Cleaner Logistics and Supply Chain 10 (2024) 100129

Contents lists available at ScienceDirect

Cleaner Logistics and Supply Chain

journal homepage: www.journals.elsevier.com/cleaner-logistics-and-supply-chain

Line balancing problem with multi-manned workstations and resource constraints: The case of electronics waste disassembly

Yin-Yann Chen[a], Pourya Pourhejazy[b,*], Tzu-Ning Liu[a]

[a] Department of Industrial Management, National Formosa University, Yunlin 632301, Taiwan
[b] Department of Industrial Engineering, UiT- The Arctic University of Norway, Lodve Langesgate 2, Narvik 8514, Norway

A R T I C L E I N F O

Keywords:
Disassembly
Resource availability
Waste from Electrical and Electronic Equipment (WEEE)
Waste management
Optimization

A B S T R A C T

The increasing public awareness of environmental protection and the scarcity of rare earth elements have made closed-loop supply chains a necessity in many sectors. In particular, recycling components and parts from end-of-life consumer electronics have drawn the attention of both academics and practitioners. Disassembly line balancing improves the cost efficiency of recycling operations and hence helps waste management businesses that operate on very narrow profit margins. This study proposes a new mathematical formulation and hybrid metaheuristics to solve the Disassembly Line Balancing Problem (DLBP) considering multi-manned workstations and resource constraints. The transformed AND/OR graph is used for prioritizing disassembly tasks in the modeling process. The method is applied for optimizing a real-world case of laptop disassembly to showcase the usefulness of the proposed approach. The performance of the developed metaheuristics is evaluated for minimizing the number of workstations, operators, and machines involved in the disassembly operations. Further, the results are analyzed through sensitivity analysis. This study is concluded by providing practical insights and suggestions for the future development of DLBPs.

1. Introduction

The depletion of rare-earth elements and other environmental concerns highlight the importance of *R*-imperatives, notably the recycling and reuse of electronics waste. In addition to these causes, electronic waste recycling has economic advantages if planned properly (Pourhejazy et al., 2021). Inappropriate handling of electronics waste has been growing by nearly 10 percent per annum as of 2010 and is projected to reach a peak of 12.1 million tons in 2030 (Sun et al., 2023). As an eco-friendly approach to facilitating the reuse of material and preventing landfills (Kerber et al., 2021; Liu et al., 2022), remanufacturing End-of-Life products is necessary for an effective transition toward cleaner logistics and supply chains. The main goals are to disassemble, sort, and recover parts/materials from End-of-Life products — which are returned through reverse logistics channels — to cope with the rising raw material costs and their unpredictable availability, reduce resource wastage, and remove hazardous parts before possible disposal.

The growing need for recycling electronics waste is apparent (Jyothi et al., 2020) while operations resource limitations remain intact and are expected to exacerbate in the post-pandemic and war economy. Disassembly consists of separating and sorting the constituents (components and sub-components) of a device and is an initial stage of recycling. Improving resource utilization in recycling carries with it a certain cachet and increases its economic viability. Disassembly Line Balancing Problem (DLBP) is one way of improving resource utilization by regulating operator and machine time while considering the takt time. DLBPs adjust the disassembly configuration for more efficient task fulfillment, which is particularly important when the system operates with limited resources and the production rate must be adjusted in order to fulfill demand (Pourhejazy, 2022). Reducing the number of disassembly workstations (Altekin, 2017; Habibi et al., 2017; Liu & Wang, 2017; Ren et al., 2017; Ren et al., 2018), balancing workload across workstations (Aydemir-Karadag & Turkbey, 2013; Wang et al., 2019), minimizing the total idle time (Kannan et al., 2017; Li et al., 2019; Xiao et al., 2017), cycle time (Kekre et al., 2003; McGovern & Gupta, 2007; Paksoy et al., 2013), the number of direction changes (Zhu et al., 2014), and CO2 saving (Igarashi et al., 2016) are the seminal examples of optimization objectives targeted in the DLBP literature.

Many extensions are developed to facilitate the industry reach of DLBPs (see the review works of (Laili et al., 2020; Özceylan et al., 2019)). New constraints were introduced to improve the effectiveness of the disassembly operations; for example, through early extraction of

* Corresponding author.
E-mail address: pourya.pourhejazy@uit.no (P. Pourhejazy).

https://doi.org/10.1016/j.clscn.2023.100129
Received 6 January 2023; Received in revised form 5 September 2023; Accepted 30 November 2023
Available online 10 December 2023

parts that are more easily accessible (Gungor & Gupta, 2001; Güngör & Gupta, 2002), those of greater monetary value and/or market demand (Jia & Shuwei, 2017; Liu & Wang, 2017; Xiao & Nie, 2017), or the parts that require shorter disassembly times (Avikal et al., 2014). Environmental-related constraints are also considered for the early extraction of parts with hazardous substances (Bentaha et al., 2014; Jia & Shuwei, 2017; J. Liu & Wang, 2017; Ren et al., 2018). Cheng et al., (2022) integrated the task correlation considerations into DLBP. Other studies addressed part characteristics (Zhang et al., 2022), machine specificity (He et al., 2022), and operational constraints; for example, execution (Liang et al., 2021), setup time (Kizilay, 2022), and tool requirement constraints (Yin et al., 2021). There still are basic assumptions that should be addressed to better reflect the real-world situation.

The existing DLBP studies assume that the workstations require one operator/machine. In practice multiple operators are often required at a workstation to handle different disassembly operations simultaneously; this feature is particularly important for disassembling large-size or high-volume End-of-Life products. Kucukkoc et al., (2020) extended the DLBP to account for more than one operator at the workstations; they developed a mixed-integer nonlinear programming model and an exact method for solving small- to medium-size instances. They did not consider resource limitations in their optimization model. Additionally, they used exact optimization, which is only applicable for solving small- to medium-size instances; metaheuristics are required to solve industry-scale problems. To address these gaps, this study investigates DLBPs with multi-manned workstations and resource availability constraints; it also develops optimization algorithms to reduce operational costs by increasing machine and operator utilization.

The main contribution of this study is to propose a new mathematical formulation for finding the optimal set of workstations, disassembly operations, machines, and operators considering resource constraints, multi-manned workstations, and workers' skills to work with various machines. The model is validated by solving small-scale instances using an exact optimization method. A real-case data is used to showcase the usefulness of the method. Hybrid Simulated Annealing (HSA) and Genetic computation-based Simulated Annealing (GSA) are further developed to solve the practical-scale problem. The numerical analysis is supplemented by a sensitivity analysis of the operational parameters to shed light on the operational aspects of the findings. The rest of this article is organized into the literature review, mathematical models, solution algorithms, numerical experiments, and conclusions in Sections 2-6, respectively.

2. Relevant literature

This section reviews the relevant literature, i.e., DLBPs with resource constraints as well as those with multi-manned workstations. Relatively limited studies accounted for resource-constrained DLBP. In one of the early studies, (Mete et al., 2016) introduced the resource-constrained DLBP, proposed a new formulation, and solved it using CPLEX. (Wang et al., 2020) studied a stochastic and sequence-dependent DLBP considering task failure and resource limitations using the Cuckoo Search algorithm. (Dong et al., 2021; Yuan et al., 2020) developed the multi-objective versions of Ant Lion and Ecological Strategy Optimization algorithms for solving DLBPs with resource constraints. (Fang & Xu, 2020) introduced a constraint-handling method for the optimization of resource-constrained DLBPs.

These studies along with the majority of other DLBP consider one operator as the single operation resource. Limited studies accounted for multi-manned and machinery resources. (Kucukkoc et al., 2020) is one of the first studies that accounted for multi-manned workstations in DLBPs. (Cevikcan et al., 2020) developed a heuristic algorithm and a new formulation to minimize the number of workers and workstations while accounting for multi-manned workstations; they do not take into account the machinery as a resource and assume that there are no restrictions on the available resources. (ÇİL, 2021) proposed an exact optimization approach for multi-manned DLBP. (Kose et al., 2023) developed a game theory-based approach for DLBP with multi-manned workstations.

Considering that improving resource efficiency is the major purpose of line balancing, notably dual-resource and limitations on the available resources, which are the most related features, should be simultaneously considered in DLBPs. Considering the possible interactions between these features, overlooking any of them may reduce the reliability of the optimization outcomes. In the most recent study, (Zhou & Bian, 2022) developed a multi-objective optimization approach considering both features. Our study is different in that (1) the proposed model considers workers and machinery as the required resources for completing disassembly operations. In this setting, only some disassembly tasks can be processed on specific machines, and only some operators can execute specific machines. The model developed by (Zhou & Bian, 2022) model did not take into account the practical characteristic that some operators can only execute specific machines in a work environment with specialized human resource divisions. (2) The proposed model minimizes the number of operators, the number of machines, and the number of workstations to reduce the involved workspace for disassembly operations. (3) The mathematical formulation is relatively compact and can be solved using commercial (exact) solvers for small- to medium-scale instances. Besides, a real case example showcases the practicability of the optimization approach. The literature review is summarized in Table 1.

3. Proposed formulation

Reuse, recycling, and disposal of end-of-life items require the extraction of the components and parts through disassembly. The disassembly procedure can be complex, hence, should be planned in a way to minimize the overhead costs (e.g., the time, labor, and tooling) while practicing operational effectiveness. For the mathematical definition of the problem, let $G = (V, A)$ represent the disassembly graph. In this graph, components/parts and the completed disassembly tasks are represented by vertices/nodes (V) and arcs (A), respectively. The starting and ending vertices in the graph represent the end-of-life items and the disassembled parts, respectively. The progression from one layer to another specifies the completion of a disassembly task. There can be different disassembly alternatives in every stage of the disassembly process, which should be planned considering various operational parameters.

The DLB operations involve simultaneous machine and operator assignments to the workstations to satisfy the operational requirements. The literature considers fixed machine type and processing time for disassembly operations. In practice, however, the operator's ability to

Table 1
Summary of the literature review.

Reference	Multi-manned	Resource-constrained	Solution method
(Mete et al., 2016)	No	Yes	Exact
(T. Wang et al., 2020)	No	Yes	Metaheuristic
(Yuan et al., 2020)	No	Yes	Metaheuristic
(Dong et al., 2021)	No	Yes	Metaheuristic
(Kucukkoc et al., 2020)	No	Yes	Exact
(Cevikcan et al., 2020)	Yes	No	Heuristic
(ÇİL, 2021)	Yes	No	Exact
(Kose et al., 2023)	Yes	No	Exact
(Zhou & Bian, 2022)	Yes	Yes	Metaheuristic
Present study	Yes	Yes	Exact and metaheuristic

perform certain tasks varies considering his/her work experience and familiarity with the machine. Multi-manned planning has the advantage of reducing the number of workstations, quicker detection of product defects, and most notably, enabling the operators to work on different tasks simultaneously, which makes it more flexible than single-person workstations. Accounting for practical constraints, like resource availability, is another practical need for obtaining dependable solutions. DLBP with multi-manned workstations and resource constraints addresses these features.

To begin with the modeling, it is assumed that the total number of operators in the workstation does not exceed the number of machines. Finally, and to better represent the DLB in practice, the priori relationships between the disassembly operations are considered in modeling the allocations. A binary nonlinear mathematical formulation of DLBP is now proposed. The following notations are used in the formulation.

Sets, indices, and parameters.

i	: Disassembly operation ($i = 1, 2, ..., I$)
k	: Node number ($k = 1, 2, ..., K$)
j, v	: Workstation ($j, v = 1, 2, ..., J$)
u	: Operator number ($u = 1, 2, ..., U$)
r	: Machine type ($r = 1, 2, ..., R$)
A_k	: The *k-th* manual node, representing the component
B_i	: The *i-th* normal node, representing the disassembly operation
t_{iu}	: Processing time of operation i by operator u
$P(A_k)$	: The set of all predecessors for the *kth* manual node
$S(A_k)$	: The set of all subsequent operations for the *kth* manual node
CT	: Cycle Time
u_{max}	: Maximum number of operators that can be configured at each workstation
r_{max}	: Maximum number of machine types that can be configured for each workstation
δ	: Maximum number of operators on the production line
M	: A positive large number

$$s_{ik} = \begin{cases} 1 : Operation B_i belongs to S(A_k) \\ 0 : Otherwise \end{cases}$$

$$r_{ir} = \begin{cases} 1 : Operation B_i can use machine r \\ 0 : Otherwise \end{cases}$$

$$o_{ur} = \begin{cases} 1 : Operator u can work with machine r \\ 0 : Otherwise \end{cases}$$

Decision variables.

$$X_{iju} = \begin{cases} 1 : If operation B_i is assigned to operator u in workstation j \\ 0 : Otherwise \end{cases}$$

$$W_j = \begin{cases} 1 : If workstation j is included in the operations \\ 0 : Otherwise \end{cases}$$

$$Z_i = \begin{cases} 1 : If operation B_i has been executed \\ 0 : Otherwise \end{cases}$$

$$Y_{jr} = \begin{cases} 1 : If machine r is assigned to workstation j \\ 0 : Otherwise \end{cases}$$

$$L_{ju} = \begin{cases} 1 : If operator u is assigned to workstation j \\ 0 : Otherwise \end{cases}$$

Eq. (1) aims to minimize the total number of operators, machines, and workstations.

$$\text{Minimize } \frac{NU - NU^{min}}{NU^{max} - NU^{min}} + \frac{NR - NR^{min}}{NR^{max} - NR^{min}} + \frac{NW - NW^{min}}{NW^{max} - NW^{min}} \quad (1)$$

where

$$NU = \sum_{j=1}^{J} \sum_{u=1}^{U} L_{ju} \quad (2)$$

$$NR = \sum_{j=1}^{J} \sum_{r=1}^{R} Y_{jr} \quad (3)$$

$$NW = \sum_{j=1}^{J} W_j \quad (4)$$

Eqs. (2) to (4) specify the total number of operators assigned to workstations, the total number of machine types to be used, and the total number of workstations on the disassembly line. Since multiple objectives are involved, it may not be meaningful to make a direct quantitative comparison. To address this issue, the minimum deviation method is used for aggregating the three sub-objective equations into a single commensurable objective function. NU^{max} is the maximum number of operators on the disassembly line; NR^{max} refers to the maximum number of machine types; NW^{max} represent the maximum number of workstations; NU, NR, and NW specify the target values for the number of operators, machine types, and workstations, respectively. Finally, the values of NU^{min}, NR^{min}, and NW^{min} are set to 1.

The objective function is subject to the following sets of constraints.

$$\sum_{i=1}^{I} (s_{ik} \times Z_i) = 1,\ for k = 1 \quad (5)$$

$$\sum_{i=1}^{I} (s_{ik} \times Z_i) = \sum_{i=1}^{I} (p_{ik} \times Z_i),\ for k = 2, \cdots, K \quad (6)$$

$$\sum_{u=1}^{U} \sum_{j=1}^{J} X_{iju} = Z_i,\ for i = 1, 2, \cdots, I \quad (7)$$

$$\sum_{u=1}^{U} \sum_{i=1}^{I} \sum_{j=1}^{v} (p_{ik} \times X_{iju}) \geq \sum_{u=1}^{U} \sum_{i=1}^{I} (s_{ik} \times X_{ivu}),\ for v = 1, 2, \cdots, J; k = 2, 3, \cdots, K \quad (8)$$

$$\sum_{u=1}^{U} \sum_{i=1}^{I} (X_{iju} \times t_{iu}) \leq CT \times \sum_{u=1}^{U} L_{ju},\ for j = 1, 2, \cdots, J \quad (9)$$

$$\sum_{i=1}^{I} \sum_{u=1}^{U} (r_{ir} \times X_{iju}) \leq M \times Y_{jr},\ for j = 1, 2, \cdots, J; r = 1, 2, \cdots, R \quad (10)$$

$$\sum_{i=1}^{I} \sum_{u=1}^{U} (r_{ir} \times X_{iju}) \geq Y_{jr},\ for j = 1, 2, \cdots, J; r = 1, 2, \cdots, R \quad (11)$$

$$\sum_{u=1}^{U} L_{ju} \geq W_j,\ for j = 1, 2, \cdots, J \quad (12)$$

$$\sum_{u=1}^{U} L_{ju} \leq u_{max} \times W_j,\ for j = 1, 2, \cdots, J \quad (13)$$

$$W_{j+1} \leq W_j,\ for j = 1, 2, \cdots, J - 1 \quad (14)$$

$$L_{ju} \leq (o_{ur} - 1) \times Y_{jr} + 1,\ for j = 1, 2, \cdots, J; u = 1, 2, \cdots, U; r = 1, 2, \cdots, R \quad (15)$$

$$\sum_{j=1}^{J} L_{ju} \leq 1,\ for u = 1, 2, \cdots, U \quad (16)$$

$$\sum_{r=1}^{R} Y_{jr} \leq r_{max},\ for j = 1, 2, \cdots, J \quad (17)$$

$$\sum_{r=1}^{R} Y_{jr} \geq \sum_{u=1}^{U} L_{ju},\ for j = 1, 2, \cdots, J \quad (18)$$

$$Y_{jr} \leq W_j,\ for j = 1, 2, \cdots, J; r = 1, 2, \cdots, R \quad (19)$$

$$\sum_{j=1}^{J} \sum_{u=1}^{U} L_{ju} \leq \delta \quad (20)$$

$$\sum_{i=1}^{I} X_{iju} \geq L_{ju},\ for j = 1, 2, \cdots, J; u = 1, 2, \cdots, U \quad (21)$$

$$\sum_{i=1}^{I} X_{iju} \leq M \times L_{ju},\ for j = 1, 2, \cdots, J; u = 1, 2, \cdots, U \quad (22)$$

$$X_{iju} \in \{0, 1\},\ \forall i, j, u \quad (23)$$

$$Z_i \in \{0, 1\},\ \forall i \quad (24)$$

$$Y_{jr} \in \{0, 1\},\ \forall j, r \quad (25)$$

$$W_j \in \{0,1\}, \quad \forall j \tag{26}$$

$$L_{ju} \in \{0,1\}, \quad \forall j, u \tag{27}$$

Eqs. (5) and (6) ensure that only one OR-successor operation is selected. Eq. (7) ensures that assignments are done for active tasks. Constraints (8) define the priority relationship to ensure that the disassembly operation is completed in the right order. Constraints (9) restrict the total processing time from exceeding the cycle time at a workstation. Constraints (10)–(11) establish a connection between variables X_{iju} and Y_{jr}, meaning that if a disassembly operation is performed in workstation j with machine type r, Y_{jr} should be activated too. Constraints (12) and (13) ensure that only active workstations are assigned with operators and that the total number of operators is less than or equal to the allowed threshold, respectively. Constraints (14) ensure that every workstation is succeeded by another workstation, except for the last workstation (i.e., $j = J$). Constraints (15) indicate that the operator assigned to each workstation can operate all machine types available in the workstation. Constraints (16) guarantee that every operator can be assigned to at most one workstation. Constraint (17) is defined to bound the total number of machines at every station. Constraints (18) indicate that the number of operators in a workstation should not be greater than the number of machine types assigned to that workstation. Constraints (19) ensure that machines are assigned to active workstations. Constraint-set (20) limits the total number of operators across the workstations. Constraints (21)–(22) link the cross-assignment X_{iju} and operator assignments L_{ju} variables. The remainder of the constraints (23)–(27) limit the variables from accepting non-binary values.

4. Solution algorithms

This section develops two metaheuristics for solving the DLBP with multi-manned workstations and resource constraints. We first elaborate on the precedence and initialization modules, which are the common elements in both metaheuristics. The chapter then details the computational steps of the algorithms.

4.1. Transformed AND/OR graph (TAOG)

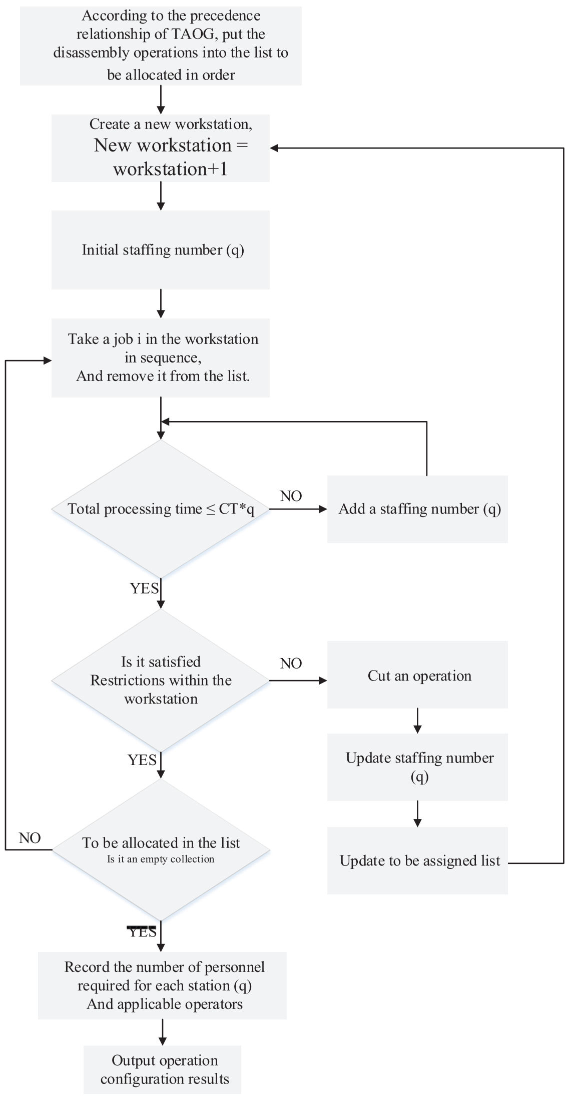
This study borrows the transformed AND/OR graph (TAOG) approach proposed by Koc et al., (2009) to define the disassembly precedence relationships. The optimization method proposed in this study is exclusively designed for DLBP. The precedence diagrams for DLBP and the Assembly Line Balancing Problem (ALBP) differ in their presentation. While the precedence relations in ALBP are primarily of the simple ’AND’ types, those in DLBP are typically categorized as either ’AND’ or ’OR’ relations. For the DLBP with multi-manned workstations and resource constraints, our study employs the transformed AND/OR graph (TAOG) to construct the model and algorithm to address the issue. For an End-of-Life product awaiting disassembly, the approach aims to determine the optimal disassembly decision and resource allocation strategy from all possible disassembly paths; the objective is to minimize the total number of operators, machines, and workstations.

AND/OR graph is an illustrative way of mapping the possible ways to break a product down into its basic subcomponents (parts) and components. In the mapping, nodes represent subcomponents/components and arcs indicate the disassembly tasks. The basic AND/OR graph only shows the possible subassemblies. The nodes in the basic AND/OR graph, which corresponds to a subassembly, are represented by an artificial node in TAOG while the arcs in the basic AND/OR graph are represented by a normal node in TAOG. The priority relationships between subassemblies (artificial nodes) and tasks (normal nodes) are shown by arcs. The artificial and normal nodes are represented by A_i and B_i, respectively. There can be multiple normal nodes before and/or after an artificial node, but only one predecessor and one successor will be handled. There are AND-type and OR-type arcs in TAOG. An AND-type arc indicates a regular precedence relationship, while arcs of OR-type permit any following arc to be selected. A small curve is used to distinguish OR-type from AND-type relationships. Using OR-type arcs in TAOG enables us to fully decompose a product considering only one subset of tasks. The outcomes of TAOG will be used for generating initial solutions in the optimization of DLBP with multi-manned workstations and resource constraints.

4.2. Generating Feasible Initial Solutions

Given input data on the disassembly case, the algorithms begin the search procedure using the initial solution that is encoded; the codes will be translated into decision variables (decoding) at the end of every iteration for evaluation purposes. A complete solution includes workstation setup, disassembly, machine, and operator configurations based on which the objective function value is calculated. The procedure for generating feasible initial solutions is as follows.

Step 1. Configure the disassembly operation. Considering the precedence relationships defined by TAOG, the computational steps in Fig. 1 are used for defining the initial disassembly path. The sub-steps are explained below.



According to the precedence relationship of TAOG, put the disassembly operations into the list to be allocated in order
Create a new workstation, New workstation = workstation+1
Initial staffing number (q)
Take a job i in the workstation in sequence, And remove it from the list.
Total processing time ≤ CT*q
NO
Add a staffing number (q)
YES
Is it satisfied Restrictions within the workstation
NO
Cut an operation
Update staffing number (q)
YES
NO
To be allocated in the list
Is it an empty collection
Update to be assigned list
YES
Record the number of personnel required for each station (q) And applicable operators
Output operation configuration results

Fig. 1. Computational steps for defining the initial disassembly path.

(a) Select the minimum operator processing time (t_{iu}) for every disassembly from all the possible processing times for that operation. Insert the disassembly tasks into the pending list considering the priori relationships in TAOG and ascending order of t_{iu}.
(b) Establish a new workstation and update the number of new workstations.
(c) Initialize the staffing number ($q = 1$) within the new workstation.
(d) Extract the first task from the pending assignment list and insert it into the new workstation.
(e) Determine whether the total processing time in the workstation is less than the given cycle time, i.e., $CT \times q$; if yes, proceed to (h); otherwise, continue to (f).
(f) Add a new operator to the workstation and return to €.
(g) Considering the resource limitations of the workstation, proceed to Step i if the following conditions are met, otherwise, go to (h). (I) The number of machine types in the workstation cannot be greater than the maximum number of configurable machine types, r_max. (II) Staffing level, q, in the workstation should be less than the number of machine types, except if there is only a single operation at the workstation. (III) The number of operators capable of handling the operations at the workstation must not be less than the staffing level (q).
(h) Remove the disassembly job from the workstation, update the staffing number, and the pending assignment list; and return to (b).
(i) If the pending list is empty, i.e., the disassembly operation has been fully configured and the result can be reported; otherwise, return to (d). To ensure that the resulting solution is feasible, and all the conditions are met, the output enters the correction module in Fig. 2.

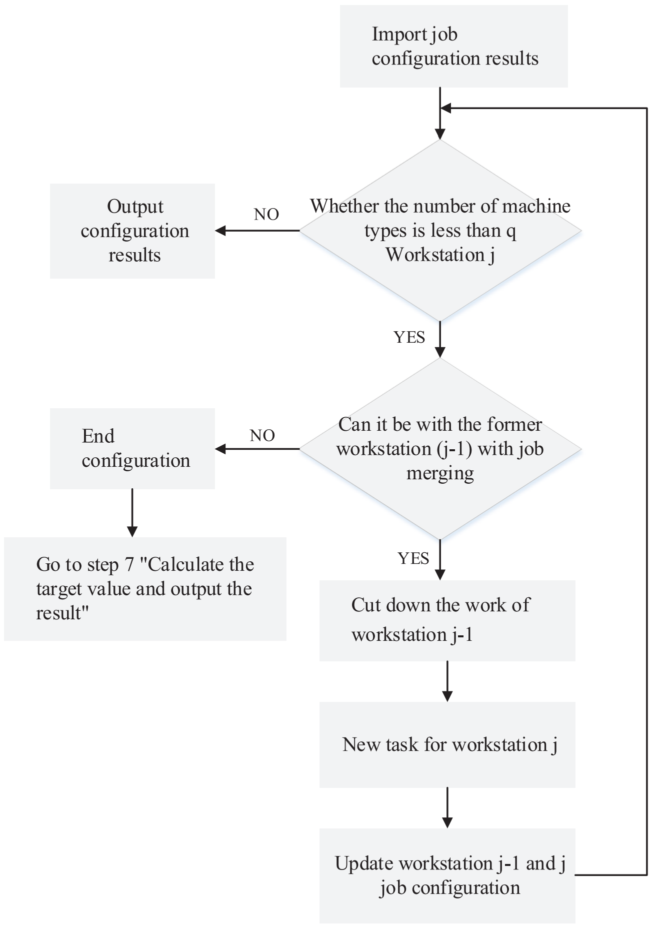

Import job configuration results
Whether the number of machine types is less than q Workstation j
NO
Output configuration results
YES
Can it be with the former workstation (j-1) with job merging
NO
End configuration
Go to step 7 "Calculate the target value and output the result"
YES
Cut down the work of workstation j-1
New task for workstation j
Update workstation j-1 and j job configuration

Fig. 2. Computational steps of the correction module.

Step 2. Machine Configuration. Considering the outcomes of Step 1, equip every workstation with the required machine type and in order.

Step 3. Operator Configuration. Different combinations of the number of operators, machine types, and workstations should be examined to select the best operator configuration; the process is summarized in Fig. 3 and explained as follows.

(a) Initialize the processing time parameters.
(b) Replace the time parameter with a large positive number (M) if the operator is not qualified (e.g., is not familiar with the machine). In so doing, the operator is unlikely to be selected for executing the task if he does not have the required skill set.
(c) Sequentially select the workstation with incomplete operator configuration.
(d) Calculate the operator processing times for the selected workstation and determine all the possible operator combinations.
(e) Look for a qualified operator combination that has not yet been assigned to any workstations and proceed to (f); otherwise, jump to (i).
(f) If the number of unassigned eligible operator combinations is greater than one, select one with a lower duplication rate; otherwise, select the unassigned eligible operator combinations. Proceed to (g).
(g) Update the processing time of the selected operator to M for all jobs; proceed to (h).
(h) Proceed to: (i) if all workstations have a complete operator configuration; otherwise, (c).
(i) Calculate the fitness value and report the operator configuration result.

Step 4. Generate a new configuration set. Determine if the disassembly operations in two adjacent workstations in the current configuration can generate new configuration combinations while satisfying the workstation conditions (feasible new configurations) in Step 1. Evaluate the feasible new combinations considering their objective values.

Step 5. Select the best configuration combination.

Step 6. Update the workstation configuration considering the outcome of Step 5.

Step 7. Report the resulting configuration and its fitness value.

4.3. *Hybrid simulated annealing*

Inspired by the annealing process of metals where transitioning to a low energy state is desired, Simulated Annealing (SA; (Kirkpatrick et al., 1983)) gets an initial feasible solution and applies a series of operators on the solution vector to move towards a (near-) optimum solution in the solution space. SA falls into the category of stochastic local search algorithms. The basic SA algorithm may not be effective for solving intractable problems considering its limited global search capabilities and high chances of falling into local optimality traps. The Hybrid Simulated Annealing (HSA) proposed in this section includes perturbation a mechanism to improve SA's performance. HSA consists of the computational steps shown in Fig. 4. A step-by-step guide to the computational procedure of HSA follows next.

Step 1. Set the initial parameters including the initial temperature (T_0), final temperature (T_f), cooling rate (q), and maximum search times in the strata (i.e., the cooling time K).

Step 2. Import the disassembly paths from TAOG. As an illustrative example, one of the dismantling paths in Fig. 5, e.g. [1, 4, 5, 9, 10, 11, 16, 17, 18] or [1, 4, 5, 9, 10, 12, 16, 17, 18, 19, 21], will be considered.

Step 3. Select one of the disassembly paths randomly.

Step 4. Decode the initial solution and set the current temperature (T_p) equal to T_0. Generate a sequence of random numbers with random

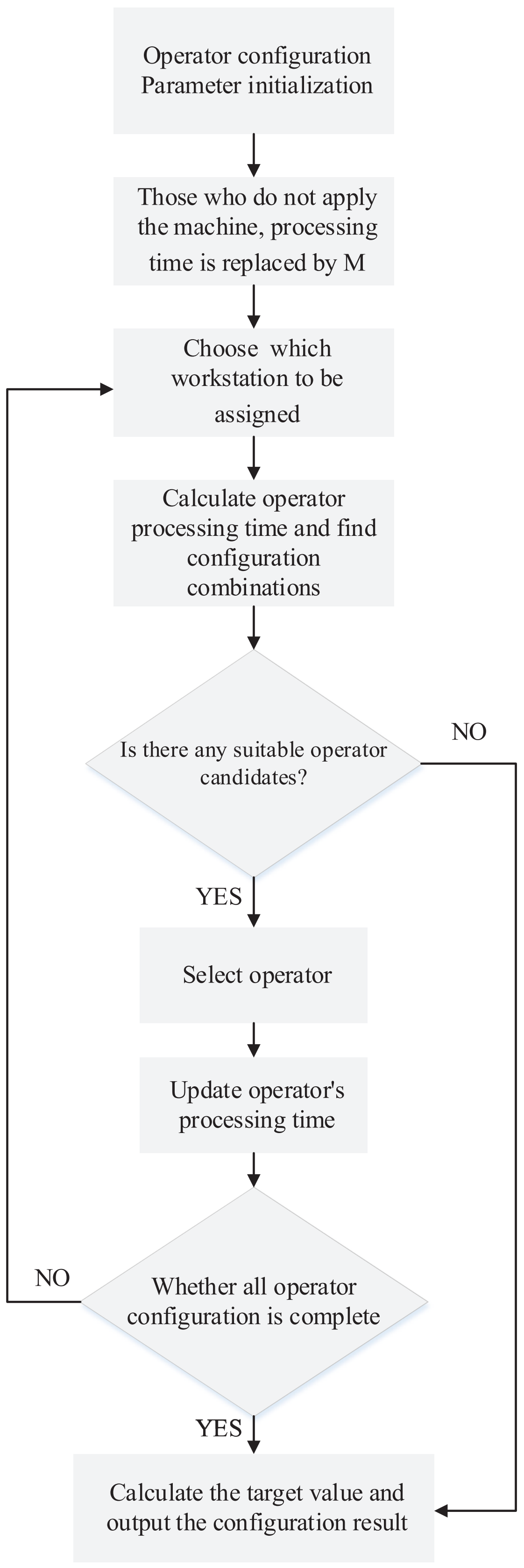

Operator configuration
Parameter initialization
Those who do not apply the machine, processing time is replaced by M
Choose which workstation to be assigned
Calculate operator processing time and find configuration combinations
Is there any suitable operator candidates?
NO
YES
Select operator
Update operator's processing time
Whether all operator configuration is complete
NO
YES
Calculate the target value and output the configuration result

Fig. 3. The operator configuration flowchart.

association to the tasks; this will be reordered through the repair mechanism to ensure that the look-ahead relationships in TAOG are applied.

The repair mechanism begins with selecting the disassembly task with no predecessors; in case of having more than or equal to two tasks, select the one that is ranked first in the original sequence (B1 in the illustrative example). Next, select the successors of the latest assigned task (B1 precedes B2 and B3); select the successor that is ranked earlier in the original sequence (B3 in the example). From the successor(s) that was not selected in the previous step and the new successors (i.e., B5 with the pre-operation of B3), select the one that comes first in the original sequence (in the example, B2 is sorted before B5, so B2 is selected). This procedure continues until all the jobs are inserted into the corrected solution. Considering 8 tasks in the illustrative example, Fig. 6 shows the outcome of this procedure.

Step 5. Calculate the fitness value considering the repaired solution vector; the following value assignments should be implemented in the first algorithm iteration; the existing (current) feasible decoding (C_{st}) = Initial decoding (I_{st}), Best Decoding (B_{st}) = Initial Decoding (I_{st}), Current feasible solution (C_{sol}) = Initial solution (I_{sol}), Best objective value (B_{sol}) = Initial objective value (I_{sol}).

Step 6. Check whether the termination condition is satisfied. If the termination criterion is met, i.e., when the current temperature (T_p) is less than or equal to the final temperature (T_f) or the best objective value (B_{sol}) remains the same for a certain number of consecutive iterations, proceed to Step 13; otherwise, go to Step 7.

Step 7. Generate a new neighborhood solution (N_{st}). Three methods are employed to generate neighborhood solutions, from which, one is selected randomly in every iteration. The following procedure is defined.

(7.1) Generate a random integer between 1 and 3.

(7.2) If the random value equals 1, go to 7.3; if = 2, go to 7.4; and if = 3, go to 7.5.

(7.3) Generate a random set of integers between 1 and the total number of tasks without repetition; then, proceed to 7.6. This step is considered the perturbation mechanism to help evade local optima. For an example of 6 tasks, the new solution would be $N_{st} = [3,2,5,4,1,6]$.

(7.4) Generate a random integer between 1 and the total number of tasks (cutting point: cp). The vector section from cp to the last element into the front of the vector; then, proceed to 7.6. In an example with six tasks, $C_{st} = [1,3,4,2,5,6]$, $cp = 3$, the new solution would be $N_{st} = [4,2,5,6,1,3]$.

(7.5) Generate two non-repetitive integer values between 1 and the total number of tasks, randomly, i.e., cp_1 and cp_2. Exchange the elements associated with cp_1 and cp_2; then, proceed to 7.6. In the example with six jobs, $C_{st} = [1,3,4,2,5,6]$, $cp_1 = 3$, $cp_2 = 5$, the resulting neighborhood solution would be $N_{st} = [1,3,5,2,4,6]$.

(7.6) Repair the new solution to ensure that the task relationships are respected.

Step 8. Calculate the fitness value of the new solution,.(N_{sol})

Step 9. Update the best-found neighboring solution. The Metropolis criterion is used to determine whether to accept the new solution. In this approach, the difference between the fitness values, $E = N_{sol} - C_{sol}$, is first calculated. If $E \leq 0$, then the new neighborhood solution is accepted and replaces the current solution, C_{sol}; otherwise, calculate a probability value through the Boltzmann distribution formula shown in Equation (28). In so doing, poor-performing neighborhood solutions will be accepted under a certain probability, i.e., $P > R$, where R is a random number. In this approach, an increase in the current temperature, T_p, reduces the likelihood of accepting poor solutions.

$$P = exp\left(-\frac{E}{T_p}\right) \tag{28}$$

Step 10. Check if the maximum number of neighborhood searches has reached (K); once the number of performed searches becomes

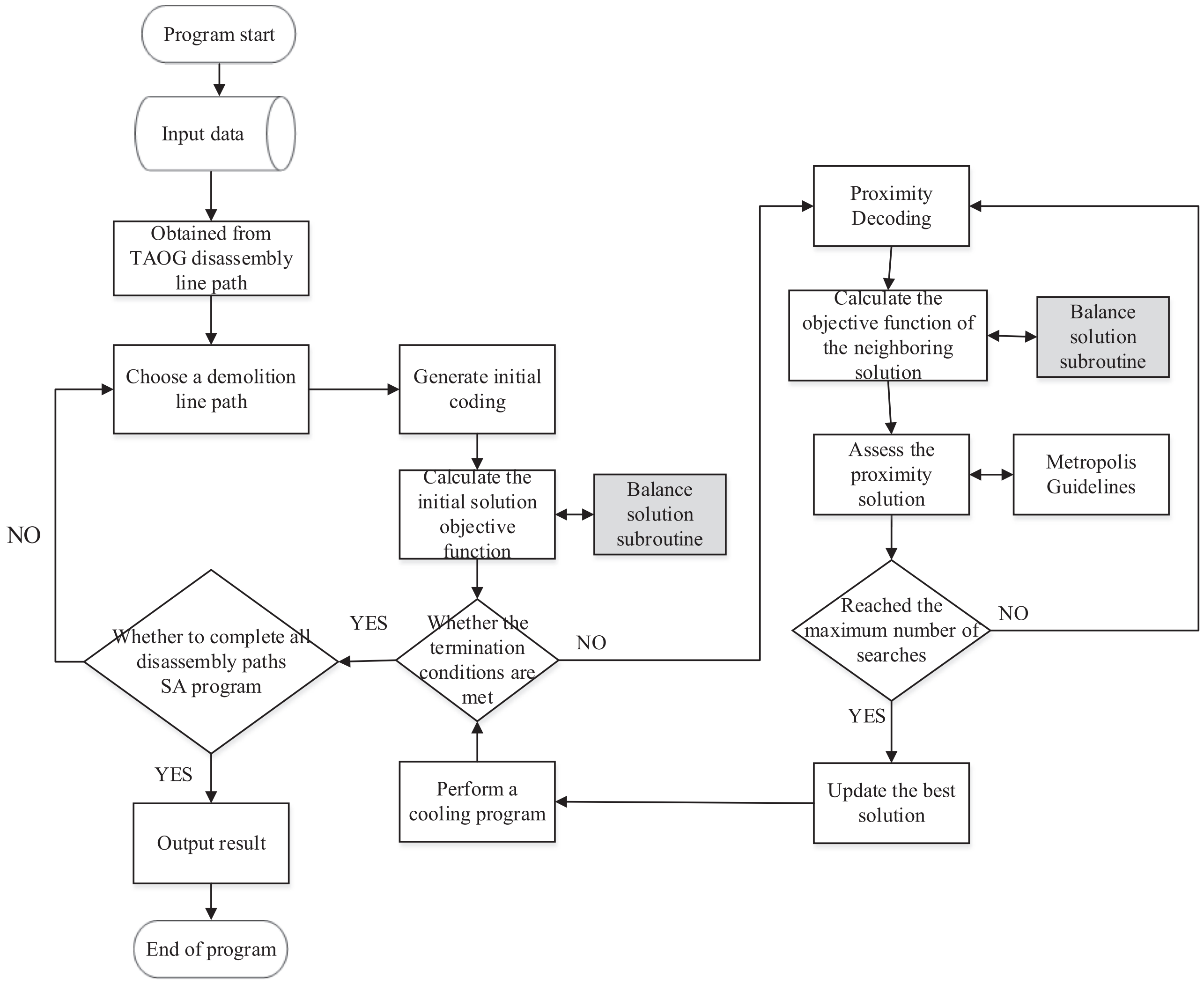

Program start
Input data
Obtained from TAOG disassembly line path
Choose a demolition line path
Generate initial coding
Calculate the initial solution objective function
Balance solution subroutine
Whether the termination conditions are met
YES
NO
Whether to complete all disassembly paths SA program
NO
YES
Output result
End of program
Perform a cooling program
Proximity Decoding
Calculate the objective function of the neighboring solution
Balance solution subroutine
Assess the proximity solution
Metropolis Guidelines
Reached the maximum number of searches
NO
YES
Update the best solution

Fig. 4. Flow chart of the hybrid simulated annealing algorithm.

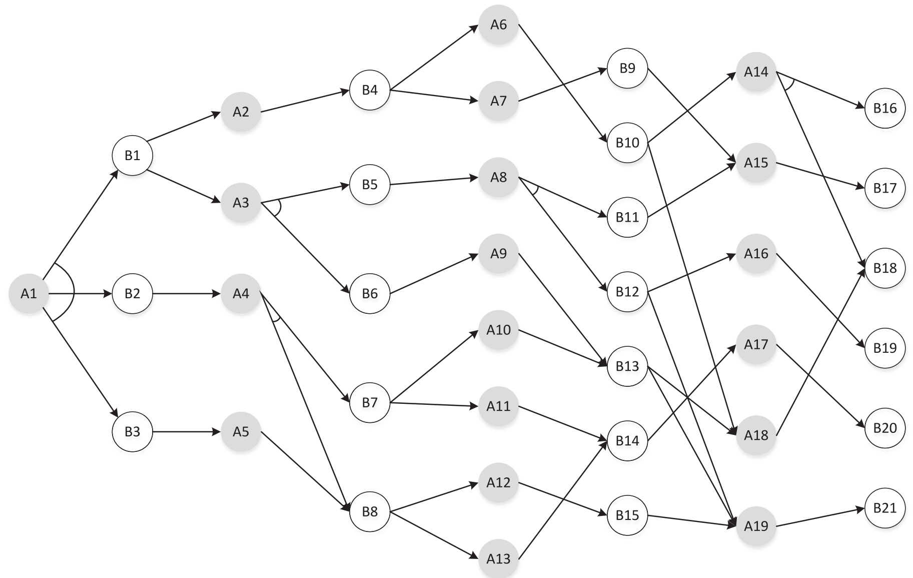


Fig. 5. TAOG of the illustrative example.

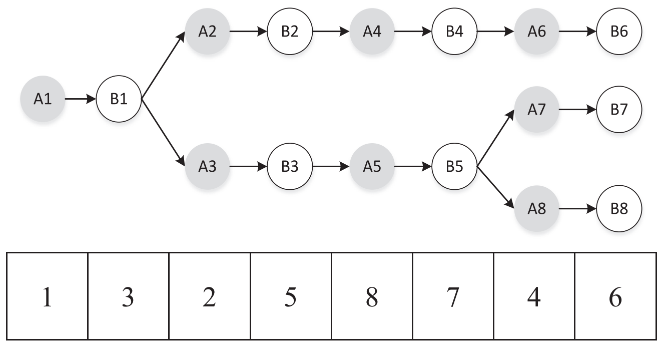


Fig. 6. Solution initialization of the illustrative example.

greater than K, stop and proceed to Step 11; otherwise, go back to Step 7 to resume the neighborhood search.

Step 11. Update the best solution. Given that the objective value is desired to be small (minimization), if the fitness of the current feasible solution (C_{sol}) is smaller than that of the best solution (B_{sol}), then $B_{sol} = C_{sol}$ and $B_{st} = C_{st}$.

Step 12. Execute the cooling procedure considering the cooling formula: Current temperature (T_p) = Current temperature (T_p) × Cooling rate (q). Return to Step 6.

Step 13. Determine whether all the disassembly paths in TAOG have been processed; if yes, proceed to Step 14; otherwise, return to Step 3.

Step 14. Report the result.

4.4. Genetic Computation-based Simulated Annealing

As an alternative solution algorithm, the Genetic-computation-based Simulated Annealing (GSA) process is developed, which combines the global search strength of the Genetic Algorithm with the local search strength of Simulated Annealing. Fig. 7 outlines the computational flow followed by a step-by-step manual of the method.

Step 1. Set the initial parameters including the initial temperature (T_0), final temperature (T_f), population size (pop), cooling rate (q), maximum search times in the strata (i.e., the cooling time K), and the crossover and mutation rates.

Step 2. Import the disassembly paths from TAOG. Select the first feasible path as a basis for repairing infeasible solutions and remove it from the list.

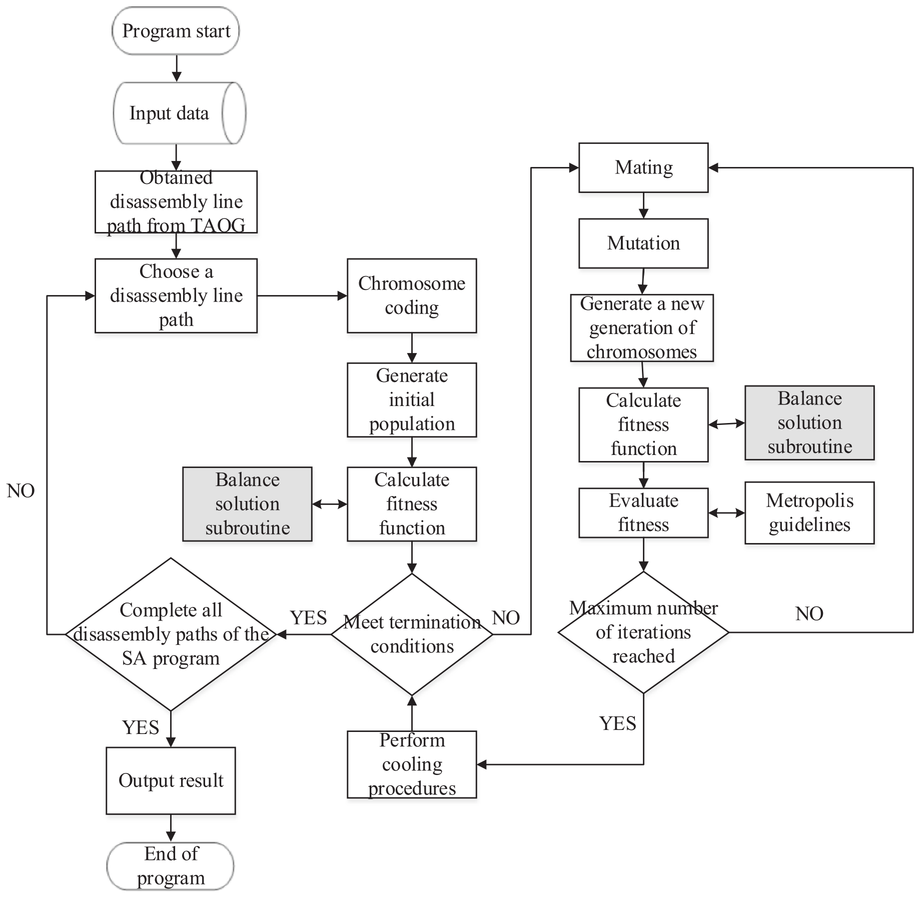
Step 3. Generate random solutions considering the population (group) size. Use the repair mechanism to ensure that every disassembly operation satisfies the precedent/antecedent constraints from TAOG.

Step 4. Encode the feasible solutions and calculate the fitness value of every individual in the population. A method like that used in HSA is considered.

Step 5. Check whether the termination condition is met. A termination condition similar to HSA (i.e., the current temperature less than or equal to the final temperature or the best solution remains the same for a certain number of iterations).

Step 6. Mating procedure. A two-point crossover function is applied to generate new solutions (offspring). In this method, two random indexes are first selected. The section between these indexes is then exchanged between the parents. Finally, the resulting offspring are corrected by replacing the duplicates with the missing values while ensuring that the precedence relationships are followed.

Taking $P_1 = [1,3,5,7,2,4,6,8]$ and $P_2 = [1,3,2,5,8,4,6,7]$ as our illustrative parents, and the cutting points of $cp_1, cp_2 = 2and5$, the resulting offsprings are $o_1 = [1,3,2,5,8,6,7,8]$ and $o_2 = [1,3,5,7,2,5,8,7]$. After a correction, the outcomes will be as follows: $o_1 = [1,3,2,5,$



Program start
Input data
Obtained disassembly line path from TAOG
Choose a disassembly line path
Chromosome coding
Generate initial population
Balance solution subroutine
Calculate fitness function
NO
Complete all disassembly paths of the SA program
YES
Meet termination conditions
NO
YES
Output result
End of program
Perform cooling procedures
Mating
Mutation
Generate a new generation of chromosomes
Calculate fitness function
Balance solution subroutine
Evaluate fitness
Metropolis guidelines
Maximum number of iterations reached
NO
YES

Fig. 7. Flowchart of the Genetic-computation-based Simulated Annealing algorithm.

$4, 6, 7, 8]$ and $o_2 = [1, 3, 4, 6, 2, 5, 8, 7]$.

Step 7. Mutation procedure. A perturbation procedure will be applied to a certain portion of the offspring (mutation rate). For this purpose, two random values from the vector are exchanged. The mutated solution is then corrected considering the precedence relationships.

Considering $o_1 = [1, 3, 2, 5, 4, 6, 7, 8]$ as an illustrative offspring, exchanging the values associated with the random indices of 2 and 7 results in the mutated solution $m_1 = [1,7,2,5,4,6,3,8]$; this vector should be corrected considering the precedence diagram shown in Fig. 4(a), which results in $m_1 = [1, 2, 4, 6, 3, 5, 7, 8]$.

Step 8. Calculate the fitness function value of the new solutions.

Step 9. Evaluate the new solutions and replace the least-fit solution (s) with the new offspring. For this purpose, the old population should be sorted considering a descending order of the fitness values. The last individual in the sorted population is then considered for the competition. The Metropolis Mechanism applied in HSA is used for deciding whether to replace the least-fit individual with the offspring.

Step 10. If the number of iterations has reached the upper value, proceed to Step 11; otherwise, return to Step 6.

Step 11. Execute the cooling procedure (Current temperature (T_p) = Current temperature (T_p) × Cooling rate (q)) and return to Step 5.

Step 12. Determine whether all the disassembly paths of TAOG have been processed; if yes, proceed to Step 13; otherwise, return to Step 2.

Step 13. Report the results.

The outcome of both algorithms includes the best-found solution code (B_{st}) and the respective objective value (B_{sol}). In the present study, the best solution contains the workstation configuration and the machine-operator configuration at every workstation. The CPU runtime of the HSA and GSA algorithms will also be reported for comparing the computational time and algorithms' efficiency.

5. Numerical experiments

The numerical experiments are conducted using MATLAB 2020b and CPLEX software on a personal computer with the following specs: Intel® Core™ i5-10210U CPU (1.60CHz) and 8 GB RAM. We first summarize the results of parameter settings, code validation, and verification of the solution algorithms. The case application then follows to analyze the findings.

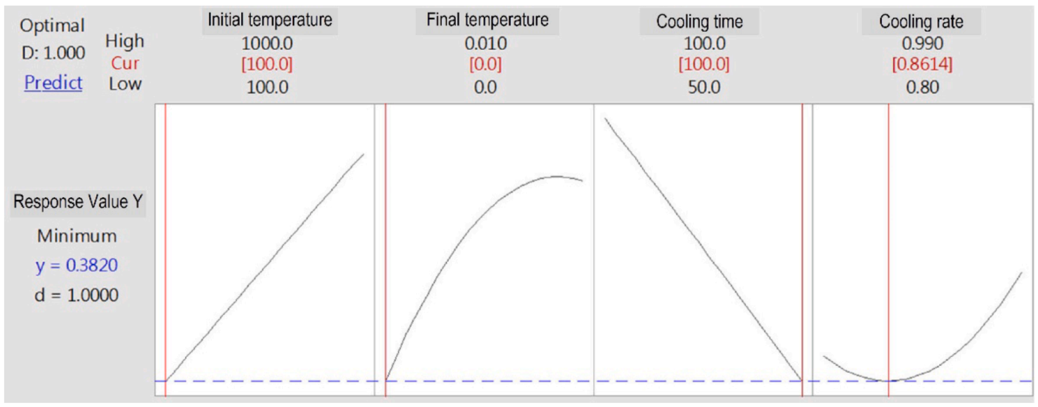
5.1. Parameter setting and model validation

The algorithm parameters' best setting should be determined before conducting the final experiments. For this purpose, the Box-Behnken design using the response surface method is considered. The calibration results analysis for the HSA and GSA algorithms are summarized in Fig. 8(a and b), respectively. On this basis, an initial temperature of 100, a final temperature of 0, and cooling parameters of 100 and 0.8614 have resulted in the best performance of HSA. Besides, the GSA parameters are set at the population size of 80, mating rate of 0.9, the mutation rate of 0.05, initial temperature of 1000, final temperature of 0.01, cooling time of 100 and cooling rate = 0.9267.

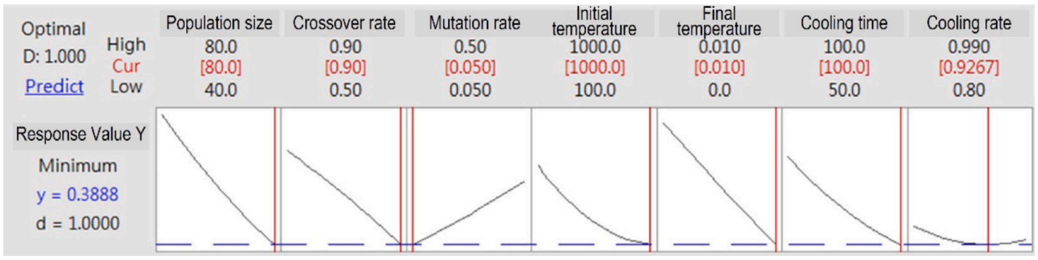
As the next step to numerical experiments, the CPLEX solver is used to solve the mathematical problem considering a small-scale test instance for validation purposes. The calibrated algorithms are also run to compare the outcomes with those obtained by the exact approach. The results are summarized in Table 2. Expectedly, the computational time of CPLEX grows exponentially with an increase in the size of the problem. Both HSA and GSA could yield the exact optimal solution obtained by the CPLEX solver, within a meaningfully shorter computational time; the matching results verify the correctness of the algorithms.

5.2. Results analysis

Data from a laptop computer disassembly factory is considered for



(a)



(b)

Fig. 8. Parameter calibration results of (a) the HSA algorithm, and (b) the GSA algorithm.

Table 2
Validation/verification results.

No.	Problem	Fitness value			Computational time (Sec)		
		CPLEX	HSA	GSA	CPLEX	HSA	GSA
1	21	0.4903	0.4903	0.4903	6.77	2.96 (-56.3 %)	6.90 (-1.9 %)
2	28	0.4502	0.4502	0.4502	36.15	8.46 (-76.6 %)	14.82 (-59 %)
3	35	0.4072	0.4072	0.4072	136.06	21.76 (-84.0 %)	35.42 (-74 %)
4	42	0.3867	0.3867	0.3867	1121.85	40.58 (-96.4 %)	41.12 (-96.3 %)
5	49	0.3653	0.3653	0.3653	16905.06	57.74 (-99.7 %)	67.89 (-99.6 %)

testing the developed optimization approach. In this case study, a total of 67 parts/components must be disassembled; Fig. 9 shows the disassembly flow diagram of the case laptop. In the disassembly line, a cycle time of 35 seconds is projected while considering five different machine types. The maximum number of machine types within each workstation is 3, and the maximum number of operators within each workstation is 3. The overall disassembly line has a limit of 20 operators. The results of optimizing the DLBP are summarized in Table 3 where HSA appeared to be more efficient while GSA yielded a better solution.

Next, sensitivity analysis is conducted to investigate the impact of parameter changes on the line balancing outcomes. For the analysis purpose, the maximum number of operators at every workstation, the operator’s skills, and cycle time changes are considered.

First, the maximum number of workstation operators is tested using alternative values while considering a fixed cycle time of 35 seconds and an upper limit of three types of machines at every station. Results obtained by GSA are shown in Table 4. Notably, the number of workstations and the total number of machines in each station show a downward trend with an increase in the maximum number of workstation operators; this may be due to the increase in machine utilization rate.

In this table, setting the maximum number of operators within each workstation to 1 has resulted in a plan employing 13 workstations, while with 2 and 3 operators per workstation, the result changes to 7 and 6 workstations, respectively. The impact of the operator’s skill to work with different machines on the line balancing results is investigated next. For this purpose, the cycle time is fixed at 35 s with a maximum of three machine types and three operators at each workstation. Results are reported in Table 5. In this table, the operators’ proficiency refers to the number of different machine types each operator is capable of working with. It is observed that the total number of workstations, the total

Table 3
Disassembly line balancing results of the case study.

	HSA	GSA
Number of selected operations (pcs)	31	31
Fitness value	0.2108	0.1958
Computational time (sec)	2005.5	3539.8
Total number of workstations (stations)	6	6
Total number of machines (pcs)	16	13
Total number of operators (persons)	13	13

Table 4
Sensitivity analysis considering the maximum number of operators.

Maximum number of operators	Total workstations	Total number of machines	Total number of operators	Fitness value
1	13	23	13	0.4736
2	7	16	13	0.2561
3	6	13	13	0.1958

Table 5
Sensitivity analysis considering the operators’ skillset.

Operators’ proficiency	Total workstations	Total number of machines	Total number of operators	Fitness value
3	7	15	14	0.2259
4	6	14	13	0.2008
5	6	13	13	0.1958

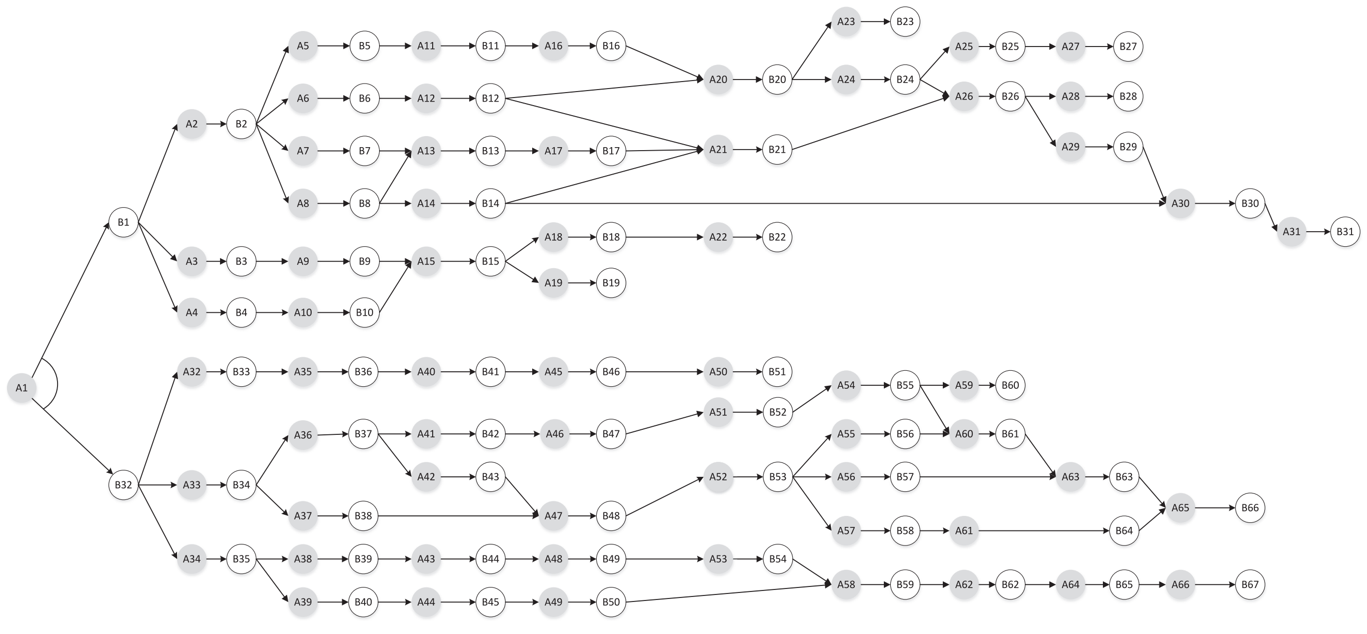


Fig. 9. Disassembly flow diagram of the case study.

number of machine types, and the total number of operators all tend to decrease when operators can work with a wider variety of machines. This is because increasing resource flexibility enables the workstations to use a wider machine/operator combination. In so doing, the number of required operators also decreases, which has resulted in better fitness values.

Finally, the maximum number of operators and machine types are fixed at 3 to analyze the impact of changing the workstation cycle times. The results are summarized in Table 6. With relaxing the cycle time, the total processing time of the operator at the workstation increases, which, expectedly, resulted in a reduction in the number of workstations and the total number of required operators. Besides, with an increase in the number of operations that can be performed at a workstation, the number of required machine types increases. Overall, the fitness value appeared to have a downward trend with increases in the cycle time.

DLBP is subject to different operational constraints; although relaxing the cycle time should theoretically enable the processing of more disassembly operations, the limited resource in the workstation blocks the change after a certain point. That is, reducing the cycle time from 1.7 to 2 times did not result in any changes where longer cycle time turns into the operator's idle time.

The findings have several practical implications. First, the analysis of results confirms that the configuration of tasks, machines, and operators within the workstations are interrelated with the resource constraints. Our results showed that more tasks can be processed within a workstation even after relaxing the cycle time settings and that the number of tasks is mostly influenced by the limitation in the number of machine types and operators. Second, the operator skills —considering the number of machines they can execute— impact the flexibility of tasks and machine configurations. A limitation in the number of skilled operators on the shop floors influences the total processing time of tasks in the workstation, the number of machine types, and the number of workstations. The redundant capacity within the workstation can be expanded as the number of skilled operators in the workstation increases. One can claim that Industry 5.0 principles can be more flexibly implemented in production environments with skilled workers while the presence of skilled workers in every workstation can also improve the human and social aspects that indirectly impact efficiency. Overall, resource efficiency in disassembly lines remains a prerequisite for the sustainable development of the remanufacturing industry.

6. Conclusions and future work

This study investigated the DLBP with multi-manned workstations and resource constraints by proposing an original mathematical model and solving it using exact and metaheuristic methods. The objective was to minimize the total number of workstations, machines, and operators while accounting for various operator skills. A real case from a laptop disassembly factory featuring multi-person workstations and resource constraints was used to showcase the applicability of the developed optimization approach. A sensitivity analysis was conducted to shed light on the practical aspects of the model.

The DLBP variant studied in this article is limited in that it does not take into account the operational uncertainties; notably, uneven flow of end-of-life items (process inputs), possible delays and disruptions along the disassembly operations, and varying quality/state of waste parts. Stochastic approaches should be developed to account for this practical feature. Additionally, line balancing integrated with collection decision variables may improve the global optimization outcomes. In many circumstances, the physical state of a part makes its disassembly unprofitable; quality evaluation features should be included in the optimization model to improve the economic viability of the disassembly activities.

As another possibility for future research, integrating line balancing of assembly and disassembly operations may be an interesting investigation with potential circularity benefits. With the increasing need for recycling, it is imaginable that third-party service providers may come into play. For such application areas, developing dynamic optimization models that allow for accepting/rejecting orders and grouping similar end-of-life items at a general products disassembly center is a worthwhile direction to investigate. In the latter example, new technology perspectives can be incorporated; for example, separating parts from end-of-life items in a way that can be used as direct feedstock in additive manufacturing-based production.

Finally, our study focused on electronic waste disassembly line balancing where there is little (if any) interference among operators compared with an assembly line studied in a recent article (Andreu-Casas et al., 2022). Interference among operators may exist in other industry situations, making it a valuable topic for extending DLBP with multi-manned workstations and resource constraints.

Table 6
Sensitivity analysis considering the cycle time changes.

Cycle Time (seconds)	Number of workstations	Total number of machines	Total number of operators	Fitness value
28	7	16	16	0.2409
35	6	13	13	0.1958
42	6	15	12	0.2008
53	5	15	9	0.1706
60	5	15	8	0.1656
70	5	15	8	0.1656

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data is provided in Fig. 9.

References

Altekin, F.T., 2017. A comparison of piecewise linear programming formulations for stochastic disassembly line balancing. International Journal of Production Research 55 (24), 7412–7434.

Andreu-Casas, E., García-Villoria, A., Pastor, R., 2022. Multi-manned assembly line balancing problem with dependent task times: a heuristic based on solving a partition problem with constraints. European Journal of Operational Research 302 (1), 96–116. https://doi.org/10.1016/j.ejor.2021.12.002.

Avikal, S., Jain, R., Yadav, H., & Mishra, P. K. (2014). A New Heuristic for Disassembly Line Balancing Problems with AND/OR Precedence Relations. *Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28-30, 2012,* 519–525.

Aydemir-Karadag, A., Turkbey, O., 2013. Multi-objective optimization of stochastic disassembly line balancing with station paralleling. Computers & Industrial Engineering 65 (3), 413–425.

Bentaha, M.L., Battaia, O., Dolgui, A., 2014. Disassembly line balancing and sequencing under uncertainty. Procedia Cirp 15, 239–244.

Cevikcan, E., Aslan, D., Yeni, F.B., 2020. Disassembly line design with multi-manned workstations: a novel heuristic optimisation approach. International Journal of Production Research 58 (3), 649–670. https://doi.org/10.1080/00207543.2019.1587190.

Cheng, C.-Y., Chen, Y.-Y., Pourhejazy, P., Lee, C.-Y., 2022. Disassembly Line Balancing of Electronic Waste Considering the Degree of Task Correlation. Electronics 11 (4), 533. https://doi.org/10.3390/electronics11040533.

ÇİL, Z.A., 2021. An exact solution method for multi-manned disassembly line design with AND/OR precedence relations. Applied Mathematical Modelling 99, 785–803. https://doi.org/10.1016/j.apm.2021.07.013.

Dong, C., Liu, P., Guo, X.W., Qi, L., Qin, S., Xu, G., 2021. Multi-Objective Ant Lion Optimizer for Stochastic Robotic Disassembly Line Balancing Problem Subject to Resource Constraints. Journal of Physics: Conference Series 2024 (1), 012014. https://doi.org/10.1088/1742-6596/2024/1/012014.

Fang, Y., Xu, H., 2020. Constraint Handling Methods for Resource-Constrained Robotic Disassembly Line Balancing Problem. Journal of Physics: Conference Series 1576 (1), 012039. https://doi.org/10.1088/1742-6596/1576/1/012039.

Gungor, A., Gupta, S.M., 2001. A solution approach to the disassembly line balancing problem in the presence of task failures. International Journal of Production Research 39 (7), 1427–1467.

Güngör, A., Gupta, S.M., 2002. Disassembly line in product recovery. International Journal of Production Research 40 (11), 2569–2589.

Habibi, M.K.K., Battaia, O., Cung, V.-D., Dolgui, A., 2017. An efficient two-phase iterative heuristic for Collection-Disassembly problem. Computers & Industrial Engineering 110, 505–514.

He, J., Chu, F., Dolgui, A., Zheng, F., Liu, M., 2022. Integrated stochastic disassembly line balancing and planning problem with machine specificity. International Journal of Production Research 60 (5), 1688–1708. https://doi.org/10.1080/00207543.2020.1868600.

Igarashi, K., Yamada, T., Gupta, S.M., Inoue, M., Itsubo, N., 2016. Disassembly system modeling and design with parts selection for cost, recycling and CO2 saving rates using multi criteria optimization. Journal of Manufacturing Systems 38, 151–164.

Jia, L., & Shuwei, W. (2017). A proposed multi-objective optimization model for sequence-dependent disassembly line balancing problem. *2017 3rd International Conference on Information Management (ICIM)*, 421–425.

Jyothi, R.K., Thenepalli, T., Ahn, J.W., Parhi, P.K., Chung, K.W., Lee, J.-Y., 2020. Review of rare earth elements recovery from secondary resources for clean energy technologies: Grand opportunities to create wealth from waste. Journal of Cleaner Production 267, 122048. https://doi.org/10.1016/j.jclepro.2020.122048.

Kannan, D., Garg, K., Jha, P.C., Diabat, A., 2017. Integrating disassembly line balancing in the planning of a reverse logistics network from the perspective of a third party provider. Annals of Operations Research 253 (1), 353–376.

Kekre, S., Rao, U.S., Swaminathan, J.M., Zhang, J., 2003. Reconfiguring a remanufacturing line at Visteon. Mexico. *Interfaces* 33 (6), 30–43.

Kerber, J.C., de Souza, E.D., Bouzon, M., Cruz, R.M., Govindan, K., 2021. Consumer behaviour aspects towards remanufactured electronic products in an emerging economy: Effects on demand and related risks. Resources, Conservation and Recycling 170, 105572. https://doi.org/10.1016/j.resconrec.2021.105572.

Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., 1983. Optimization by simulated annealing. Science 220 (4598), 671–680.

Kizilay, D., 2022. A novel constraint programming and simulated annealing for disassembly line balancing problem with AND/OR precedence and sequence dependent setup times. Computers & Operations Research 146, 105915. https://doi.org/10.1016/j.cor.2022.105915.

Koc, A., Sabuncuoglu, I., Erel, E., 2009. Two exact formulations for disassembly line balancing problems with task precedence diagram construction using an AND/OR graph. IIE Transactions 41 (10), 866–881. https://doi.org/10.1080/07408170802510390.

Kose, Y., Cevikcan, E., Ertemel, S., Murat, M., 2023. Game theory-oriented approach for disassembly line worker assignment and balancing problem with multi-manned workstations. Computers & Industrial Engineering 181, 109294. https://doi.org/10.1016/j.cie.2023.109294.

Kucukkoc, I., Li, Z., Li, Y., 2020. Type-E disassembly line balancing problem with multi-manned workstations. Optimization and Engineering 21 (2), 611–630. https://doi.org/10.1007/s11081-019-09465-y.

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

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

Liang, J., Guo, S., Du, B., Li, Y., Guo, J., Yang, Z., Pang, S., 2021. Minimizing energy consumption in multi-objective two-sided disassembly line balancing problem with complex execution constraints using dual-individual simulated annealing algorithm. Journal of Cleaner Production 284, 125418. https://doi.org/10.1016/j.jclepro.2020.125418.

Liu, C., Chen, J., Cai, W., 2022. Data-Driven Remanufacturability Evaluation Method of Waste Parts. IEEE Transactions on Industrial Informatics 18 (7), 4587–4595. https://doi.org/10.1109/TII.2021.3118466.

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

McGovern, S.M., Gupta, S.M., 2007. A balancing method and genetic algorithm for disassembly line balancing. European Journal of Operational Research 179 (3), 692–708.

Mete, S., Abidin Çil, Z., Özceylan, E., Ağpak, K., 2016. Resource Constrained Disassembly Line Balancing Problem. IFAC-PapersOnLine 49 (12), 921–925. https://doi.org/10.1016/j.ifacol.2016.07.893.

Özceylan, E., Kalayci, C.B., Güngör, A., 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), 4805–4827. https://doi.org/10.1080/00207543.2018.1428775.

Paksoy, T., Güngör, A., Özceylan, E., Hancilar, A., 2013. Mixed model disassembly line balancing problem with fuzzy goals. International Journal of Production Research 51 (20), 6082–6096.

Pourhejazy, P., 2022. Production Management and Supply Chain Integration. In *The Palgrave Handbook of Supply Chain Management*. In: Sarkis, J. (Ed.), The Palgrave Handbook of Supply Chain Management. Springer International Publishing, Cham, pp. 1–26.

Pourhejazy, P., Zhang, D., Zhu, Q., Wei, F., Song, S., 2021. Integrated E-waste transportation using capacitated general routing problem with time-window. Transportation Research Part e: Logistics and Transportation Review 145, 102169. https://doi.org/10.1016/j.tre.2020.102169.

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

Ren, Y., Zhang, C., Zhao, F., Tian, G., Lin, W., Meng, L., Li, H., 2018. Disassembly line balancing problem using interdependent weights-based multi-criteria decision making and 2-Optimal algorithm. Journal of Cleaner Production 174, 1475–1486.

Sun, B., Li, B., Ma, S., Zhu, M., Dong, C., Xiang, M., Cheng, H., Yu, Y., 2023. The recycling potential of unregulated waste electrical and electronic equipment in China: Generation, economic value, and cost-benefit analysis. Journal of Cleaner Production 402, 136702. https://doi.org/10.1016/j.jclepro.2023.136702.

Wang, T., Guo, X., Liu, S., Qi, L., & Zhao, Z. (2020). A Stochastic Sequence-dependent Multi-objective Disassembly Line Balancing Model Subject to Task Failure and Resource Constraint via Multi-objective Cuckoo Search. *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 700–705. https://doi.org/10.1109/SMC42975.2020.9283012.

Wang, K., Li, X., Gao, L., 2019. A multi-objective discrete flower pollination algorithm for stochastic two-sided partial disassembly line balancing problem. Computers & Industrial Engineering 130, 634–649.

Xiao, S., Nie, S.K., 2017. An improved adaptive multi-objective particle swarm optimization for disassembly line balancing problem. International Journal of Research in Engineering and Science 5 (5), 55–61.

Xiao, S., Wang, Y., Yu, H., Nie, S., 2017. An entropy-based adaptive hybrid particle swarm optimization for disassembly line balancing problems. Entropy 19 (11), 596.

Yin, T., Zhang, Z., Jiang, J., 2021. A Pareto-discrete hummingbird algorithm for partial sequence-dependent disassembly line balancing problem considering tool requirements. Journal of Manufacturing Systems 60, 406–428. https://doi.org/10.1016/j.jmsy.2021.07.005.

Yuan, G., Yang, Y., Pham, D.T., 2020. Multiobjective Ecological Strategy Optimization for Two-Stage Disassembly Line Balancing With Constrained-Resource. IEEE Access 8, 88745–88758. https://doi.org/10.1109/ACCESS.2020.2994065.

Zhang, Y., Zhang, Z., Guan, C., Xu, P., 2022. Improved whale optimisation algorithm for two-sided disassembly line balancing problems considering part characteristic indexes. International Journal of Production Research 60 (8), 2553–2571. https://doi.org/10.1080/00207543.2021.1897178.

Zhou, B., Bian, J., 2022. A bi-objective salp swarm algorithm with sine cosine operator for resource constrained multi-manned disassembly line balancing problem. Applied Soft Computing 131, 109759. https://doi.org/10.1016/j.asoc.2022.109759.

Zhu, X., Zhang, Z., Hu, J., 2014. An ant colony optimization algorithm for multi-objective disassembly line balancing problem. Zhongguo Jixie China Mechanical Engineering 25 (8), 1075–1079.