International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 10 Issue 6 ǁ 2022 ǁ PP. 1705-1716

Invasive Weed Optimization Algorithm for Solving Multi-Objective Sequence-Dependent U-Shaped Disassembly Line Balancing Problem

Pengfei Yao[1], Surendra M. Gupta[1, *]
[1]Department of Mechanical and Industrial Engineering
Northeastern University Boston, MA, 02115, USA.
Corresponding author: Surendra M. Gupta

ABSTRACT
Environmental protection has become a global problem since the deteriorating environment greatly threatens human existence. Due to rapid developments in technology and manufacturing methods, the number of waste products is increasing faster than ever. Product recovery is one of the most effective strategies to deal with the waste problem. Compared to the traditional landfilling, product recovery adds green concept in dealing with the waste problem as well as retrieving valuable parts and materials from end-of-life (EOL) products. Popular choices in product recovery are remanufacture, reuse, and recycle. Disassembly is an unavoidable step in remanufacturing and is an essential step in its profit-based objectives. With a higher line efficiency and flexibility, U-shaped disassembly line has many more advantages compared to the traditional straight-line disassembly line. In order to optimize the disassembly operations, it is necessary to solve the disassembly line balancing problem (DLBP). The DLBP involves optimally allocating tasks/parts to workstations within the domain of cycle time and precedence relationship constraints. Characteristics of a U-shaped disassembly line give this layout more chances to find optimal tasks assignment. A mixed-integer non-linear programming model (MINLP) with four different objectives and constraints can be formulated to solve such problem. However, since the DLBP is an NP-hard problem, a novel meta-heuristic algorithm called invasive weed optimization algorithm (IWO) is applied on a straight-line and a U-shaped disassembly line separately. IWO is based on the concept of natural selection (survival of the fittest). Two sets of instances are applied to test the performance of the proposed IWO algorithm. Case studies show that the proposed MINLP model can find optimal solution(s) for small-size instances, and the U-shaped disassembly line performs better than the traditional straight-line disassembly line. A comparative study demonstrates that the proposed IWO algorithm is superior in comparison to other meta-heuristic algorithms reported in the literature.
KEYWORDS: *Remanufacturing, Disassembly Line Balancing, U-Shaped Disassembly line, Invasive Weed optimization (IWO).*

Date of Submission: 12-06-2022 Date of acceptance: 26-06-2022

I. INTRODUCTION

Manufacturing prosperity greatly helps the development of society and technology at some degree. One obvious situation is an increasing number of products are manufactured much faster than ever before [1]. Especially using electronic and electricity products are becoming an important part of daily life of individuals. One unavoidable problem which is troubling global individuals and counties is the number of waste products are getting extremely large. The concept of environmentally conscious manufacturing and product recovery was first highlighted by Gungor and Gupta [2]. which aspires to add green manufacturing to the whole life cycle of a product. With the sustainable development and green manufacturing conscious are becoming a responsibility of industry and society [3], landfilling as a traditional way dealing with waste problem is not widely usable. Product recovery is a smart strategy which adding green concept into waste problem [4]. The nature of product recovery is to minimize the amount of waste through different processes which contain reuse, remanufacturing and recycle. Remanufacturing plays an essential role in retrieving valuable parts/materials from end-of-life (EOL) products and create profits step by step. Disassembly is one of the most important steps in remanufacturing, it aims to disassemble EOL products into subassemblies and/or parts. After EOL products entering disassembly process, disassembly tasks are started. Disassembly tasks are operated on a paced disassembly line via linked workstations and operators and/or intelligent robotics are accessed working in the workstation [5]. Therefore, balancing the disassembly line should be getting much more attention.

Disassembly line balancing problem (DLBP) was, for the first time, proposed by research Gungor and Gupta [6], which aims at optimally allocating tasks to workstations with the domain of cycle time constraint and precedence relationship constraint. There are four popular types of a disassembly lines, namely, straight-line, parallel, U-shaped, and two-sided. A U-shaped disassembly line is much more productive and efficient compared to a straight-line configuration since operators and/or intelligent robotics can work across the workstation [7]. Cycle time constraint ensures that there will not happen line stoppage situation and precedence relationship constraint indicate that the sequence rules between disassembly tasks. Sequence dependent relationship is a special type of precedence relationship since it is much more complicated and sequence dependencies should be considered between target tasks [8, 9].

After the pioneering work by Gungor and Gupta [6], DLBP has becoming an active research area and many approaches and heuristics are applied on a disassembly line. Based on the collection data of research Ozceylan et al. [10], DLBP is considered as a multi-criteria decision-making problem [11] by some research. Models and solution approaches contain linear programming [12, 13], non-linear programming [14], heuristic [15, 16], genetic algorithm [17], Greedy algorithm [18], ant colony optimization [19-21], hill climbing, simulated annealing [22], particle swarm optimization [23], artificial bee colony [8], artificial fish swarm optimization [24], small world optimization [25], invasive weed optimization [26], tabu search [27], reinforcement learning technique [28], teaching-learning-based optimization [7], fish school search optimization [29], network-based shortest route model [30], multi-criteria decision-making [31].

The rest of paper is structured as follows: literature review is included in the next 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

There are four typical line types of a disassembly line, viz., straight-line, parallel, U-shaped and two-sided. Previous studies focus more on a straight-line layout, whereas U-shaped DLBP is not considered that many [10]. In 2008, the first research of U-shaped DLBP was studied by Agrawal and Tiwari [19] and since then, U-shaped layout is getting much more attentions. Exact methods, heuristics, and meta-heuristics are continuing applied on a U-shaped disassembly line. Yao and Gupta [5, 7, 21, 25, 26, 29] has for the first time, introduced five meta-heuristic algorithms on U-shaped layout, 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 algorithm (FSS) which expand the field of approaches on DLBP. Sequence-dependent U-shaped DLBP (SUDLBP) was first studied in Li, Kucukkoc, and Zhang [9] and iterated local search strategy was used to help. Wang, Gao, and Li [32] and Li and Janardhanan [33] considered partial disassembly on a U-shaped disassembly line. These two studies enlarge the horizon of UDLBP research. One obvious difference between U-shaped and straight line is operators and/or intelligent robotics can work across the workstation, this may improve line efficiency and productivity.

Considering of uncertainty of real-world instances, sequence dependent situation should be taken into account. According to the research Kalayci and Gupta [8], interactions between tasks will affect task processing times which is called sequence-dependent situation. Notice that sequence dependencies between related tasks are known before disassembling.

McGovern and Gupta [14, 15] mathematically proved that DLBP belongs to NP-hard class problem. Therefore, to obtain near-optimal solutions for large-size instances, heuristics and meta-heuristics are continually introduced in DLBP research field. IWO algorithm is a novel optimization meta-heuristic algorithm which was originally inspired from the nature of weed. Great searching ability makes IWO algorithm is suitable in solving DLBP. For all reasons above, the main contributions to the literature are listed as follows:

(1) Considering of the evaluation of one objective is nonlinear, a mixed-integer nonlinear programming (MINLP) mathematical model is formulated to help solve multi-objective U-shaped DLBP.

(2) Invasive weed optimization algorithm (IWO), has for the first time, applied to help find near-optimal solutions of sequence-dependent U-shaped DLBP (SUDLBP). It has a great ability to get a suitable balance between exploration and exploitation.

(3) A comprehensive comparative study is conducted on two sets of instances in this paper to evaluate the performance of developed MINLP model and the proposed IWO algorithm. The first instance set contains two small-size cases, and the second instance set includes 47 cases and many of them are large-size instances. Computational tests show that U-shaped disassembly line has greater performance for line smoothness and number of workstations compared with traditional straight-line disassembly line. Total of 10 algorithms are involved to compare the performance of IWO algorithm, and the comparative study demonstrates that the proposed IWO algorithm outperforms other meta-heuristic algorithms on many aspects.

III. PROBLEM DEFINITION

This section first introduces sequence-dependent U-shaped DLBP (SUDLBP) and then presents the proposed MINLP model and detailed constraints.

3.1 Problem description

DLBP is much more complicated than assembly line balancing problem (ALBP). One reason for that is the precedence relationship in ALBP is only AND precedence relationship, but in DLBP, especially real-world instances, there may exist AND precedence, OR precedence, and complex AND/OR precedence relationships. One task can be disassembled only when all its AND predecessors has been disassembled or at least one of its OR predecessors has been moved based on its precedence relationship. Fig. 1 presents a small-size instance, task 2 and task 3 are OR predecessors of task 4 and task 3 and task 4 are AND predecessors of task 5. To optimally allocate disassembly tasks on workstations, a suitable and efficient model should be proposed.

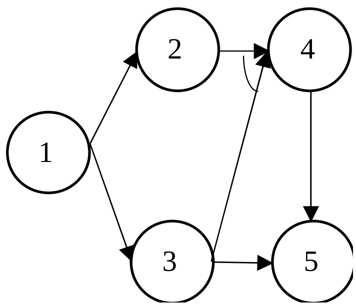

Fig. 1. Precedence relationship of a 5-part instance

3.2 Mathematical model

This section presents a mixed-integer non-linear programming model with core constraints. Before introducing the MINLP model, assumptions and notations should be presented. The assumptions are as follows:

1. EOL products are only one type and number of products is enough.
2. Products should be disassembled completely.
3. Task processing time is known and deterministic.
4. Task failure is not considered.

The notations and decision variables utilized in the model are given as follows.

Notations	
i,j	Task index, $i, j = 1,2, \ldots, N$
M	Number of workstations
m	Workstation (sub-station) index, $m = 1,2, \ldots, 2M$
t_i	Processing/removal time of task i
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
s_i	Position number of task *i* in sequence

Notice that the reason why the maximum amount of m is 2*M is that on a U-shaped line, one workstation has two directions which are entrance side and exit side, therefore, for computation, one workstation is divided into two sub-stations. A feasible solution of a supposed 8-part instance is shown in Fig.2. In Fig.2, four workstations are divided into 8 sub-stations, therefore $M = 4, N = 8$, e.g., workstation 1 has two sub-stations which are sub-station 1 and sub-station 8. The sequence of this feasible solution is 1, 3, 6, 4, 8, 7, 2, 5.

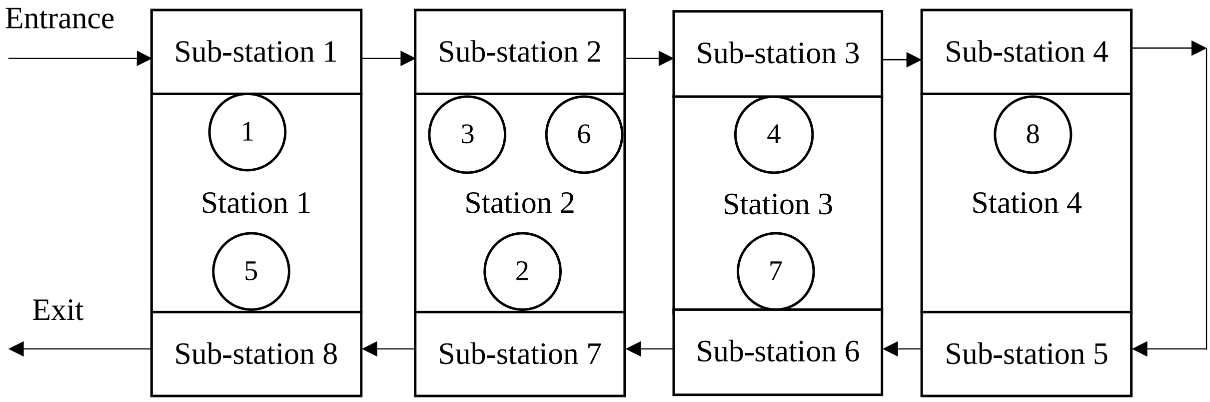

Fig.2 A feasible solution on a U-shaped disassembly line

There are four different objective functions in this paper, viz., minimizing number of workstations, increasing line smoothness, removing hazardous part early, and removing high demand part early. Objective functions are presented as follows:

Objectives:

Min $F_1 = \sum_{m=1}^{M} ws_m$ (1)

Min $F_2 = \sum_{m=1}^{M} (CT - T_m)^2$ (2)

Min $F_3 = \sum_{i=1}^{N} (s_i * h_i)$ (3)

Min $F_4 = \sum_{i=1}^{N} (s_i * d_i)$ (4)

The first objective is to minimize the number of workstations which is a cost-based consideration. Equation (2) is a non-linear function with the goal of optimally increasing line smoothness. Equation (3) presents removal of hazardous part(s) early and equation (4) tries to remove high demand part(s) early. There are two popular ways to classify optimal or near-optimal solutions, viz., pareto optimal strategy [5, 7, 21, 25, 26, 29] and hierarchy method. In this paper, hierarchy method is utilized to compare near-optimal solutions and F_1 has the highest priority and F_2 weighs lower than F_1.

Constraints:

$\sum_{m=1}^{M} \left(x_{im} + x_{i,2M+1-m}\right) = 1$ (5)

$\sum_{i=1}^{N} \left(x_{im} + x_{i,2M+1-m}\right) \geq 1$ (6)

$CT \geq T_m$ (7)

$x_{im} \leq \sum_{n=1}^{m} x_{jn} \quad \forall i, m; \ \forall j \in ANDP(i)$ (8)

$x_{im} \leq \sum_{j \in ORP(i)} \sum_{n=1}^{m} x_{jn} \quad \forall m, \forall i \in ORPT$ (9)

$w_{ij} + w_{ji} = 1 \quad \forall i, j \ and \ i < j$ (10)

$T_m = \sum_{i=1}^{N} t_i \times \left(x_{im} + x_{i,2M+1-m}\right) + \sum_{i=1}^{N} \sum_{j=1}^{N} sd_{ji} \times (x'_{imj} + x'_{i,2M+1-m,j}) \quad \forall i, j$ (11)

$s_i = N - \sum_{j=1}^{N} w_{ij} \ \forall i$ (12)

Constraint (5) and (6) consider the situation of a workstation, they ensure that one task can only be operated in one sub-station and one sub-station can disassemble one or more tasks, separately. Constraint (7) strictly ensures that line stoppage will not happen which is the cycle time constraint. Constraint (8) and (9) consider different types of precedence relationship and these constraints ensure that this MINLP model can solve complicated precedence relationship problem. Constraint (10) determines the value of w_{ij}, if $w_{ij} = 1$, task i will be disassembled before task j. Constraint (11) and (12) are the calculation of total task processing times of a sub-station and sequence number respectively.

IV. INVASIVE WEED OPTIMIZATION ALGORITHM (IWO)

Since DLBP belongs to NP-hard class, therefore dealing with large-size instances, meta-heuristic algorithms perform strong searching ability and can find near-optimal solutions in a relative short computational time. This section presents encoding and decoding procedures of IWO algorithm. IWO was originally proposed in research [34] which has a strong global and local searching ability. The basic searching procedure of IWO are as follows:

Step 1: Initialize randomly generated weeds in the entire search space.

Step 2: Evaluate fitness of the whole population members.

Step 3: Allow each population member to produce a few seeds with better population members produce more seeds (i.e., reproduction).

Step 4: The generated seeds are distributed over the search space by normally distributed random numbers with mean equal to zero but varying variance (i.e., spatial dispersal).
Step 5: When the weed population exceeds the upper limit, perform competitive exclusion.
Step 6: Check the termination criteria.

4.1 Encoding and decoding

In this paper, the encoding of task permutation is same with research Kalayci and Gupta [8, 20, 22, 23, 27]; Li, Kucukkoc, and Zhang [9]. Table 1 and Fig. 3 present information and precedence relationship of an 8-part PC instance respectively, which is acquired from research Kalayci and Gupta [27]. Dashed line in Fig. 3 represents 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. 4.

Table 1. Information of the 8-part PC 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

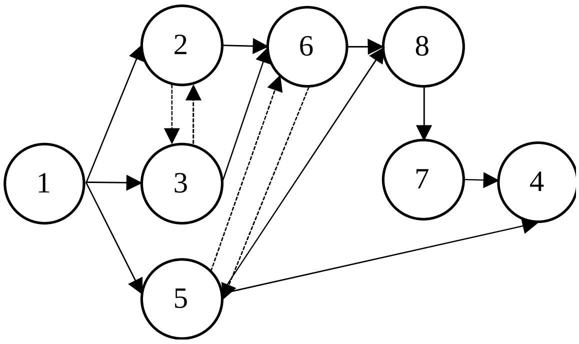

Fig. 3. Precedence relationship of the 8-part PC instance

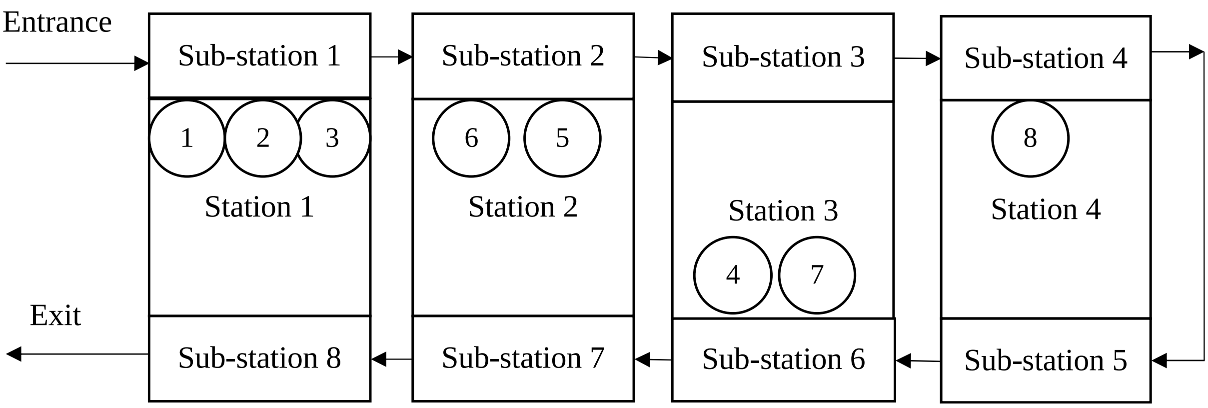

Fig. 4. Task assignment of a feasible solution for 8-part PC instance

As shown in Fig. 4, the task permutation is 1, 2, 3, 6, 5, 4, 7, 8, but the task sequence is 1, 2, 3, 6, 5, 8, 7, 4, which are different. Therefore, a decoding procedure of IWO is proposed to help transfer task permutation into task sequence. For this small-size instance, Table 2 and Table 3 presents objective values and calculation processes of 4 objectives. Task 1, 2, and 3 are assigned at sub-station 1, task 6 and 5 are allocated at sub-station 2, task 8 is disassembled at sub-station 4, and task 7 and 4 are operated at sub-station 6. The number of workstations is 4, and based on the idle times of each workstation, the value of minimizing total of idle times is

20. Since there is no hazardous task, value of the third objective is 0. According to the model, the value of removing high demand part early is 19145.

Table 2. Task allocation of a feasible solution for 8-part PC instance

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	8	36	36	4
	Sub-station 5	-	-		

Table 3. Objective values and calculation processes of the feasible solution

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

The decoding procedure in SUDLBP is different from that in SDLBP, since workstations are divided into sub-stations. Detailed decoding procedure of SUDLBP is presented in Algorithm 1 as follows.

Algorithm 1. Decoding process for SUDLBP

Start
Step 1: If all tasks are assigned, terminate procedure; otherwise, execute step 2.
Step 2: Open a new workstation.
Step 3: Add task(s), whose predecessor(s) has been assigned to the entrance side, to the available task set A_{en}; Add task(s), whose successor(s) has been assigned to the exit side, to the available task set A_{ex}.
Step 4: Add the task in $A_{en}$to the assignable task set AS_{en} on the entrance side with the domain of cycle time constraint; 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. COMPUTATIONAL STUDY AND RESULTS

Two instance sets are utilized in this section to help test the performance of IWO algorithm and the proposed model. The first instance set contains two small-size instances which are taken from previous research Kalayci and Gupta [8]. 47 instances are concluded in the second instance set. The smallest-size instance (Mertens) has 7 tasks, and the largest-size instance (Barthol 2) has 148 tasks. Also, the comparation results of traditional straight-line and U-shaped line are presented. Moreover, IWO algorithm is compared with 9 other algorithms which include hill-climbing algorithm (HC) [14, 15], late acceptance hill-climbing algorithm (LAHC) [35], 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).

5.1 Case study

There are two instances in this section which are 10-part instance (P10) and 25-part telephone instance (P25). Information of two instances is acquired from research Kalayci and Gupta [8]. Table 4 and Fig. 5 present related information of P10 instance. Also, 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 proposed IWO algorithm is applied on straight-line and U-shaped line separately and tested on each line type 20 times. The cycle time for P10 is predetermined as 40. Table 5 clearly presents that the best value of F_1 is 5 in both line types, whereas U-shape line obtains smaller value on F_2. Since hierarchy method is applied in this paper, it is obvious that U-shaped layout performs better. As the standard deviation of all objectives are 0.00, it might be concluded that IWO is robust in solving this instance.

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

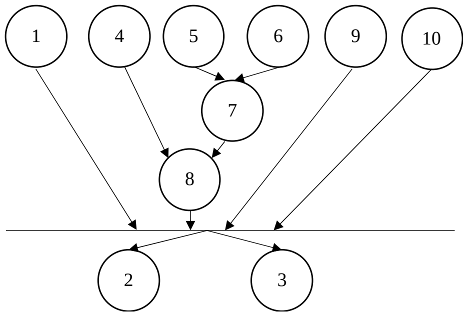

Fig. 5. Precedence relationship of P10

Table 5. Results for P10 instance by IWO algorithm

Line type	Algorithm	Evaluation	F_1	F_2	F_3	F_4
SDLBP	IWO	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	IWO	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

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 $sd_{25,22} = 2$. Fig. 6 and Table 6 show related information of P25 instance. The proposed IWO algorithm was applied on a U-shaped line and a straight line 20 times separately.

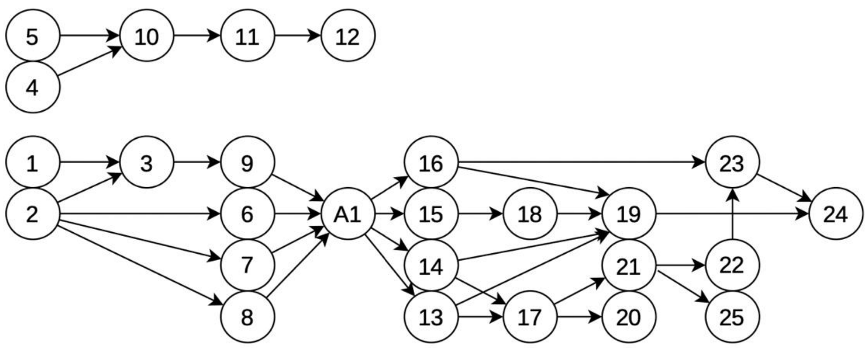

Fig. 6. Precedence relationship of P25 instance

Table 6. Database 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

From Table 7, it is clear that solutions on U-shaped line performs better than that on straight-line layout, especially considering removing hazardous and high demand parts early. Again, for the first two objectives, IWO obtains the same best results reported in Li, Kucukkoc, and Zhang [9], this illustrates that IWO is effective in solving SUDLBP. Also, U-shaped layout indeed can improve line efficiency and productivity in some respects.

Table 7. Results of P25 for two layouts

Line type	Algorithm	Evaluation	F_1	F_2	F_3	F_4
SDLBP	IWO	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	IWO	Best value	10	9	76	910
		Avg. value	10.00	9.00	77.84	914.80
		S. D	0.00	0.00	1.61	5.94

5.2 Comparative study

This section first compares IWO algorithm with genetic algorithm with variable neighborhood search method (VNSGA) [17] and iterated local search approach (ILS) [9] on two aspects. Best results of three algorithms and listed in Table 8. Notice that, in Table 8, results for VNSGA and ILS algorithm are taken from above mentioned research and the first two objectives are taken into consideration.

Table 8. Comparison between VNSGA, ILS and IWO

Instance	N	CT	VNSGA (SDLBP)		ILS (SDLBP)		IWO (SDLBP)		ILS (SUDLBP)		IWO (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	29
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	8.53E+05	7	7.99E+05	7	8.07E+05
Gunther	35	41	14	1519	14	1735	14	1759	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	1.90E+06	5	6	5	6
Tonge	70	168	22	2152	22	1756	22	1962	22	1672	22	1720
Tonge	70	170	22	3002	22	2660	22	2730	21	204	21	568
Tonge	70	173	22	5196	21	1081	22	5196	21	745	21	891
Tonge	70	179	21	3459	20	312	21	2304	20	262	20	294
Tonge	70	182	20	968	20	912	20	956	20	854	20	882
Wee-Mag	75	46	35	983	34	399	34	385	34	349	34	343
Wee-Mag	75	47	33	148	33	116	33	122	33	106	33	112

Wee-Mag	75	49	32	189	32	163	32	167	32	155	32	159
Wee-Mag	75	50	32	347	32	333	32	333	32	327	32	327
Wee-Mag	75	52	31	455	31	443	31	455	31	431	31	441
Arcus1	83	3985	20	9.34E+05	20	9.22E+05	20	9.08E+05	20	8.14E+05	20	8.12E+05
Arcus1	83	5048	16	1.76E+06	16	1.76E+06	16	1.76E+06	16	1.67E+06	16	1.65E+06
Arcus1	83	5853	14	2.79E+06	14	2.79E+06	14	2.79E+06	13	1.16E+04	13	1.15E+05
Arcus1	83	6842	12	4.26E+06	12	4.25E+06	12	4.19E+06	12	3.43E+06	12	3.41E+05
Arcus1	83	7571	11	5.37E+06	11	5.54E+06	11	5.37E+06	11	5.37E+06	11	5.37E+06
Arcus1	83	8412	10	7.09E+06	10	7.83E+06	10	7.17E+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	1.39E+07	7	1.10E+01	7	1.09E+07
Lutz2	89	15	34	63	34	61	34	63	33	10	33	10
Lutz3	89	150	12	2050	12	2256	12	2230	11	6	11	6
Mukherjee	94	201	23	12057	23	14853	23	12975	21	13	21	13
Mukherjee	94	301	15	10137	15	10137	15	10137	14	6	15	2107
Arcus2	111	5755	27	2.58E+06	27	2.40E+06	27	2.36E+06	27	1.06E+06	27	1.04E+06
Arcus2	111	7520	21	3.00E+06	21	2.97E+06	21	2.99E+06	21	2.75E+06	21	2.81E+06
Arcus2	111	8847	18	4.38E+06	18	4.59E+06	18	4.41E+06	18	4.41E+06	18	4.40E+06
Arcus2	111	10027	16	6.33E+06	16	6.39E+06	16	6.33E+06	16	6.42E+06	16	6.31E+06
Arcus2	111	10743	15	7.76E+06	15	7.82E+06	15	7.79E+06	15	7.81E+06	15	7.76E+06
Arcus2	111	11378	14	5.76E+06	14	5.72E+06	14	5.72E+06	14	5.68E+06	14	5.68E+06
Arcus2	111	11570	14	9.86E+06	14	1.02E+07	14	9.90E+06	14	9.63E+06	14	9.60E+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	365	51	243	51	289
Barthol2	148	89	50	1174	49	425	49	573	48	74	48	97
Barthol2	148	91	49	1179	48	504	48	464	47	67	47	65
Barthol2	148	95	47	1279	46	454	46	448	45	53	45	52

From Table 8, IWO algorithm is applied on U-shaped line and straight line separately. It is clear that on a straight-line layout, IWO found 42 same and 5 smaller values compared with VNSGA, and IWO obtained 45 same objective values compared with ILS considering minimizing number of workstations. Also, for increasing line smoothness, IWO found 18 same and 19 better values compared with VNSGA, and IWO obtained 14 same and 19 better results compared with ILS. For SUDLBP, IWO found 46 same and 1worse objective values on F_1, and 18 same and 16 better results on F_2 compared with ILS. Based on comparation results, it might be concluded that IWO has a superior performance, and it is observed that U-shaped disassembly line performs much better than traditional straight-line layout especially on large-size instances.

Table 9. Performance of 10 algorithms on the first objective

Instance	N	CT	HC	LAHC	SA	TS	GA	ABC	BA	PSO	ILS	IWO
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	12.15	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	5.85	5.5	6.0	5.85	5.65	5.85	5.2	5.2
Tonge	70	168	22	22	22	22	22.05	22	22.05	22	22	22
Tonge	70	170	21.95	21.95	22.05	21.85	21.95	22	22	21.95	21.8	21.8
Tonge	70	173	21	21	21.15	21	21	21	21	21	21	21
Tonge	70	179	20	20	20	20	20	20.5	20	20	20	20
Tonge	70	182	20	20	20	20	20	20	20	20	20	20
Wee-Mag	75	46	34	34	34.15	34	34	34.5	34.95	34.05	34	34
Wee-Mag	75	47	33	33	33	33	33	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	32	32	32	32	32	32	32
Wee-Mag	75	52	31	31	31	31.05	31	31	31	31	31	31
Arcus1	83	3985	20	20	20.4	20	20	20.4	20	20	20	20
Arcus1	83	5048	16	16	16	16	16	16	16	16	16	16
Arcus1	83	5853	13	13	13.4	13	13.85	14	13	13.7	13	13
Arcus1	83	6842	12	12	12	12	12	12	12	12	12	12
Arcus1	83	7571	11	11	11	11	11	11	11	11	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	8.3	8	8	8	8	8	7.8	8
Lutz2	89	15	33	33	33.3	33	33.5	33	33.3	33	33	33
Lutz3	89	150	11	11	11	11	11.4	11.25	11	11	11	11
Mukherjee	94	201	21.25	21.2	21.3	22	21.7	22	21.75	21.75	21.25	21.2
Mukherjee	94	301	14	14	14.15	14.3	14.9	15.15	14.75	14.5	14	15
Arcus2	111	5755	27	27	27	27	27	27	27	27	27	27
Arcus2	111	7520	21	21	21	21	21	21	21	21	21	21
Arcus2	111	8847	18	18	18	18	18.15	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	51.95	51.5	51.7	51.7	51.15	51.05	51	51
Barthol2	148	89	49	48.9	48.7	49	49.15	49	49	49.05	48.75	48.7
Barthol2	148	91	48	47.8	47.75	48	48	48	48	48.3	47.6	47.5
Barthol2	148	95	45.9	45.85	46.15	46	45.95	46	45.7	46	45.65	45.5

Table 10. Performance of 10 algorithms on the second objective

Instance	N	CT	HC	LAHC	SA	TS	GA	ABC	BA	PSO	ILS	IWO
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	29.6	29.7	29.7	30.3	30.2	29.7	29.1	29.3
Roszieg	25	16	3.2	3.9	3	3.2	3	3	3.5	3	3	3
Heskiaoff	28	216	634.8	636.4	629.7	633.4	629.7	630.3	628.7	629.5	629.1	629.0
Buxey	29	30	8.4	15.8	9.4	7.9	7.7	6.9	7.1	6.2	6.5	6.2
Lutzl	32	2357	838157	830279	836797	871356	810743	814624	855693	837661	804475	811792
Gunther	35	41	13	13.4	14.1	13.7	14.0	13.3	13.5	13.1	13.1	13.0
Kilbridge	45	62	6.2	8.9	6.3	6.1	6	6	6	6.2	6	6
Hahn	53	2806	1E+06	1E+06	1E+06	997632	1E+06	932610	1E+06	964275	344411	285907
Tonge	70	168	1805.5	1811.3	2105.6	1895.3	1796.8	1919.3	1791.7	1765.9	1783.0	1796.4
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	1356.4	1089.1	992.4	1405.3	1239.5	892.5	954.1	933.9
Tonge	70	179	325.6	518.5	442.6	397.5	1053.4	291.4	315.8	287.4	290.8	330.5
Tonge	70	182	934	1685.7	1105.2	939.4	890.1	1207.9	971.5	934.7	879.9	904.9
Wee-Mag	75	46	475.4	457.5	417.4	387.2	568.3	621.3	379.1	390.6	426.7	373.4
Wee-Mag	75	47	128.5	118.0	117.9	125.4	126.5	133.8	128.1	119.5	117.3	120.1
Wee-Mag	75	49	159.9	159.5	159.4	162.7	160.3	174.5	156.3	167.9	159.3	160.5
Wee-Mag	75	50	337.8	331.5	337.2	332.9	340.5	336.6	339.1	335.7	330.5	330.9
Wee-Mag	75	52	446.9	444.4	445.6	452.1	439.4	447.3	452.2	443.7	437.8	448.0
Arcus1	83	3985	838896	835347	893527	902607	843561	839215	863780	829735	827898	818734
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	36721	19405	68927	1E+06	2E+06	12786	11974
Arcus1	83	6842	4E+06	4E+06	3E+06	4E+06	4E+06	4E+06	4E+06	4E+06	4E+06	4E+06
Arcus1	83	7571	6E+06	6E+06	8E+06	6E+06	6E+06	7E+06	7E+06	6E+06	6E+06	6E+06
Arcus1	83	8412	1E+07	1E+07	1E+07	1E+07	9E+06	9E+06	9E+06	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	4E+07	4E+07	5E+07	4E+07	4E+07	3E+07	3E+07
Lutz2	89	15.0	10.3	16.5	110.9	15.2	13.7	12.0	11.3	10.7	10.1	10.5
Lutz3	89	150	6.4	10.7	302.0	6.9	7.4	87.5	6.9	473.9	6.6	6.5
Mukherjee	94	201	588.25	475.1	1749.4	1839.7	2457.2	985.4	1624.3	2057.9	564.35	589.4
Mukherjee	94	301	14.4	16.5	3873.1	24.8	2196.6	4237.7	6108.3	5426.0	9.6	2457.5
Arcus2	111	5755	1E+06	2E+06	1E+06	1E+06	2E+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	6E+06	5E+06	5E+06	5E+06	5E+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	8E+06	8E+06	9E+06	9E+06	8E+06	8E+06	8E+06
Arcus2	111	11378	6E+06	6E+06	6E+06	6E+06	6E+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	373.5	428.4	550.1	283.9	315.4	315.0	257.4	305.2
Barthol2	148	89	371.2	346.0	371.5	420.9	573.4	309.0	853.2	462.9	294.65	287.2
Barthol2	148	91	414.0	362.4	523.4	425.0	471.2	580.9	402.7	356.3	281.3	265.7
Barthol2	148	95	419.4	396.95	473.2	379.4	593.1	478.5	671.3	1024.5	311.65	324.3

Table 9 and Table 10 present detailed comparation results of 10 algorithms on a U-shaped disassembly line in terms of F_1 and F_2 respectively. Notice that part data is acquired from related research study and SA, TS, GA, ABC, BA, and PSO are re-implemented 20 times. From Table 9 and Table 10, IWO performs better than many of these algorithms especially on solving large-size instances. In terms of F_2, IWO obtains 30 best values and part of objective values is same with other results. It is concluded that IWO has a strong searching ability, and it shows superior performance in solving SUDLBP.

VII. CONCLUSION

Environmental protection concept is widely accepted in all countries and green manufacturing method gets a great development recently. As an important step in remanufacturing, disassembly is becoming an active research area and DLBP is getting much more attentions. This study provides the first study of applying IWO algorithm on a U-shaped layout with the considering of sequence dependent situation. The encoding and decoding procedures of IWO algorithm help tackle SUDLBP. In the meanwhile, the proposed MINLP model is capable of solving large-size instances.

Based on the case studies and comparative study, U-shaped layout allows more task assignments and has a better performance compared with traditional straight-line layout. In the future, U-shaped, parallel, and two-sided line are worth of studying. Also, it is concluded that IWO algorithm has a strong ability in solve DLBP or even SUDLBP and comparation results illustrate that IWO algorithm is superior especially on increasing line smoothness. Novel meta-heuristic algorithms can be applied on a disassembly and improved previous approaches are of interest for researchers to explore.

REFERENCES

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

[2]. 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.

[3]. 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.

[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]. 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.

[6]. 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.

[7]. 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.

[8]. 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.

[9]. 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.

[10]. 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.

[11]. McGovern, S. M., and S. M. Gupta. 2011. The Disassembly Line: Balancing and Modelling. New York: McGraw Hill. ISBN 9780071622875.

[12]. Mete, S., Çil, Z.A., Ağpak, K., Özceylan, E. and Dolgui, A., 2016. A solution approach based on beam search algorithm for disassembly line balancing problem. *Journal of Manufacturing Systems*, *41*, pp.188-200.

[13]. Altekin, F.T., 2016. A piecewise linear model for stochastic disassembly line balancing. *IFAC-PapersOnLine*, *49*(12), pp.932-937.

[14]. 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.

[15]. 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.

[16]. Avikal, S., Jain, R. and Mishra, P., 2013. A heuristic for U-shaped disassembly line balancing problems. *MIT International Journal of Mechanical Engineering*, *3*(1), pp.51-56.

[17]. 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.

[18]. McGovern, S.M. and Gupta, S.M., 2003, October. Greedy algorithm for disassembly line scheduling. *In SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme-System Security and Assurance (Cat. No. 03CH37483)* (Vol. 2, pp. 1737-1744). IEEE.

[19]. 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.

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

[21]. 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.

[22]. 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.

[23]. 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.

[24]. Zhang, Z., Wang, K., Zhu, L. and Wang, Y., 2017. A Pareto improved artificial fish swarm algorithm for solving a multi-objective fuzzy disassembly line balancing problem. *Expert Systems with Applications*, *86*, pp.165-176.

[25]. 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.

[26]. 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.

[27]. 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.

[28]. Tuncel, E., Zeid, A. and Kamarthi, S., 2014. Solving large scale disassembly line balancing problem with uncertainty using reinforcement learning. *Journal of Intelligent Manufacturing*, *25*(4), pp.647-659.

[29]. 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.

[30]. Hezer, S. and Kara, Y., 2015. A network-based shortest route model for parallel disassembly line balancing problem. *International Journal of Production Research*, *53*(6), pp.1849-1865.

[31]. Avikal, S., Sharma, S., Kalra, J.S., Varma, D. and Pandey, R., 2016. A Fuzzy AHP Approach for Calculating the Weights of Disassembly Line Balancing Criteria. In *Proceedings of Fifth International Conference on Soft Computing for Problem Solving* (pp. 723-727). Springer, Singapore.

[32]. 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*, pp.1-22.

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

[34]. Mehrabian, A.R. and Lucas, C., 2006. A novel numerical optimization algorithm inspired from weed colonization. *Ecological informatics*, *1*(4), pp.355-366.

[35]. 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.