Profit-oriented partial disassembly line design: dealing with hazardous parts and task processing times uncertainty

Mohand Lounes Bentaha, Alexandre Dolgui, Olga Battaïa, Robert Riggs, Jack Hu

Profit-oriented partial disassembly line design: dealing with hazardous parts and task processing times uncertainty

Mohand Lounes Bentaha[a*], Alexandre Dolgui[b], Olga Battaïa[c], Robert J. Riggs[d,e] and Jack Hu[d,f]

[a] *DISP Laboratory (EA 4570), Université Lyon 2, Lyon, France;* [b] *École des Mines de Nantes, IRCCyN (UMR CNRS 6597), Nantes Cedex 3, France;* [c] *Institut Supérieur de l'Aéronautique et de l'Espace, Toulouse, France;* [d] *Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, MI, USA;* [e] *Department of Industrial Engineering, Clemson University, Clemson, SC, USA;* [f] *Department of Mechanical Engineering, The University of Michigan, Ann Arbor, MI, USA*

(*Received 26 May 2017; accepted 5 December 2017*)

This paper addresses the problem of profit-oriented disassembly line design and balancing considering partial disassembly, presence of hazardous parts and uncertainty of task processing times. Few papers have studied the stochastic disassembly line balancing problem and existing approaches have focused on heuristic and metaheuristic methods. Most existing work has concentrated on complete disassembly where task times are assumed to be normal random variables and where AND/OR graphs are not considered. The objective of this paper is the design of a serial line that obtains the maximum revenue and then balances the workload under uncertainty. The processing time of a disassembly task is assumed to be a random variable with any known probability distribution. An AND/OR graph is used to model the precedence relationships among tasks. Stochastic programming models and exact-based solution approaches combining the L-shaped algorithm and Monte Carlo sampling techniques are proposed. The relevance and applicability of the proposed models and solution methods are shown by solving efficiently a set of disassembly problem instances from the literature.

Keywords: product recovery; disassembly; line design and balancing; uncertainty; Monte Carlo sampling; sample average approximation

1. Introduction

Product recovery is becoming more important due to its social, environmental and economic benefits. Product recovery preserves resources by reducing the consumption of virgin raw materials, water and energy. In addition, this process plays a key role in minimising the amount of waste sent to landfills and diminishing air and water pollution (Ashby 2012). Through recycling, remanufacturing and reuse, product recovery aims to retrieve valuable parts and materials from discarded or End of Life (EOL) products. A mandatory step and most challenging part of product recovery process is disassembly.

Disassembly is a revalorising process, a methodical and organised separation of parts and materials of outdated products for recycling, remanufacturing and reuse. A disassembly line is more suitable than a single workstation or a disassembly cell to carry out disassembly operations with a higher productivity rate and automated disassembly. Figure 1 illustrates forward and reverse logistics flows and positions disassembly in the closed loop logistics (Ma et al. 2011). This study particularly focuses on design of disassembly systems and more specifically on disassembly line design.

Although disassembly seems to be the reverse of assembly, the disassembly process has unique characteristics which make it more complex than assembly. Gupta and Güngör (2001) provide a comparison of operational and technical considerations of assembly and disassembly lines. Indeed, the most obvious difference is the flow process which is convergent for assembly and is divergent for disassembly. In a disassembly environment, a product is broken down into many parts and subassemblies whose qualities, quantities and reliabilities cannot be controlled as in an assembly environment. The assembly process has to be complete while the disassembly process, due to technical and economic restrictions, does not have to be carried out completely and hence disassembly is usually a partial process (Lambert 2002). For example, irreversible connections of components of a product can be considered as a technical restriction with the disassembly cost being greater than the revenue obtained from retrieved parts as an economic restriction. The structure and quality of EOL products are very uncertain and even the number of components in such products cannot be predicted. Consumers may remove certain components or parts before disposing of the product or these components may be unfastened during their period of use. Moreover, an EOL product may contain certain hazardous parts necessitating special handling at a workstation of a disassembly line. These unique complex characteristics make disassembly processes and especially disassembly lines more challenging. Therefore,

*Corresponding author. Email: mohand.bentaha@univ-lyon2.fr

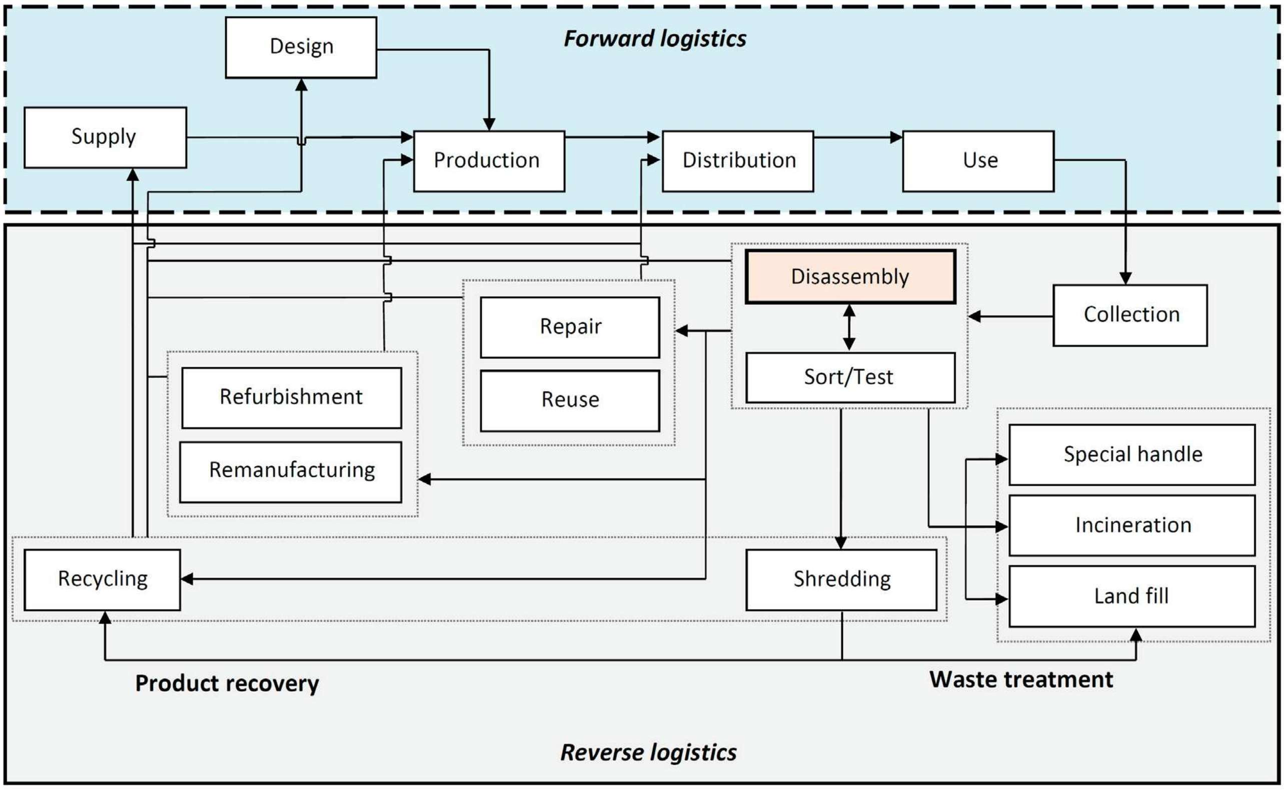

Figure 1. Forward and reverse logistics (Ma et al. 2011).

efficient decision-making tools are needed in order to optimise their performance and cost effectiveness. Such tools must take into account the high degree of uncertainty in the structure and the quality of the products to be disassembled.

This paper considers in particular the design and balancing of disassembly lines under uncertainty. Such a line consists of an ordered sequence of workstations connected by a material handling system, which allows the transportation of work-pieces (parts and subassemblies) from one workstation to another. Certain parts or subassemblies may be hazardous and require a particular treatment incurring a supplementary cost. As mentioned above, the disassembly process is typically partial. The optimisation problem considered aims to assign a given set of disassembly operations (tasks) to an ordered sequence of workstations while respecting precedence and cycle time constraints under uncertainty. The case of partial disassembly is studied and the objective is to maximise the profit produced by the line. The line profit is calculated as the difference between the positive revenue generated by recovered parts, and the line operation cost taken as negative revenue. The latter includes the workstation operation costs, additional costs of workstations handling hazardous parts of the EOL product, as well as penalty costs caused by the cycle time constraint violations. This objective allows the minimisation of the number of workstations of the line, which in turn minimises the total idle time. Then, for this minimum number of workstations, an optimal balance is determined under uncertainty for the resulting disassembly line. The obtained balance ensures that idle times are as similar as possible for all the line workstations. This scheme of lexicographic optimisation is motivated for economic reasons. In fact, the revenue generated by EOL products recovery constitutes one of the most important attractions for investment in disassembly lines. Thus, the criterion of line profit is considered as dominant for disassembly lines.

In order to help designers to design economically viable disassembly lines and therefore to facilitate the implementation of disassembly end of life scenarios in practice, this paper develops an efficient procedure to design disassembly lines under uncertainty brought by the EOL products. The added value of this work is the ability to assess the disassembly profit or cost for a product at the end of life. Such an assessment may help to take the decisions not only for EOL options but even at the product design stage. The proposed model is rather complete and is capable of treating at the same time partial disassembly, presence of hazardous material and uncertainty of task processing times.

The remainder of the paper is organised as follows: Section 2 provides an overview of the relevant literature on disassembly line design and balancing. Section 3 follows with a formal description of the considered optimisation problem. Section 4 presents the developed solution algorithm that combines the L-shaped method with Monte Carlo sampling techniques.

Section 5 is dedicated to the analysis of numerical experiments and Section 6 concludes the paper and suggests further research directions.

2. Previous work

The disassembly line design problem was introduced by Güngör and Gupta (1999). This problem called DLBP consists of the assignment of disassembly tasks to workstations of a disassembly line with the objective of optimising some measures of effectiveness (time, labour or money).

To deal with the deterministic DLBP, heuristic and metaheuristic approaches were developed. An iterative heuristic using branch and bound technique was developed in Lambert and Gupta (2005) to deal with the line balancing problem subjected to sequence dependent costs. A multi-objective heuristic for U-shaped DLBP was developed in Avikal, Jain, and Mishra (2013). The authors considered several performance criteria in a lexicographic order: minimise the workstation idle times, maximise the priority of removing hazardous components and maximise the priority of removing high-demand components. Two multi-objective metaheuristics, a distributed agent ant system and an uninformed deterministic search, for the design and balancing of disassembly lines were developed and compared in McGovern and Gupta (2005). Other multi-objective formulations of the DLBP were presented in McGovern and Gupta (2006), Ding et al. (2010) where the objectives are also ordered lexicographically. Ant colony-based approaches were developed for this problem. Several metaheuristic approaches were developed to solve problems of disassembly process planning, disassembly scheduling and sequence dependent disassembly line balancing, respectively, a Petri Net-based approach (Tiwari et al. 2002), constraint-based simulated annealing approach (Prakash, Ceglarek, and Tiwari 2012) and particle swarm optimisation, a tabu search, ant colony optimisation (Kalayci and Gupta 2013). A beam search-based approach for DLBP was proposed in Mete et al. (2016) to minimise the number of workstations. Tang and Zhou (2006) developed a Petri net approach where a heuristic was employed to maximise the line productivity. Qualitative and quantitative comparisons of different heuristics and metaheuristics for DLBP, including genetic algorithm, ant colony optimisation method, greedy algorithm, greedy/hill-climbing, greedy/2-opt hybrid heuristics and hunter-killer heuristics, were undertaken in McGovern and Gupta (2007).

Mathematical programming formulations and exact solution approaches were also proposed for the DLBP. Altekin, Kandiller, and Ozdemirel (2008) developed an integer programming formulation for profit maximisation for the case of partial disassembly. Koc, Sabuncuoglu, and Erel (2009) considered DLBP with the objective of minimising the number of workstations. Two exact approaches based on mixed integer and dynamic programmes were developed. To balance a mixed-model disassembly line, a linear physical programming approach was proposed in Ilgin, Akçay, and Araz (2017). A comprehensive survey on production line design and balancing particularly on disassembly can be found in the literature (Ilgin and Gupta 2010; Battaïa and Dolgui 2013; Bentaha, Battaïa, and Dolgui 2015a).

Few studies in the literature, however, have studied the DLBP under uncertainty (Bentaha, Battaïa, and Dolgui 2015a). In Riggs, Battaïa, and Hu (2015), the joint precedence graph approach, known from assembly line balancing, was adapted for disassembly line balancing in order to deal with EOL product states. A fuzzy coloured Petri net model with a heuristic solution method was proposed by Turowski and Morgan (2005) to deal with the human factors (uncertainty of disassembly task times) and product condition. A collaborative ant colony algorithm for stochastic mixed-model U-shaped disassembly line balancing was developed in Agrawal and Tiwari (2006). Task times were assumed stochastic with known normal probability distributions. A self-guided ants metaheuristic was proposed in Tripathi et al. (2009) for the disassembly line sequencing problem, where fuzzy optimisation model was developed with the objective of maximising the net revenue of the disassembly process. Tuncel, Zeid, and Kamarthi (2014) used a Monte Carlo-based reinforcement learning technique to solve the multi-objective DLBP under demand variations of the EOL products. Güngör and Gupta (2001) proposed a heuristic to deal with task failures caused by defective parts or joints in the EOL product. A MIP-based predictive-reactive approach to deal with task failures was also developed in Altekin and Akkan (2011). In these two studies, a task failure can be seen as a disruption and the authors proposed remedial actions to deal with such short-term disturbances. A first study to deal dynamically with real time production line disturbances is proposed in Antoine et al. (2016). A binary bi-objective non-linear programme was developed in Aydemir-Karadag and Turkbey (2013) for disassembly line design and balancing under uncertainty of the task times. Disassembly task times were assumed to be independent random variables with known normal probability distributions. Complete disassembly was considered and a genetic algorithm was designed to solve the problem.

This literature review shows that there are few papers that have dealt with stochastic DLBP and these are either restricted to the study of demand fluctuations, task failures, condition of the EOL products or stochastic task times (as normal random variables) with only heuristic/metaheuristic solution methods being proposed without information about the solution quality. Many of these works have not considered the case of partial disassembly and have not used an AND/OR graph to model the precedence relationships among tasks. It is shown in the literature (Koc, Sabuncuoglu, and Erel 2009) that the integration of the AND/OR graphs in the DLBP formulation allowed better solutions to be obtained compared to the use of AND precedence diagrams. Most of the AND/OR graphs used in the literature are part-based diagrams, and such graphs do not exploit the precedence relations among tasks and parts of the EOL product. The existing solution methods are unadapted to simultaneously take into account other sources of uncertainty, and only consider task times that are normally distributed.

To bridge the gap, we developed in Bentaha et al. (2014), Bentaha, Battaïa, and Dolgui (2014a, 2014b, 2014c, 2014d), Bentaha, Battaïa, and Dolgui (2015b), Bentaha et al. (2015) mathematical models and exact solution approaches to design disassembly (or assembly) lines under uncertainty of the task processing times. Partial disassembly has been taken into account and complex and/or precedence relationships among tasks have been integrated.

In Bentaha et al. (2014), Bentaha, Battaïa, and Dolgui (2014a, 2014b, 2014c), uncertainty was modelled using workstation expectation times. This problem was introduced in Bentaha et al. (2014). In Bentaha, Battaïa, and Dolgui (2014a), the joint problem of disassembly line design and tasks sequencing was studied. In Bentaha, Battaïa, and Dolgui (2014c), a Lagrangian relaxation was proposed to maximise the disassembly line profit. In Bentaha, Battaïa and Dolgui (2014b), the line balancing problem has been undertaken under the assumption of the fixed number of workstations. In Bentaha, Battaïa, and Dolgui (2015b), Bentaha et al. (2015), the goal was to guarantee a certain operational level defined by the designer. Uncertainty was modelled using joint probabilistic cycle time constraints. In Bentaha, Battaïa, and Dolgui (2014d), uncertainty was modelled using the notion of recourse cost. In other words, this study aimed in minimising the line stoppage costs caused by the task processing time uncertainties. In Bentaha, Battaïa, and Dolgui (2014d), the objective was the line cost minimisation. The line cost includes the operation costs for workstations and penalty costs generated by the cycle time constraint violations. Thus, revenue generated by recovered parts was not considered. However, as mentioned in Section 1, the revenue generated by EOL products recovery constitutes one of the most important attractions for investment in disassembly lines. In addition, handling hazardous parts and balancing the workload of the resulted line in Bentaha, Battaïa, and Dolgui (2014d) were not considered.

The present paper deals with the profit-oriented partial DLBP under uncertainty of the disassembly task times (SP-DLBP) which are assumed to be random variables with known probability distributions. Task times can be modelled using any probability distribution. To deal with this uncertainty, a stochastic programme is developed. An AND/OR graph is used to model the precedence relations among tasks. The objective is to maximise the net revenue produced by the line. As mentioned in Section 1, the line profit is calculated as the difference between the positive revenue generated by recovered parts, and the line operation cost, considered as negative revenue. The cost includes the workstation operation costs, additional costs of workstations handling hazardous parts of the EOL product, as well as penalty costs incurred by the cycle time constraint violations. This work addresses the aforementioned shortcomings in Bentaha, Battaïa, and Dolgui (2014d).

The optimisation procedure proposed in this paper consists of two phases: (1) a minimum number of workstations for the line is found by the maximisation of the line revenue and (2) for this minimum number of workstations, an optimal balance is determined under uncertainty. An exact-based solution method is developed. The method integrates efficient Monte Carlo sampling techniques with the L-shaped algorithm. The developed solution method computes lower and upper bounds for the problem solutions and provides the corresponding confidence intervals as well as optimality gaps. In addition, the proposed solution approach deals with any probability distribution for task times and can be extended to deal with additional stochastic parameters without additional complexities. A detailed problem formulation is given in the next section.

3. Problem definition and formulation

In this section, the SP-DLBP and the graph used to model disassembly alternatives of an EOL product and relationships among tasks and subassemblies are defined (Section 3.1). The first phase of the SP-DLBP is formulated in Section 3.2 while the second phase is formulated in Section 3.3.

3.1 *AND/OR graph and disassembly alternatives*

The SP-DLBP consists of the assignment of the disassembly tasks, a set I, of an EOL product to an ordered sequence of workstations, a set J, while satisfying precedence and cycle time constraints under uncertainty of the task times. The cycle time (C_T) is the amount of time allocated to each station to complete its assigned tasks. It is the ratio of the planning period length to the number of products that need to be disassembled in order to meet the demand.

The solution method developed in this paper considers the following assumptions: a single type discarded product is to be partially (or completely) disassembled on a straight paced line; the EOL products are sufficiently available and their demand quantities are deterministic; all received EOL products contain all their parts with no addition or removing of components. Disassembly tasks of every product have to be assigned to the workstations in the same manner. All products are assumed to be of the same type. Certain components of a product are hazardous. Task times are assumed to be random variables with known probability distributions. A disassembly task can be performed by all workstations but only by one at a time.

Each component or subassembly of a product has a certain recovery (resale) value which is considered the same for all components or subassemblies of the same type. This is particularly the case if the resale value of a given component or subassembly is only based on the contained materials. In this paper, the study is focused on the variability in disassembly task processing times. A fixed cost is allocated per operating time unit of a workstation with an additional fixed cost per operating time unit of a workstation handling hazardous parts.

All possible disassembly alternatives of an EOL product are modelled using an AND/OR graph (Homem de Mello and Sanderson 1990; Koc, Sabuncuoglu, and Erel 2009; Bentaha, Battaïa, and Dolgui 2015b). Such a graph represents explicitly the precedence relationships existing among tasks and parts or subassemblies (Bentaha, Battaïa, and Dolgui 2014d; Bentaha et al. 2015). The use of an AND/OR graph is illustrated with a rigid caster example (see Figure 2 and Table 1).

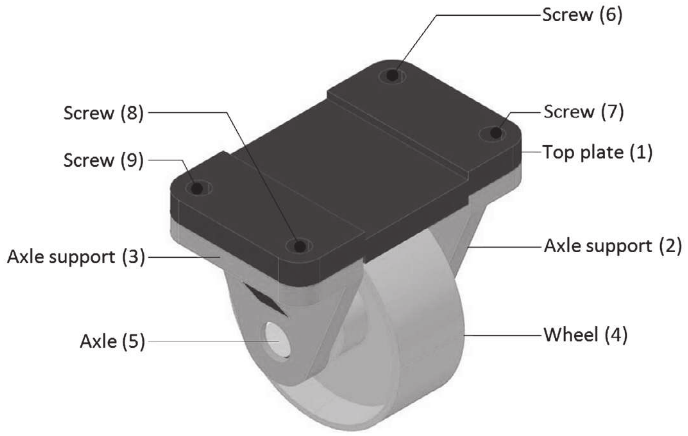

Figure 2. A rigid caster example.

Table 1. The rigid caster associated disassembly tasks and the corresponding subassemblies and/or components.

Task	Subassembly	Components	Task	Subassembly	Components
1	1:5,7:9	6	17	1,3,9	8
2	1:6,8,9	7	18	1,3,8	9
3	1:7,9	8	19	1,3,9	7
4	1:8	9	20	1,2,7	3;4;5;9
5	1,3,8,9	2;4;5;7	21	1,3,8	2;4;5;7
6	1:5,7,9	8	22	1,2,7	3;4;5;8
7	1:5,7,8	9	23	1,3,9	2;4;5;6
8	1,3,8,9	2;4;5;6	24	1,2,6	3;4;5;9
9	1:6,9	8	25	1,3,8	2;4;5;6
10	1:6,8	9	26	1,2,6	3;4;5;8
11	1:5,7,9	6	27	1,2,7	6
12	1:6,9	7	28	1,6,6	7
13	1,2,6,7	3;4;5;9	29	–	1;3;9
14	1:5,7,8	6	30	–	1;3;8
15	1:6,8	7	31	–	1;2;7
16	1,2,6,7	3;4;5;8	32	–	1;2;6

The product to be disassembled is composed of nine components as shown in Figure 2. The AND/OR graph in Figure 3 corresponds to the rigid caster example and is constructed as follows: each subassembly is represented by a node labelled $A_k, k \in K$. For example, node A_0 represents the rigid caster which can be noted as '1:9', A_1 represents the subassembly '1:5,7:9', etc. (see Table 1). Subassemblies with one component are not represented; set K contains the indices of all possible subassemblies that can be generated by the tasks from I. Each node labelled $B_i, i \in I$, represents a disassembly task. For instance, node B_1 represents disassembly task '1', B_2 represents disassembly task '2', etc. Two types of arcs define the precedence relations between the subassemblies and disassembly tasks: AND and OR. The first type imposes a mandatory precedence relation and the second type is employed for optional precedence dependencies where only one option is chosen for a final solution. For example, if a disassembly task generates two subassemblies or more, then it is related to these subassemblies by AND-type arcs (in bold in Figure 3). If several concurrent disassembly tasks may be performed on a subassembly, this subassembly is related to these disassembly tasks by OR-type arcs. A sink node s is introduced and linked with dummy arcs to all disassembly tasks. Thus, if the dummy task s is assigned to a workstation, the disassembly process is finished (partial or complete disassembly). Table 1 summarises all the possible disassembly tasks that can be performed on the rigid caster in Figure 2. For each task, the corresponding generated components and/or subassemblies are given.

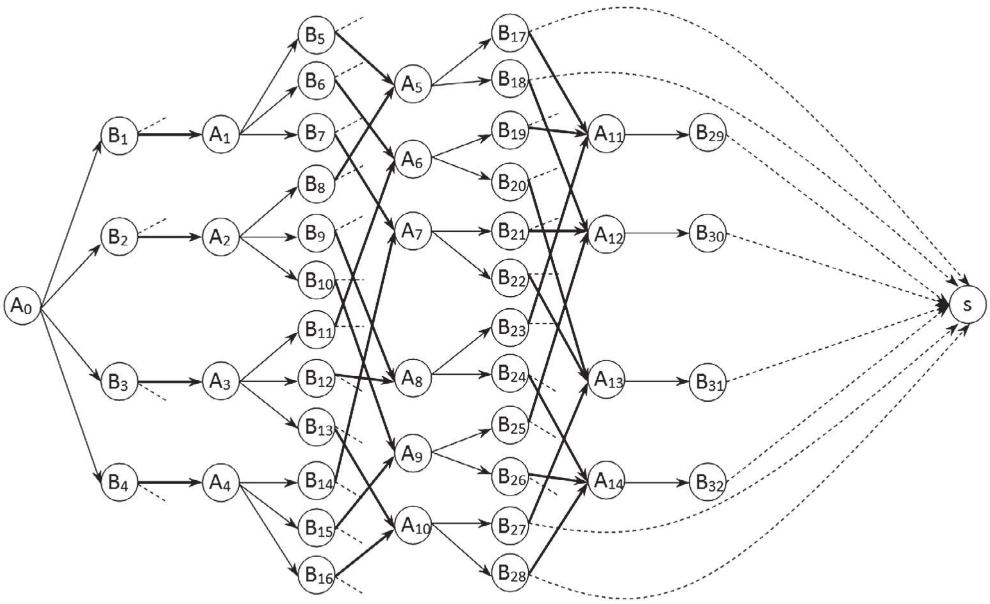

Figure 3. A rigid caster AND/OR graph.

3.2 *First phase: line profit maximisation*

As mentioned earlier, disassembly task times $t_i, i \in I$, are assumed to be stochastic variables with known probability distributions. They are represented by a random vector $\tilde{\xi} = (\tilde{t}_1, \tilde{t}_2, \ldots, \tilde{t}_\mathsf{N})$ from a set $\Xi \subset \mathbb{R}_+^\mathsf{N}$.

Let $\tilde{t}_i = t_i(\tilde{\xi}), i \in I$. A disassembly task $i \in H \subset I$ is called hazardous if its execution generates a hazardous subassembly or component. Disassembly tasks have to be assigned to the workstations of the line under precedence constraints given by the AND/OR graph, i.e. exactly one alternative sequence of the disassembly process is chosen, and at the same time the line profit is maximised. In this paper, the line profit is the difference between the positive revenue generated by recovered parts and the line operation cost taken as negative revenue. The line cost includes the workstation operation costs, additional costs of workstations handling hazardous parts and penalty costs incurred by workstation overloads. Each time the duration of tasks assigned to one workstation exceeds the given line cycle time C_T, a corrective action is needed, and its cost is included in the objective function.

To formulate the SP-DLBP, the following notation is used.

3.2.1 *Sets and parameters*

- I: set of disassembly task indices: $I = \{1, 2, \ldots, \mathsf{N}\}$, $\mathsf{N} \in \mathbb{N}^*$;
- J: set of workstation indices: $J = \{1, 2, \ldots, \mathsf{M}\}$, $\mathsf{M} \in \mathbb{N}^*$;
- H: hazardous disassembly task index set;
- L: set of part indices: $L = \{1, 2, \ldots, \mathsf{L}\}$, $\mathsf{L} \in \mathbb{N}^*$;
- K: set of indices for the generated subassemblies: $K = \{0, 1, \ldots, \mathsf{K}\}$, $\mathsf{K} \in \mathbb{N}$;

L_i: set of retrieved parts by the execution of disassembly task B_i, $i \in I$;
A_k: a subassembly, $k \in K$;
B_i: a disassembly task, $i \in I$;
s: the AND/OR graph's sink node;
r_l: revenue generated by part l, $l \in L$;
F_c: fixed cost per operating time unit of a workstation, $F_c > 0$;
q: time unit fixed recourse cost;
C_T: cycle time, $C_T > 0$;
C_h: additional cost per time unit for stations handling hazardous parts, $C_h > 0$;
P_k: set of indices for predecessors of A_k, $k \in K$, i.e. $P_k = \{i \mid B_i \text{ precedes } A_k\}$;
S_k: set of indices for successors of A_k, $k \in K$, $S_k = \{i \mid A_k \text{ precedes } B_i\}$.

3.2.2 *Decision variables*

$$x_{ij} = \begin{cases} 1, & \text{if disassembly task } B_i \text{ is assigned} \\ & \text{to workstation } j; \\ 0, & \text{otherwise.} \end{cases} \qquad h_j = \begin{cases} 1, & \text{if a hazardous task is assigned} \\ & \text{to workstation } j; \\ 0, & \text{otherwise.} \end{cases}$$

$$x_{\mathsf{s}j} = \begin{cases} 1, & \text{if the dummy task } \mathsf{s} \text{ is assigned} \\ & \text{to workstation } j; \\ 0, & \text{otherwise.} \end{cases}$$

$y_j(\tilde{\xi}) \geqslant 0$, $j \in J$, measures the amount of time exceeding C_T if there is any.

For the problem described, the following mathematical model has been developed.

3.2.3 *Stochastic mixed integer programme*

$$\max \left\{ \sum_{i\in I}\sum_{j\in J}\sum_{l\in L_i} r_l \cdot x_{ij} - \left[C_T\Big(F_c \cdot \sum_{j\in J} j \cdot x_{\mathsf{s}j} + C_h \cdot \sum_{j\in J} h_j\Big) + \mathbb{E}_{\tilde{\xi}}\Big(q \cdot \sum_{j\in J} y_j(\tilde{\xi})\Big)\right]\right\} \tag{I}$$

s.t.

$$\sum_{i\in S_0}\sum_{j\in J} x_{ij} = 1 \tag{1}$$

$$\sum_{j\in J} x_{ij} \leqslant 1, \forall i \in I \tag{2}$$

$$\sum_{i\in S_k}\sum_{j\in J} x_{ij} \leqslant \sum_{i\in P_k}\sum_{j\in J} x_{ij}, \forall k \in K\backslash\{0\} \tag{3}$$

$$\sum_{i\in S_k} x_{iv} \leqslant \sum_{i\in P_k}\sum_{j=1}^{v} x_{ij}, \forall k \in K\backslash\{0\}, \forall v \in J \tag{4}$$

$$\sum_{j\in J} x_{\mathsf{s}j} = 1 \tag{5}$$

$$\sum_{j\in J} j \cdot x_{ij} \leqslant \sum_{j\in J} j \cdot x_{\mathsf{s}j}, \forall i \in I \tag{6}$$

$$h_j \geqslant x_{ij}, \forall j \in J, \forall i \in H \tag{7}$$

$$\sum_{i\in I} t_i(\tilde{\xi}) \cdot x_{ij} - y_j(\tilde{\xi}) \leqslant C_T, \forall j \in J \tag{8}$$

$$(x_{\mathsf{s}j}, x_{ij}, h_j) \in \mathfrak{X} \subseteq \{0, 1\}^{|J|\cdot(|I|+2)} \tag{9}$$

$$y_j(\tilde{\xi}) \geqslant 0, \forall j \in J \tag{10}$$

The terms of the objective function represent, respectively, the earned profit of retrieved parts, the cost of operating workstations, the additional cost for handling hazardous parts and the expected recourse cost $\mathbb{E}_{\tilde{\xi}}$ with respect to the distribution of $\tilde{\xi}$.

If the dummy task s is assigned to workstation j, then j defines the number of processed stations. Constraint (1) imposes the selection of only one disassembly task, i.e. OR-successor, to begin the disassembly process. Without loss of generality, we assume that at least one task is required for the disassembly process. Naturally, this task has to be selected from the alternative disassembly tasks realisable for the initial EOL product A_0. Only one alternative has to be selected. The following disassembly options will be determined by the corresponding A-node and B-nodes connected to it. Constraint set (2) indicates that a task is to be assigned to at most one workstation. Constraints (3) ensure that only one OR-successor is selected. Constraint set (4) defines the precedence relations among tasks. Constraint (5) imposes the assignment of the dummy task s to one station. Constraints (6) ensure that all disassembly tasks are assigned to lower or equal-indexed workstations than the one to which s is assigned. Constraints (7) ensure the value of h_j to be 1 if at least one hazardous task is assigned to a workstation j. Constraints (8) represent the cycle time limitation. The variable $y_j(\tilde{\xi})$, $j \in J$ measures the amount of time exceeding C_T if there is any at each opened workstation. Finally, sets (9) and (10) represent constraints on the possible values of the decision variables.

Let Z be the set of all possible solutions of programme (**I**). An assignment of the disassembly tasks to workstations of the line is always possible and the number of the possible assignments is finite, i.e. $Z \neq \varnothing$ and finite. It is straightforward that $0 \leqslant \mathbb{E}_{\tilde{\xi}}\Big(q \cdot \sum_{j \in J} y_j(\tilde{\xi})\Big) < \infty$.

Since Z is not empty and the objective function defines finite values, then problem (**I**) possesses an optimal solution.

Let x be a vector of decision variables $x_{ij}, x_{\mathsf{s}j}, h_j, \forall i \in I, \forall j \in J$ and $X = \{x|$ constraints (1)-(7) and (9) are satisfied$\}$. If $\tilde{\xi}$ has a finite-discrete distribution $\{(\xi_\ell, p_\ell),\ \ell \in \mathfrak{D}, p_\ell > 0, \forall \ell \in \mathfrak{D}\}$, where $\mathfrak{D} = \{1, 2, \ldots, \mathtt{D}\}$, $\mathtt{D} \in \mathbb{N}^*$ and p_ℓ is the realisation probability of ξ_ℓ of $\tilde{\xi}$, then programme (**I**) is an ordinary linear programme with a dual decomposition structure. Programme (**II**) given below represents a deterministic equivalent version of programme (**I**) and shows its particular block structure; each programme (11) can be seen as a block. The programme (**II**) is a two-stage stochastic linear mixed integer programme with fixed recourse (Birge and Louveaux 1997). This same programme, i.e. (**II**), will be used in the solution method.

$$\max \left\{ \sum_{i \in I} \sum_{j \in J} \sum_{l \in L_i} r_l \cdot x_{ij} - \left[C_T \Big(F_c \cdot \sum_{j \in J} j \cdot x_{\mathsf{s}j} + C_h \cdot \sum_{j \in J} h_j \Big) + \mathcal{Q}^{\mathtt{D}}(x) \right] \right\} \tag{II}$$

s.t. $x \in X$

$$\text{where } \mathcal{Q}^{\mathtt{D}}(x) = \sum_{\ell=1}^{\mathtt{D}} p_\ell \cdot \mathcal{Q}(x, \xi_\ell)$$

$$\text{and } \mathcal{Q}(x, \xi_\ell) = \min \Big\{ q \cdot \sum_{j \in J} y_j(\xi_\ell) \Big| \sum_{i \in I} t_i(\xi_\ell) \cdot x_{ij} - y_j(\xi_\ell) \leqslant C_T,\ y_j(\xi_\ell) \geqslant 0, \forall j \in J \Big\},\ \ell \in \mathfrak{D} \tag{11}$$

Depending on the number of realisations of $\tilde{\xi}$, i.e. $\mathtt{D}$, programme (**II**) may become very large in scale, but its block structure can be exploited efficiently by specially designed algorithms such as the L-shaped algorithm (Ahmed and Shapiro 2002). The L-shaped method, proposed by Van Slyke and Wets (1969), is a variant of Benders' decomposition. The initial problem (**I**) is decomposed into a master problem and a subproblem. In this paper, the 0–1 binary decision variables, $x \in \mathfrak{X}$, in the master problem are called master variables and the continuous variables, $y_j(\xi)$, $\xi \in \Xi, \forall j \in J$, of the subproblem are called subproblem variables.

3.3 *Second phase: idle time leveling*

In the first phase, a number of workstations $\mathtt{m}^* = j^* \leqslant \mathsf{M}$ (where $x_{\mathsf{s}j^*} = 1$) and a subset of tasks $I^* \subset I$ (a disassembly alternative) are determined under uncertainty of the task times. An example of such a disassembly alternative and a selected subset I^* of disassembly tasks is illustrated in Figure 4. The selected alternative (in bold) is represented by an AND-graph and $I^* = \{B_2, B_9, B_{23}\}$. There are no more OR-relations since a decision of disassembling partially a product is made. For this reason, only direct precedence relationships among tasks are considered, i.e. generated subassemblies $\{A_0, A_2, A_8\}$ are simply deleted. The simple graph in Figure 5 is then created.

In the second phase, a balance measure is optimised. This measure seeks to assign the disassembly tasks I^* to the $\mathtt{m}^*$ stations ensuring (if possible) similar idle time at each workstation, i.e. total idle time balancing or levelling. Let $J^* = \{1, 2, \ldots, \mathtt{m}^*\}$,

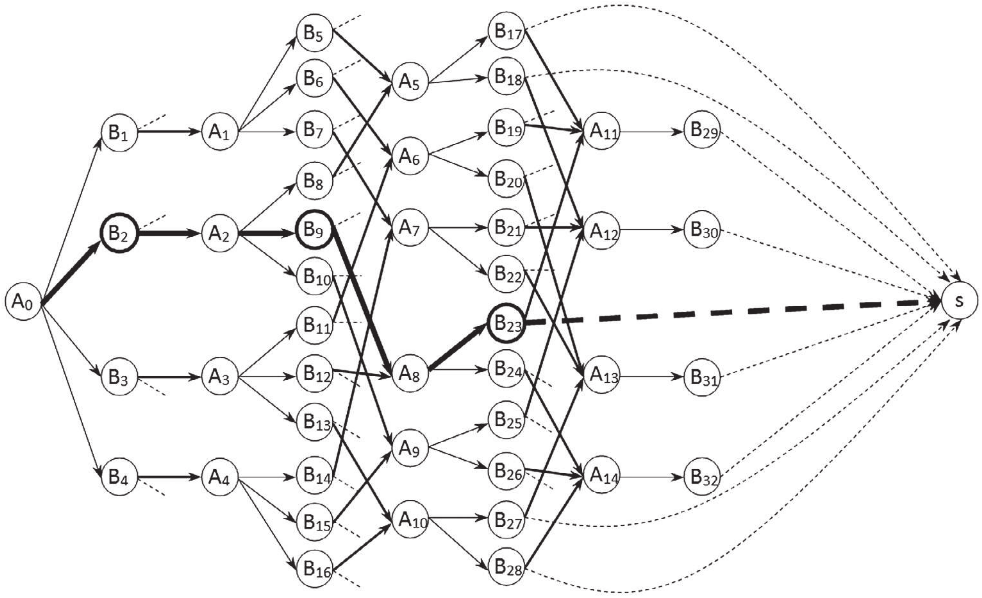

Figure 4. A selected disassembly alternative.

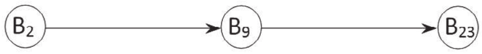

Figure 5. Obtained simple AND-graph.

$ST_j(\tilde{\xi}) = \sum_{i \in I^*} t_i(\tilde{\xi}) \cdot x_{ij}, \forall j \in J^*$ and $\text{Pred}(i) = \{i' \in I^* | \, i' \text{ precedes } i\}, i \in I^*$. The maximum of the expectations of the differences among all workstations workloads is minimised in the non-linear programme (**III**) as follows.

$$\min \max_{\forall j, j' \in J^*, j \neq j'} \mathbb{E}_{\tilde{\xi}} \left(\left| ST_j(\tilde{\xi}) - ST_{j'}(\tilde{\xi}) \right| \right) \quad \textbf{(III)}$$

s.t.

$$\sum_{j \in J^*} x_{ij} = 1, \forall i \in I^* \quad (12)$$

$$\sum_{j \in J^*} j \cdot x_{i'j} \leqslant \sum_{j \in J^*} j \cdot x_{ij}, \forall i \in I^*, \forall i' \in \text{Pred}(i) \quad (13)$$

$$x_{ij} \in \{0, 1\}, \forall i \in I^*, \forall j \in J^* \quad (14)$$

Constraints (12) ensures that a task is assigned to only one workstation. Constraint set (13) models the precedence relationships among tasks.

Let x be a vector of decision variables $x_{ij}, \forall i \in I^*, \forall j \in J^*$ and $X = \{x |$ constraints (12), (13) and (14) are satisfied$\}$. Let $\mathcal{B} = \{1, 2, \ldots, \mathtt{B}\}, \mathtt{B} \in \mathbb{N}^*$ and $\mathtt{S}_l(\tilde{\xi}) = \left(ST_j(\tilde{\xi}) - ST_{j'}(\tilde{\xi}), j, j' \in J^*, j \neq j' \right), l \in \mathcal{B}$, where $\mathtt{B} = \binom{|J^*|}{2}$. The programme (**IV**) given below represents an equivalent version of programme (**III**).

$$\min Y \quad \textbf{(IV)}$$

$$\text{s.t. } x \in X$$

$$Y \geqslant \mathbb{E}_{\tilde{\xi}} \left(\left| \mathtt{S}_l(\tilde{\xi}) \right| \right), \forall l \in \mathcal{B}$$

Note that the recourse cost $\mathbb{E}_{\tilde{\xi}}\left(q \cdot y_j(\tilde{\xi})\right)$, $j \in J^*$, may change for some workstations, caused by the possible reassignment of disassembly tasks I^* in the second phase. Thus, the total recourse cost $\mathbb{E}_{\tilde{\xi}}\left(q \cdot \sum_{j\in J} y_j(\tilde{\xi})\right)$ may increase or decrease. No reassignment of hazardous tasks will be considered.

Let x, x' be the optimal solutions of problems (**I**) and (**III**), and $\mathcal{Q}^{\mathrm{D}}(x)$ and $\mathcal{Q}^{\mathrm{D}}(x')$ the corresponding recourse costs, respectively. The solution x' is retained if

$$\eta = \frac{\mathcal{Q}^{\mathrm{D}}(x') - \mathcal{Q}^{\mathrm{D}}(x)}{\mathcal{Q}^{\mathrm{D}}(x)} \times 100 \leqslant \varrho$$

Otherwise, x' is not considered and x is retained (ϱ is a percentage fixed by the decision-maker). In other words, x' is retained if the percentage of the recourse cost's increase in phase 2 does not exceed a certain percentage, i.e. ϱ, fixed by the decision-maker.

The solution approach proposed is detailed in the next section.

4. Solution method

Computing the expected value $\mathbb{E}_{\tilde{\xi}}\left(q \cdot \sum_{j\in J} y_j(\tilde{\xi})\right)$ for a given task assignment to workstations is quite difficult, since it requires numerical integration of implicitly defined probability density functions of variables $y_j(\xi)$, $\xi \in \Xi, \forall j \in J$. Even for the discrete distribution of $\tilde{\xi}$, the number of linear programmes of type (11) to solve may increase exponentially. Although the exact evaluation of the expectation term in (**I**) is possible, its optimisation presents serious difficulties (Santoso et al. 2005). Indeed, $\mathbb{E}_{\tilde{\xi}}\left(q \cdot \sum_{j\in J} y_j(\tilde{\xi})\right)$ is implicitly defined. In order to deal with these difficulties, the so-called Sample Average Approximation (SAA) method is used (Kleywegt, Shapiro, and Homem-De-Mello 2001). It combines Monte Carlo sampling techniques detailed in Section 4.1 with the L-shaped algorithm presented in Section 4.2.1.

4.1 *Monte Carlo sampling*

As introduced earlier, disassembly task times are modelled by a random vector $\tilde{\xi}$ varying over a set $\Xi \subset \mathbb{R}_+^{\mathsf{N}}$ (of a given probability space $(\Xi, \mathcal{F}, P)$ introduced by $\tilde{\xi}$). The integral

$$\mathbb{E}_{\tilde{\xi}}\left(q \cdot \sum_{j\in J} y_j(\tilde{\xi})\right) = \mathbb{E}_{\tilde{\xi}}\left[\mathcal{Q}(x, \tilde{\xi})\right] = \int_{\Xi} \left(q \cdot \sum_{j\in J} y_j(\tilde{\xi})\right) \mathrm{d}P$$

represents the expected value $\mathbb{E}_{\tilde{\xi}}\left[\mathcal{Q}(x, \tilde{\xi})\right]$ of the function

$$\mathcal{Q}(x, \xi) = \min \left\{ q \cdot \sum_{j\in J} y_j(\xi) | \sum_{i\in I} t_i(\xi) \cdot x_{ij} - y_j(\xi) \leqslant C_T, y_j(\xi) \geqslant 0, \forall j \in J \right\}, \xi \in \Xi$$

A Monte Carlo estimation $\mathcal{Q}^{\lambda}(x)$ of the expected value $\mathbb{E}_{\tilde{\xi}}\left[\mathcal{Q}(x, \tilde{\xi})\right]$ is obtained with a random generation of a λ-sample $\left(\xi_1, \xi_2, \ldots, \xi_\lambda\right)$ of the random vector $\tilde{\xi}$, using computer generated pseudo-random numbers. We have then: $\mathcal{Q}^{\lambda}(x) = \frac{1}{\lambda} \sum_{\ell=1}^{\lambda} \mathcal{Q}(x, \xi_\ell)$. Random variable $\mathcal{Q}^{\lambda}(x, \tilde{\xi})$ defined by $\mathcal{Q}^{\lambda}(x, \tilde{\xi}) = \frac{1}{\lambda} \sum_{\ell=1}^{\lambda} \mathcal{Q}(x, \tilde{\xi}_\ell)$ represents the Monte Carlo estimator of $\mathbb{E}_{\tilde{\xi}}\left[\mathcal{Q}(x, \tilde{\xi})\right]$. It is an unbiased estimator for $\mathbb{E}_{\tilde{\xi}}\left[\mathcal{Q}(x, \tilde{\xi})\right]$.

4.2 *Solution method of the first phase*

4.2.1 *L-shaped algorithm*

The main idea of the L-shaped method is to approximate the non-linear term in the objective function of the stochastic programme with fixed recourse (**I**). The fixed recourse is called simple recourse (Kall and Wallace 1994) if the matrix of recourse variable coefficients in programme (11) has the following form: $(\boldsymbol{I}, -\boldsymbol{I})$, where $\boldsymbol{I}$ represents the identity matrix.

The SP-DLBP has a simple recourse. In fact, we have:

$\mathcal{A} = \{a | a = W\mathsf{y} = \boldsymbol{I}y' - \boldsymbol{I}y, y, y' \geqslant 0\} = \mathbb{R}^{|J|}$, i.e. the second stage programme is always feasible for any feasible solution of the first phase and for any realisation scenario ξ of $\tilde{\xi}$;

$\mathcal{Q}(x, \xi) = \min\left\{\mathsf{q}^{\mathsf{T}} y \,|\, \boldsymbol{I}y' - \boldsymbol{I}y = \hbar(\xi) - T(\xi)x, y, y' \geqslant 0\right\}, \forall x \in X, \forall \xi \in \Xi$, defines the matrix formulation of the second-stage programme, where $\mathsf{q} = (q, \ldots, q)^{\mathsf{T}}$, $\hbar(\xi) = (C_T, \ldots, C_T)^{\mathsf{T}}$ and $T(\xi) = (T, \ldots, T)^{\mathsf{T}}, T = (t_1, \ldots, t_{\mathsf{N}}), \forall \xi \in \Xi$, represent, respectively, the recourse vector, the cycle time vector and the technological matrix.

The result above (i.e. SP-DLBP has a simple recourse) can be proved as follows:
$\forall x \in X, \forall \xi \in \Xi$, we have

$$\begin{aligned}\sum_{i\in I} t_i(\xi)\cdot x_{ij} - y_j(\xi) \leqslant C_T &\iff \sum_{i\in I} t_i(\xi)\cdot x_{ij} - y_j(\xi) + y'_j(\xi) = C_T \\ &\iff y'_j(\xi) - y_j(\xi) = C_T - \sum_{i\in I} t_i(\xi)\cdot x_{ij} \\ &\quad\;\; y_j(\xi), y'_j(\xi) \geqslant 0, \forall j \in J\end{aligned}$$

Since $y_j(\xi), y'_j(\xi) \geqslant 0, \forall j \in J, \forall \xi \in \Xi$, we have then, $\forall j \in J, \forall \xi \in \Xi$, $y'_j(\xi) - y_j(\xi) \in \mathbb{R}$. It follows from the above that $\mathcal{A} = \{a | a = W\mathsf{y} = \boldsymbol{I} y' - \boldsymbol{I} y, y, y' \geqslant 0\} = \mathbb{R}^{|J|}$, i.e. $\forall \xi \in \Xi, \forall x \in X$, the second stage programme

$$\mathcal{Q}(x,\xi) = \min\Big\{q\cdot\sum_{j\in J} y_j(\xi) | \sum_{i\in I} t_i(\xi)\cdot x_{ij} - y_j(\xi) \leqslant C_T, y_j(\xi) \geqslant 0, \forall j \in J\Big\}$$

is feasible.

Since the problem addressed in the L-shaped algorithm is a minimisation one and since $\max f(x) = -\min\big(-f(x)\big)$ where $f(x)$ is a function of decision variables x, in order to solve problem (I) its equivalent minimisation version

$$\min\left\{C_T\Big(F_c\cdot\sum_{j\in J} j\cdot x_{\mathsf{s}j} + C_h\cdot\sum_{j\in J} h_j\Big) - \sum_{i\in I}\sum_{j\in J}\sum_{l\in L_i} r_l\cdot x_{ij} + \mathbb{E}_{\tilde{\xi}}\Big(q\cdot\sum_{j\in J} y_j(\tilde{\xi})\Big)\right\} \tag{15}$$

is used. The optimal value of problem (I) is then the symmetric value of (15). Consider the minimisation version of programme (I) and the matrix formulation of problem (II) given below, where $c^{\mathsf{T}}x = C_T\Big(F_c\cdot\sum_{j\in J} j\cdot x_{\mathsf{s}j} + C_h\cdot\sum_{j\in J} h_j\Big) - \sum_{i\in I}\sum_{j\in J}\sum_{l\in L_i} r_l\cdot x_{ij}$ and $\hbar_\ell = (C_T,\ldots,C_T)^{\mathsf{T}}, \forall \ell \in \mathfrak{D}$.

$$\begin{aligned}&\min\ \Big\{c^{\mathsf{T}}x + \mathcal{Q}^{\mathrm{D}}(x), x \in X\Big\}\\ &\text{where } \mathcal{Q}^{\mathrm{D}}(x) = \sum_{\ell=1}^{\mathrm{D}} p_\ell\cdot\mathcal{Q}(x,\xi_\ell) \text{ and}\\ &\mathcal{Q}(x,\xi_\ell) = \min\big\{\mathsf{q}^{\mathsf{T}}\mathsf{y} \,|\, W\mathsf{y} = \hbar_\ell - T(\xi_\ell)x, \mathsf{y} \geqslant 0\big\}\end{aligned}$$

In the L-shaped algorithm given below, counters h and υ are used, respectively, for the optimality cuts and for the algorithm iterations; E_ν and e_ν are defined in the algorithm. In the objective function of the master problem, a new variable φ is introduced. It verifies the inequality $\varphi \geqslant \mathcal{Q}^{\mathrm{D}}(x)$. Since $\mathcal{Q}^{\mathrm{D}}(x)$ is implicitly defined by a relatively large number of optimisation problems, the master programme is not directly solved with this inequality. The L-shaped algorithm processes as follows.

L-shaped algorithm

***Step* 0**. Set $\mathsf{h} = \upsilon = 0$

***Step* 1**. Set $\upsilon = \upsilon + 1$. Solve the master programme:

$$\begin{aligned}&\min\ \Big\{c^{\mathsf{T}}x + \varphi\Big\}\\ &\text{s.t.}\\ &\quad x \in X\\ &\quad E_\nu\cdot x + \varphi \geqslant e_\nu, \nu = 1,2,\ldots,\mathsf{h}\\ &\quad x \text{ binary}, \varphi \geqslant 0\end{aligned} \tag{16}$$

Let $(x^\upsilon, \varphi^\upsilon)$ be an optimal solution.

***Step* 2**. For $\ell = 1,2,\ldots,\mathrm{D}$, solve the subprogramme:

$$\begin{aligned}&\min \mathcal{W} = \mathsf{q}^{\mathsf{T}}\cdot\mathsf{y}\\ &\text{s.t.}\\ &\quad W\mathsf{y} = \hbar_\ell - T(\xi_\ell)\cdot x^\upsilon\\ &\quad \mathsf{y} \geqslant 0\end{aligned}$$

Let ω_ℓ^υ be the simplex multipliers associated with an optimal solution of problem ℓ above; define

$$E_{\mathsf{h}+1} = \sum_{\ell \in \mathfrak{D}} p_\ell \cdot (\omega_\ell^\upsilon)^\mathsf{T} \cdot T_\ell$$

and

$$e_{\mathsf{h}+1} = \sum_{\ell \in \mathfrak{D}} p_\ell \cdot (\omega_\ell^\upsilon)^\mathsf{T} \cdot \hbar_\ell$$

Let $\theta^\upsilon = e_{\mathsf{h}+1} - E_{\mathsf{h}+1} \cdot x^\upsilon$. If $\varphi^\upsilon \geqslant \theta^\upsilon$, stop: x^υ is an optimal solution.

Else, generate an optimality cut of type (16), set $\mathsf{h} = \mathsf{h} + 1$, add constraint of type (16) and return to ***Step* 1**.

4.2.2 *Sample average approximation procedure*

The SAA strategy deals with an approximate computation of the expected recourse cost. It is used to compute lower and upper bounds of the objective function for problem (**I**). Therefore, the optimality gap and statistical confidence intervals on the quality of the approximate solutions are evaluated.

The SAA strategy is implemented as follows: for a random sample of $\tilde{\xi}$, generated using MCS and having size Λ, the term $\mathbb{E}_{\tilde{\xi}}\big[\mathcal{Q}(x, \tilde{\xi})\big]$ is approximated by the sample average function $\mathcal{Q}^\Lambda(x) = \frac{1}{\Lambda} \cdot \sum_{\ell=1}^{\Lambda} \mathcal{Q}(x, \xi_\ell)$. Thus, problem (**I**) is approximated by problem (**II**) where $p_\ell = \frac{1}{\Lambda}$, $\ell = 1, 2, \ldots, \Lambda$. Then, problem (**II**) is solved with the L-shaped algorithm.

Let $\gamma_\Lambda,\ x_\Lambda$ and $\gamma^*,\ x^*$ be the optimal objective values and optimal solutions of problems (**II**) and (**I**), respectively. We have then,

$$\gamma^* = \min_{x \in X} \left\{ f(x) = c^\mathsf{T} x + \mathbb{E}_{\tilde{\xi}}\big[\mathcal{Q}(x, \tilde{\xi})\big] \right\}$$

and

$$\gamma_\Lambda = \min_{x \in X} \left\{ f_\Lambda(x) = c^\mathsf{T} x + \frac{1}{\Lambda} \cdot \sum_{\ell=1}^{\Lambda} \mathcal{Q}(x, \xi_\ell) \right\}$$

Note that

$$\mathbb{E}\big(f_\Lambda(x)\big) = c^\mathsf{T} x + \frac{1}{\Lambda} \cdot \mathbb{E}\left(\sum_{\ell=1}^{\Lambda} \mathcal{Q}(x, \xi_\ell) \right) = f(x)$$

The strong law of large numbers implies that $\gamma_\Lambda \longrightarrow_{\Lambda \to \infty} \gamma^*$ with probability 1. Moreover, in Kleywegt, Shapiro, and Homem-De-Mello (2001) it is shown that, under certain weak conditions, as Λ increases, x_Λ converges to x^* with probability approaching 1 exponentially fast. In order to get a good approximate solution to problem (**I**), the sample size Λ is determined as

$$\Lambda \geqslant \frac{3\sigma_{\max}^2}{\varepsilon^2} \log\left(\frac{|\mathfrak{X}|}{\alpha}\right) \tag{17}$$

where $\varepsilon > 0$ and $\alpha \in]0, 1[$, $\sigma_{\max}^2$ represents the maximum variance of function differences on $\mathcal{Q}^\Lambda(x)$ (Kleywegt, Shapiro, and Homem-De-Mello 2001); $\mathfrak{X}$, as introduced in constraint (9) of programme (**I**), defines the set of the possible values of decision variables x.

This sample size is sufficient to get an ε-solution $\widehat{x}_\Lambda$ of problem (**I**) with a probability at least equal to $(1 - \alpha)$, i.e. a solution with an absolute optimality gap ε. Even if it may be too conservative in practice, estimate (17) shows that the required sample size Λ is linear in the number of disassembly tasks and workstations: $|\mathfrak{X}| \leqslant 2^{|J| \cdot (|I|+2)} = 2^{\mathsf{M} \times (\mathsf{N}+2)}$ and $|\mathfrak{X}| \leqslant 2^{\mathsf{M} \times (\mathsf{N}+2)} \Rightarrow \log |\mathfrak{X}| \leqslant (\log 2) \cdot \mathsf{M} \times (\mathsf{N} + 2)$.

Practically, the SAA strategy consists of generating Ω independent random samples $(\xi_1^n, \xi_2^n, \ldots, \xi_\Lambda^n)$, $n = 1, 2, \ldots, \Omega$, of a modest sample size Λ. Then, estimates of lower and upper bounds and the optimality gap depending on Ω and Λ are computed. In this study, Ω and Λ are chosen so as to obtain approximate solutions of a reasonable quality.

4.2.3 *SAA procedure*

Step 1. Lower Bound Estimation

Generate Ω independent random samples $(\xi_1^n, \xi_2^n, \ldots, \xi_\Lambda^n), n = 1, 2, \ldots, \Omega$ of size Λ each. Solve the corresponding problem (**II**) with the L-shaped algorithm, compute an optimal solution x_Λ^n and the corresponding objective value γ_Λ^n, for each value of n. Compute a lower bound: $LB_{\Lambda\Omega} = \frac{1}{\Omega} \cdot \sum_{n=1}^{\Omega} \gamma_\Lambda^n$.

Let X^* be a set of the optimal solutions of problem (**I**). Then,

$$\gamma_\Lambda \leqslant \min_{x \in X^*} f_\Lambda(x) \text{ and } \mathbb{E}(\gamma_\Lambda) \leqslant \mathbb{E}\big(\min_{x \in X^*} f_\Lambda(x)\big) \leqslant \min_{x \in X^*} \mathbb{E}\big(f_\Lambda(x)\big) = \gamma^*$$

Since $\mathbb{E}\big(LB_{\Lambda\Omega}\big) = \frac{1}{\Omega} \cdot \mathbb{E}\left(\sum_{n=1}^{\Omega} \gamma_\Lambda^n\right) = \mathbb{E}(\gamma_\Lambda)$, we have then, $LB_{\Lambda\Omega} \leqslant \gamma^*$. The variance $\sigma^2_{LB_{\Lambda\Omega}}$ of $LB_{\Lambda\Omega}$ is estimated as follows:

$$\sigma^2_{LB_{\Lambda\Omega}} = \frac{1}{\Omega(\Omega - 1)} \cdot \sum_{n=1}^{\Omega} (\gamma_\Lambda^n - LB_{\Lambda\Omega})^2$$

By the application of CLT, we have: $LB_{\Lambda\Omega} \rightsquigarrow \mathcal{N}\left(\mathbb{E}(\gamma_\Lambda), \frac{\sigma_{LB}}{\sqrt{\Omega}}\right), \sigma_{LB} = \sqrt{\text{Var}(\gamma_\Lambda)}$. A confidence interval of the level $(1 - \alpha)$ for $\mathbb{E}(\gamma_\Lambda)$ is then given by:

$$\left[LB_{\Lambda\Omega} - \frac{z_{\alpha/2} \cdot \sigma_{LB_{\Lambda\Omega}}}{\sqrt{\Omega}}, LB_{\Lambda\Omega} + \frac{z_{\alpha/2} \cdot \sigma_{LB_{\Lambda\Omega}}}{\sqrt{\Omega}}\right]$$

where z_α is the quantile of the standard normal distribution $\mathcal{N}(0, 1)$ verifying $P\Big(\mathcal{N}(0, 1) \leqslant z_\alpha\Big) = 1 - \alpha$.

Step 2. Upper Bound Estimation

Generate a sample $(\xi_1, \xi_2, \ldots, \xi_\lambda)$ of size λ independent from the Ω generated samples in *Step 1*. Let x_Λ^{n} be a feasible solution obtained in *Step 1*; x_Λ^{n} should be the one for which $f_\lambda(x)$ is minimum. Since x_Λ^{n} is a feasible solution of problem (**I**), then $f(x_\Lambda^{\mathsf{n}}) \geqslant \gamma^*$. Compute the following upper bound: $UB_\lambda = f_\lambda(x_\Lambda^{\mathsf{n}})$.

UB_λ is an unbiased estimator of $f(x_\Lambda^{\mathsf{n}})$: $\mathbb{E}\Big(f_\lambda(x_\Lambda^{\mathsf{n}})\Big) = f(x_\Lambda^{\mathsf{n}})$, we have then $UB_\lambda \geqslant \gamma^*$. The variance $\sigma^2_{UB_\lambda}$ of UB_λ can be estimated with:

$$\sigma^2_{UB_\lambda} = \frac{1}{\lambda(\lambda - 1)} \cdot \sum_{\ell=1}^{\lambda} (c^\mathsf{T} x_\Lambda^{\mathsf{n}} + \mathcal{Q}(x_\Lambda^{\mathsf{n}}, \xi_\ell) - UB_\lambda)^2$$

By the application of the CLT, we have: $UB_\lambda \rightsquigarrow \mathcal{N}\left(f(x_\Lambda^{\mathsf{n}}), \frac{\sigma_{UB}}{\sqrt{\lambda}}\right), \sigma_{UB} = \sqrt{\text{Var}\big(f_\lambda(x_\Lambda^{\mathsf{n}})\big)}$.

A confidence interval of the level $(1 - \alpha)$ of $f(x_\Lambda^{\mathsf{n}})$ is then given by:

$$\left[UB_\lambda - \frac{z_{\alpha/2} \cdot \sigma_{UB_\lambda}}{\sqrt{\lambda}}, UB_\lambda + \frac{z_{\alpha/2} \cdot \sigma_{UB_\lambda}}{\sqrt{\lambda}}\right], \ P\Big(\mathcal{N}(0, 1) \leqslant z_\alpha\Big) = 1 - \alpha$$

Step 3. Optimality Gap Estimation

Here, the optimality gap is the difference between upper and lower bound values:

$OG_{\lambda\Omega\Lambda} = UB_\lambda - LB_{\Omega\Lambda}$. Note that $\mathbb{E}\big(OG_{\lambda\Omega\Lambda}\big) = f(x_\Lambda^{\mathsf{n}}) - \mathbb{E}(\gamma_\Lambda) \geqslant f(x_\Lambda^{\mathsf{n}}) - \gamma^*$. It follows that $OG_{\lambda\Omega\Lambda}$ is a biased estimator of the optimality gap, it overestimates $\big(f(x_\Lambda^{\mathsf{n}}) - \gamma^*\big)$. Its bias $\big(\gamma^* - \mathbb{E}(\gamma_\Lambda)\big)$ is monotonically decreasing in Λ (Mak, Morton, and Wood 1999; Norkin, Ruszczynski, and Pflug 1998). The variance $\sigma^2_{OG_{\lambda\Omega\Lambda}}$ of $OG_{\lambda\Omega\Lambda}$ can be estimated with:

$$\sigma^2_{OG_{\lambda\Omega\Lambda}} = \sigma^2_{LB_{\Lambda\Omega}} + \sigma^2_{UB_\lambda}$$

The flowchart in Figure 6 summarises the steps of the SAA method. The random samples are generated using MCS.

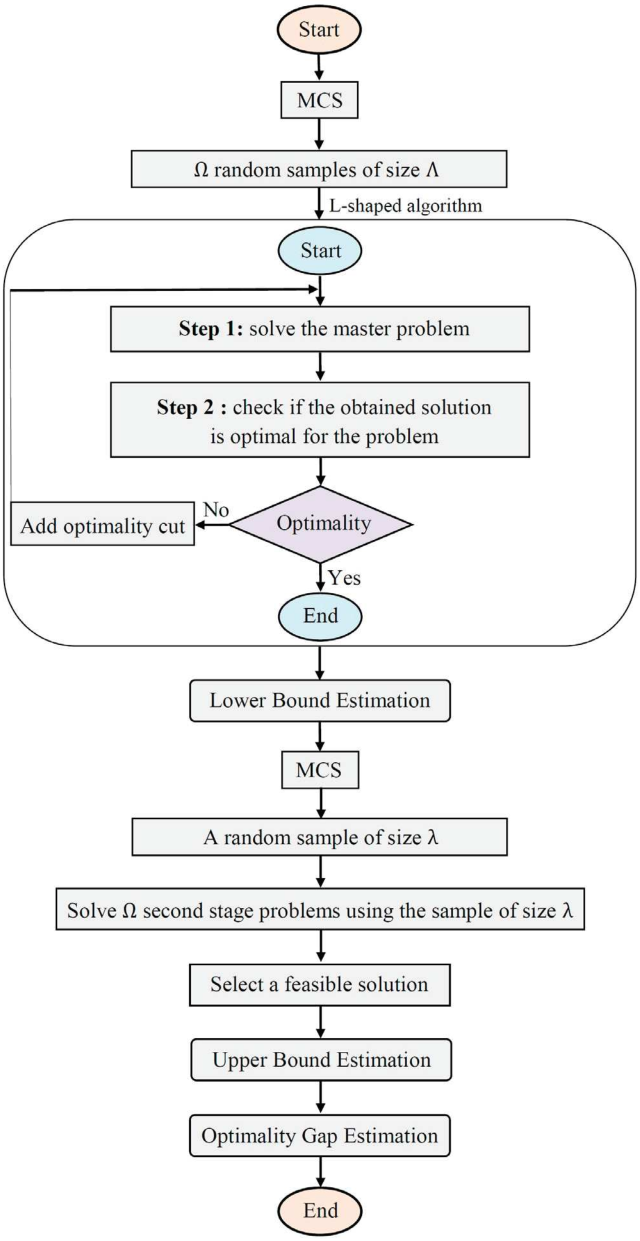

Figure 6. Flowchart of the SAA procedure.

4.3 *Solution method of the second phase*

In this subsection, a lower and upper bounding schemes are defined in order to approximate problem (**IV**).

4.3.1 *Upper bound of programme* (**IV**)

In the same way as in Section 4.1, let $S_l(\tilde{\xi}) = |\mathrm{S}_l(\tilde{\xi})|$ and consider the random variable $S_l^{\lambda}(\tilde{\xi}) = \frac{1}{\lambda}\sum_{\ell=1}^{\lambda} S_l(\tilde{\xi}_\ell)$, $l \in \mathcal{B}$. Then, $S_l^{\lambda}(\tilde{\xi}), l \in \mathcal{B}$, represents an unbiased estimator of $\mathbb{E}_{\tilde{\xi}}\left(S_l(\tilde{\xi})\right), l \in \mathcal{B}$:

$$\mathbb{E}_{\tilde{\xi}}\left[S_l^{\lambda}(\tilde{\xi})\right] = \mathbb{E}\left(\frac{1}{\lambda}\sum_{\ell=1}^{\lambda} S_l(\tilde{\xi}_\ell)\right) = \frac{1}{\lambda}\sum_{\ell=1}^{\lambda}\mathbb{E}\left(S_l(\tilde{\xi}_\ell)\right) = \mathbb{E}_{\tilde{\xi}}\left[S_l(\tilde{\xi})\right]$$

Let $z_{\ell l} \geqslant S_l(\xi_\ell) = |\mathrm{S}_l(\xi_\ell)|, \ell = 1, 2, \ldots, \lambda, \forall l \in \mathcal{B}$. Using a λ-sample $(\xi_1, \xi_2, \ldots, \xi_\lambda)$ of the random vector $\tilde{\xi}$, the optimal value of linear programme (**V**) below is an approximation of the optimal value of programme (**IV**).

$$\begin{aligned} &\min Y \\ &\text{s.t. } x \in X \\ &\quad Y \geqslant \frac{1}{\lambda} \cdot \sum_{\ell=1}^{\ell=\lambda} z_{\ell l}, \forall l \in \mathcal{B} \\ &\quad -z_{\ell l} \leqslant \mathrm{S}_l(\xi_\ell) \leqslant z_{\ell l}, \forall l \in \mathcal{B}, \ell = 1, 2, \ldots, \lambda \qquad (18) \\ &\quad Y \geqslant 0, z_{\ell l} \geqslant 0, \ell = 1, 2, \ldots, \lambda, \forall l \in \mathcal{B} \end{aligned} \tag{V}$$

Recall that, in phase 2, $X = \{x|$ constraints (12)–(14) are satisfied$\}$. Constraint set (18) can be replaced with:

$$\begin{aligned} &u_{\ell l} + \mathrm{S}_l(\xi_\ell) = z_{\ell l}, \forall l \in \mathcal{B}, \ell = 1, 2, \ldots, \lambda \\ &0 \leqslant u_{\ell l} \leqslant 2z_{\ell l}, \forall l \in \mathcal{B}, \ell = 1, 2, \ldots, \lambda \end{aligned}$$

Proposition 1 *The optimal value of problem* (**V**) *is an upper bound of the optimal value of problem* (**IV**).

Proof. Since $z_{\ell l} \geqslant S_l(\xi_\ell), \ell = 1, 2, \ldots, \lambda, \forall l \in \mathcal{B}$, then

$$\sum_{\ell=1}^{\ell=\lambda} z_{\ell l} \geqslant \sum_{\ell=1}^{\ell=\lambda} S_l(\xi_\ell), \forall l \in \mathcal{B}$$

and

$$Y \geqslant \frac{1}{\lambda} \cdot \sum_{\ell=1}^{\ell=\lambda} z_{\ell l} \geqslant \frac{1}{\lambda} \cdot \sum_{\ell=1}^{\ell=\lambda} S_l(\xi_\ell), \forall l \in \mathcal{B}$$

□

4.3.2 *Lower bound of programme* (**IV**)

The optimal value of programme (**VI**) given below defines an approximation of the optimal value of programme (**IV**).

$$\begin{aligned} &\min \max_{l \in \mathcal{B}} \left| \mathbb{E}_{\tilde{\xi}}\left(\mathrm{S}_l(\tilde{\xi})\right) \right| \\ &\text{s.t. } x \in X \end{aligned} \tag{VI}$$

Proposition 2 *The optimal value of problem* (**VI**) *is a lower bound of the optimal value of problem* (**IV**).

Proof. Obvious since $\left|\mathbb{E}_{\tilde{\xi}}\left(\mathrm{S}_l(\tilde{\xi})\right)\right| \leqslant \mathbb{E}_{\tilde{\xi}}\left(\left|\mathrm{S}_l(\tilde{\xi})\right|\right), l \in \mathcal{B}$ □

The programme (**VI'**) below represents an equivalent version of programme (**VI**).

$$\begin{aligned} &\min Y \\ &\text{s.t. } x \in X \\ &\quad -Y \leqslant \mathbb{E}_{\tilde{\xi}}\left(\mathrm{S}_l(\tilde{\xi})\right) \leqslant Y, \forall l \in \mathcal{B} \\ &\quad Y \geqslant 0 \end{aligned} \tag{VI'}$$

Using a λ-sample $(\xi_1, \xi_2, \ldots, \xi_\lambda)$ of the random vector $\tilde{\xi}$, the expectation $\mathbb{E}_{\tilde{\xi}}\left(\mathrm{S}_l(\tilde{\xi})\right)$ can be approximated with $\frac{1}{\lambda}\sum_{\ell=1}^{\lambda} \mathrm{S}_l(\xi_\ell), l \in \mathcal{B}$.

5. Numerical results

The first phase, i.e. profit maximisation with the SAA method, and the second one, i.e. idle time levelling with programmes (**V**) and (**VI'**), were implemented with C++ (under Linux) and ILOG CPLEX 12.4 was used to solve the different models on a PC with a Pentium(R) Dual-Core CPU T4500, 2.30 GHz and 3GB RAM. These two sequential phases have been applied to the problem instance illustrated in Figure 3 and to six instances available in the literature. These six instances are used as benchmarking problems and contain process alternatives for disassembly. The names of the problem instances were, respectively, composed of the first letters of authors' names and the year of publication, i.e. BBD13 (Bentaha, Battaïa, and Dolgui 2014a), BBD13b (Bentaha, Battaïa, and Dolgui 2014c), KSE09 (Koc, Sabuncuoglu, and Erel 2009), L99a and L99b from (Lambert 1999) and MJKL11 from (Ma et al. 2011). BBD13a corresponds to the rigid caster product instance (see Figures 2 and 3). The input data for each problem instance are given in Table 2. The columns 'AND-relations' report the number of disassembly tasks with no successor in subcolumn 0, with one AND-type arc in subcolumn 1 and with two AND-type arcs in subcolumn 2. The column 'arcs' gives the total number of AND-type and OR-type arcs.

Table 2. Instances from the literature used as benchmark problems in the numerical experiments.

	N	K	L	Arcs	AND-relations			M	C_T
					0	1	2		
MJKL11	37	22	33	76	4	27	6	10	35
BBD13a	32	14	23	60	4	28	0	4	0.80
L99a	30	18	28	60	2	26	2	9	30
BBD13b	25	11	27	49	4	18	3	4	91
KSE09	23	13	20	47	4	14	5	6	20
L99b	20	13	23	41	5	9	6	9	5.5
BBD13	10	5	12	18	3	6	1	3	0.51

Table 3 reports the optimisation results of phase 1 obtained for the processed instances using Monte Carlo sampling for different values of Λ and λ. The number of generated samples Ω was 10, the recourse cost q was fixed at 5, operating workstation cost F_c at 3, additional cost for stations handling hazardous parts $C_h = 2$, the confidence intervals level was 95%, and 25% of tasks were taken as hazardous. The remaining parameters $r_l, l \in L$ and $\tilde{t}_i, i \in I$ were randomly generated:

• $r_l \in [0, r_{\max}]$, where r_l is the revenue of part $l, l \in L$ and $r_{\max}$ is its maximum value; $r_l, l \in L$ is generated uniformly in the interval $[0, r_{\max}]$. The value of $r_{\max}$ depends on the instance used. As an example, the $r_{\max}$ value of BBD13 is 12.

• $\tilde{t}_i$ is assumed to be a random variable with a normal distribution of known mean μ_i and standard deviation σ_i, where $\sigma_i = \mu_i/5$; and $\tilde{t}_i$ is the processing time of the disassembly task $i, i \in I$.

Although using different probability distributions for task times is possible, they were assumed, without loss of generality, to be normally distributed. Columns 'LB' and 'UB' report the lower and upper bound values and the corresponding confidence intervals, respectively. Column 'OG' reports the optimality gap $OG_{\lambda\Omega\Lambda}$ and the associated standard deviation $\sigma_{OG_{\lambda\Omega\Lambda}}$. Finally, columns '$o$-tasks', '$s$-tasks', '$n$-stations', '$h$-stations', 'CPU time' report the original number of tasks of the selected disassembly alternative, the number of selected tasks of the selected alternative, the optimal number of workstations, the number of hazardous workstations with the corresponding rank for each station, and the resolution time in seconds, respectively.

The numerical results in Table 3 show that a higher chosen value of Λ resulted in a higher quality solution. In particular, solutions of instances MJKL11 and L99b represent good quality solutions since their optimality gap values and the corresponding standard deviations are significantly reduced. The resolution time, however, illustrated in Figure 7 increases considerably as the value of Λ increases. This depends on the complexity of the instance (e.g. the number of tasks with 2 AND-type arcs) but is due mainly to the escalating number of subproblems to solve for each master problem. Figure 8 illustrates the convergence of $LB_{\Lambda\Omega}$ and UB_{λ} values for the instance L99b.

In order to show the ability of the proposed approach to handle different probability distributions for the task times, instances BBD12 and L99a are solved for three different cases as illustrated in Table 4. In the first case, each task time $\tilde{t}_i, i \in I$ is assumed to follow a triangular distribution $\mathcal{T}(a_i, b_i, c_i), i \in I$ where $b_i = \mu_i, \forall i \in I$; $a_i, c_i, i \in I$ are chosen randomly. In the second one, each task time is assumed to follow a continuous uniform distribution $\mathcal{U}(a_i, c_i), i \in I$ where a_i, c_i have the same values as for triangular distributions. In the third case, a mix of normal, triangular and uniform distributions is considered. Approximatively, 40% of task times are assumed to follow normal distributions, 40% triangular distributions and 20% uniform distributions.

All the returned solutions were the same as the solutions obtained in Table 3 for instances BBD12 and L99a; CPU times are of the same order of magnitude. The only difference is the objective values of the returned solutions (which are slightly greater) and their associated confidence intervals. These values depend on the used probability distributions of the task processing times.

Table 5 reports the optimisation results obtained for phase 2 using Monte Carlo sampling for different values of λ. The values of the remaining parameters were the same as in phase 1. Columns 'Upper bound' and 'Lower bound' report the optimisation results of programmes (**V**) and (**VI'**), respectively. Sub-column 'Reassignment' indicates the value 0 if the

Table 3. Obtained results: profit maximisation using normal probability distributions for the task times.

	Λ, λ	LB	UB	OG	o-tasks	s-tasks	n-stations	h-stations	CPU time
	100, 200	34.214 ± 0.192	34.032 ± 0.384	−0.182, 2.785	7	5	2	(1 : 1)	101
	300, 600	33.753 ± 0.088	33.865 ± 0.127	0.112, 1.591	7	5	2	(1 : 1)	738
MJKL11	600, 900	33.942 ± 0.096	33.826 ± 0.085	−0.116, 1.307	7	5	2	(1 : 1)	2836
	800, 1200	33.921 ± 0.093	33.782 ± 0.064	−0.139, 1.139	7	5	2	(1 : 1)	4506
	1000, 1500	34.027 ± 0.034	34.063 ± 0.051	0.036, 1.006	7	5	2	(1 : 1)	7122
	100, 200	180.400 ± 0.010	180.420 ± 1.778	0.020, 12.831	4	2	1	(1 : 1)	44
	300, 600	180.425 ± 0.008	180.446 ± 0.592	0.021, 7.396	4	2	1	(1 : 1)	333
BBD13a	600, 900	180.428 ± 0.003	180.424 ± 0.394	−0.004, 6.037	4	2	1	(1 : 1)	1103
	800, 1200	180.434 ± 0.003	180.423 ± 0.296	−0.011, 5.227	4	2	1	(1 : 1)	1973
	1000, 1500	180.425 ± 0.002	180.439 ± 0.236	0.014, 4.675	4	2	1	(1 : 1)	3057
	100, 200	499.168 ± 0.402	499.614 ± 5.191	0.446, 37.461	9	7	3	–	133
	300, 600	499.628 ± 0.184	501.170 ± 1.727	1.542, 21.591	9	7	3	–	1104
L99a	600, 900	500.096 ± 0.148	500.376 ± 1.151	0.280, 17.624	9	7	3	–	3852
	800, 1200	500.176 ± 0.146	499.462 ± 0.863	−0.714, 15.262	9	7	3	–	7791
	1000, 1500	500.136 ± 0.141	500.403 ± 0.691	0.267, 13.649	9	7	3	–	11309
	100, 200	208.532 ± 1.056	209.080 ± 2.779	0.548, 20.125	4	3	2	–	65
	300, 600	204.841 ± 0.694	206.941 ± 0.924	2.100, 11.599	4	3	2	–	549
BBD13b	600, 900	207.779 ± 0.476	205.562 ± 0.618	−2.217, 9.491	4	3	2	–	1863
	800, 1200	207.859 ± 0.463	204.477 ± 0.462	−3.382, 8.209	4	3	2	–	3069
	1000, 1500	207.089 ± 0.268	207.443 ± 0.369	0.354, 7.317	4	3	2	–	4826
	100, 200	896.117 ± 0.142	895.766 ± 8.842	−0.351, 63.802	6	4	2	(1 : 2)	53
	300, 600	895.988 ± 0.068	895.582 ± 2.942	−0.406, 36.774	6	4	2	(1 : 2)	457
KSE09	600, 900	895.692 ± 0.053	895.901 ± 1.961	0.209, 30.018	6	4	2	(1 : 2)	1393
	800, 1200	895.806 ± 0.050	895.910 ± 1.471	0.104, 25.992	6	4	2	(1 : 2)	2415
	1000, 1500	895.680 ± 0.053	895.823 ± 1.176	0.143, 23.246	6	4	2	(1 : 2)	3845
	100, 200	75.261 ± 0.006	75.055 ± 0.742	−0.206, 5.353	8	6	3	–	235
	300, 600	75.204 ± 0.008	75.197 ± 0.247	−0.007, 3.085	8	6	3	–	2422
L99b	600, 900	75.206 ± 0.003	75.246 ± 0.165	0.040, 2.518	8	6	3	–	8731
	800, 1200	75.232 ± 0.005	75.225 ± 0.123	−0.007, 2.180	8	6	3	–	14913
	1000, 1500	75.216 ± 0.007	75.223 ± 0.099	0.007, 1.950	8	6	3	–	24439
	100, 200	92.917 ± 0.006	92.895 ± 0.923	−0.022, 6.658	3	3	2	(1 : 2)	40
	300, 600	92.914 ± 0.004	92.894 ± 0.307	−0.020, 3.837	3	3	2	(1 : 2)	328
BBD12	600, 900	92.910 ± 0.004	92.916 ± 0.205	0.006, 3.132	3	3	2	(1 : 2)	1103
	800, 1200	92.913 ± 0.003	92.906 ± 0.153	−0.007, 2.712	3	3	2	(1 : 2)	1981
	1000, 1500	92.912 ± 0.002	92.932 ± 0.123	0.020, 2.426	3	3	2	(1 : 2)	3205

solution of phase 2 remains the same as in phase 1 and indicates the value 1 if this solution is different. The last column gives the optimality gap defined by $Gap = \frac{UB-LB}{LB} \cdot 100$.

The results in Table 5 show that the proposed lower and upper bounding schemes of the line balancing phase returned solutions of good quality, except for instance BBD13b where the solution is of an average quality. For each value of λ, the returned solution was the same for both the upper and lower bounds. CPU time of the lower bound is better due to the reduced number of constraints and variables of programme (V') compared to programme (V). Except for instance L99b, the solution of phase 1 stays unchanged in phase 2. This can be explained by the fact that each unchanged solution is represented by a simple path (see Figure 5) for its precedence graph. In such a case, no reassignment is possible. All instances were solved in a few seconds due to the elimination of hard cycle time constraints. Cycle time constraints are not needed in phase 2 since the number of workstations is fixed. The selected alternative and the corresponding selected tasks for instance L99b are illustrated in Figure 9. Note that this selected alternative is represented by a tree.

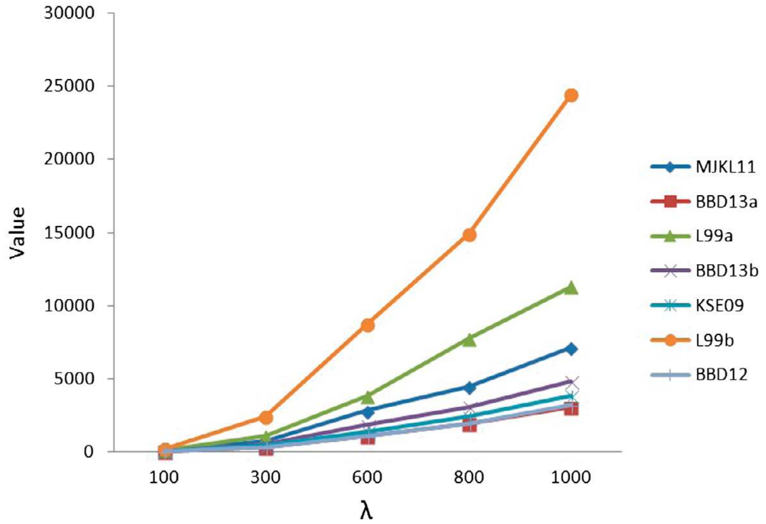

Figure 7. CPU time of phase 1 (in seconds).

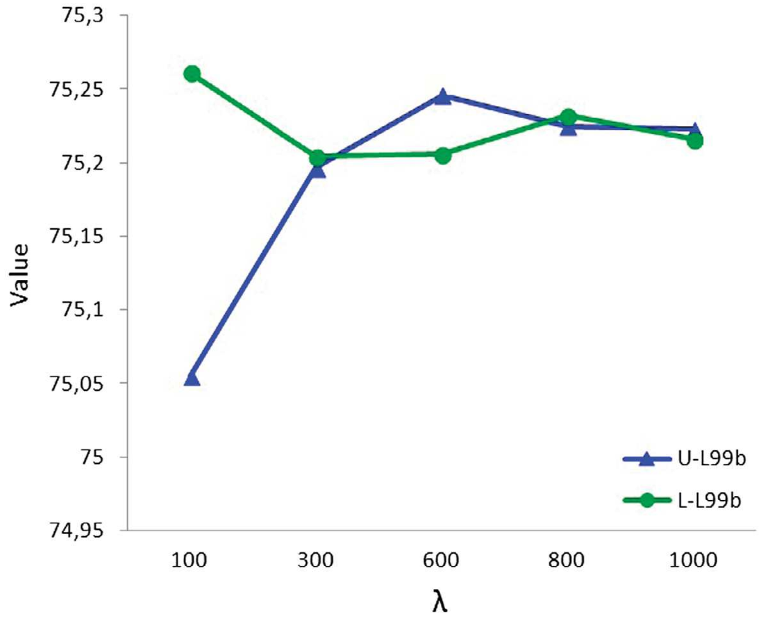

Figure 8. UB and LB convergence for instance L99b.

Table 6 establishes a comparison of the solution quality (in terms of line profit) for different values of λ between the solution of instance L99b in phase 1, i.e. x, and its solution in phase 2, i.e. x'. The column 'Estimated recourse cost' reports the recourse cost of workstations 1, 2, 3 in subcolumns 'R_{W1}', 'R_{W2}', 'R_{W3}', respectively and 'R_{total}' represents the total recourse cost. Variable η and parameter ϱ, introduced in Section 3.3, indicate the percentage recourse cost increases in phase 2 and the percentage the recourse cost is allowed to increase, respectively.

The comparison between x and x' is illustrated in Figure 10, whose objective is to prove the importance and the added value of phase 2. As shown in Table 6, solution x' of phase 2 is notably better than solution x of phase 1. In fact, the three workstations are better balanced with x' than with x, and the recourse cost in the second phase is decreased by about 85% when compared to the recourse cost from the first phase. Equivalently, the recourse cost of phase 2 represents only 15% of the recourse cost of phase 1. As a consequence, $\eta \ll \varrho$ and x' is to be retained as the best and final solution. CPU time increases

Table 4. Obtained results: profit maximisation using different probability distributions for the task times.

	Λ, λ	LB	UB	OG	o-tasks	s-tasks	n-stations	h-stations	CPU time	
	100,200	507.069 ± 0.136	508.520 ± 5.188	1.451,37.436	9	7	3	–	134	
	300,600	507.086 ± 0.113	507.597 ± 1.726	0.511,21.578	9	7	3	–	1088	
L99a	600,900	506.797 ± 0.108	506.482 ± 1.151	-0.315,17.614	9	7	3	–	3874	
	800,1200	506.924 ± 0.089	507.072 ± 0.863	0.148,15.252	9	7	3	–	7704	
	1000,1500	506.912 ± 0.066	506.772 ± 0.690	-0.140,13.640	9	7	3	–	11370	Triangular distribution
	100,200	93.251 ± 0.005	93.278 ± 0.923	0.027,6.658	3	3	2	(1:2)	42	
	300,600	93.276 ± 0.003	93.274 ± 0.307	-0.002,3.837	3	3	2	(1:2)	354	
BBD12	600,900	93.269 ± 0.002	93.278 ± 0.205	0.009,3.132	3	3	2	(1:2)	1141	
	800,1200	93.271 ± 0.002	93.261 ± 0.153	-0.010,2.712	3	3	2	(1:2)	1853	
	1000,1500	93.264 ± 0.001	93.266 ± 0.123	0.002,2.426	3	3	2	(1:2)	3287	
	100,200	504.650 ± 0.370	506.233 ± 5.189	1.583,37.448	9	7	3	–	132	
	300,600	504.987 ± 0.156	505.917 ± 1.727	0.930,21.583	9	7	3	–	1070	
L99a	600,900	505.207 ± 0.120	505.489 ± 1.151	0.282,17.617	9	7	3	–	3911	
	800,1200	505.230 ± 0.114	505.063 ± 0.863	-0.167,15.255	9	7	3	–	7650	
	1000,1500	504.737 ± 0.104	505.224 ± 0.690	0.487,13.644	9	7	3	–	11251	Uniform distribution
	100,200	93.431 ± 0.008	93.447 ± 0.923	0.016,6.658	3	3	2	(1:2)	39	
	300,600	93.443 ± 0.005	93.443 ± 0.307	0.000,3.837	3	3	2	(1:2)	343	
BBD12	600,900	93.434 ± 0.002	93.446 ± 0.205	0.012,3.132	3	3	2	(1:2)	1186	
	800,1200	93.431 ± 0.002	93.435 ± 0.153	0.004,2.712	3	3	2	(1:2)	1897	
	1000,1500	93.438 ± 0.002	93.445 ± 0.123	0.007,2.426	3	3	2	(1:2)	3258	
	100,200	503.173 ± 0.462	503.806 ± 5.190	0.633,37.456	9	7	3	–	132	
	300,600	503.434 ± 0.220	503.936 ± 1.727	0.502,21.588	9	7	3	–	1125	
L99a	600,900	502.822 ± 0.169	503.877 ± 1.151	1.055,17.623	9	7	3	–	3888	
	800,1200	502.990 ± 0.174	503.139 ± 0.863	0.149,15.260	9	7	3	–	7662	
	1000,1500	503.034 ± 0.163	503.412 ± 0.691	0.378,13.648	9	7	3	–	11298	Mix of distributions
	100,200	92.948 ± 0.008	92.902 ± 0.923	-0.046,6.658	3	3	2	(1:2)	45	
	300,600	92.956 ± 0.008	92.977 ± 0.307	0.021,3.837	3	3	2	(1:2)	356	
BBD12	600,900	92.960 ± 0.003	92.961 ± 0.205	0.001,3.132	3	3	2	(1:2)	1118	
	800,1200	92.949 ± 0.004	92.954 ± 0.153	0.005,2.712	3	3	2	(1:2)	1938	
	1000,1500	92.949 ± 0.002	92.964 ± 0.123	0.015,2.426	3	3	2	(1:2)	3160	

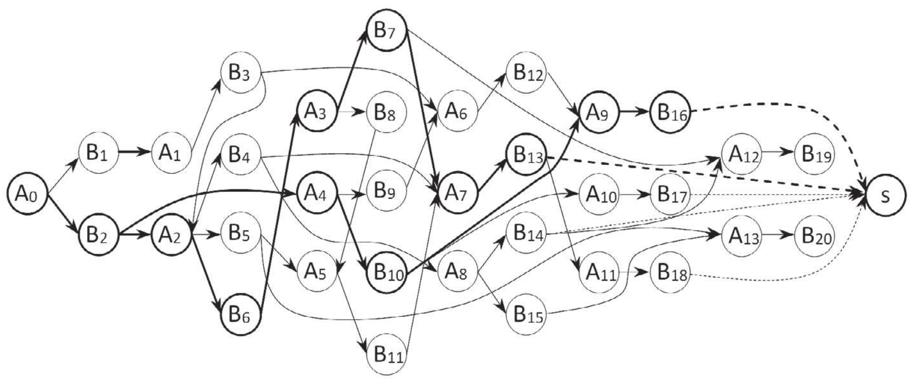

Figure 9. Selected alternative and disassembly tasks for instance L99b.

Table 5. Obtained results: idle time leveling.

		Upper bound			Lower bound			Gap(%)
		UB	Reassignment	CPU time λ	LB	Reassignment	CPU time	
MJKL11	500	16.03	0	0.016	15.86	0	0.001	1.07
	1000	16.18	0	0.047	15.74	0	0.020	2.79
	1500	15.82	0	0.031	15.82	0	0.020	0
BBD13a	500	0	0	0.000	0	0	0.001	0
	1000	0	0	0.000	0	0	0.001	0
	1500	0	0	0.000	0	0	0.001	0
L99a	500	7.679	0	26.22	7.092	0	0.500	8.27
	1000	7.523	0	90.57	7.011	0	0.08	7.30
	1500	7.644	0	269.19	6.709	0	0.06	13.94
BBD13b	500	22.95	0	0.078	17.95	0	0.001	27.85
	1000	23.00	0	0.172	18.88	0	0.020	21.82
	1500	23.72	0	0.296	18.72	0	0.001	26.71
KSE09	500	6.533	0	0.016	6.532	0	0.020	0.01
	1000	6.539	0	0.062	6.451	0	0.001	1.36
	1500	6.526	0	0.034	6.448	0	0.001	1.21
L99b	500	1.904	1	44.73	1.803	1	0.100	5.60
	1000	1.791	1	198.64	1.789	1	0.06	0.11
	1500	1.827	1	543.09	1.776	1	0.08	2.87
BBD12	500	0.498	0	0.030	0.493	0	0.001	1.01
	1000	0.499	0	0.020	0.497	0	0.001	0.40
	1500	0.499	0	0.030	0.497	0	0.020	0.40

Table 6. Recourse cost analysis of instance L99b.

		Estimated recourse cost						
	λ	R_{W1}	R_{W2}	R_{W3}	R_{total}	$\eta(\%)$	$\varrho(\%)$	CPU time
x	5000	0	0.00006	0.25006	0.25012	−85.75	5	24
	7500	0	0.00008	0.23578	0.23586			54
	10000	0	0.00005	0.24539	0.24544	−85.07	5	94
	12500	0	0.00010	0.24623	0.24633			144
	15000	0	0.00007	0.24874	0.24881	−85.80	5	204
x'	5000	0.00948	0	0.02617	0.03565			25
	7500	0.00809	0	0.02712	0.03521	−85.24	5	53
	10000	0.00883	0	0.02601	0.03484			94
	12500	0.00873	0	0.02763	0.03636	−85.64	5	145
	15000	0.00847	0	0.02727	0.03574			204

insignificantly compared to the increasing value of λ, which is due to the continuity of the linear programmes solved for each value of λ. Indeed, if $\lambda = 100$, then 100 continuous linear programmes of the same type as in step 2 of the L-shaped algorithm are solved.

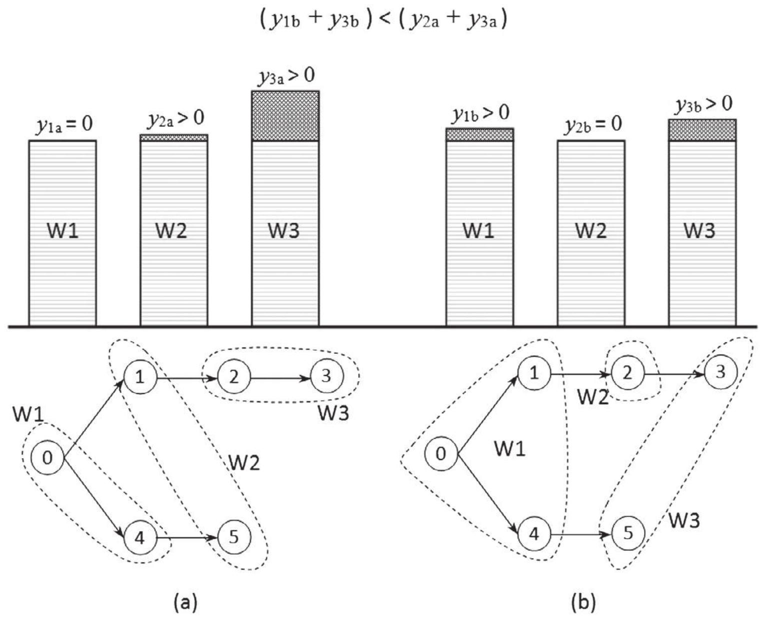

Figure 10. Idle time levelling and recourse cost decreasing for instance L99b: (a) tasks assignment and profit maximisation in phase 1; (b) tasks reassignment and idle time levelling in phase 2.

6. Conclusion and future research directions

In this paper, profit-oriented disassembly line design and balancing problems were studied under uncertainty. The case of partial disassembly and the presence of hazardous parts are considered and disassembly task times were assumed to be random variables with known probability distributions. To solve the defined problems with an assessment of solution quality, two phases were addressed. In the first phase, a two-stage stochastic linear mixed integer programme with fixed recourse and the SAA strategy were proposed for line design. The SAA method combined the L-shaped algorithm with Monte Carlo sampling techniques. This procedure provided lower and upper bounds for the optimal value of the optimisation problem solved and associated 95% confidence intervals. The corresponding optimality gaps and deviations were also provided. In the second phase, a min–max formulation and an upper bound mixed integer programme were developed in order to balance, under uncertainty, the line derived from the first phase.

The SAA procedure from the line design phase and the lower and upper bound MIPs from the line balancing phase were evaluated by a set of instances from the literature. The obtained results of phase 1 have shown that high-quality solutions require large Monte Carlo samples and a resolution time which increases considerably with the increase of sample sizes. All instances were solved in less than four hours, except one which was solved in just under seven hours with our previously specified machine. Since the installation of a disassembly line is usually based on a long-term decision, a resolution time of several hours for the problem of disassembly line design and balancing is acceptable. In the line balancing phase, all problem instances were solved in very little time. The numerical results have shown that the solution from phase 1 remained optimal in phase 2 for the majority of problem instances excluding one instance for which the line balance could be improved, resulting in a decrease of the recourse cost. This proved the added value of the line balancing phase.

The obtained results show the applicability of the developed optimisation model in real disassembly context. Indeed, the computational time is acceptable enough for giving to the designer the opportunity to generate different design alternatives depending on the profit expected from the retrieved parts. This model helps to make a decision on the disassembly alternative to be realised in the line and on the assignment of these tasks to workstations. Therefore, the choice between complete or partial disassembly can be made on the basis of the economic arguments. The modelling process presented can be easily adapted for real life cases like End of Life Vehicles or Waste Electrical and Electronic Equipment (WEEE). Undertaking such case studies is one of our next research objectives.

The presented study opens numerous further research directions for problem modelling and algorithmic enhancements. We believe that the computing time at the first solution stage can be reduced by generating new optimality cuts, using a computational grid in order to parallelise the optimisation process or by implementing penalty function techniques. The developed model should also be extended to more sophisticated line configurations, for example, with parallel workstations and consider multi-product types with different end-of-life states.

Disclosure statement

No potential conflict of interest was reported by the authors.

ORCID

Mohand Lounes Bentaha http://orcid.org/0000-0001-6564-3435
Alexandre Dolgui http://orcid.org/0000-0003-0527-4716
Olga Battaïa http://orcid.org/0000-0002-5367-7846

References

Agrawal, S., and M. K. Tiwari. 2006. "A Collaborative Ant Colony Algorithm to Stochastic Mixed-model U-shaped Disassembly Line Balancing and Sequencing Problem." *International Journal of Production Research* 46: 1405–1429.

Ahmed, S., and A. Shapiro. 2002. "The Sample Average Approximation Method for Stochastic Programs with Integer Recourse." *SIAM Journal of Optimization* 12: 479–502.

Altekin, F. T., and C. Akkan. 2011. "Task-failure-driven Rebalancing of Disassembly Lines." *International Journal of Production Research* 50: 4955–4976.

Altekin, F. T., L. Kandiller, and N. E. Ozdemirel. 2008. "Profit-oriented Disassembly-line Balancing." *International Journal of Production Research* 46: 2675–2693.

Antoine, M., B. E. H. Hind, R. C. K. Wahiba, and B. M. Lounes. 2016. "Iterated Local Search for Dynamic Assembly Line Rebalancing Problem." *IFAC-PapersOnLine* 49: 515–519.

Ashby,, M. F. 2012. *Materials and the Environment, Eco-informed Material Choice*. 2nd ed. Boston, MA: Butterworth-Heinemann.

Avikal, S., R. Jain, and P. Mishra. 2013. "A Heuristic for U-shaped Disassembly Line Balancing Problems." *MIT International Journal of Mechanical Engineering* 3: 51–56.

Aydemir-Karadag, A., and O. Turkbey. 2013. "Multi-objective Optimization of Stochastic Disassembly Line Balancing with Station Paralleling." *Computers & Industrial Engineering* 65: 413–425.

Battaïa, O., and A. Dolgui. 2013. "A Taxonomy of Line Balancing Problems and Their Solution Approaches." *International Journal of Production Economics* 142: 259–277.

Bentaha, M. L., O. Battaïa, and A. Dolgui. 2014a. "Disassembly Line Balancing and Sequencing under Uncertainty." *Procedia CIRP* 15: 239–244.

Bentaha, M. L., O. Battaïa, and A. Dolgui. 2014b. "Disassembly Line Balancing Problem with Fixed Number of Workstations under Uncertainty." In *19th IFAC World Congress*, 3522–3526. South Africa: Cape Town.

Bentaha, M. L., O. Battaïa, and A. Dolgui. 2014c. "Lagrangian Relaxation for Stochastic Disassembly Line Balancing Problem." *Procedia CIRP* 17: 56–60.

Bentaha, M. L., O. Battaïa, and A. Dolgui. 2014d. "A Sample Average Approximation Method for Disassembly Line Balancing Problem under Uncertainty." *Computers & Operations Research* 51: 111–122.

Bentaha, M. L., O. Battaïa , and A. Dolgui. 2015a. "A Bibliographic Review of Production Line Design and Balancing under Uncertainty." In *IFAC-PapersOnLine*. Vol. 48, 15th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2015, Ottawa, Canada, 70–75.

Bentaha, M. L., O. Battaïa, and A. Dolgui. 2015b. "An Exact Solution Approach for Disassembly Line Balancing Problem under Uncertainty of the Task Processing Times." *International Journal of Production Research* 53: 1807–1818.

Bentaha, M. L., O. Battaïa, A. Dolgui, and S. J. Hu. 2014e. "Dealing with Uncertainty in Disassembly Line Design." *CIRP Annals – Manufacturing Technology* 63: 21–24.

Bentaha, M. L., O. Battaïa, A. Dolgui, and S. J. Hu. 2015c. "Second Order Conic Approximation for Disassembly Line Design with Joint Probabilistic Constraints." *European Journal of Operational Research* 247: 957–967.

Birge, J. R., and F. Louveaux. 1997. *Introduction to Stochastic Programming*. New York: Springer-Verlag New York Inc.

Ding, L. P., Y. X. Feng, J. R. Tan, and Y. C. Gao. 2010. "A New Multi-objective Ant Colony Algorithm for Solving the Disassembly Line Balancing Problem." *The International Journal of Advanced Manufacturing Technology* 48: 761–771.

Güngör, A., and S. M. Gupta. 1999. "Disassembly Line Balancing." In *Proceedings of the 1999 Annual Meeting of the Northeast Decision Sciences Institute*, Newport, RI, 193–195.

Güngör, A., and S. M. Gupta. 2001. "A Solution Approach to the Disassembly Line Balancing Problem in the Presence of Task Failures." *International Journal of Production Research* 39: 1427–1467.

Gupta, S., and A. Güngör. 2001. "Product Recovery Using a Disassembly Line: Challenges and Solution." In *Electronics and the Environment, 2001. Proceedings of the 2001 IEEE International Symposium*, Denver, CO, 36–40.

Homem de Mello, L., and A. Sanderson. 1990. “AND/OR Graph Representation of Assembly Plans.” *IEEE Transactions on Robotics and Automation* 6: 188–199.
Ilgin, M. A., H. Akçay, and C. Araz. 2017. “Disassembly Line Balancing Using Linear Physical Programming.” *International Journal of Production Research* 55: 6108–6119. doi:10.1080/00207543.2017.1324225.
Ilgin, M. A., and S. M. Gupta. 2010. “Environmentally Conscious Manufacturing and Product Recovery (ECMPRO): A Review of the State of the Art.” *Journal of Environmental Management* 91: 563–591.
Kalayci, C. B., and S. M. Gupta. 2013. “A Tabu Search Algorithm for Balancing a Sequence-dependent Disassembly Line.” *Production Planning & Control* 25 (2): 149–160.
Kall, P., and S. W. Wallace. 1994. *Stochastic Programming*. 2nd ed. Chichester: John Wiley & Sons.
Kleywegt, A. J., A. Shapiro, and T. Homem-De-Mello. 2001. “The Sample Average Approximation Method for Stochastic Discrete Optimization.” *SIAM Journal of Optimization* 12: 47–502.
Koc, A., I. Sabuncuoglu, and E. Erel. 2009. “Two Exact Formulations for Disassembly Line Balancing Problems with Task Precedence Diagram Construction Using an AND/OR graph.” *IIE Transactions* 41: 866–881.
Lambert, A. J. D. 1999. “Linear Programming in Disassembly/Clustering Sequence Generation.” *Computers & Industrial Engineering* 36: 723–738.
Lambert, A. J. D. 2002. “Determining Optimum Disassembly Sequences in Electronic Equipment.” *Computers & Industrial Engineering* 43: 553–575.
Lambert, A. J. D., and S. M. Gupta. 2005. “A Heuristic Solution for the Disassembly Line Balancing Problem Incorporating Sequence Dependent Costs.” *Proceedings of SPIE 5997, Environmentally Conscious Manufacturing V, 59970A*, Boston, MA.
Ma, Y. S., H. B. Jun, H. W. Kim, and D. H. Lee. 2011. “Disassembly Process Planning Algorithms for End of Life Product Recovery and Environmentally Conscious Disposal.” *International Journal of Production Research* 49: 7007–7027.
Mak, W. K., D. P. Morton, and R. K. Wood. 1999. “Monte Carlo Bounding Techniques for Determining Solution Quality in Stochastic Programs.” *Operations Research Letters* 24: 47–56.
McGovern, S. M., and S. M. Gupta. 2005. “Uninformed and Probabilistic Distributed Agent Combinatorial Searches for the Unary NP-complete Disassembly Line Balancing Problem.” In *Proceedings of SPIE 5997, Environmentally Conscious Manufacturing V, 59970B*, Boston, MA.
McGovern, S. M., and S. M. Gupta. 2006. “Ant Colony Optimization for Disassembly Sequencing with Multiple Objectives.” *The International Journal of Advanced Manufacturing Technology* 30: 481–496.
McGovern, S. M., and S. M. Gupta. 2007. “Combinatorial Optimization Analysis of the Unary NP-complete Disassembly Line Balancing Problem.” *International Journal of Production Research* 45: 4485–4511.
Mete, S., Z. A. Çil, K. Ağpak, E. Ağpak, and A. Dolgui. 2016. “A Solution Approach Based on Beam Search Algorithm for Disassembly Line Balancing Problem.” *Journal of Manufacturing Systems* 41: 188–200.
Norkin, V. I., A. Ruszczynski, and G. C. Pflug. 1998. “A Branch and Bound Method for Stochastic Global Optimization.” *Mathematical Programming* 83: 425–450.
Prakash, P., D. Ceglarek, and M. K. Tiwari. 2012. “Constraint-based Simulated Annealing (CBSA) Approach to Solve the Disassembly Scheduling Problem.” *The International Journal of Advanced Manufacturing Technology* 60: 1125–1137.
Riggs, R. J., O. Battaïa, and S. J. Hu. 2015. “Disassembly Line Balancing under High Variety of End of Life States Using a Joint Precedence Graph Approach.” *Journal of Manufacturing Systems* 37: 638–648.
Santoso, T., S. Ahmed, M. Goetschalckx, and A. Shapiro. 2005. “A Stochastic Programming Approach for Supply Chain Network Design under Uncertainty.” *European Journal of Operational Research* 167: 96–115.
Tang, Y., and M. C. Zhou. 2006. “A Systematic Approach to Design and Operation of Disassembly Lines.” *IEEE Transactions on Automation Science and Engineering* 3: 324–329.
Tiwari, M. K., N. Sinha, S. Kumar, R. Rai, and S. K. Mukhopadhyay. 2002. “A Petri Net-based Approach to Determine the Disassembly Strategy of a Product.” *International Journal of Production Research* 40: 1113–1129.
Tripathi, M., S. Agrawal, M. K. Pandey, R. Shankar, and M. Tiwari. 2009. “Real World Disassembly Modeling and Sequencing Problem: Optimization by Algorithm of Self-Guided Ants (ASGA).” *Robotics and Computer-Integrated Manufacturing* 25: 483–496.
Tuncel, E., A. Zeid, and S. Kamarthi. 2014. “Solving Large Scale Disassembly Line Balancing Problem with Uncertainty Using Reinforcement Learning.” *Journal of Intelligent Manufacturing* 25: 647–659.
Turowski, M., and M. Morgan. 2005. “Disassembly Line Design with Uncertainty.” *2005 IEEE International Conference on Systems, Man and Cybernetics* 1 :954–959.
Van Slyke, R. M., and R. Wets. 1969. “L-shaped Linear Programs with Applications to Optimal Control and Stochastic Programming.” *SIAM Journal on Applied Mathematics* 17: 638–663.