L-Shaped Algorithm for Stochastic Disassembly Line Balancing Problem

Mohand Lounes Bentaha, Olga Battaïa, Alexandre Dolgui

7th IFAC Conference on Manufacturing Modelling, Management, and Control
International Federation of Automatic Control
June 19-21, 2013. Saint Petersburg, Russia

L-shaped Algorithm for Stochastic Disassembly Line Balancing Problem

M. L. Bentaha * **O. Battaïa** * **A. Dolgui** *

* *École Nationale Supérieure des Mines, EMSE-FAYOL, CNRS UMR6158, LIMOS, F–42023 Saint–Étienne, France (e-mail: {bentaha,battaia,dolgui}@emse.fr).*

Abstract: Disassembly Line Balancing Problem considered here both chooses the best disassembly alternative for an end of life product and assigns the corresponding disassembly tasks to the workstations of the line with the aim to reduce the line cost. Precedence and cycle time constraints are observed. Task times are assumed stochastic with known normal probability distributions. The line cost includes the investment and operation costs for workstations as well as penalty costs generated by the cycle time constraints violations. To deal with uncertainties, a stochastic linear mixed integer formulation is developed. To solve efficiently the problem, L-shaped algorithm combined with Latin Hypercube Sampling is proposed.

Keywords: Disassembly, Disassembly Line Design, DLBP, Stochastic Programming, Sample Average Approximation, L–shaped Method, LHS, Sustainable manufacturing.

1. INTRODUCTION

Disassembly plays an important role in the efficient management of the waste electrical and electronic equipment as well as end-of-life vehicles. The industrial interest to this process is increasing not only due to the environmental awareness of consumers and governments, but also due to its economical attractiveness (Zorpas and Inglezakis (2012), Mayyas et al. (2012), Güngör and Gupta (1999)). Disassembly aims at the separation of an End of Life (EOL) product into its components or sub-assemblies for product and material recovery. Product recovery includes the parts reuse and remanufacturing, while material recovery consists of recycling and material reuse. The disassembly line is the most efficient solution for disassembling complex products (like vehicles) and small products available in large quantities (like cell phones). Installing a disassembly line is a long term capital investment. Therefore, a particular attention has to be paid to its design stage. A mathematical model for optimizing line cost is proposed in this paper organized as follows. Section 2 presents a problem formulation. Section 3 describes a solution method developed. Section 4 presents an illustrative example. conclusions are given in Section 5.

2. PROBLEM FORMULATION

The case of complete disassembly of a single type of EOL product is considered. All possible alternatives for disassembly process are given by an And/Or Graph (AOG), (Ma et al. (2011), Altekin et al. (2008), Koc et al. (2009)). An example for such a graph is given in Figure 1. To simplify the graph, without information loss, subassemblies with one component are not represented. Each subassembly of the product to be disassembled is represented by an auxiliary node $A_k, k \in K$, in the AOG, and each basic node $B_i, i \in I$, represents a disassembly task. Two types of arcs define the precedence relations between the subassemblies and the disassembly tasks: And–type and Or–type arcs. For example, if a disassembly task generates two subassemblies, or more, then it is related to these subassemblies by And–type arcs (in bold on the Figure 1). If, for a given subassembly, one or more disassembly tasks can be performed, but only one must be chosen, this subassembly is related to these disassembly tasks by Or–type arcs. In order to calculate the number of workstations (after optimization), a dummy task S is introduced into the precedence graph as a sink node, see Figure 1. Disassembly task times $t_i, i \in I$, are assumed stochastic with known normal probability distributions: mean μ and variance σ^2 are known for each task time, *i.e.* $t_i \sim \mathcal{N}(\mu_i, \sigma_i)$, $t_i > 0$, $i \in I$; (Dolgui and Proth (2010), Silverman and Carter (1986)). It is modeled by a random vector $\tilde{\xi} = (t_1, t_2, \ldots, t_{\mathsf{N}})$ varying over a set $\Xi \subset \mathbb{R}^{\mathsf{N}}_+$; consider a given probability space $(\Xi, \mathcal{F}, P)$ introduced by the random vector $\tilde{\xi}$. Let $t_i = \zeta_i(\tilde{\xi}), i \in I$. The disassembly tasks have to be assigned to the workstations of the line in such a way that precedence constraints given by And/Or graph are respected and at the same time the line cost is minimized. The line cost includes two components: first one is related to the cost of workstations used, the second

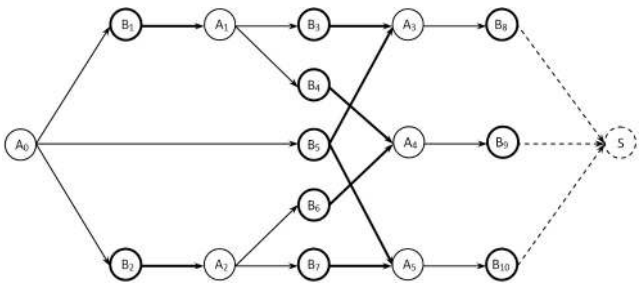

Fig. 1. And/Or precedence graph

one is entailed by exceeding cycle time C. In fact, each time the duration of tasks assigned to one workstation exceeds the given line cycle time, a corrective action is needed, and its cost is included in the objective function. For the problem described, the following mathematical model has been developed.

Stochastic Linear Mixed Integer Program

Parameters

I: Disassembly tasks' index set: $I = \{1, 2, \ldots, \mathsf{N}\}$, $\mathsf{N} \in \mathbb{N}^*$;
J: Workstations' index set: $J = \{1, 2, \ldots, \mathsf{M}\}$, $\mathsf{M} \in \mathbb{N}^*$;
K: Index set of the And/Or precedence graph's auxiliary nodes: $K = \{0, 1, \ldots, \mathsf{K}\}$, $\mathsf{K} \in \mathbb{N}$;
A_k: Auxiliary node of the And/Or graph, $k \in K$;
B_i: Disassembly task, $i \in I$;
S: The And/Or graph's sink node, $t_{\mathsf{S}} = 0$;
F_c: Fixed cost per unit time of operating the workstations;
C: Cycle time, $C > 0$;
P_k: Predecessors index set of A_k, $k \in K$, *i.e.* $P_k = \{i|\ B_i$ preceeds $A_k\}$;
S_k: Successors index set of A_k, $k \in K$, $S_k = \{i|\ A_k$ preceeds $B_i\}$.

Decision Variables

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

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

$$z_j = \begin{cases} C, & \text{if } x_{\mathsf{S}j} = 1; \\ 0, & \text{otherwise.} \end{cases}$$

The objective considered in this paper is to minimize the line cost including fixed workstation operating costs and recourse costs, $q_j, j \in J$ (per unit time), caused by exceeding C. A recourse variable $y_j(\tilde{\xi}), j \in J$, measures the amount of time exceeding C if there is any. The following model is used for the Stochastic DLBP presented.

SLMIP

$$\min \left\{ F_c \cdot \sum_{j \in J} j \cdot z_j + \mathbb{E}_{\tilde{\xi}} \left(\sum_{j \in J} q_j \cdot y_j(\tilde{\xi}) \right) \right\} \qquad (\mathbf{I})$$

s.t.

$$z_j = C \cdot x_{\mathsf{S}j}, \forall j \in J \qquad (1)$$

$$\sum_{i \in S_0} \sum_{j \in J} x_{ij} = 1 \qquad (2)$$

$$\sum_{j \in J} x_{ij} \leqslant 1, \forall i \in I \qquad (3)$$

$$\sum_{i \in S_k} \sum_{j \in J} x_{ij} = \sum_{i \in P_k} \sum_{j \in J} x_{ij}, \forall k \in K \backslash \{0\} \qquad (4)$$

$$\sum_{i \in S_k} x_{iv} \leqslant \sum_{i \in P_k} \sum_{j=1}^{v} x_{ij}, \forall k \in K \backslash \{0\}, \forall v \in J \qquad (5)$$

$$\sum_{j \in J} x_{\mathsf{S}j} = 1 \qquad (6)$$

$$\sum_{j \in J} j \cdot x_{ij} \leqslant \sum_{j \in J} j \cdot x_{\mathsf{S}j}, \forall i \in I \qquad (7)$$

$$\sum_{i \in I} \zeta_i(\tilde{\xi}) \cdot x_{ij} - y_j(\tilde{\xi}) \leqslant C, \forall j \in J \qquad (8)$$

$$z_j \in \{0, C\}, \forall j \in J \qquad (9)$$

$$x_{\mathsf{S}j}, x_{ij} \in \{0, 1\}, \forall i \in I, \forall j \in J \qquad (10)$$

$$y_j(\tilde{\xi}) \geqslant 0, \forall j \in J \qquad (11)$$

The term of the objective function represents the cost of operating workstations and the expected recourse cost. If the dummy task S is assigned to station j, which defines the number of processed workstations, then $\sum_{j \in J} j \cdot z_j = j \cdot C$ and the workstations operating cost becomes $j \cdot (F_c \cdot C)$. Constraints (1) ensure the value of z_j to be C when dummy task S is assigned to station j. Constraint (2) imposes the selection of only one disassembly task (Or–successor) to begin the disassembly process. Constraint set (3) indicates that a task is to be assigned to at most one workstation. Constraints (4) ensure that only one Or–successor is selected. Constraint set (5) defines the precedence relations among tasks: the selected successor is assigned to upper–indexed station (or the same) than the one to which the selected predecessor is assigned. Constraint (6) imposes the assignment of the dummy task S to one station. Constraints (7) ensure that all the disassembly tasks, preceding S, are assigned to lower or equal–indexed workstations than the one to which S is assigned. The constraints (8) enforces the station operating time to remain within the cycle time *under uncertainty*. Finally, sets (9)–(11) represent the trivial constraints. $\mathbb{E}_{\tilde{\xi}}$ stands for the expected value with the respect to the distribution of $\tilde{\xi}$:

$$\mathbb{E}_{\tilde{\xi}} \left(\sum_{j \in J} q_j \cdot y_j(\tilde{\xi}) \right) = \int_{\Xi} \left(\sum_{j \in J} q_j \cdot y_j(\tilde{\xi}) \right) \mathrm{d}P \qquad (12)$$

Note that the integral (12) makes the program (**I**) a nonlinear one, that is one source of difficulty in solving this problem.

Let x be a vector representing a solution of problem (**I**) and $X = \{x|$ constraints (1)–(7), (9)–(10) are satisfied$\}$. The program (**II**), given below, represents an equivalent stochastic version of program (**I**). It is a *two–stage stochastic linear mixed integer program with deterministic recourse*, (Birge and Tang (1993), Kall and Wallace (1994)).

$$\min \left\{ F_c \cdot \sum_{j \in J} j \cdot z_j + \mathbb{E}_{\tilde{\xi}} \big[\mathcal{Q}(x, \tilde{\xi}) \big] \right\} \qquad (\mathbf{II})$$

s.t.

$$x \in X$$

where for an outcome ξ of $\tilde{\xi}$, we have:

$$\mathcal{Q}(x,\xi) = \min\left\{\sum_{j\in J} q_j \cdot y_j(\xi)\right\} \qquad (\mathbf{II'})$$

s.t.

$$\sum_{i\in I} \zeta_i(\xi)\cdot x_{ij} - y_j(\xi) \leqslant C, \forall j \in J \qquad (13)$$
$$y_j(\xi) \geqslant 0, \forall j \in J$$

Let $L = \{1, 2, \ldots, \mathsf{L}\}$, $\mathsf{L} \in \mathbb{N}^*$. If $\tilde{\xi}$ has a finite discrete distribution $\{(\xi^l, p_l),\ l \in L\}, p_l > 0, \forall l \in L$, p_l is the realization probability of ξ^l of $\tilde{\xi}$, then (**I**) is an ordinary linear program with a *dual decomposition structure*, as shown below.

Deterministic Equivalent

$$\min\left\{F_c \cdot \sum_{j\in J} j\cdot z_j + \sum_{l=1}^{\mathsf{L}} p_l \sum_{j\in J} q_j \cdot y_j(\xi^l)\right\} \qquad (\mathbf{III})$$

s.t.

$$\sum_{i\in I} \zeta_i(\xi^l)\cdot x_{ij} - y_j(\xi^l) \leqslant C, \forall j \in J, \forall l \in L$$
$$y_j(\xi^l) \geqslant 0, \forall j \in J, \forall l \in L$$
$$x \in X$$

Depending on the number of realizations of $\tilde{\xi}$, *i.e.* L, this linear mixed integer program may become very large in scale, but its particular block structure can be exploited by specially designed algorithms such as the *L-shaped* method, (Ahmed and Shapiro (2002), Birge (1997); Birge and Tang (1993), Kall and Wallace (1994), Ntaimo (2013)).

3. SOLUTION METHOD

As mentioned earlier, computing the expected value $\mathbb{E}_{\tilde{\xi}}\big[\mathcal{Q}(x,\tilde{\xi})\big]$, for a given disassembly task assignment to workstations, is quite impossible. Even, for discrete distributions of $\tilde{\xi}$, the number of linear programs of type (**II'**) to solve may be a huge one. Although the exact evaluation of the expectation term in (**II**) is possible, its optimization presents serious difficulties, (Birge and Louveaux (1997), Santoso et al. (2005)). Indeed, $\mathbb{E}_{\tilde{\xi}}\big[\mathcal{Q}(x,\tilde{\xi})\big]$ is implicitly defined. To deal with these difficulties, we present a solution strategy combining a method, latin hypercube sampling, with the L–shaped algorithm. We begin by introducing the latin hypercube sampling and the L–shaped algorithm, then, the solution strategy developed is detailed.

3.1 Latin Hypercube Sampling

The Latin hypercube sampling is a Monte Carlo technique, having the efficient manner to sample the range of each random variable. Under the assumption of random variables independence, latin hypercube sampling generates, in the following manner, a sample of size Λ from $\tilde{\xi} = (t_1, t_2, \ldots, t_{\mathsf{N}})$ in consistency with the probability space $(\Xi, \mathcal{F}, P)$, (Helton and Davis (2003), Helton et al. (2006)).

The range of each variable $t_i, i = 1, 2, \ldots, \mathsf{N}$, is divided into Λ disjoint intervals of equal probability and one value is randomly selected from each interval. Then, the Λ values obtained for t_1 are paired at random, without replacement, with the Λ values obtained for t_2. These Λ pairs are randomly combined without replacement with the Λ values of t_3 to form Λ triples. This process is continued until a set of Λ N–tuples is formed. These N–tuples constitute the Latin hypercube sample.

3.2 The L–shaped Algorithm

The main idea of the L–shaped method is to approximate the nonlinear term in the objective function of the two–stage stochastic problems, (Birge (1997), Birge and Tang (1993)). Consider the formulation (**II**) of the SDLBP. Assume a finite realizations set Ξ of the stochastic vector $\tilde{\xi}$ such as $|\Xi| = \mathsf{L}$. The L-shaped algorithm, adapted to the SDLBP, processes as follows. Counters $\mathsf{r}, \mathsf{h}, \upsilon$ are respectively used for the feasibility and optimality cuts as well for the algorithm iterations. Constraint (16) is the matrix version of the cycle time constraints (13); $\boldsymbol{I}$ represents the identity matrix.

L–shaped Algorithm

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

***Step* 1**. Set $\upsilon = \upsilon + 1$. Solve the following LP:

$$\min\left\{F_c \cdot \sum_{j\in J} j\cdot z_j + \varphi\right\}$$

s.t.

$$x \in X$$
$$D_\nu \cdot x \geqslant d_\nu, \nu = 1, 2, \ldots, \mathsf{r} \qquad (14)$$
$$E_\nu \cdot x + \varphi \geqslant e_\nu, \nu = 1, 2, \ldots, \mathsf{h} \qquad (15)$$
$$x \text{ mixed binary}, \varphi \geqslant 0$$

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

***Step* 2**. For $l = 1, 2, \ldots, \mathsf{L}$, solve the following LP:

$$\min Z = a^{\mathsf{T}} \cdot u$$

s.t.

$$T_l \cdot x^\upsilon - \boldsymbol{I}\cdot y - \boldsymbol{I}\cdot u \leqslant C \qquad (16)$$
$$y \geqslant 0, u \geqslant 0$$

$a = (1, \ldots, 1)^{\mathsf{T}}$, until for some l the optimal value $Z > 0$. In this case, let σ^υ be the associated simplex multipliers; define

$$D_{\mathsf{r}+1} = (\sigma^\upsilon)^{\mathsf{T}} \cdot T_l$$

and

$$d_{\mathsf{r}+1} = (\sigma^\upsilon)^{\mathsf{T}} \cdot h_l$$

in order to generate constraint called a *feasibility cut* of type (14). Set $\mathsf{r} = \mathsf{r} + 1$, add constraint type (14) and return to ***Step* 1**. If $\forall l \in L, Z = 0,$ go to ***Step* 3**.

***Step* 3**. For $l = 1, 2, \ldots, \mathsf{L}$, solve the LP:

$$\min \mathcal{W} = q_l^{\mathsf{T}} \cdot y$$

s.t.

$$T_l \cdot x^\upsilon - \boldsymbol{I}\cdot y \leqslant C$$
$$y \geqslant 0$$

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

$$E_{\mathsf{h}+1} = \sum_{l \in L} p_l \cdot (\omega_l^{\upsilon})^{\mathsf{T}} \cdot T_l$$

$$e_{\mathsf{h}+1} = \sum_{l \in L} p_l \cdot (\omega_l^{\upsilon})^{\mathsf{T}} \cdot h_l$$

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* (constraint) of type (15), set $\mathsf{h} = \mathsf{h} + 1$, add constraint type (15) and return to ***Step* 1**.

This method approximated (12) using an outer linearization. Two types of constraints (cuts) are sequentially added:
1) feasibility cuts (14): if for a given solution $\widehat{x} \in X$, it exists $\xi \in \Xi$ such that the problem (**II'**) isn't feasible, then, a constraint of type (14) cutting $\widehat{x}$ is generated.
2) optimality cuts (15): in ***Step* 1** of the L-shaped algorithm, a new variable, φ, is introduced. It verifies the inequality $\varphi \geqslant \sum_{l=1}^{\mathsf{L}} p_l \cdot \mathcal{Q}(x, \xi^l)$. Since each $\mathcal{Q}(x, \xi)$ is implicitly defined by an optimization problem, the program in ***Step* 1** is solved without this inequality. If for a given feasible solution, $(\widehat{x}, \widehat{\varphi})$, of ***Step* 2**, we have $\widehat{\varphi} \geqslant \sum_{l=1}^{\mathsf{L}} p_l \cdot \mathcal{Q}(\widehat{x}, \xi^l)$, then, $(\widehat{x}, \widehat{\varphi})$ is an optimal solution, otherwise, a constraint of type (15) cutting $(\widehat{x}, \widehat{\varphi})$ is generated.

3.3 Sample Average Approximation Strategy

The SAA strategy deals with an approximate computation of the expected recourse cost. It allows the computation of lower and upper bounds of the optimal solution of problem (**II**). A random sample of $\tilde{\xi}$ generated by the LHS having size Λ, is considered. Then, the term $\mathbb{E}_{\tilde{\xi}}\big[\mathcal{Q}(x, \tilde{\xi})\big]$ is approximated by the sample average function $\frac{1}{\Lambda} \cdot \sum_{l=1}^{\Lambda} \mathcal{Q}(x, \xi^l)$. Thus, the problem (**II**) is approximated by the problem (**III**) while $p_l = \frac{1}{\Lambda}, l = 1, 2, \ldots, \Lambda$. The problem (**III**) is solved with the L–shaped algorithm.

SAA Procedure

***Step* 1**. Lower Bound Estimation:
generate, using LHS, Ω independent random samples $(\xi_1^n, \xi_2^n, \ldots, \xi_{\Lambda}^n), n = 1, 2, \ldots, \Omega$; Λ is the size of each sample. Solve the corresponding problem (**III**) with the L–shaped algorithm $\forall n = 1, 2, \ldots, \Omega$. Compute an optimal solution x_{Λ}^n and the corresponding objective value γ_{Λ}^n, for each value of n.

Compute a lower bound: $LB_{\Lambda\Omega} = \dfrac{1}{\Omega} \cdot \sum_{n=1}^{\Omega} \gamma_{\Lambda}^n$.

If γ^* represents the optimal objective value of (**II**), then, we have $\mathbb{E}(\gamma_{\Lambda}) \leqslant \gamma^*$ and $\gamma_{\Lambda} \longrightarrow_{\Lambda \to \infty} \gamma^*$ with probability 1; γ_{Λ} repesents the optimal objective value of problem (**III**), where $\mathsf{L} = \Lambda$ and $p_l = 1/\Lambda$, $l = 1, 2, \ldots, \Lambda$. As $LB_{\Lambda\Omega}$ is an unbiased estimator of $\mathbb{E}(\gamma_{\Lambda})$, we have $LB_{\Lambda\Omega} \leqslant \gamma^*$, (Mak et al. (1999)), Santoso et al. (2005).

The variance $\sigma^2_{LB_{\Lambda\Omega}}$ of $LB_{\Lambda\Omega}$ is calculated as follows:

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

***Step* 2**. Upper Bound Estimation:
generate, using LHS, a sample $(\xi_1, \xi_2, \ldots, \xi_{\Lambda'})$ of size Λ' independent from those generated in ***Step* 1**. Let $x_{\Lambda'}^n$ be a feasible solution obtained in ***Step* 1**; $x_{\Lambda'}^n$ should be the one for which the objective value of problem (**III**) is minimum, with $\mathsf{L} = \Lambda'$, $p_l = 1/\Lambda'$, $l = 1, 2, \ldots, \Lambda'$. Let $c_{\Lambda'}^n = F_c \cdot \sum_{j \in J} j \cdot z_j$

and $\mathcal{Q}_{\Lambda'}^n = \dfrac{1}{\Lambda'} \cdot \sum_{l=1}^{\Lambda'} \mathcal{Q}(x_{\Lambda'}^n, \xi^l)$, under the solution $x_{\Lambda'}^n$.

Compute an upper bound: $UB_{\Lambda'} = c_{\Lambda'}^n + \mathcal{Q}_{\Lambda'}^n$.

$UB_{\Lambda'}$ is an unbiased estimator of $\gamma^*_{x_{\Lambda'}^n}$ where $\gamma^*_{x_{\Lambda'}^n} = c_{\Lambda'}^n + \mathbb{E}_{\tilde{\xi}}\big[\mathcal{Q}(x_{\Lambda'}^n, \tilde{\xi})\big]$; $\gamma^*_{x_{\Lambda'}^n} \geqslant \gamma^*$, we have then, $UB_{\Lambda'} \geqslant \gamma^*$.

The variance $\sigma^2_{UB_{\Lambda'}}$ of $UB_{\Lambda'}$ can be estimated with:

$$\sigma^2_{UB_{\Lambda'}} = \frac{1}{\Lambda' - 1} \cdot \sum_{l=1}^{\Lambda'} (c_{\Lambda'}^n + \mathcal{Q}(x_{\Lambda'}^n, \xi^l) - UB_{\Lambda'})^2.$$

4. ILLUSTRATIVE EXAMPLE

The SAA procedure has been applied to the compass example illustrated on Figure 2. The product is made of seven components: (1) wheel, (2) left leg, (3) right leg, (4) left fixation screw, (5) lead, (6) tip and (7) right fixation screw. The ways of completely disassembling the product is modeled with the AOG shown on the Figure 1. The data associated with the product are grouped in Table 1 below. The method presented in this study was implemented in Microsoft Visual C++ 2008. ILOG CPLEX 12.4 was used to solve the model on a PC with Pentium(R) Dual-Core CPU 2.30 GHz and 3Go RAM.

Solution results

Lower and upper bounds values, after optimization for the compass example, are summarized in Table 2. Table 3 represents the solution corresponding to the upper bound. Note that just a subset of disassembly tasks is selected to completely disassemble the compass.

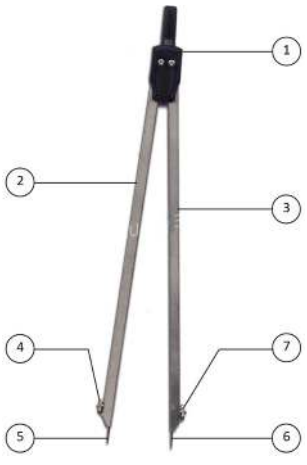

Fig. 2. Compass example

Table 1. The sample product associated data.

Task	μ	σ	subassembly	component
1	0.21	0.05	$1:5$	$6;7$
2	0.21	0.05	$1:3,6,7$	$4;5$
3	0.50	0.10	$2,4,5$	$1;3$
4	0.21	0.05	$1:3$	$4;5$
5	0.50	0.10	$2,4,5/3,6,7$	1
6	0.21	0.05	$1:3$	$6;7$
7	0.50	0.10	$3,6,7$	$1;2$
8	0.21	0.05	–	$2;4;5$
9	0.50	0.10	–	$1;2;3$
10	0.21	0.05	–	$3;6;7$
K, M, N	F_c	C	$q_j, \forall j \in J$	$\Omega, \Lambda, \Lambda'$
$5,3,10$	5	0.51	7	$20,30,50$

Table 2. Objective value results

LB	UB	$c^n_{\Lambda'}$	$\mathcal{Q}^n_{\Lambda'}$
$LB_{\Lambda\Omega} = 5.253$	$UB_{\Lambda'} = 5.340$	5.100	0.240
$\sigma^2_{LB_{\Lambda\Omega}} = 0.000$	$\sigma^2_{UB_{\Lambda'}} = 0.533$		

Under the upper bound solution, the number of workstations is 2. Tasks $\{2, 6\}$ are assigned to the 1^{st} workstation and task 9 to the second one. The overall idle time of the disassembly line, if mean time of each task selected is considered, is 0.10. This solution was provided in 10.5 seconds.

Table 3. Upper bound solution

Task	μ	σ	subassembly	component
2	0.21	0.05	$1:3,6,7$	$4;5$
6	0.21	0.05	$1:3$	$6;7$
9	0.50	0.10	–	$1;2;3$

5. CONCLUSION

In this paper, the DLBP was studied under uncertainty and a two–stage stochastic linear mixed integer program with fixed recourse was developed. Disassembly task times were assumed random variables with known normal probability distributions. To solve the problem efficiently, the SAA strategy was proposed. This strategy combined the L–shaped algorithm and the LHS Monte Carlo technique. The SAA procedure provided lower and upper bounds of the optimal solution for the SDLBP. Deviations of these bounds were also provided. To illustrate the SAA strategy, a case example was presented.

Our next objective is to test the SAA strategy on a data set of industrial problems in order to evaluate its practical efficiency. Such an evaluation should show if the developed method can afford the solution of the real–size problems characterized by a large number of alternatives and subassemblies in reasonable computational time. Further, the case of partial disassembly under uncertainty will be investigated.

REFERENCES

Ahmed, S. and Shapiro, A. (2002). The Sample Average Approximation Method for Stochastic Programs with Integer Recourse. Technical report, School of Industrial & Systems Engineering, Georgia Institute of Technology.

Altekin, F.T., Kandiller, L., and Ozdemirel, N.E. (2008). Profit-oriented disassembly-line balancing. *International Journal of Production Research*, 46(10), 2675–2693.

Birge, J.R. (1997). Stochastic programming computation and applications. *INFORMS Journal on Computing*, 9(2), 111–133.

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

Birge, J.R. and Tang, H. (1993). L–shaped method for two stage problems of stochastic convex programming. Technical report, Department of Industrial and Operations Engineering, University of Michigan.

Dolgui, A. and Proth, J.M. (2010). *Supply Chain Engineering: Useful Methods and Techniques*. Springer London Ltd, London, 1 edition.

Güngör, A. and Gupta, S.M. (1999). Issues in environmentally conscious manufacturing and product recovery : a survey. *Computers & Industrial Engineering*, 36, 811–853.

Helton, J. and Davis, F. (2003). Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. *Reliability Engineering & System Safety*, 81(1), 23–69.

Helton, J., Johnson, J., Sallaberry, C., and Storlie, C. (2006). Survey of sampling-based methods for uncertainty and sensitivity analysis. *Reliability Engineering & System Safety*, 91(10-11), 1175–1209.

Kall, P. and Wallace, S.W. (1994). *Stochastic Programming*. John Wiley & Sons, Chichester, 2 edition.

Koc, A., Sabuncuoglu, I., and 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.

Ma, Y.S., Jun, H.B., Kim, H.W., and Lee, D.H. (2011). Disassembly process planning algorithms for end-of-life product recovery and environmentally conscious disposal. *International journal of production research*, 49(23), 7007–7027.

Mak, W.K., Morton, D.P., and Wood, R. (1999). Monte Carlo bounding techniques for determining solution quality in stochastic programs. *Operations Research Letters*, 24(1–2), 47–56.

Mayyas, A., Qattawi, A., Omar, M., and Shan, D. (2012). Design for sustainability in automotive industry: A comprehensive review. *Renewable and Sustainable Energy Reviews*, 16(4), 1845–1862.

Ntaimo, L. (2013). Fenchel decomposition for stochastic mixed-integer programming. *Journal of Global Optimization*, 55, 141–163.

Santoso, T., Ahmed, S., Goetschalckx, M., and Shapiro, A. (2005). A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167(1), 96–115.

Silverman, F.N. and Carter, J.C. (1986). A cost-based methodology for stochastic line balancing with intermittent line stoppages. *Management Science*, 32(4), 455–463.

Zorpas, A.A. and Inglezakis, V.J. (2012). Technology in Society Automotive industry challenges in meeting EU 2015 environmental standard. *Technology in Society*, 1–29.