International Journal of Latest Engineering and Management Research (IJLEMR)
ISSN: 2455-4847
www.ijlemr.com || Volume 07 - Issue 05 || May 2022 || PP. 13-26

Teaching-Learning-Based Optimization Algorithm for Solving U-Shaped Sequence-Dependent Disassembly Line Balancing Problem with Multiple Objectives

Pengfei Yao[1], Surendra M. Gupta[1, *]
[1]*(Department of Mechanical and Industrial Engineering, Northeastern University, USA)*

Abstract: Increased environmental awareness by individuals and governments have brought environmental consciousness into focus. Stricter regulations discourage traditional landfilling methods to deal with end-of-life (EOL) products. In recent years, product recovery is gaining popularity as it is one of the most ecofriendly methods in dealing with EOL products. Remanufacturing is a popular method of product recovery and has gained recognition in recent years. Disassembly plays an important first step in remanufacturing and aims to separate EOL products into subassemblies/parts and materials to achieve predefined objectives. Disassembly tasks are performed on a paced disassembly line with workstations linked together. One of the most essential tasks is to optimally balance the disassembly line. Operators and/or intelligent robotics on a U-shaped disassembly line can work across workstations, which is the most important advantage of a U-shaped disassembly line over a straight-line configuration. This characteristic improves line efficiency and smoothness. A mixed-integer non-linear programming (MINLP) model can be formulated to address four different objectives, viz., minimum number of workstations, maximum line smoothness, early removal of hazardous part(s), and early removal of highly demanded part(s). Considering the NP-hard nature of the DLBP, a novel optimization meta-heuristic algorithm, namely, teaching-learning-based optimization (TLBO) is proposed in this paper to obtain near-optimal solutions. Two sets of instances are used to help test the performance of TLBO and the difference between U-shaped and straight-line layouts. Case studies and comparative studies illustrate that the proposed TLBO algorithm is suitable for use in the DLBP field and it has a superior performance against other known algorithms. In addition, U-shaped layout improves line efficiency and smoothness compared to the traditional straight-line configuration.

Keywords: Remanufacturing, Disassembly Line Balancing, U-Shaped Disassembly line, Teaching-learning-based optimization (TLBO).

I. Introduction

Development of manufacturing and economic in one hand greatly boosts product prosperity, but in another hand, it accelerates the process of product becomes end-of-life products. The fast updated mechanical and electrical products (MEP), such as TV sets, refrigerators, laptops, have a shorted lifecycle than before [1]. Environmental protection consciousness increased since pollution and waste problem are accelerating in this world[2]. Contrast to traditional landfilling method, product recovery is one of the most efficient and ecofriendly methods in sustainable development and it avoids serious pollution and wastage of available resources[3]. Remanufacturing as an important process in product recovery gains more attentions recently and it is an efficient approach to deal with EOL products and resource recovery[4,5,6]. Disassembly is the first and one of the most crucial steps in remanufacturing which aims to physically separately EOL products into subassemblies/parts that are worthy to be future remanufactured, reused, or recycled[7]. EOL products are operated on a paced disassembly line, and there are four typical types of a disassembly line, viz., straight-line, U-shaped, parallel, and two-sided. To professionally disassemble EOL products within the consideration of objectives, balancing the disassembly line is important.

The first research of disassembly line balancing problem (DLBP) is studied by Gungor and Gupta [8]. Since then, DLBP has become an active research area and obtains rapid development. DLBP contains consideration of types of EOL products, uncertainty of EOL products, disassembly line, objectives, constrains, algorithms, mathematical model, case studies. Therefore, the nature of DLBP is optimal assignment of disassembly tasks at workstation with the domain of objectives and constraints[9,10]. Different objectives will affect near-optimal solutions, and singe profit-based or cost-based objective is not enough in research and industrial manufacturing [11]. In this paper, four objectives are considered, viz., minimizing number of workstations, minimizing total idle time, removing of hazardous parts early, and removing high demand parts early. Two basic constraints used in this paper are cycle time and precedence relationship constraints. Cycle time constraint should be strictly followed to avoid line stoppage [12]. Precedence relationship constraint describes sequence order rules among tasks, and this paper utilized AND/OR graph which is commonly used in

DLBP field. In fact, mutual interference between tasks is exist in real situations which is a kind of uncertainty of disassembly task processing time [13-19]. In real world disassembly process, there are two modes of disassembly which are complete and partial disassembly. Complete disassembly requires that all EOL parts should be disassembled, and partial disassembly only requires to disassemble some parts with hazardous or high-demand index, and rest of parts are left for other consideration.

Optimization approaches and algorithms are gradually introduced on DLBP field and especially sequence-dependent DLBP (SDLBP) after research Kalayci and Gupta [9]. Approaches and algorithms include artificial bee colony algorithm [9], improved discrete artificial bee colony algorithm [20], particle swarm optimization algorithm[13], ant colony optimization [14], simulated annealing algorithm[15], river formation dynamics approach[16], tabu search algorithm [17], hybrid genetic algorithm[18], iterated local search method[19], improved cuckoo search algorithm [21], pareto-discrete hummingbird algorithm [1].

The rest of paper is structured as follows: literature review is included in the second section. The section that follows introduces the detailed disassembly line balancing problem and introduces a mixed-integer non-linear programming (MINLP) model and related constrains. This is followed by a section that covers detailed results and comparison of the performance of several algorithms. The last section provides the conclusion and directions for future research.

II. Literature Review

After the concept of DLBP proposed, scholars conducted in-depth research on DLBP field and extended it in different aspects.

2.1 Line layout

There are four main line configurations of a disassembly, viz., straight-line, parallel, U-shaped, and two-sided. According to the data of Ozceylan et al. [22], most of the studies focus on straight-line layout, research on U-shaped, parallel, and two-sided lines are limited. Agrawal and Tiwari [23] first applied a collaborative ant colony algorithm (ACO) on a U-shaped disassembly line, and since then, U-shaped line has become an attractive research area. Most recently, in 2021, Yao and Gupta [6,10,24-27] has for the first time proposed and introduced five novel meta-heuristics on a U-shaped disassembly line, viz., cat swarm optimization (CSO), small world optimization (SWO), ant colony optimization (ACO), invasive weed optimization (IWO), teaching-learning-based optimization (TLBO), and fish school search optimization (FSS) and tested the performance of proposed algorithm with many meta-heuristics.Wang et al. [28] implemented a meta-heuristic approach which combined multi-criterion decision making (MCDM) and variable neighborhood search (VNS) on a U-shaped layout. Li et al. [29]improved a two-phase artificial bee colony algorithm (ABC) and an original bee algorithm (BA) to tackle U-shaped DLBP with the considering of multiple objectives. Research on special disassembly like sequence-dependent and partial U-shaped DLBP are extremely limited. Sequence-dependent U-shaped problem was first studied by Li, Kucukkoc, and Zhang in 2019, and they applied an iterated local search method which is easy to implement and test. Wang, Gao, and Li [30] and Li and Janardhanan [31] considered partial disassembly on a U-shaped disassembly line.

2.2 Disassembly level

Complete and partial disassembly are two modes based on disassembly level. Complete disassembly aims to disassemble all the parts of EOL products, whereas partial disassembly only requires removing needed parts of EOL products. According to the research [22,32], most of the DLBP related studies focus on complete disassembly and the number of partial disassembly is limited. For partial disassembly, Ren et al. [33] introduced an improved gravitational search algorithm. Bentaha et al. [34] studied profit oriented partial DLBP with uncertain task processing times. Wang, Gao, and Li, in 2020, considered partial destructive mode of a U-shaped disassembly line. For all above reasons, the main contributions of this paper are listed as follows:

(1) A mixed-integer non-linear programming (MINLP) model is established to deal with sequence-dependent U-shaped DLBP (SUDLBP) which is capable of solving complex precedence relationships.
(2) Since DLBP is proven to be NP-hard problem [35,36], a novel meta-heuristic algorithm, teaching-learning-based optimization algorithm (TLBO) is introduced to help solve SUDLBP in a reasonable computation time. TLBO as a nature-inspired algorithm is a population-based approach which is simple to be implemented [37].
(3) Small-size instances and large-size benchmark instances are utilized to test the proposed model and algorithm. Case study illustrates greater performance of U-shaped line against tradition straight-line configuration and results of comparative study verify that TLBO performs much better than other compared meta-heuristic algorithms.

III. Problem Definition

DLBP presumes to optimally assign tasks to workstations with the following of multiple constraints. Cycle time as mentioned above, should be strictly followed, since task failure is not accepted in this paper. Precedence relationship contains three different types, viz., AND precedence, OR precedence, and complex AND/OR precedence relationships. Notice that the proposed MINLP model is capable of solving complex precedence relationships. This section presents assumptions, notations, and mathematical model of SUDLBP.

3.1 Assumptions

Assumptions should be noticed before proposing the model which is listed as follows:

(1) To fully figure out the disassembly process, all parts of EOL product should be disassembled.
(2) Sequence-dependent relationships is existing and actual task processing time of should be reconsidered.
(3) The input EOL products are enough, and all these products are similar.
(4) Line stoppage will not happen since cycle time constrain is strictly followed.

3.2 Notations

Notations	
i,j	Serial number of tasks, $i, j = 1,2, \dots, N$
M	Number of workstations
m	Workstation (sub-station) index, $m = 1,2, \dots, 2M$
t_i'	Processing/removal time of task i, which is not interfered by other tasks
t_i	Actual processing time of task i, which should add sequence dependency
h_i	Binary variable, 1, if task i is hazardous; 0, otherwise
d_j	Demand value of task j
ANDP(i)	Set of AND predecessor of task i
ORP(i)	Set of OR predecessor of task i
CT	Cycle time
T_m	Total task processing times of workstation m
sd_{ij}	Sequence dependent time between task j and task i
F_a	Objective function, $a = 1,2,3,4$
Decision variables	
x_{im}	Binary variable, 1, if task i is assigned to sub-station m; 0, otherwise
x_{imj}	Binary variable, 1, if task is assigned to sub-station m and is operated before task j; 0, otherwise
w_{ij}	Binary variable, 1, if task i is operated before task j; 0, otherwise
ws_m	Binary variable, 1, if workstation m is opened; 0, otherwise
l_i	Position number of task *i* in sequence

Notice that, on a U-shaped disassembly line, workstation is divided into two sub-stations to help classify side of the assigned tasks. Fig. 1 presents a solution on a U-shaped disassembly line. It is clear that there are total 4 workstations, and each workstation is divided into two sub-stations. For example, workstation 3 contains sub-station 3 and 6. The task sequence for this example is 1, 2, 4, 5, 8, 6, 3, 9, and 7 and the position number for task 6 is $l_6 = 6$.

3.3 Objectives

As mentioned in previous sections, single profit-based objective is not enough for real world industrial case. Different directions should be considered to expand the research field. In this paper, four objectives are used. Minimizing number of workstations aims to decrease equipment costs, which is a cost-based consideration. Minimizing total idle times has a goal of making workload uniform and improving line efficiency. Hazardous and high demand parts are two important issues in disassembly. For environmental protection and creating profit, removing hazardous and high demand parts early are two objectives in this paper.

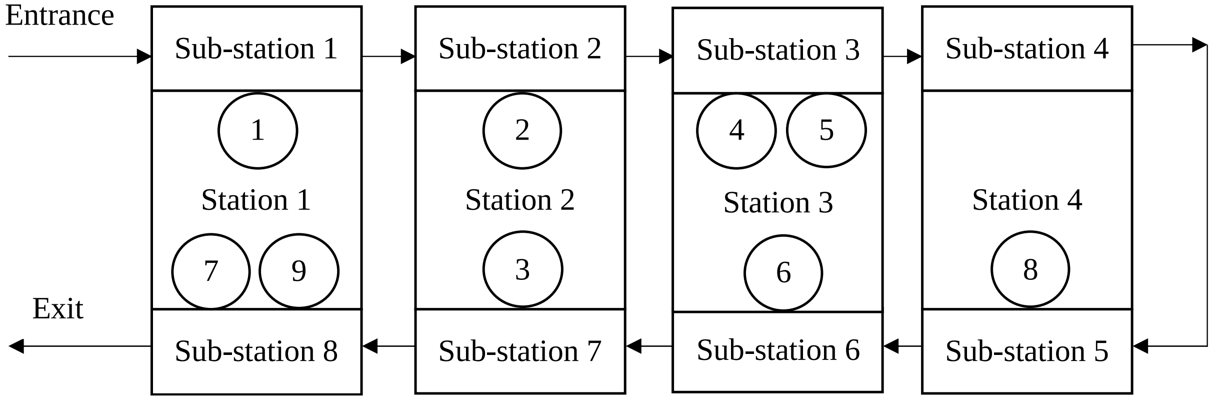

Fig. 1. An example of a solution on U-shaped disassembly line.

Min $F_1 = \sum_{m=1}^{M} ws_m$ (1)
Min $F_2 = \sum_{m=1}^{M} (CT - T_m)^2$ (2)
Min $F_3 = \sum_{i=1}^{N} (l_i * h_i)$ (3)
Min $F_4 = \sum_{i=1}^{N} (l_i * d_i)$ (4)
Subject to:
$\sum_{m=1}^{M} \left(x_{im} + x_{i,2M+1-m}\right) = 1$ (5)
$\sum_{i=1}^{N} \left(x_{im} + x_{i,2M+1-m}\right) \geq 1$ (6)
$CT \geq T_m$ (7)
$x_{im} \leq \sum_{n=1}^{m} x_{jn} \ \forall i, m;\ \forall j \in ANDP(i)$ (8)
$x_{im} \leq \sum_{j \in ORP(i)} \sum_{n=1}^{m} x_{jn} \ \forall m, \forall i \in ORPT$ (9)
$w_{ij} + w_{ji} = 1 \forall i, j \ and \ i < j$ (10)
$T_m = \sum_{i=1}^{N} t_i \times \left(x_{im} + x_{i,2M+1-m}\right) + \sum_{i=1}^{N} \sum_{j=1}^{N} sd_{ji} \times (x'_{imj} + x'_{i,2M+1-m,j}) \ \forall i, j$ (11)
$l_i = N - \sum_{j=1}^{N} w_{ij} \ \forall i$ (12)

Equation (1) and (2) describe minimization number of workstations and increasing line smoothness respectively. Equation (3) and (4) aim to removing hazardous and high demand parts early respectively. Constraint (5) requires that one task should only be assigned to one sub-station. Constraint (6) means there can be one or more tasks in each sub-station. Constraint (7) presents cycle time rule which means task processing time of each workstation should not exceed cycle time. Constraint (8) and (9) together allow that this model can solve complex precedence relationships. Constraint (10) introduces that two disassembly orders of task i and j. If task i is removed before task j, then $w_{ij} = 1, w_{ji} = 0$, otherwise, $w_{ji} = 0, w_{ij} = 0$. Constraint (11) is the calculation process of total task processing time of workstation m. Constraint (12) calculates sequence number of task i.

IV. Teaching-Learning-Based Optimization (TLBO)

For an optimization problem with NP-hard characteristic, exact methods cannot provide enough near-optimal solutions in a reasonable computation time, thus meta-heuristic algorithms can be proposed to solve this problem. The proposed TLBO algorithm was originally introduced by Rao et al., which is a population-based approach. TLBO algorithm utilizes two phases to proceed to the global solution which are ‘Teacher phase’ and ‘Learner phase’. The mechanism of ‘Teacher phase’ is learners get knowledge from teachers and ‘Learner phase’ aims at learning by the interaction between learners. According to the research [37,38], TLBO is a simple algorithm to be implemented and it shows great balance of exploration and exploitation. Basic steps of TLBO algorithm are listed as follows.

4.1 Steps of TLBO algorithm

Step 1: Defining the optimization problem and initializing the optimization parameters.
Step 2: Initializing the population.
Step 3: Starting teacher phase where the main activity is learners learning from their teacher.
Step 4: Starting learner phase where the main activity is learners further tune their knowledge through the interaction with their peers.

Step 5: Evaluating stopping criteria. Terminate the algorithm in the maximum generation number is reached; otherwise return to Step 3 and the algorithm continues.

4.2 Encoding and decoding

Task permutation for encoding follows the same rule with research [9,13-19]. Fig. 2 and Table 1 present information of an 8-part instance. From Fig. 2, dashed lines between task No.2 and 3 and task No.5 and 6 represent sequence-dependent relationship should be considered between connected tasks. Sequence dependencies of this instance are provided as follows: $sd_{23} = 2, sd_{32} = 4, sd_{56} = 1, sd_{65} = 3$. Task assignment of one feasible solution of this small-size instance is shown in Fig. 3. Table 2 and Table 3 present a feasible solution and related calculation processes of objective values. Notice that task permutation for the feasible solution in Fig. 2 is 1, 2, 3, 6, 5, 4, 7, and 8, but actual task sequence is 1, 2, 3, 6, 5, 8, 7, and 4. It is clear that task permutation and actual task sequence is different and for different instances, these two task orders may exist difference. Therefore, a suitable and effective decoding procedure is important to transfer encoding into a feasible solution.

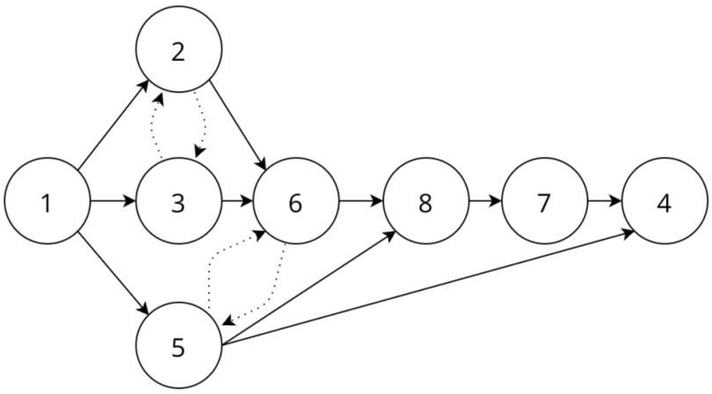

Fig. 2 Precedence relationship of 8-part instance

Table 1. EOL product information of 8-part instance

Task	Part title	Task removal time	Hazardous index	demand
1	PC top cover	14	No	360
2	Floppy drive	10	No	500
3	Hard drive	12	No	620
4	Back plane	18	No	480
5	PCI cards	23	No	540
6	RAM modules	16	No	750
7	Power supply	20	No	295
8	Motherboard	36	No	720

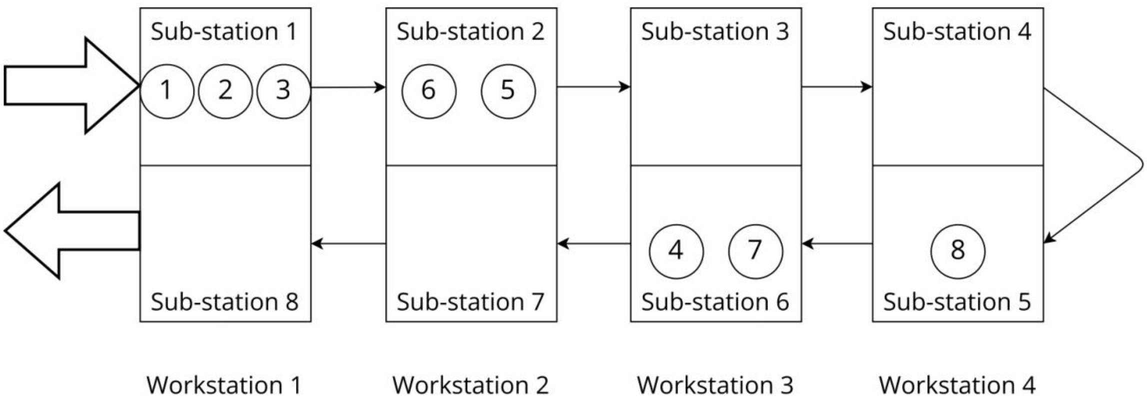

Fig. 3 Feasible solution for 8-part instance

In Table 2, there are four workstations, and each workstation contains two sub-stations. For example, workstation 1 is divided into sub-station 1 and 8 and sub-station 1 operates task 1, 2 and 3. For sub-station 8, no task is assigned in it, therefore, task processing time for it is 0. Idle time for each workstation is listed in the last column. Idle time for workstation 1 and 2 is 0, for workstation 3 is 2 and for workstation 4 is 4. Table 3 presents objective values of this feasible solution.

Table 2. Task assignment and important issues

Workstation number	Sub-station number	Task number	Task processing time	Total task processing time	Idle time
Workstation 1	Sub-station 1	1,2,3	14,10+4,12	40	0
	Sub-station 8	-	-		
Workstation 2	Sub-station 2	6,5	16+1,23	40	0
	Sub-station 7	-	-		
Workstation 3	Sub-station 3	-	-	38	2
	Sub-station 6	7,4	20,18		
Workstation 4	Sub-station 4	-	-	36	4
	Sub-station 5	8	36		

Table 3. Calculation of objective values

Objective	Value
F_1	4
F_2	$0+0+2^2+4^2=20$
F_3	0 (No hazardous task)
F_4	1*360+2*500+3*620+4*750+5*540+6*720+7*295+8*480=19145

The procedure of SUDLBP is complicated than that of SDLBP, since two sides of the workstation are considered. Algorithm 1 presents decoding steps of SUDLBP. In decoding procedure, A_{en} is the available task set for entrance side and A_{ex} is the available task set for exit side. Also, AS_{en} and AS_{ex} represent entrance side assignable task set and exit side assignable task set respectively.

Algorithm1. Decoding procedure

Start

Step 1: If all tasks are disassembled, terminate procedure; otherwise, execute step 2.

Step 2: Open a new workstation for task assignment.

Step 3: (1) Add task(s), whose predecessor(s) has been assigned to the entrance side, to the available task set A_{en} .(2) Add task(s), whose successor(s) has been assigned to the exit side, to the available task set A_{ex}.

Step 4: (1) Add the task in A_{en} to the assignable task set AS_{en} on the entrance side with the domain of cycle time constraint. (2) Add the task in A_{ex} to the assignable task set AS_{ex} on the exit side with the domain of cycle time constraint. % For an assignable task, it can be assigned only the total task processing time of this workstation is less than or equal to the given cycle time with the considering of sequence dependency.

Step 5: If both two assignable task sets AS_{en} and AS_{ex} are empty, go back to step 1; otherwise, execute step 6.

Step 6: Select the task with higher priority of task permutation and allocate it to the entrance or exit side based on the situation; go back to step 3.

End

V. Case Study and Comparative Study

This section provides computational studies to test the searching ability and suitability of proposed model and algorithm. Benchmark problems are utilized and include two instance sets. The first instance set has two small-size cases which are acquired from research Kalayci and Gupta in 2013. The second instance set contains 47 cases with different problem size. The smallest-size Mertens instance is a 7-part problem and the largest-size Barthol 2 has 148 parts. Notice that, in this paper, hierarchy method is used to find near-optimal solutions and the first objective has highest priority and the fourth objective has lowest priority. The proposed TLBO algorithm is applied on a straight-line and a U-shaped line separately for each benchmark instances, and on each line TLBO run 20 times respectively.

5.1 Case study

The first case is a 10-part PC instance (P10) and the second case contains 25 parts (P25). Table 4 and Fig. 4 present product information of P10, and Table 5 and Fig. 5 introduce basic task data of P25. In Table 4, it is clear that task 7 is the only hazardous task. Cycle time for P10 is 40 and sequence dependencies of P10 instance are as follows: $sd_{1,4} = 1$, $sd_{4,1} = 4$, $sd_{2,3} = 2$, $sd_{3,2} = 3$, $sd_{4,5} = 4$, $sd_{5,4} = 2$, $sd_{5,6} = 2$, $sd_{6,5} = 4$, $sd_{6,9} = 3$, and $sd_{9,6} = 1$. The second case is containing 25 parts and the cycle time of P25 instance is 18. Sequence dependencies of P25 instance are shown as follows: $sd_{4,5} = 2$, $sd_{5,4} = 1$, $sd_{6,7} = 1$, $sd_{7,6} = 2$, $sd_{6,9} = 2$, $sd_{9,6} = 1$, $sd_{7,8} = 1$, $sd_{8,7} = 2$, $sd_{13,14} = 1$, $sd_{14,13} = 2$, $sd_{14,15} = 2$, $sd_{15,14} = 1$, $sd_{20,21} = 1$, $sd_{21,20} = 2$, $sd_{22,25} = 1$, and $sd_{25,22} = 2$. Table 6 and Table 7 present detailed results of P10 and P25.

Table 4. Data of 10-part instance

Task number	Part removal time	Hazardous index	Demand
1	14	No	0
2	10	No	500
3	12	No	0
4	17	No	0
5	23	No	0
6	14	No	750
7	19	Yes	295
8	36	No	0
9	14	No	360
10	10	No	0

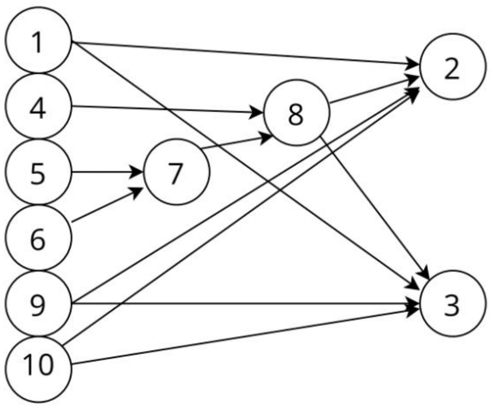

Fig. 4 Precedence relationship among 10 tasks

Table 5. Information for P25 instance

Task number	Part name	Part removal time	Hazardous index	Demand value
1	Antenna	3	1	4
2	Battery	2	1	7
3	Antenna guide	3	0	1
4	Bolt (Type 1) A	10	0	1
5	Bolt (Type 1) B	10	0	1
6	Bolt (Type 2) 1	15	0	1
7	Bolt (Type 2) 2	15	0	1
8	Bolt (Type 2) 3	15	0	1
9	Bolt (Type 2) 4	15	0	1
10	Clip	2	0	2
11	Rubber Seal	2	0	1
12	Speaker	2	1	4
13	White Cable	2	0	1
14	Red/Blue Cable	2	0	1
15	Orange Cable	2	0	1
16	Metal Top	2	0	1

17	Front Cover	2	0	2
18	Back Cover	3	0	2
19	Circuit Board	18	1	8
20	Plastic Screen	5	0	1
21	Keyboard	1	0	4
22	LCD	5	0	6
23	Sub-keyboard	15	1	7
24	Internal IC Board	2	0	1
25	Microphone	2	1	4

In Table 6 and Table 7, best and average value of 20-time runs, and standard deviation of results are listed. In Table 6, results on a U-shaped line got better results against increasing line smoothness and removing high demand parts early. In Table 7, all the compared objective values on a U-shaped line are better or equal to that on a straight-line configuration. Therefore, it may conclude that U-shaped layout can provide more task assignments and improve line efficiency. Notice that TLBO got the same best results compared with algorithm used in research Li, Kucukkoc, and Zhang in 2019, thus illustrating TLBO is capable of solving DLBP.

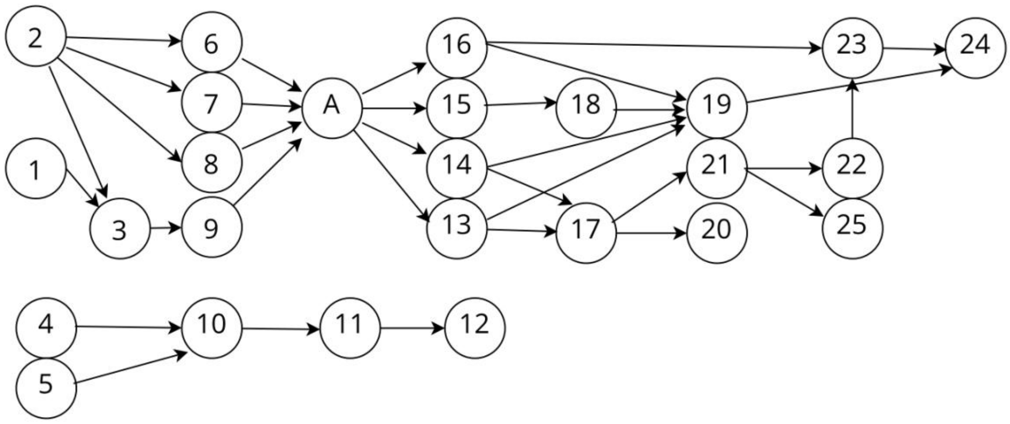

Fig. 5 Precedence relationship of P25 instance

Table 6. Performance of different layouts for P10

Line type	Algorithm	Evaluation	F_1	F_2	F_3	F_4
SDLBP	TLBO	Best value	5	67	5	9605
		Avg. value	5.00	67.00	5.00	9605.00
		S. D	0.00	0.00	0.00	0.00
SUDLBP	TLBO	Best value	5	61	6	8880
		Avg. value	5.00	61.00	6.00	8880.00
		S. D	0.00	0.00	0.00	0.00

Table 7. Performance of different layouts for P25

Line type	Algorithm	Evaluation	F_1	F_2	F_3	F_4
SDLBP	TLBO	Best value	10	9	80	925
		Avg. value	10.00	9.00	80.00	925.00
		S. D	0.00	0.00	0.00	0.00
SUDLBP	TLBO	Best value	10	9	76	884
		Avg. value	10.00	9.00	77.45	908.75
		S. D	0.00	0.00	2.32	9.08

5.2 Comparative study

This section provides comparation results of different algorithms. To test the performance of TLBO on a straight-line and a U-shaped line, a genetic algorithm combined with variable neighborhood search approach (VNSGA) [18] and an iterated local search method (ILS) [19] are utilized. Table 8 listed detailed best results of the first two objectives, and results of VNSGA and ILS are acquired from above mentioned research. In Table 8, TLBO is compared with VNSGA and ILS on straight-line layout and also, on U-shaped line TLBO is compared with ILS. For SDLBP, in terms of F_1,TLBO obtains 7 better and 0 worse results against VNSGA, and 47 same solutions against ILS, also, in terms of F_2, TLBO obtains 23 better solutions against VNSGA, and 19 better solutions against ILS. For SUDLBP, TLBO has 47 same results against ILS in terms of F_1, and 12 better results against ILS in terms of F_2. It may conclude that the great searching ability of TLBO make it suitable for SDLBP and SUDLBP. Also, results on U-shaped layout are much better against solutions on traditional straight-line configuration. Therefore, U-shaped layout improve line efficiency and smoothness again.

Table 8. Performance of VNSGA, ILS, and TLBO

Instance	N	CT	VNSGA (SDLBP)		ILS (SDLBP)		TLBO (SDLBP)		ILS (SUDLBP)		TLBO (SUDLBP)	
			F_1	F_2	F_1	F_2	F_1	F_2	F_1	F_2	F_1	F_2
Mertens	7	7	5	10	5	10	5	10	5	10	5	10
Bowman	8	20	5	149	5	149	5	149	4	13	4	13
Jaeschke	9	7	7	26	7	28	7	28	7	28	7	26
Jackson	11	10	5	6	5	6	5	6	5	4	5	4
Mansoor	11	94	2	5	2	5	2	5	2	5	2	5
Mitchell	21	15	8	31	8	43	8	31	8	29	8	31
Roszieg	25	16	8	5	8	5	8	5	8	3	8	3
Heskiaoff	28	216	5	628	5	630	5	628	5	628	5	628
Buxey	29	30	12	118	12	122	12	122	11	6	11	6
Lutzl	32	2357	7	8.13E+05	7	8.47E+05	7	8.33E+05	7	7.99E+05	7	8.05E+05
Gunther	35	41	14	1519	14	1735	14	1629	12	13	12	13
Kilbridge	45	62	9	6	9	6	9	6	9	6	9	6
Hahn	53	2806	6	1.87E+06	6	1.91E+06	6	1.89E+06	5	6	5	6
Tonge	70	168	22	2152	22	1756	22	1926	22	1672	22	1824
Tonge	70	170	22	3002	22	2660	22	2872	21	204	21	366
Tonge	70	173	22	5196	21	1081	21	2134	21	745	21	985
Tonge	70	179	21	3459	20	312	20	674	20	262	20	278
Tonge	70	182	20	968	20	912	20	968	20	854	20	902
Wee-Mag	75	46	35	983	34	399	34	563	34	349	34	367
Wee-Mag	75	47	33	148	33	116	33	126	33	106	33	116
Wee-Mag	75	49	32	189	32	163	32	163	32	155	32	155
Wee-Mag	75	50	32	347	32	333	32	341	32	327	32	331
Wee-Mag	75	52	31	455	31	443	31	449	31	431	31	443
Arcus1	83	3985	20	9.34E+05	20	9.22E+05	20	9.20E+05	20	8.14E+05	20	8.10E+05
Arcus1	83	5048	16	1.76E+06	16	1.76E+06	16	1.76E+06	16	1.67E+06	16	1.67E+06
Arcus1	83	5853	14	2.79E+06	14	2.79E+06	14	2.77E+06	13	1.16E+04	13	2.55E+06
Arcus1	83	6842	12	4.26E+06	12	4.25E+06	12	4.25E+06	12	3.43E+06	12	3.37E+05
Arcus1	83	7571	11	5.37E+06	11	5.54E+06	11	5.49E+06	11	5.37E+06	11	5.35E+06
Arcus1	83	8412	10	7.09E+06	10	7.83E+06	10	7.21E+06	10	7.93E+06	10	7.09E+06
Arcus1	83	8898	9	2.14E+06	9	2.15E+06	9	2.14E+06	9	2.13E+06	9	2.12E+06
Arcus1	83	10816	8	1.49E+07	8	3.75E+07	8	1.37E+07	7	1.10E+01	7	1.11E+07
Lutz2	89	15	34	63	34	61	34	61	33	10	33	10
Lutz3	89	150	12	2050	12	2256	12	1890	11	6	11	6
Mukherjee	94	201	23	12057	23	14853	23	14091	21	13	21	13
Mukherjee	94	301	15	10137	15	10137	15	10137	14	6	14	402
Arcus2	111	5755	27	2.58E+06	27	2.40E+06	27	2.54E+06	27	1.06E+06	27	1.08E+06
Arcus2	111	7520	21	3.00E+06	21	2.97E+06	21	2.98E+06	21	2.75E+06	21	2.74E+06
Arcus2	111	8847	18	4.38E+06	18	4.59E+06	18	4.42E+06	18	4.41E+06	18	4.38E+06
Arcus2	111	10027	16	6.33E+06	16	6.39E+06	16	6.37E+06	16	6.42E+06	16	6.35E+06
Arcus2	111	10743	15	7.76E+06	15	7.82E+06	15	7.76E+06	15	7.81E+06	15	7.75E+06
Arcus2	111	11378	14	5.76E+06	14	5.72E+06	14	5.74E+06	14	5.68E+06	14	5.70E+06
Arcus2	111	11570	14	9.86E+06	14	1.02E+07	14	9.80E+06	14	9.63E+06	14	9.59E+06
Arcus2	111	17067	9	1.14E+06	9	1.14E+06	9	1.14E+06	9	1.14E+06	9	1.14E+06
Barthol2	148	85	52	906	51	293	51	417	51	243	51	271
Barthol2	148	89	50	1174	49	425	49	477	48	74	48	95
Barthol2	148	91	49	1179	48	504	48	499	47	67	47	67
Barthol2	148	95	47	1279	46	454	46	449	45	53	45	51

TLBO algorithm is compared with 9 other algorithms which include hill-climbing algorithm (HC), late acceptance hill-climbing algorithm (LAHC) [39], simulated annealing algorithm (SA), tabu search algorithm

(TS), genetic algorithm (GA), artificial bee colony algorithm (ABC), bee algorithm (BA), particle swarm optimization (PSO), and iterated local search optimization (ILS). Notice that Table 9 and Table 10 provide comparation average values of 10 algorithms on F_1 and F_2 respectively, and results of HC, LAHC, and ILS are taken from mentioned research directly. Results of SA, TS, GA, ABC, BA, and TLBO are implemented 20 times on a U-shaped layout.

Table 9. Results of different algorithms on minimizing number of workstations

Instance	N	CT	HC	LAHC	SA	TS	GA	ABC	BA	PSO	ILS	TLBO
Mertens	7	7	5	5	5	5	5	5	5	5	5	5
Bowman	8	20	4	4	4	4	4	4	4	4	4	4
Jaeschke	9	7	7	7	7	7	7	7	7	7	7	7
Jackson	11	10	5	5	5	5	5	5	5	5	5	5
Mansoor	11	94	2	2	2	2	2	2	2	2	2	2
Mitchell	21	15	8	8	8	8	8	8	8	8	8	8
Roszieg	25	16	8	8	8	8	8	8	8	8	8	8
Heskiaoff	28	216	5	5	5	5	5	5	5	5	5	5
Buxey	29	30	11	11.05	11	11	11	11.15	11	11	11	11
Lutzl	32	2357	7	7	7	7	7	7	7	7	7	7
Gunther	35	41	12	12	12	12	12	12	12	12	12	12
Kilbridge	45	62	9	9	9	9	9	9	9	9	9	9
Hahn	53	2806	5.7	5.65	5.5	5.65	5.85	5.85	5.5	5.8	5.2	5.25
Tonge	70	168	22	22	22	22.15	22	22.05	22	22	22	22
Tonge	70	170	21.95	21.95	22.00	21.95	21.95	22.00	21.95	21.5	21.8	21.75
Tonge	70	173	21	21	21	21.15	21	21	21.3	21	21	21
Tonge	70	179	20	20	20	20	20.3	20	20.5	20	20	20
Tonge	70	182	20	20	20	20	20	20	20	20	20	20
Wee-Mag	75	46	34	34	34	34.3	34	34	34.15	34	34	34
Wee-Mag	75	47	33	33	33	33	33.5	33	33	33	33	33
Wee-Mag	75	49	32	32	32	32	32	32	32	32	32	32
Wee-Mag	75	50	32	32	32.3	32	32	32	32	32	32	32
Wee-Mag	75	52	31	31	31	31	31	31	31.05	31	31	31
Arcus1	83	3985	20	20	20	20	20	20	20	20	20	20
Arcus1	83	5048	16	16	16	16	16	16	16	16	16	16
Arcus1	83	5853	13	13	13	13.5	13	13.75	13	13	13	13
Arcus1	83	6842	12	12	12.3	12	12	12	12	12	12	12
Arcus1	83	7571	11	11	11	11	11	11	11	11.25	11	11
Arcus1	83	8412	10	10	10	10	10	10	10	10	10	10
Arcus1	83	8898	9	9	9	9.15	9	9	9	9	9	9
Arcus1	83	10816	8	8	8.05	8	8	7.95	8	8	7.8	7.95
Lutz2	89	15	33	33	33	33.25	33.25	33	33	33.15	33	33
Lutz3	89	150	11	11	11.25	11	11	11	11	11.05	11	11
Mukherjee	94	201	21.25	21.2	21.50	21.95	21.95	22.00	21.75	21.75	21.25	21.25
Mukherjee	94	301	14	14	14.95	15.05	14.30	14.75	14.25	14.15	14	14.3
Arcus2	111	5755	27	27	27	27	27	27	27	27	27	27
Arcus2	111	7520	21	21	21	21.15	21	21	21	21	21	21
Arcus2	111	8847	18	18	18	18	18	18	18	18	18	18
Arcus2	111	10027	16	16	16	16	16	16	16	16	16	16
Arcus2	111	10743	15	15	15	15	15	15	15	15	15	15
Arcus2	111	11378	14	14	14	14	14	14	14	14	14	14
Arcus2	111	11570	14	14	14	14	14	14	14	14	14	14
Arcus2	111	17067	9	9	9	9	9	9	9	9	9	9
Barthol2	148	85	51	51	51.05	51.00	51.75	51.95	51.05	51.00	51.00	51.00
Barthol2	148	89	49	48.9	48.95	49.00	48.95	49.15	49.00	49.15	48.75	48.75
Barthol2	148	91	48	47.8	47.90	48.00	48.00	48.00	48.15	48.10	47.60	47.50
Barthol2	148	95	45.9	45.85	46.15	46.00	46.00	46.00	45.85	46.15	45.65	45.60

Table 10. Results of different algorithms in terms of minimizing total idle times

Instance	N	CT	HC	LAHC	SA	TS	GA	ABC	BA	PSO	ILS	TLBO
Mertens	7	7	10	10	10	10	10	10	10	10	10	10
Bowman	8	20	13	13	13	13	13	13	13	13	13	13
Jaeschke	9	7	28	28	28	28	28	28	28	28	28	28

Jackson	11	10	4	4	4	4	4	4	4	4	4	4
Mansoor	11	94	5	5	5	5	5	5	5	5	5	5
Mitchell	21	15	30.7	31	30.5	30.7	29.9	30.3	29.9	29.5	29.1	31.0
Roszieg	25	16	3.2	3.9	3.0	3.1	3.0	3.0	3.0	3.0	3.0	3.0
Heskiaoff	28	216	634.8	636.4	630.4	629.9	631.0	630.5	629.3	628.7	629.1	628.4
Buxey	29	30	8.4	15.8	8.4	6.8	7.2	6.9	6.7	6.4	6.5	6.4
Lutzl	32	2357	838157	830279	821497	835620	808962	814793	823537	826561	804475	812253
Gunther	35	41	13	13.4	13.7	13.5	13.9	13.2	13.6	13.2	13.1	13.1
Kilbridge	45	62	6.2	8.9	6.1	6..0	6.3	6.0	6.0	6.2	6.0	6.0
Hahn	53	2806	1E+06	1E+06	1E+06	1E+06	1E+06	841705	1E+06	1E+06	344411	306257
Tonge	70	168	1805.5	1811.3	1920.5	1837.9	1801.6	1792.9	2093.4	1847.0	1783.0	1906.4
Tonge	70	170	2690.9	2651.8	3037.9	2590.6	2437.1	2985.6	3250.5	2574.8	2159.8	2437.9
Tonge	70	173	1088.8	1719.7	1109.3	1205.6	1005.0	1492.3	1334.8	947.2	954.1	1026.4
Tonge	70	179	325.6	518.5	314.2	375.4	792.3	305.6	324.1	293.4	290.8	295.5
Tonge	70	182	934	1685.7	895.6	899.0	935.2	919.8	1025.4	887.3	879.9	925.4
Wee-Mag	75	46	475.4	457.5	387.6	412.5	396.0	457.1	421.9	396.7	426.7	382.5
Wee-Mag	75	47	128.5	118.0	125.9	119.4	109.5	125.7	119.4	120.4	117.3	125.1
Wee-Mag	75	49	159.9	159.5	159.5	160.2	159.5	161.7	158.4	160.9	159.3	159.2
Wee-Mag	75	50	337.8	331.5	335.7	334.5	339.8	352.4	341.5	337.4	330.5	332.1
Wee-Mag	75	52	446.9	444.4	449.5	473.2.1	440.4	453.5	442.5.2	448.6	437.8	450.2
Arcus1	83	3985	838896	835347	873257	839674	840938	840206	853791	828563	827898	815724
Arcus1	83	5048	2E+06	2E+06	2E+06	2E+06	2E+06	2E+06	2E+06	2E+06	2E+06	2E+06
Arcus1	83	5853	13515	19389	2E+06	2E+06	2E+06	20453	2E+06	2E+06	12786	2E+06
Arcus1	83	6842	4E+06	4E+06	4E+06	4E+06	3E+06	3E+06	4E+06	4E+06	4E+06	4E+06
Arcus1	83	7571	6E+06	6E+06	7E+06	6E+06	6E+06	6E+06	6E+06	6E+06	6E+06	6E+06
Arcus1	83	8412	1E+07	1E+07	9E+06	1E+07	1E+07	9E+06	1E+07	1E+07	1E+07	9E+06
Arcus1	83	8898	2E+06	2E+06	2E+06	2E+06	2E+06	2E+06	2E+06	2E+06	2E+06	2E+06
Arcus1	83	10816	4E+07	4E+07	5E+07	5E+07	4E+07	4E+07	4E+07	3E+07	3E+07	3E+07
Lutz2	89	15.0	10.3	16.5	17.5	15.8	17.4	10.9	11.9	10.5	10.1	10.1
Lutz3	89	150	6.4	10.7	245.3	7.0	8.4	10.2	22.9	6.6	6.6	6.4
Mukherjee	94	201	588.25	475.1	1099.8	1325.4	609.8	897.4	2125.3	1742.6	564.35	507.4
Mukherjee	94	301	14.4	16.5	2335.4	19.8	973.6	2794.5	3393.1	2893.0	9.6	758.4
Arcus2	111	5755	1E+06	2E+06	1E+06	1E+06	1E+06	1E+06	1E+06	1E+06	1E+06	1E+06
Arcus2	111	7520	3E+06	3E+06	3E+06	3E+06	3E+06	3E+06	3E+06	3E+06	3E+06	3E+06
Arcus2	111	8847	5E+06	5E+06	5E+06	5E+06	5E+06	5E+06	5E+06	6E+06	5E+06	5E+06
Arcus2	111	10027	7E+06	7E+06	7E+06	7E+06	7E+06	8E+06	7E+06	7E+06	7E+06	7E+06
Arcus2	111	10743	8E+06	8E+06	9E+06	1E+07	8E+06	9E+06	9E+06	8E+06	8E+06	8E+06
Arcus2	111	11378	6E+06	6E+06	6E+06	6E+06	7E+06	6E+06	6E+06	6E+06	6E+06	6E+06
Arcus2	111	11570	1E+07	1E+07	1E+07	1E+07	1E+07	1E+07	1E+07	1E+07	1E+07	1E+07
Arcus2	111	17067	1E+06	1E+06	1E+06	1E+06	1E+06	1E+06	1E+06	1E+06	1E+06	1E+06
Barthol2	148	85	259.8	258.4	273.1	328.5	259.4	293.7	451.4	309.1	257.4	292.5
Barthol2	148	89	371.2	346.0	417.5	429.5	773.1	375.0	396.2	334.9	294.65	312.6
Barthol2	148	91	414.0	362.4	479.1	455.2	392.2	489.3	427.5	339.8	281.3	237.5
Barthol2	148	95	419.4	396.95	720.5	426.5	492.3	862.4	451.0	624.3	311.65	305.4

From Table 9 and Table 10, TLBO performs much better than other algorithms, and it obtains 42 best or same results against other algorithms in terms of F_1, and 33 best or same average values compared with others in terms of F_2. Therefore, the proposed TLBO algorithm has a superior performance and based on hierarchy method, TLBO outperforms other algorithms in finding near-optimal solutions.

VII. Conclusion

Disassembly line balancing problem is one of the most active research topics in industrial since environmental protection consciousness is accepted gradually by individuals and governments. Product recovery and remanufacturing not only solve pollution and waste problem but also create profits from EOL products. This paper has for the first-time proposed teaching-learning-based optimization (TLBO) algorithm on a disassembly line with the consideration of multiple objectives, U-shaped layout, and sequence-dependent situation. In the meanwhile, a mixed-integer non-linear programming model is introduced to deal with SUDLBP with the ability of solving complex precedence relationship. Case studies and comparative studies provide enough evidence of superior performance of TLBO and improved efficiency by using U-shaped layout.

In the future, with the consideration of large-size instances, combined approaches and improved algorithms can be more interesting especially adding high-quality initial population method to a meta-heuristic algorithm. Complex disassembly layout like U-shaped, parallel, and two-sided are attractive to explore. Also, during the COVID-19 pandemic, automatic and/or intelligent robotics-controlled workstations may expand the DLBP field.

References

[1] Yin, T., Zhang, Z. and Jiang, J., 2021. A Pareto-discrete hummingbird algorithm for partial sequence-dependent disassembly line balancing problem considering tool requirements. *Journal of Manufacturing Systems*, *60*, pp.406-428.

[2] Yin, T., Zhang, Z., Zhang, Y., Wu, T. and Liang, W., 2022. Mixed-integer programming model and hybrid driving algorithm for multi-product partial disassembly line balancing problem with multi-robot workstations. *Robotics and Computer-Integrated Manufacturing*, *73*, p.102251.

[3] Cheng, C.Y., Chen, Y.Y., Pourhejazy, P. and Lee, C.Y., 2022. Disassembly Line Balancing of Electronic Waste Considering the Degree of Task Correlation. *Electronics*, *11*(4), p.533.

[4] Gao, Y., Lou, S., Zheng, H. and Tan, J., 2021. A data-driven method of selective disassembly planning at end-of-life under uncertainty. *Journal of Intelligent Manufacturing*, pp.1-21.

[5] Akpinar, M.E., Ilgin, M.A. and Aktaş, H., 2021. Disassembly Line Balancing by Using Simulation Optimization. *Alphanumeric Journal*, *9*(1), pp.63-84.

[6] Yao, P. and Gupta, S. M., 2021a. Cat Swarm Optimization Algorithm for Solving Multi-Objective U-Shaped Disassembly Line Balancing Problem. *Proceedings of the International Conference on Remanufacturing*, March 24-25, pp. 222-230.

[7] Gungor, A. and Gupta, S.M., 1999b. Issues in environmentally conscious manufacturing and product recovery: a survey. *Computers & Industrial Engineering*, 36(4), pp.811-853.

[8] Gungor, A., & Gupta, S. M., 1999a. Disassembly line balancing. *Proceedings of the 1999 Annual Meeting of the Northeast Decision Sciences Institute, Newport*, Rhode Island, March 24-26, pp.193-195.

[9] Kalayci, C.B. and Gupta, S.M., 2013a. Artificial bee colony algorithm for solving sequence-dependent disassembly line balancing problem. *Expert Systems with Applications*, *40*(18), pp.7231-7241.

[10] Yao, P. and Gupta, S. M., 2021b. Small World Optimization Algorithm for Solving Multi-Objective U-Shaped Disassembly Line Balancing Problem, *Proceedings of the 2021 Annual Meeting of the Northeast Decision Sciences Institute*, Virtual, March 26-27, 659-668.

[11] Gungor, A., Gupta, S.M., Pochampally, K. and Kamarthi, S.V., 2001, February. Complications in disassembly line balancing. In *Environmentally conscious manufacturing* (Vol. 4193, pp. 289-298). International Society for Optics and Photonics.

[12] Bentaha, M.L., Battaïa, O. and Dolgui, A., 2014. Lagrangian relaxation for stochastic disassembly line balancing problem. *Procedia Cirp*, *17*, pp.56-60.

[13] Kalayci, C.B. and Gupta, S.M., 2013b. A particle swarm optimization algorithm with neighborhood-based mutation for sequence-dependent disassembly line balancing problem. *The International Journal of Advanced Manufacturing Technology*, *69*(1), pp.197-209.

[14] Kalayci, C.B. and Gupta, S.M., 2013c. Ant colony optimization for sequence-dependent disassembly line balancing problem. *Journal of Manufacturing Technology Management*.

[15] Kalayci, C.B. and Gupta, S.M., 2013d. Simulated annealing algorithm for solving sequence-dependent disassembly line balancing problem. *IFAC Proceedings Volumes*, *46*(9), pp.93-98.

[16] Kalayci, C.B. and Gupta, S.M., 2013e. River formation dynamics approach for sequence-dependent disassembly line balancing problem. *Reverse supply chains: issues and analysis*, pp.289-312.

[17] Kalayci, C.B. and Gupta, S.M., 2014. A tabu search algorithm for balancing a sequence-dependent disassembly line. *Production Planning & Control*, *25*(2), pp.149-160.

[18] Kalayci, C.B., Polat, O. and Gupta, S.M., 2016. A hybrid genetic algorithm for sequence-dependent disassembly line balancing problem. *Annals of Operations Research*, *242*(2), pp.321-354.

[19] Li, Z., Kucukkoc, I. and Zhang, Z., 2019. Iterated local search method and mathematical model for sequence-dependent U-shaped disassembly line balancing problem. *Computers & Industrial Engineering*, *137*, p.106056.

[20] Liu, J. and Wang, S., 2017. Balancing disassembly line in product recovery to promote the coordinated development of economy and environment. *Sustainability*, *9*(2), p.309.

[21] Zhou, R., Guo, X., Fu, Y. and Qi, L., 2019, October. Solving Sequence-dependent Disassembly Line Balancing Problem with Improved Cuckoo Search Algorithm. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)* (pp. 403-408). IEEE.

[22] Özceylan, E., Kalayci, C.B., Güngör, A. and Gupta, S.M., 2019. Disassembly line balancing problem: a review of the state of the art and future directions. *International Journal of Production Research, 57*(15-16), pp.4805-4827.

[23] Agrawal, S., and Tiwari, M.K., 2008. A collaborative ant colony algorithm to stochastic mixed-model U-shaped disassembly line balancing and sequencing problem. *International Journal of Production Research*, *46*(6), pp.1405-1429.

[24] Yao, P. and Gupta, S. M., 2021c. Ant Colony Optimization Algorithm for Solving U-Shaped Disassembly Line Balancing Problem with Multiple Objectives, *Proceedings of the 4th International Conference on Innovative Studies of Contemporary Sciences*, Tokyo, Japan, July 29-31, pp. 21-26.

[25] Yao, P. and Gupta, S. M., 2021d. Invasive Weed Optimization Algorithm for Solving Multi-Objective U-Shaped Disassembly Line Balancing Problem", *Proceedings of the 12th International Congress on Mathematics, Engineering and Natural Sciences*, Paris, France, July 9-11, pp. 286-292.

[26] Yao, P. and Gupta, S. M., 2021e. Teaching-Learning-Based Optimization Algorithm for Solving Multi-Objective U-Shaped Disassembly Line Balancing Problem, *Proceedings of the 5th International New York Conference on Evolving Trends in Interdisciplinary Research and Practices*, Manhattan, New York City, October 3-5, pp. 21-28.

[27] Yao, P. and Gupta, S. M., 2021f. Fish School Search Algorithm for Solving Multi-Objective U-Shaped Disassembly Line Balancing Problem, *Proceedings of the Latin American International Conference on Natural and Applied Sciences*, Villahermosa, Mexico, November 5-6, pp. 44-52.

[28] Wang, Y., Xie, Y., Ren, Y. and Zhang, C., 2021, February. A MCDM-Based Meta-Heuristic Approach for U-shaped Disassembly Line Balancing Problem. In *Journal of Physics: Conference Series* (Vol. 1828, No. 1, p. 012159). IOP Publishing.

[29] Li, Z., Kucukkoc, I., Tang, Q. and Zhang, Z., 2021. Models and two-phase bee algorithms for multi-objective U-shaped disassembly line balancing problem. *Optimization and Engineering*, pp.1-32.

[30] Wang, K., Gao, L. and Li, X., 2020. A multi-objective algorithm for U-shaped disassembly line balancing with partial destructive mode. *Neural Computing and Applications*, *32*(16), pp.12715-12736.

[31] Li, Z. and Janardhanan, M.N., 2021. Modelling and solving profit-oriented U-shaped partial disassembly line balancing problem. *Expert Systems with Applications*, *183*, p.115431.

[32] Laili, Y., Li, Y., Fang, Y., Pham, D.T. and Zhang, L., 2020. Model review and algorithm comparison on multi-objective disassembly line balancing. *Journal of Manufacturing Systems*, *56*, pp.484-500.

[33] Ren, Y., Yu, D., Zhang, C., Tian, G., Meng, L. and Zhou, X., 2017. An improved gravitational search algorithm for profit-oriented partial disassembly line balancing problem. *International Journal of Production Research*, *55*(24), pp.7302-7316.

[34] Bentaha, M.L., Dolgui, A., Battaïa, O., Riggs, R.J. and Hu, J., 2018. Profit-oriented partial disassembly line design: dealing with hazardous parts and task processing times uncertainty. *International Journal of Production Research*, *56*(24), pp.7220-7242.

[35] McGovern, S.M. and Gupta, S.M., 2007a. A balancing method and genetic algorithm for disassembly line balancing. *European journal of operational research*, *179*(3), pp.692-708.

[36] McGovern, S.M. and Gupta, S.M., 2007b. Combinatorial optimization analysis of the unary NP-complete disassembly line balancing problem. *International Journal of Production Research*, *45*(18-19), pp.4485-4511.

[37] Rao, R.V., Savsani, V.J. and Vakharia, D.P., 2011. Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. *Computer-aided design*, *43*(3), pp.303-315.

[38] Singh, R.K., Singh, A.R. and Yadav, R.K., 2021, September. Disassembly Line Balancing Using Recursive Optimization in Presence of Task-Failure. In *IFIP International Conference on Advances in Production Management Systems* (pp. 430-440). Springer, Cham.

[39] Yuan, B., Zhang, C. and Shao, X., 2015. A late acceptance hill-climbing algorithm for balancing two-sided assembly lines with multiple constraints. *Journal of Intelligent Manufacturing*, *26*(1), pp.159-168.