}^R + \min(d_i - t - r_{iw}, 0) \right\}$ for reprocessing workstations; and

$i^* = \text{argmin}_{i \in N_{wt}} \left\{ t_i^A + \min(d_i - t - r_{iw}, 0) \right\}$ for reassembly workstations.

The priority rules explained above can be used in two ways. First, one priority rule is used across disassembly, reprocessing, and reassembly workstations, i.e., single rule approach. Second, different rules are used across the three workstation types, i.e., rule combination approach. Note that the approach of Kim et al. [19] is a rule combination in which the sequence of disassembling products is determined by a priority rule and the reprocessing and reassembly schedules are determined according to the sequence of disassembling products, i.e., FCFS in reprocessing and reassembly workstations.

4 Simulation experiments

To test the performances of the priority rules, simulation experiments were done on various random instances, and the results are reported in this section. The simulation model, together with the priority rules, was coded in C++, and the tests were done on a workstation with Intel Core i5 processor operating at 2.80 GHz.

For the simulation experiments, we generated two types of test instances, i.e., agreeable and non-agreeable instances, where the agreeable instances have the property that due dates increase as the sums of disassembly, reprocessing, and reassembly times increase. The agreeable instances were considered to test the performance of EDD schedules because if the SPT sequence is identical to the EDD sequence, it is optimal for the basic single machine scheduling problem that minimizes the total tardiness. For evaluating the results, we use the relative deviation index, where the relative deviation index for a test instance is defined as

$$(TT_a - TT_{\text{best}})/(TT_{\text{worst}} - TT_{\text{best}})$$

where TT_a is the total tardiness obtained from rule (or rule combination) a, TT_{worst} is the worst one for that instance among those obtained from all rules (or rule combinations), and TT_{best} is the best one for that instance among those obtained from all rules (or rule combinations). Note that the CPU seconds are not reported here since all the test instances were solved within 1 s.

For each of the two instance types (agreeable and non-agreeable), 320 instances were generated randomly according to the method of Kim et al. [19], i.e., 10 instances for each of 32 combinations of 2 levels of due date tightness (loose and tight), 4 levels of the number of products (20, 40, 60, and 80), 2 levels for the number of components (3/5 and 7/9), and 2 levels for the number of reassembly workstations (2 and 3). Recall that a component is reprocessed at its dedicated reprocessing line, and hence, the number of reprocessing lines is the same as the number of components for a product type.

The detailed data were generated as follows. The processing times were generated from $DU(1100)$, where $DU(a, b)$ denotes the discrete uniform distribution with range $[a, b]$. Also, the due dates of remanufactured products were generated from

$$DU(C_i \cdot (1-T-R/2), C_i \cdot (1-T+R/2))$$

where C_i denotes the estimated completion time of product i, which was calculated as

$$t_i^D + \max\left\{ \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l_j} t_{ijk}^R}{m}, \max_{j,k}\left\{ t_{ijk}^R \right\} \right\} + \max\left\{ \frac{\sum_{i=1}^{n} t_i^A}{w}, \max\left\{ t_i^A \right\} \right\}$$

Also, the tardiness factor T and the relative due date range R were set to 0.2 for the instances with loose due dates and 0.6 for the instances with tight due dates, respectively.

Test results are summarized in Tables 1 and 2 that show relative deviation indices of all priority rules tested for the instances with loose and tight due dates, respectively. As can be seen in Table 1, EDD dominates the others for all the agreeable and non-agreeable test instances with loose due dates. On the other hand, Table 2 shows that no one priority rule dominates the others for the instances with tight due dates. Specifically, EDD, LPT, and CR are better than the others in overall averages for the agreeable test instances and LPT, COVERT, EDD, and CR for the non-agreeable test instances. To show the statistical differences among the priority rules, the Duncan's multiple range test was done for all the test instances, and the result is given in Table 3. It can be seen from the table that EDD and CR are better than the others for both agreeable and non-agreeable test instances.

Table 1 Test results for the instances with loose due dates

NP[a]	NC[b]	NR[c]	Priority rules							
			SPT	EDD	SLACK	MDD	CR	COVERT	RMDD	LPT
(a) Agreeable instances										
20	3	2	0.11(0.00,1.00)[d]	0.00(0.00,0.00)	0.20 (0.00,1.00)	0.14(0.00,0.69)	0.08(0.00,0.24)	0.85(0.00,1.00)	0.14(0.00,0.69)	0.40(0.00,1.00)
		3	0.33(0.00,1.00)	0.00(0.00,0.00)	0.35 (0.00,0.78)	0.35(0.00,1.00)	0.22(0.00,0.51)	0.81(0.00,1.00)	0.35(0.00,1.00)	0.39(0.00,0.98)
	5	2	0.38(0.00,0.80)	0.00(0.00,0.00)	0.31 (0.00,1.00)	0.30(0.00,1.00)	0.13(0.00,0.61)	0.89(0.43,1.00)	0.30(0.00,1.00)	0.33(0.00,1.00)
		3	0.40(0.11,1.00)	0.00(0.00,0.00)	0.47 (0.00,0.96)	0.41(0.00,0.61)	0.24(0.00,0.64)	0.88(0.43,100)	0.41(0.06,1.00)	0.56(0.00,1.41)
40	3	2	0.06(0.00,0.35)	0.00(0.00,0.00)	0.02 (0.00,0.23)	0.10(0.00,1.00)	0.02(0.00,0.2)	0.81(0.00,1.00)	0.05(0.00,0.50)	0.08(0.00,0.52)
		3	0.18(0.00,0.70)	0.00(0.00,0.00)	0.01 (0.00,0.05)	0.06(0.00,0.19)	0.07(0.00,0.18)	0.89(0.20,1.00)	0.06(0.00,0.19)	0.41(0.00,1.00)
	5	2	0.15(0.00,0.94)	0.00(0.00,0.00)	0.00 (0.00,0.00)	0.16(0.00,1.00)	0.12(0.00,1.00)	0.66(0.00,1.00)	0.16(0.00,1.00)	0.47(0.00,1.00)
		3	0.25(0.00,1.00)	0.00(0.00,0.00)	0.02 (0.00,0.14)	0.08(0.00,0.2)	0.03(0.00,0.24)	0.88(0.39,1.00)	0.08(0.00,0.20)	0.25(0.00,1.00)
60	7	2	0.44(0.00,1.00)	0.00(0.00,0.00)	0.03 (0.00,0.14)	0.11(0.00,0.48)	0.07(0.00,0.20)	0.74(0.00,1.00)	0.11(0.00,0.48)	0.64(0.00,1.00)
		3	0.47(0.14,0.97)	0.00(0.00,0.00)	0.23 (0.00,0.82)	0.31(0.01,0.77)	0.08(0.00,0.38)	0.93(0.57,1.00)	0.31(0.01,0.77)	0.72(0.32,1.00)
	9	2	0.96(0.64,1.00)	0.00(0.00,0.00)	0.04 (0.00,0.13)	0.10(0.00,0.23)	0.04(0.00,0.11)	0.65(0.24,0.99)	0.10(0.00,0.23)	0.54(0.00,1.00)
		3	0.80(0.45,1.00)	0.00(0.00,0.00)	0.31 (0.04,0.88)	0.35(0.10,0.78)	0.08(0.00,0.28)	0.77(0.50,1.00)	0.35(0.10,0.78)	0.76(0.27,1.00)
80	7	2	0.49(0.00,1.00)	0.00(0.00,0.00)	0.01 (0.00,0.10)	0.05(0.00,0.27)	0.05(0.00,0.22)	0.70(0.00,1.00)	0.05(0.00,0.27)	0.47(0.00,1.00)
		3	0.62(0.16,1.00)	0.00(0.00,0.00)	0.16 (0.00,0.47)	0.20(0.00,0.47)	0.05(0.00,0.20)	0.91(0.38,1.00)	0.20(0.00,0.47)	0.63(0.25,1.00)
	9	2	1.00(1.00,1.00)	0.00(0.00,0.00)	0.00 (0.00,0.00)	0.01(0.00,0.07)	0.02(0.00,0.13)	0.40(0.24,0.79)	0.01(0.00,0.07)	0.23(0.00,0.45)
		3	0.11(0.00,1.00)[d]	0.00(0.00,0.00)	0.20 (0.00,1.00)	0.14(0.00,0.69)	0.08(0.00,0.24)	0.85(0.00,1.00)	0.14(0.00,0.69)	0.40(0.00,1.00)
Average			0.47	**0.00**	0.14	0.18	0.08	0.78	0.18	0.47
(b) Non-agreeable instances										
20	3	2	0.35(0.00,1.00)	0.00(0.00,0.00)	0.18(0.00,0.55)	0.20(0.00,0.53)	0.19(0.00,0.78)	0.71(0.00,1.00)	0.20(0.00,0.53)	0.59(0.00,1.00)
		3	0.70(0.21,1.00)	0.00(0.00,0.00)	0.31(0.00,0.74)	0.43(0.00,0.92)	0.15(0.00,0.36)	0.64(0.00,1.00)	0.43(0.00,0.92)	0.68(0.20,1.00)
	5	2	0.45(0.00,1.00)	0.00(0.00,0.00)	0.12(0.00,0.64)	0.25(0.00,0.84)	0.05(0.00,0.29)	0.65(0.00,1.00)	0.25(0.00,0.84)	0.50(0.00,1.00)
		3	0.56(0.01,1.00)	0.00(0.00,0.00)	0.30(0.00,1.00)	0.45(0.03,0.97)	0.19(0.00,0.45)	0.90(0.63,1.00)	0.45(0.03,0.97)	0.60(0.09,1.00)
40	3	2	0.25(0.00,1.00)	0.00(0.00,0.00)	0.01(0.00,0.12)	0.04(0.00,0.24)	0.04(0.00,0.42)	0.93(0.30,1.00)	0.04(0.00,0.24)	0.18(0.00,0.48)
		3	0.36(0.12,0.90)	0.00(0.00,0.00)	0.04(0.00,0.21)	0.10(0.00,0.33)	0.06(0.00,0.22)	0.76(0.13,1.00)	0.10(0.00,0.33)	0.52(0.00,1.00)
	5	2	0.31(0.00,1.00)	0.00(0.00,0.00)	0.03(0.00,0.37)	0.12(0.00,0.47)	0.06(0.00,0.48)	0.58(0.00,1.00)	0.12(0.00,0.47)	0.35(0.00,1.00)
		3	0.45(0.00,1.00)	0.00(0.00,0.00)	0.03(0.00,0.11)	0.16(0.00,0.35)	0.05(0.00,0.29)	0.84(0.32,1.00)	0.16(0.04,0.35)	0.47(0.00,1.00)
60	7	2	0.84(0.51,1.00)	0.00(0.00,0.00)	0.03(0.00,0.11)	0.11(0.03,0.20)	0.04(0.00,0.21)	0.64(0.27,1.00)	0.11(0.03,0.20)	0.86(0.56,1.00)
		3	0.85(0.49,1.00)	0.00(0.00,0.00)	0.29(0.02,0.56)	0.37(0.06,0.57)	0.13(0.00,0.43)	0.86(0.58,1.00)	0.37(0.06,0.57)	0.70(0.25,1.00)
	9	2	0.97(0.71,1.00)	0.00(0.00,0.00)	0.01(0.00,0.08)	0.12(0.03,0.26)	0.07(0.00,0.35)	0.62(0.24,1.00)	0.12(0.03,0.26)	0.50(0.22,0.92)
		3	0.89(0.48,1.00)	0.00(0.00,0.00)	0.21(0.02,0.55)	0.36(0.12,0.71)	0.09(0.00,0.18)	0.71(0.26,1.00)	0.36(0.12,0.71)	0.71(0.36,1.00)
80	7	2	0.67(0.32,1.00)	0.00(0.00,0.00)	0.01(0.00,0.04)	0.07(0.00,0.16)	0.02(0.00,0.08)	0.62(0.19,1.00)	0.07(0.00,0.16)	0.72(0.30,1.00)
		3	0.78(0.37,1.00)	0.00(0.00,0.00)	0.10(0.00,0.36)	0.15(0.00,0.36)	0.04(0.00,0.10)	0.78(0.31,1.00)	0.15(0.00,0.36)	0.70(0.21,1.00)

Table 1 (continued)

NP[a]	NC[b]	NR[c]	Priority rules							
			SPT	EDD	SLACK	MDD	CR	COVERT	RMDD	LPT
9	2		1.00(1.00,1.00)	0.00(0.00,0.00)	0.00(0.00,0.02)	0.03(0.00,0.08)	0.02(0.00,0.14)	0.37(0.08,0.89)	0.03(0.00,0.08)	0.25(0.11,0.48)
	3		0.96(0.86,1.00)	0.00(0.00,0.00)	0.08(0.01,0.15)	0.12(0.03,0.18)	0.06(0.00,0.20)	0.72(0.28,1.00)	0.12(0.03,0.18)	0.65(0.06,1.00)
Average		0.65	**0.00**	0.11	0.19	0.08	0.66	0.19	0.56	

[a] Number of products

[b] Number of components (parallel reprocessing lines)

[c] Number of reassembly workstations

[d] Average relative deviation index out of 10 instances (minimum and maximum in parenthesis)

The next test was done to compare the best priority rule with the previous approach of Kim et al. [19] in which the disassembly schedule is determined by a priority rule and the resulting reprocessing and reassembly schedules are determined according to the disassembly schedule. From a preliminary test, we found that EDD outperforms the others and hence it was compared with the previous one with the EDD rule in disassembly and the FCFS rule in reprocessing and reassembly, i.e., EDD-FCFS-FCFS, for the test instances with tight due dates, and the results are summarized in Table 4 that shows the percentage improvements over the previous one. It can be seen from the table that a significant amount of improvement can be obtained from the approach proposed in this study, i.e., 64.9% for agreeable and 59.9% for non-agreeable test instances in overall averages. However, there was no difference between the two approaches for the instances with loose due dates.

Another test was done to compare two different approaches that use the priority rules, i.e., single rule approach that uses the same priority rule over disassembly, reprocessing, and reassembly shops and rule combination approach that uses different rules over the three subsystems. For the rule combination approach, we selected SPT for disassembly workstation, EDD for reprocessing lines, and LPT for reassembly workstations from a preliminary test. In summary, we compared EDD under the single rule approach with SPT-EDD-LPT under the rule combination approach, and the test results are summarized in Table 5 that shows the percentage improvements of SPT-EDD-LPT over EDD for the instances with tight due dates (agreeable and non-agreeable). As can be seen in the table, SPT-EDD-LPT outperforms EDD significantly. In particular, the outperformance is larger for the non-agreeable instances, i.e., 33.8% for agreeable and 61.1% for non-agreeable instances. However, there was no difference between the two approaches for the instances with loose due dates.

The final test was done to show the performance of the best rule combination SPT-EDD-LPT in the absolute sense, i.e., gaps from the optimal solution values or lower bounds obtained by solving the mixed integer programming formulations using CPLEX 12.5 under the time limit of 3600 s. For this test, 160 small sized instances were randomly generated for the tight due date case, i.e., 10 instances of each of 16 combinations of 4 levels of the number of products (5, 8, 10, and 13), 2 level of number of components (2 and 3), and 2 levels of the number of reassembly workstations (1 and 2). The data were generated using the method explained earlier. Test results are summarized in Table 6 that shows the number of test instances that the CPLEX gave the optimal solutions, its CPU seconds, and the gaps from the optimal solution values or lower bounds for the best rule combination SPT-EDD-LPT. As expected, the CPU seconds of CPLEX increase sharply as the problem size increases. In fact, some test instances with 13 products could not be solved using the CPLEX within 3600 s. Also, it can be

Table 2 Test results for the instances with tight due dates

NP	NC	NR	Priority rules							
			SPT	EDD	SLACK	MDD	CR	COVERT	RMDD	LPT
(a) Agreeable instances										
20	3	2	0.30(0.00,0.90)	0.22(0.00,0.97)	0.95(0.58,1.00)	0.94(0.58,1.00)	0.29(0.08,0.84)	0.70(0.41,1.00)	0.10(0.00,0.25)	0.24(0.00,0.79)
		3	0.23(0.00,0.54)	0.19(0.00,0.71)	0.91(0.44,1.00)	0.91(0.44,1.00)	0.33(0.00,0.84)	0.47(0.06,1.00)	0.91(0.44,1.00)	0.13(0.00,0.41)
	5	2	0.46(0.07,0.79)	0.19(0.00,1.00)	0.99(0.97,1.00)	0.97(0.89,1.00)	0.26(0.00,1.00)	0.41(0.05,0.87)	0.97(0.89,1.00)	0.16(0.00,0.53)
		3	0.43(0.22,0.65)	0.19(0.00,0.59)	1.00(1.00,1.00)	0.98(0.93,1.00)	0.30(0.00,0.55)	0.41(0.01,0.58)	0.98(0.93,1.00)	0.14(0.00,0.37)
40	3	2	0.39(0.01,0.86)	0.09(0.00,0.57)	0.99(0.96,1.00)	0.95(0.87,1.00)	0.35(0.00,0.75)	0.49(0.06,0.92)	0.95(0.87,1.00)	0.17(0.00,0.34)
		3	0.46(0.29,0.91)	0.02(0.00,0.12)	0.97(0.88,1.00)	0.95(0.85,1.00)	0.22(0.02,0.67)	0.41(0.18,0.76)	0.95(0.85,1.00)	0.19(0.00,0.43)
	5	2	0.74(0.43,1.00)	0.12(0.00,0.39)	0.93(0.66,1.00)	0.91(0.79,1.00)	0.14(0.00,0.48)	0.31(0.00,0.61)	0.91(0.79,1.00)	0.11(0.00,0.3)
		3	0.73(0.19,1.00)	0.13(0.00,0.55)	0.90(0.41,1.00)	0.86(0.39,1.00)	0.12(0.00,0.44)	0.34(0.00,0.56)	0.86(0.39,1.00)	0.1(0.00,0.34)
60	7	2	0.74(0.48,1.00)	0.04(0.00,0.12)	0.91(0.68,1.00)	0.89(0.68,1.00)	0.08(0.00,0.19)	0.10(0.00,0.29)	0.89(0.68,1.00)	0.07(0.00,0.19)
		3	0.74(0.15,1.00)	0.05(0.00,0.20)	0.89(0.60,1.00)	0.87(0.59,1.00)	0.09(0.00,0.21)	0.09(0.03,0.20)	0.87(0.59,1.00)	0.02(0.00,0.12)
	9	2	0.79(0.55,1.00)	0.03(0.00,0.11)	0.94(0.63,1.00)	0.92(0.59,1.00)	0.06(0.00,0.18)	0.10(0.00,0.20)	0.92(0.59,1.00)	0.04(0.00,0.16)
		3	0.78(0.52,1.00)	0.09(0.00,0.24)	0.89(0.63,1.00)	0.89(0.64,1.00)	0.10(0.00,0.21)	0.09(0.00,0.20)	0.89(0.64,1.00)	0.06(0.00,0.18)
80	7	2	0.93(0.72,1.00)	0.03(0.00,0.18)	0.95(0.82,1.00)	0.89(0.66,0.96)	0.09(0.00,0.36)	0.10(0.00,0.26)	0.89(0.66,0.96)	0.06(0.00,0.24)
		3	0.92(0.75,1.00)	0.00(0.00,0.06)	0.93(0.74,1.00)	0.92(0.68,1.00)	0.08(0.00,0.25)	0.14(0.03,0.22)	0.92(0.68,1.00)	0.09(0.00,0.22)
	9	2	0.81(0.57,1.00)	0.01(0.00,0.09)	0.90(0.56,1.00)	0.87(0.55,0.98)	0.08(0.00,0.24)	0.12(0.00,0.22)	0.87(0.55,0.98)	0.06(0.00,0.20)
		3	0.77(0.04,1.00)	0.23(0.00,1.00)	0.78(0.02,1.00)	0.77(0.02,1.00)	0.14(0.00,0.99)	0.11(0.00,0.57)	0.77(0.02,1.00)	0.07(0.00,0.51)
Average			0.64	**0.10**	0.93	0.91	**0.17**	0.27	0.85	**0.11**
(b) Non-agreeable instances										
20	3	2	0.23(0.00,0.86)	0.52(0.04,0.83)	0.94(0.58,1.00)	0.90(0.58,1.00)	0.48(0.12,0.89)	0.62(0.14,1.00)	0.19(0.00,0.61)	0.11(0.00,0.62)
		3	0.14(0.00,0.31)	0.47(0.12,1.00)	0.84(0.44,1.00)	0.83(0.44,1.00)	0.51(0.00,1.00)	0.40(0.03,1.00)	0.83(0.44,1.00)	0.05(0.00,0.16)
	5	2	0.40(0.00,0.79)	0.33(0.00,1.00)	0.92(0.56,1.00)	0.95(0.56,1.00)	0.59(0.00,0.72)	0.33(0.00,0.56)	0.95(0.56,1.00)	0.13(0.00,0.58)
		3	0.33(0.16,0.60)	0.28(0.00,0.49)	1.00(1.00,1.00)	0.93(0.67,1.00)	0.42(0.00,0.72)	0.30(0.01,0.53)	0.93(0.67,1.00)	0.01(0.00,0.09)
40	3	2	0.31(0.00,0.76)	0.60(0.32,1.00)	0.94(0.69,1.00)	0.91(0.77,1.00)	0.56(0.00,0.82)	0.40(0.06,0.82)	0.91(0.77,1.00)	0.09(0.00,0.31)
		3	0.36(0.00,0.79)	0.33(0.00,0.87)	0.98(0.91,1.00)	0.91(0.80,1.00)	0.34(0.06,0.65)	0.29(0.00,0.70)	0.91(0.80,1.00)	0.03(0.00,0.12)
	5	2	0.70(0.31,1.00)	0.22(0.00,0.57)	0.95(0.68,1.00)	0.87(0.62,0.99)	0.34(0.00,0.64)	0.26(0.00,0.53)	0.87(0.62,0.99)	0.06(0.00,0.28)
		3	0.70(0.10,1.00)	0.23(0.00,0.58)	0.90(0.41,1.00)	0.79(0.35,1.00)	0.18(0.00,0.46)	0.29(0.03,0.51)	0.79(0.35,1.00)	0.03(0.00,0.18)
60	7	2	0.74(0.42,1.00)	0.05(0.00,0.18)	0.88(0.42,1.00)	0.88(0.54,1.00)	0.11(0.00,0.37)	0.08(0.00,0.43)	0.88(0.54,1.00)	0.04(0.00,0.19)
		3	0.73(0.13,1.00)	0.15(0.00,0.32)	0.88(0.55,1.00)	0.90(0.60,1.00)	0.14(0.06,0.22)	0.08(0.01,0.2)	0.90(0.60,1.00)	0.00(0.00,0.04)
	9	2	0.80(0.48,1.00)	0.07(0.00,0.17)	0.93(0.58,1.00)	0.92(0.54,1.00)	0.08(0.00,0.17)	0.07(0.00,0.14)	0.92(0.54,1.00)	0.00(0.00,0.02)
		3	0.78(0.48,1.00)	0.09(0.00,0.29)	0.90(0.61,1.00)	0.89(0.62,1.00)	0.12(0.00,0.24)	0.07(0.00,0.19)	0.89(0.62,1.00)	0.03(0.00,0.18)
80	7	2	0.97(0.89,1.00)	0.04(0.00,0.20)	0.95(0.87,1.00)	0.93(0.78,1.00)	0.10(0.00,0.21)	0.07(0.00,0.14)	0.93(0.78,1.00)	0.03(0.00,0.10)
		3	0.9(0.69,1.00)	0.14(0.03,0.22)	0.92(0.62,1.00)	0.92(0.57,1.00)	0.10(0.00,0.22)	0.07(0.00,0.21)	0.92(0.57,1.00)	0.02(0.00,0.12)

Table 2 (continued)

NP	NC	NR	Priority rules							
			SPT	EDD	SLACK	MDD	CR	COVERT	RMDD	LPT
	9	2	0.82(0.50,1.00)	0.08(0.00,0.20)	0.88(0.67,1.00)	0.88(0.57,1.00)	0.11(0.00,0.25)	0.08(0.00,0.20)	0.88(0.57,1.00)	0.02(0.00,0.06)
		3	0.93(0.81,1.00)	0.06(0.00,0.20)	0.91(0.56,1.00)	0.88(0.53,1.00)	0.09(0.00,0.22)	0.07(0.01,0.14)	0.88(0.53,1.00)	0.03(0.00,0.14)
	Average		0.62	**0.23**	0.92	0.89	**0.27**	**0.22**	0.85	**0.04**

See the footnotes of Table 1

Table 3 Results of the Duncan's multiple range test

Priority rules	Agreeable instances					Non-agreeable instances				
	Clusters					Clusters				
EDD	1					1				
CR	1					1	2			
LPT		2					2	3		
RMDD			3					3	4	
COVERT			3						4	5
SLACK			3						4	5
MDDT			3						4	5
SPT			3							5

seen from the table that the gaps of SPT-EDD-LPT are quite high because it is based on the simple rule based scheduling approach. Nevertheless, it is better than the other priority rules and rule combinations tested in this study and useful for the situations that the decision time is very critical.

5 Concluding remarks

This study considered the scheduling problem in remanufacturing systems with a single disassembly workstation, parallel flow-shop-type reprocessing lines, and parallel reassembly workstations. The problem is to determine the sequence of products to be disassembled at the disassembly workstation, the sequence of components to be reprocessed at each workstation of the flow-shop-type reprocessing lines, and the allocation and sequence of the products to be reassembled at the parallel reassembly workstations. Unlike the previous study, we considered the due date-based objective of minimizing the total tardiness and parallel workstations at the reassembly process. A mixed integer programming model was developed to represent the problem mathematically, and then, a priority rule based solution approach was proposed. To test the performances of various priority rules, simulation experiments were done on a number of test instances,

Table 4 Improvement over the previous approach of Kim et al. [19]

Number of products	Agreeable instances	Non-agreeable instances
20	36.3[a]	24.3
40	61.1	45.1
60	85.3	84.6
80	76.9	85.7
Average	64.9	59.9

[a] Average percentage improvement of EDD over EDD-FCFS-FCFS out of 40 agreeable instances with tight due dates

Table 5 Improvement over the single rule approach

Number of products	Agreeable instances	Non-agreeable instances
20	39.0[a]	51.4
40	50.2	84.0
60	28.2	62.3
80	17.8	46.6
Average	33.8	61.1

[a] Average percentage improvement of SPT-EDD-LPT (rule combination) over EDD (single rule) out of 40 agreeable test instances with tight due dates

and the best priority rules were identified. In particular, we showed from additional tests that the solution approach proposed in this study outperforms the previous one that determines the reprocessing and reassembly schedules according to the sequence of disassembling products significantly and also the rule combination approach that uses different rules on disassembly, reprocessing, and reassembly shops outperforms the single rule approach that uses the same rule over the three subsystems.

Table 6 Test results for small sized instances

NP	NC	NR	CPLEX		SPT-EDD-LPT
			NO[a]	CPU[b]	Gap[c]
5	2	1	10	0.1	0.52
		2	10	0.1	0.30
	3	1	10	0.1	0.39
		2	10	0.1	0.29
8	2	1	10	7.8	0.94
		2	10	2.3	0.76
	3	1	10	3.5	1.00
		2	10	2.3	1.16
10	2	1	9	262.4	3.88
		2	10	3.8	0.90
	3	1	9	21.4	2.27
		2	10	12.2	1.24
13	2	1	7	104.4	4.13
		2	9	224.7	2.54
	3	1	8	820.3	3.63
		2	10	595.9	2.21
Average			9.5		1.71

See the footnotes of Table 1

[a] Number of instances that the CPLEX gave the optimal solutions out of 10 instances within 3600 s

[b] Average CPU second for the instances that the optimal solution can be obtained out of 10 instances

[c] Average gap from optimal solution values or lower bounds out of 10 instances

This study can be extended in several directions. In the theoretical aspect, it is needed to develop optimal algorithms after characterizing the problem properties. Also, various meta-heuristics can be developed to improve the solution quality. Finally, for practical applications, other considerations, such as sequence-dependent setups, non-dedicated reprocessing lines, finite buffers, and uncertainties, can be additionally incorporated into the problem considered in this study.

References

1. Steinhilper R (1998) Remanufacturing: the Ultimate Form of Recycling. Fraunhofer IRB Verlag
2. Lund RT (1984) Remanufacturing. Technol Rev 87:19–29
3. Guide VDR Jr (2000) Production planning and control for remanufacturing: industry practice and research needs. J Oper Manag 18:467–483
4. Sundin E, Bras B (2005) Making functional sales environmentally and economically beneficial through product remanufacturing. J Clean Prod 33:913–925
5. Lee CY, Cheng TC, Lin BMT (1993) Minimizing the makespan in the 3-machine assembly-type flowshop scheduling problem. Manag Sci 39:616–625
6. Al-Anzi FS, Allahverdi A (2007) A self-adaptive differential evolution heuristic for two stage assembly scheduling problem to minimize maximum lateness with setup times. Eur J Oper Res 182:80–94
7. Al-Anzi FS, Allahverdi A (2009) Heuristics for a two-stage assembly flow shop with bicriteria of maximum lateness and makespan. Comput Oper Res 36:2682–2689
8. Allahverdi A, Al-Anzi FS (2009) The two-stage assembly scheduling problem to minimize total completion time with setup times. Comput Oper Res 36:2740–2747
9. Shokrollahpour E, Zandieh M, Dorri B (2011) A novel imperialist competitive algorithm for bi-criteria scheduling of the assembly flowshop problem. Int J Prod Res 49:3087–3103
10. Choi H-S, Kim J-S, Lee D-H (2011) Real-time scheduling for reentrant hybrid flow shops: a decision tree based mechanism and its application to a TFT-LCD line. Expert Syst Appl 38:3514–3521
11. Chamnanlor C, Sethanan K, Chien C-F, Gen M (2013) Hybrid genetic algorithms for solving reentrant flow-shop scheduling with time windows. Industrial Engineering & Management Systems 12:306–316
12. Yang J (2015) Hybrid flow shop with parallel machines at the first stage and dedicated machines at the second stage. Industrial Engineering & Management Systems 14:22–31
13. Linn R, Zhang W (1999) Hybrid flow shop scheduling: a survey. Comput Ind Eng 37:57–61
14. Guide VDR Jr (1995) A simulation model of drum-buffer-rope for production planning and control at a naval aviation depot. SIMULATION 65:157–168
15. Guide VDR Jr (1996) Scheduling using drum-buffer-rope in a remanufacturing environment. Int J Prod Res 34:1081–1091
16. Guide VDR Jr (1997) Scheduling with priority dispatching rules and drum-buffer-rope in a recoverable manufacturing system. Int J Prod Econ 53:101–116
17. Guide VDR Jr (1997) An evaluation of order release strategies in a remanufacturing environment. Comput Oper Res 24:37–47

18. Guide VDR Jr, Kraus ME, Srivastava R (1997) Scheduling policies for remanufacturing. Int J Prod Econ 48:187–204
19. Kim M-G, Yu J-M, Lee D-H (2015) Scheduling algorithms for remanufacturing systems with parallel flow-shop-type reprocessing lines. Int J Prod Res 53:1819–1831
20. Kizilkaya E, Gupta SM (1998) Material flow control and scheduling in a disassembly environment. Comput Ind Eng 35:93–96
21. Udomsawat G, Gupta SM (2005) Multi-kanban mechanism for appliance disassembly. Proceedings of the SPIE International Conference on Environmentally Conscious Manufacturing: 30–41
22. Yu J-M, Kim J-S, Lee D-H (2011) Scheduling algorithms to minimize the total family flow time for job shops with job families. Int J Prod Res 49:6885–6903
23. Kim J-S, Park J-H, Lee D-H (2016) Iterated greedy algorithms to minimize total family flow time for job shop scheduling with job families and sequence-dependent set-ups. Eng Optim. doi:10.1080/0305215X.2016.1261247
24. Du J, Leung JYT (1990) Minimizing total tardiness on one processor in NP-hard. Math Oper Res 15:483–495
25. Baker KR (1974) Introduction to sequencing and scheduling. John Wiley and Sons, New York