The disassembly line balancing problem of type II

Wang Shuwei[1], Guo Xiuping[1,*], Liu Jia[2]
1. School of Economics & Management, Southwest Jiaotong University, Chengdu, China
2. Business School, Qingdao University of Technology, Qingdao, China
wangshuwei@my.swjtu.edu.cn, gxp@home.swjtu.edu.cn, liujia1109@sina.cn

***Abstract*—Product disassembly is a vital stage for industrial recycling and remanufacturing which generates the desired parts/subassemblies by means of separation of a product into its elements. After disassembly, reusable parts/subassemblies are cleaned, refurbished, tested and directed to the part/subassembly inventory for remanufacturing process. The recyclable materials are sold to raw-material suppliers, while the residuals are sent to landfills. The disassembly line is the most suitable setting to disassemble products on a large scale. Therefore, designing and balancing disassembly line is important to improve the disassembly efficiency and realize the disassembly industrialization. In this study, we deal with the problem of disassembly line balancing with fixed number of workstations. A disassembly line balancing problem of type II (DLBP-II) is presented to minimize the cycle time and ensure the balance of workloads among workstations, and an improved variable neighborhood search algorithm is proposed to solve it. Finally, the validity of the presented model and the performance of the proposed algorithm are tested by different scale instances.**

Keywords- disassembly; disassembly line balancing problem; type II; cycle time; variable neighborhood search

I. INTRODUCTION

Stricter environmental laws have forced many more manufacturers to recycle and remanufacture their post-consumed products. Besides, lower production cost and customer demand are also the main driving forces of manufacturers to recover the end-of-life (EOL) products. Disassembly is a key process in product recovery, since it allows for the methodical extraction of the desired components and materials from the EOL products for recycling, remanufacturing, or reuse [1]. For improving disassembly efficiency and promoting disassembly industrialization, disassembly line is the best way to carry out disassembly operations. Therefore, the disassembly line balancing problem (DLBP) has recently attracted much attention from both academia and industry.

The DLBP can be defined as the task assignments to an ordered sequence of workstations, while satisfying all the disassembly precedence relations and optimizing the effectiveness of some measures [2]. Gungor and Gupta firstly introduced the DLBP, and presented a heuristic method for minimizing the number of opened workstations [3]. Later, Gungor and Gupta described the DLBP as a multi-objective combinatorial problem [4]. To deal with the DLBP, some simple heuristic methods were developed to attain near-optimal disassembly sequences, such as 2-opt heuristic, greedy algorithm [5~6]. Traditional mathematical programming techniques were also proposed for the DLBP [7~9]. However, the DLBP is NP-complete [10], thus the meta-heuristic algorithms are more suitable for solving large-scale DLBP. Ant colony optimization (ACO) [11~12], genetic algorithm (GA) [13~14], particle swarm optimization (PSO) [15] and artificial bee colony (ABC) [16~18] etc. were applied to the DLBP for finding an (near) optimal solution.

Including the literatures mentioned above, most of the research on DLBP is minimization the number of opened workstations under the condition that the cycle time of disassembly line is known. There are few researches in the DLBP with fixed number of workstations to minimize the cycle time. Similar to assembly line balancing problem [19], the DLBP with known cycle time is the first type of DLBP (DLBP-I), while the DLBP with fixed number of workstations is the second type of DLBP (DLBP-II). In the real world, when the disassembly line is established, the number of workstations is fixed. The task assignments to fixed number of workstations should be balanced in order to minimize the cycle time for improving the productivity of the disassembly line. Therefore, in this study we consider the problem of disassembly line balancing with fixed number of workstations, and an optimization model of DLBP-II is presented to minimize the cycle time and ensure the balance of workloads among workstations

The rest of this paper is organized as follows. Problem definition and formulation is given in Section 2. Section 3 introduces the VNS algorithm briefly, and computational study is given in Section 4. Finally, conclusions are presented in Section 5.

II. PROBLEM DEFINITION AND FORMULATION

A simple disassembly example of a six-task product is depicted in Fig. 1. The figure illustrates the tasks (nodes), precedence relationships (solid line arrows), and task removal times (numbers in parentheses). In the disassembly process, the tasks distributed among the workstations should satisfy the precedence relationship between tasks. For a sequence {1, 3, 5, 2, 4, 6}, task 5 is disassembled firstly, and then assign task 3. After that assign task 5 followed by task 2, and so on. This sequence satisfies the precedence constrains, so it is a feasible disassembly sequence which can disassemble this product completely.

The DLBP-II is to seek a disassembly sequence of EOL product, which can assign the tasks among the fixed number of workstations evenly, while satisfying all the disassembly precedence relations and minimizing the cycle time and other goals.

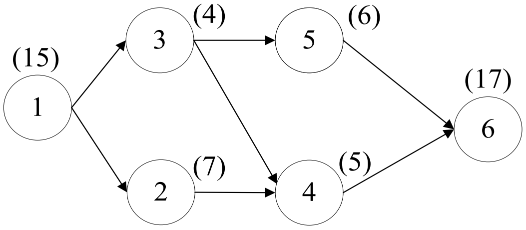


Fig. 1. A simple example of a six-task product

In this study, the DLBP-II is concerned with the paced disassembly line for a single product type that is to be completely disassembled. Model assumptions include the following: (1) the supply of the EOL products is infinite; (2) the quantity of parts is known and constant; (3) each part (task) is distributed to only one workstation; (4) the line is paced; (5) part removal time is deterministic and constant; (6) each product has no deletions, physical defects, additions or modification; (7) all the parts are removed from the EOL product completely; (8) When disassembled, each part must satisfy the precedence relations; (9) the processing time of each workstation must not exceed the cycle time.

Based on the above assumptions, a mixed integer non-linear programming model is presented. The parameters and decision variables are given below:

- Parameters:

n Number of tasks, which is equal to the number of parts; i.e., $i, j = 1, \ldots, n$

m Fixed number of workstations; i.e., $k = 1, \ldots, m$

t_i Removal time of part i

CT Cycle time; maximum time available at each workstation

P_{ij} Binary value; $P_{ij}=1$ if part i is the direct predecessor of task j, else $P_{ij}=0$

ST_k Working time of workstation k

IT_k Idle time of workstation k

- Decision variables:

$$x_{jk} = \begin{cases} 1 & \text{if task } j \text{ is assigned to workstation } k \\ 0 & \text{otherwise} \end{cases}$$

- The mathematical formulation for the DLBP-II is given as follows:

$$\min f_1 = CT = \max\{ST_k \mid 1 \le k \le m\} \qquad (1)$$

$$\min f_2 = \sum_{k=1}^{m} IT_k^{\,2} \qquad (2)$$

Subject to:

$$\sum_{k=1}^{m} x_{ik} = 1 \quad i = 1, 2, \cdots, n \qquad (3)$$

$$x_{ik} \le \sum_{k=1}^{m} x_{jk} \quad \forall P_{ij} = 1, \; i, j = 1, 2, \cdots, n \qquad (4)$$

$$\sum_{i=1}^{n} t_i \times x_{ik} \le CT \quad k = 1, 2, \cdots, m \qquad (5)$$

$$\sum_{i=1}^{n} x_{ik} \ge 1 \quad k = 1, 2, \cdots, m \qquad (6)$$

In this model, there are two objectives to be achieved. Objective (1) minimizes the cycle time to improve the productivity of the disassembly line. Objective (2) minimizes the smoothness index which ensures the balance of workload at each workstation, i.e., distributes idle times across the fixed number of workstations evenly.

Constraint (3) guarantees that each task is assigned to only one workstation. Constraint (4) imposes that all the disassembly precedence relationships should be satisfied. Constraint (5) forces the working time of each workstation to be no more than the cycle time. Constraint (6) ensures that all the workstations are opened.

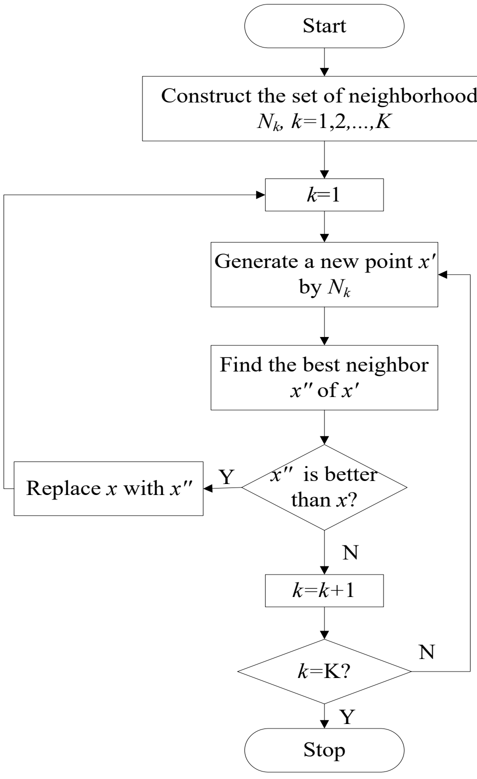


Fig. 2. The flowchart of the basic VNS

III. VNS ALGORITHM FOR DLBP-II

Variable Neighborhood Search (VNS) algorithm is a meta-heuristic as a general framework to solve combinatorial optimization problems. Its basic idea is systematic change of neighborhood structures within the search procedure. Let us define $N_k(k= 1,\ldots,K)$ as a series of predefined neighborhood structures, and $N_k(x)$ as the neighbors of a solution x generated by N_k [20]. The steps of the basic VNS is given in Fig. 2.

Unlike many other meta-heuristics, the basic VNS is simple and require few parameters. Therefore, it can provide good solutions in simpler ways than other meta-heuristics. Despite its simplicity it proves to be effective.

A. Coding and decoding

In the DLBP-II, the solution is a sequence of tasks assigned to workstations, so the permutation based representation is utilized to denote a solution in this paper. Each integer represents a disassembly task. As shown in Fig. 1, there is a product with six tasks, so the length of the solution string is 6. A permutation {1, 3, 2, 5, 4, 6} denotes a feasible sequence which satisfies all the disassembly precedence relations.

To decode a solution, the tasks need to be sequentially assigned to the fixed number of workstations while satisfying the working time of each workstation to be no more than the cycle time. There is a disassembly line consisting of 3 workstations. Suppose the cycle time is 17s. Take the solution {1, 3, 2, 5, 4, 6} for example, the decoding procedure is given as follows. First, open workstation 1 (WS_1) and assign task 1 to WS_1. The working time (ST) and idle time (IT) of WS_1 are ST_1=15s and IT_1=2s, respectively. Then assign task 3. Because the processing time of task 3 is larger than IT_1 (t_3=4s>IT_1=2s), so a new workstation (WS_2) is required to be opened, and the task 3 is assigned to WS_2. After that assign task 2 followed by task 5, and so on. The decoding result of this solution is shown in Fig. 3. The working times of 3 workstations are 15s, 17s, 20s respectively. The actual cycle time is the maximum working times of workstations, so f_1=20s. And then we can calculate $f_2=(20-15)^2+(20-17)^2+(20-20)^2=34$.

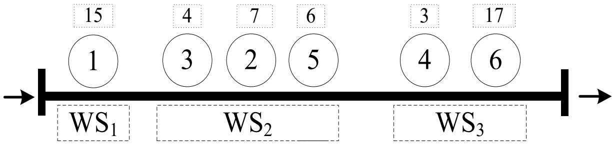


Fig. 3. Assignment of tasks to workstations

B. Neighborhood structures

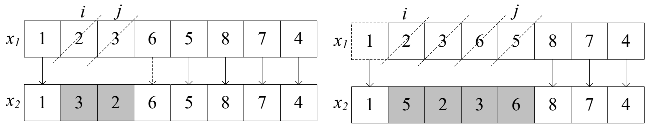
Neighborhood structures are the important part for VNS, which can be used to produce new solutions. To improve the current solution and further expand the search space, the VNS uses the idea of systematical changes of neighborhood structures. Considering the solution representation, in this study, we define and use the following four neighborhood operators.

(1) Swaps operator (N_1): This operator randomly selects two positions (in the solution vector) i and j with $i \neq j$ and swaps the task located in positions i and j. Fig. 4(a) shows an example, i=2, j=3.

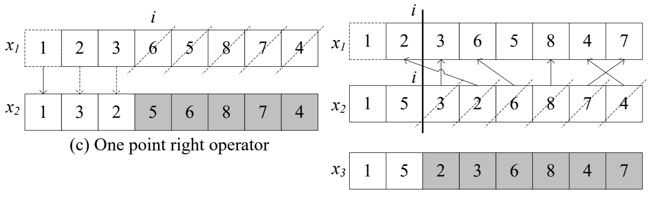
(2) Insertions operator (N_2): This operator consists of randomly selecting positions i and j with $i \neq j$ and relocating the task from position i to position j. See Fig. 4(b), where task 5 is relocated from position 5 to position 2.

(3) One point right operator (N_3): This operator randomly selects one position i, and then reconstructs a new subsequence starting from this position according to precedence relationships while keeping the rest unchanged (Fig. 4(c)).

(4) Right crossover operator (N_4): This operator randomly selects a cross position i in solution x_1 and x_2. Solution x_2 keeps the positions of the tasks which are assigned before this cross point while reordering the left sub-sequence according to the order of solution x_1. We can see from the Fig. 4(d) that solution x_2 is reordered from position 3.



(a) swap operator (b) insert operator

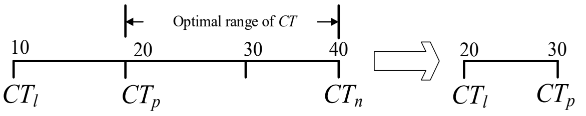


(c) One point right operator (d) right crossover operator

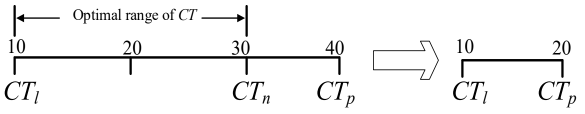
Fig. 4. Neighborhood structures

C. Cycle time adjustment

In order to maximize the productivity of the disassembly line, the minimum of the cycle time should be obtained. In this study, a boundary strategy based on binary search shown in Fig. 5 is introduced to improve the search efficiency. The cycle time often start at theoretical minimum value of cycle time which is equal to $\max\{\sum t_i/m,\ \max\{t_i\ |1\leq i\leq n\}\}$. The are two cases in the search process: If $CT_n>CT_p$, then $CT_l=CT_p$ and $CT_p=(CT_n+CT_p)/2$, else $CT_l=CT_l$ and $CT_p=(CT_l+CT_n)/2$.



(a) $CT_n>CT_p$



(b) $CT_n \leq CT_p$

CT_l: the lower boundary of cycle time
CT_p: the postulated value of cycle time
CT_n: the real value of cycle time

Fig. 5. A strategy based on binary search

IV. ILLUSTRATIVE EXAMPLES

In this section, the experiments are carried out to evaluate the validity of the proposed optimization model and the performance of the improved VNS algorithm. Currently, there are only three benchmark instances for DLBP in the open literature, including a 10-part product instance (P10), a 25-part cellular telephone instance (P25) and a 47-part laptop instance (P47). The precedence relationships and other detail knowledge database of the three instances can be obtained in [17~18]. All the experiments were implemented on an Intel(R) CoreI5 2.88 GHz PC with 8 GB RAM, and the proposed VNS was programmed in Microsoft Visual C++ 6.0. In order to calculate the average as statistics for the solution quality, each instance is carried out 30 times in 3s, 15s, 75s respectively. The results are shown in Table I.

Table I. Average (AVG) and standard deviation (SD) for objectives

Instance	Number of workstations	f_1		f_2	
		AVG	SD	AVG	SD
P10	2	81	0	1	0
	3	58	0	13	0
	4	46	0	117	0
	5	36	0	43	0
	6	36	0	503	0
P25	6	27	0	14.67	26.39
	7	25.03	0.98	97.70	89.10
	8	21.13	1.86	37.60	84.84
	9	19.03	2.99	51.43	134.34
	10	18	0	81.20	21.04
P47	4	214.37	3.60	3.93	41.23
	5	172.10	2.17	9.63	44.06
	6	144.43	4.17	31.87	148.24
	7	124.33	4.08	49.33	200.98
	8	110.53	7.31	149.20	499.32
	9	104	0	901.93	634.89

For the 10-part product instance, the solution space is relatively small, so we can use exhaustive search method to find the optimal solution. The VNS are always able to attain the optimal solutions when the number of workstations change from 2 to 6. Table II depicts an optimal solution sequence which tasks are assigned among 5 workstations. The objective function values of this optimal solution are found to be: f_1=36, f_2=43. From Fig. 6, we can see as the number of workstations increases, the cycle time decreases. When the cycle time keep unchanged, the smoothness index increases with the increase in the number of workstations, which illustrates that the balance of tasks assigned on workstations becomes worse. Therefore, in disassembly process, the balance of workload at each workstation should be considered in order to improve disassembly efficiency.

For the P25 and P47 instances, as the number of parts increase, the solution space increases exponentially (The entire solution space could be 25! for p25 and 47! for p47). Therefore, it is impractical to apply the exhaustive search method to find an optimal solution in such huge search space. Thus, the near-optimal solutions obtained by the meta-heuristics in reasonable computation times should be accepted. As shown in table I, the VNS is able to find the near optimal solutions, although the solutions are not the same in each experiment. Because the solution space of P47 is much larger than P25, the value of P47 is bigger than P25 in term of standard deviation. Based on the above discussion, the VNS can search a near optimal disassembly sequence in reasonable time.

Table II. An optimal disassembly sequence for P10 assigned among 5 workstations

Task	Workstion				
	1	2	3	4	5
10	10				
5	23				
6		14			
7		19			
1			14		
4			17		
8				36	
9					14
3					12
2					10
Total time	33	33	31	36	36
Idle time	3	3	5	0	0
$f_1 = 36; f_2 = 3^2 + 3^2 + 5^2 + 0^2 + 0^2 = 43$					

V. CONCLUSIONS

In this study, we presented a DLBP-II optimization model which considers two objective functions, including minimization of the cycle time and minimization of smoothness index. Then an improved VNS algorithm is proposed to obtain the (near) optimal disassembly sequences. Computational results demonstrate the validity of the presented model and the effectiveness of the proposed algorithm for solving the DLBP-II.

For further research, incomplete or partial disassembly should be considered due to environmental and economic factors. Some other meta-heuristics or hybrid approaches can be applied to further improve the solution. It would also be worthwhile studying two-sided disassembly lines in the light of huge EOL products, such as buses and trucks.

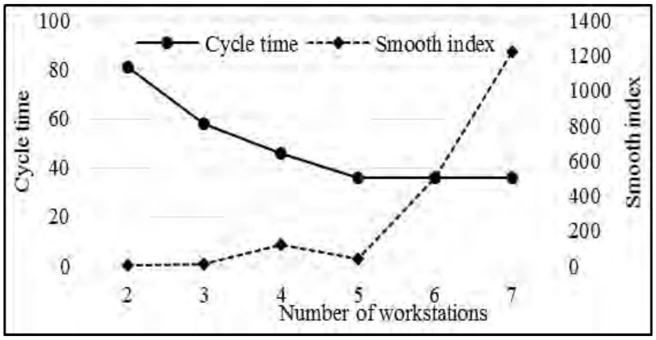


Fig. 6. The change of cycle time and smoothness index

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China under Grant No. 71471151 and Fundamental Research Funds for the Central Universities under Grant No. 26816WCX04.

REFERENCES

[1] M. A. Ilgin, S. M. Gupta, "Environmentally conscious manufacturing and product recovery (ECMPRO): A review of the state of the art". Journal of Environmental Management, vol. 91, issue 3, pp. 563-591, 2010.

[2] E. Zceylan, T. Paksoy, "Reverse supply chain optimisation with disassembly line balancing". International Journal of Production Research, vol. 51, issue 20, pp.5985-6001, 2013.

[3] A. Gungor, S. M. Gupta, "Disassembly line balancing". In Proceedings of the 1999 Annual Meeting of the Northeast Decision Sciences Institute, pp. 24-26, 1999a.

[4] A. Gungor, S. M. Gupta, "Disassembly line in product recovery". International Journal of Production Research, vol. 40, issue. 40, pp. 2569-2589, 2002.

[5] S. M. Mcgovern, S. M. Gupta, "Greedy algorithm for disassembly line scheduling". In IEEE International Conference on Systems, Man and Cybernetics, vol.1732, pp. 1737-1744, 2003.

[6] S. M. Mcgovern, S. M. Gupta, "2-opt heuristic for the disassembly line balancing problem". In Photonics Technologies for Robotics, Automation, and Manufacturing, pp. 71-84, 2004.

[7] F. T. Altekin, C. Akkan, "Task-failure-driven rebalancing of disassembly lines". International Journal of Production Research, vol. 50, issue 18, pp. 4955-4976, 2012.

[8] F. T. Altekin, L. Kandiller, and N. E. Ozdemirel, "Profit-oriented disassembly-line balancing". International Journal of Production Research, vol. 46, issue 10, pp. 2675-2693, 2008.

[9] A. Koc, I. Sabuncuoglu, and E. Erel, "Two exact formulations for disassembly line balancing problems with task precedence diagram construction using an AND/OR graph". Iie Transactions, vol. 41, issue 10, pp. 866-881, 2009.

[10] S. M. Mcgovern, S. M. Gupta, "Combinatorial optimization analysis of the unary NP-complete disassembly line balancing problem". International Journal of Production Research, vol. 45, issue 18-19, pp. 4485-4511, 2007.

[11] C. Lu, H. Z. Huang, and J. Y. H. Fuh, et al., "A multi-objective disassembly planning approach with ant colony optimization algorithm". Proceedings of the Institution of Mechanical Engineers Part B Journal of Engineering Manufacture, vol. 222, isuue 11, pp. 1465-1474, 2008.

[12] L. P. Ding, Y. X. Feng, and J. R. Tan, et al. "A new multi-objective ant colony algorithm for solving the disassembly line balancing problem". International Journal of Advanced Manufacturing Technology, vol. 48, issue 5, pp. 761-771, 2010.

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

[14] A. Aydemir-Karadag, O. Turkbey, "Multi-objective optimization of stochastic disassembly line balancing with station paralleling". Computers & Industrial Engineering, vol. 65 issue 3, pp. 413-425, 2013.

[15] C. B. Kalayci, S. M. Gupta, "A particle swarm optimization algorithm with neighborhood-based mutation for sequence-dependent disassembly line balancing problem". The International Journal of Advanced Manufacturing Technology, vol. 69 issue 1, pp. 197-209, 2013a.

[16] C. B. Kalayci, A. Hancilar, and A. Gungor, et al., "Multi-objective fuzzy disassembly line balancing using a hybrid discrete artificial bee colony algorithm". Journal of Manufacturing Systems, vol. 37, pp. 672-682, 2014.

[17] J. Liu, S. Wang, "Balancing disassembly line in product recovery to promote the coordinated development of economy and environment". Sustainability, vol. 9, issue 3, pp. 309, 2017.

[18] C. B. Kalayci, S. M. Gupta, "Artificial Bee Colony Algorithm for Solving Sequence-Dependent Disassembly Line Balancing Problem". Expert Systems with Applications, vol. 40, issue 18, pp. 7231–7241, 2013b.

[19] P R. McMullen, "Using simulated annealing to solve a multi-objective assembly line balancing problem with parallel workstations". International Journal of Production Research, vol. 36 issue 10, pp. 2717-2741, 1998.

[20] N. Mladenović, P. Hansen, "Variable neighborhood search". Computers & Operations Research, vol. 24, issue 11, pp. 1097-1100, 1997.