Journal of Intelligent Manufacturing (2021) 32:837–854
https://doi.org/10.1007/s10845-020-01592-z

End-of-life product disassembly with priority-based extraction of dangerous parts

Michela Dalle Mura[1] · Francesco Pistolesi[2] · Gino Dini[1] · Beatrice Lazzerini[2]

Received: 15 November 2019 / Accepted: 19 May 2020 / Published online: 13 June 2020

Abstract
The amount of electronic waste generated in the world is impressive. The USA alone yearly throw away 9.4 million tons of electronic devices: only 12.5% is recycled. One way to reduce this massive impact on the environment is to disassemble these devices with the aim of reusing and recycling as many parts as possible. Disassembling end-of-life products is a complex industrial process that may pose workers at risk because some parts of the product may contain dangerous materials. It is thus crucial to design efficient, sustainable and secure disassembly lines. This paper presents a multi-objective formulation of the Disassembly Line Balancing Problem (DLBP) which promotes efficiency and includes a new objective that increases the level of safety. The efficiency is guaranteed by balancing the idle times of the workstations, and by maximizing the profit and the level of feasibility of a disassembly sequence, which means disassembling the product as much as possible. Safety is maximized by extracting each dangerous part with a priority that is higher the more dangerous the part is. The most dangerous parts can thus be quickly removed from the product, thereby eliminating the exposure to the greatest risks. The disassembly continues with the execution of the tasks that remove the parts that are gradually less dangerous. Along with the DLBP formulation, this paper presents a genetic algorithm purposely designed to solve the problem. Two real-world case studies are discussed which entail the disassembly of a TV monitor and an air conditioner.

Keywords Combinatorial optimization · Disassembly line balancing · Electronic waste · Evolutionary computation · Genetic algorithm · Metaheuristic · Multi-objective optimization · Product reuse and recycling · Risk · Safety · Selective disassembly · Sustainability

Introduction

Technology has made great strides in recent years. This has generated an outstanding increase in sales of consumer goods which has progressively led to products that quickly become obsolescent (Habibi et al. 2014; Riggs et al. 2015; Özceylan et al. 2018). It is estimated that the volume of waste due to end-of-life (EOL) products will have achieved 54 million tonnes by 2025: recovering EOL products is thus crucial (Huang et al. 2015).

De-manufacturing is the first step to recover EOL products (Colledani et al. 2014) by means of *disassembly* and *shredding* processes (Colledani and Tolio 2013). Disassembly performs a series of *tasks* that remove from the product the parts that contain materials to reuse and components to resell as spare parts. On the other hand, shredding processes separate the valuable materials from each other, for reuse and recycling (Chern et al. 2015). The choice between disassembly and shredding is generally made on the basis of cost–benefit analyses. For example, when products do not contain any component to use as a spare part, they typically undergo shredding. Instead, an EOL product that contains hazardous materials must be disassembled to remove the dangerous parts from it: These parts are then properly disposed (McGovern and Gupta 2006a, b).

Hybrid options exist that combine disassembly and shredding. For example, EOL products may be disassembled to extract the dangerous parts and the components to resell as spare parts; the other components may then be shredded.

When disassembling products, it is possible to carry out this process incompletely, and stop it as soon as it achieves

✉ Francesco Pistolesi
francesco.pistolesi@unipi.it

1 Department of Civil and Industrial Engineering, University of Pisa, Largo Lazzarino, 1 – 56122 Pisa, Italy

2 Department of Information Engineering, University of Pisa, Largo Lazzarino, 1 – 56122 Pisa, Italy

the maximum profit (*partial disassembly*), or when specific parts have been removed from the product (*selective disassembly*). Disassembly processes are generally performed by manual and labour-intensive systems equipped with specific tools (McGovern and Gupta 2006a, b) that may be assisted by smart robots and sensors (Vongbunyong et al. 2013). The *disassembly line* is the best way to organize these systems when disassembling large volumes of products that come from waste collection facilities (McGovern and Gupta 2006a, b). In order to design efficient lines, there is the need to decide the number of workstations to set up, which tasks must be performed at each workstation, and in what order, so as to satisfy the precedence relationships among the tasks. This is the *Disassembly Line Balancing Problem (DLBP)*, an NP-hard combinatorial optimization problem, with multiple objectives (McGovern and Gupta 2006a, b).

Many recent works that deal with the DLBP in the literature prove an increasing interest in looking for efficient algorithms for EOL disassembly (Kalayci et al. 2016; Mete et al. 2016; Liu et al. 2018; Zhang et al. 2017; Özceylan et al. 2018). The existing works split into *single-objective (SO)* and *multi-objective (MO)* approaches. The main purpose of several SO techniques is profit maximization. For example, Bentaha et al. (2014) proposed an exact method based on integer programming and Monte Carlo sampling, with a factor of uncertainty that affects the quality level of the returned parts. Among the SO contributions, the minimization of the number of workstations was considered by Mete et al. (2016), who used a beam search-based approach. In Riggs et al. (2015), the DLBP is instead solved by minimizing the difference in execution time at the various workstations, considering EOL products in various wear conditions.

Regarding the MO approaches, the methods that make the scalarization of the objectives are popular in the literature. For example, Tuncel et al. (2014) minimized the number of workstations of the disassembly line, while trying to anticipate as much as possible the extraction of the hazardous and high-demand components. Among the bio-inspired techniques, genetic algorithms (GAs) and hybrid methods based on GAs have been shown to achieve better results compared to other evolutionary counterparts (Pistolesi et al. 2018). Kalayci et al. (2016) proposed a hybrid method that combines a variable neighborhood search with a GA, in order to minimize the number of workstations, balance the disassembly line, and quickly remove from the product the hazardous parts and those that are in high demand. These objectives had already been considered by McGovern and Gupta (2006a, b), along with the minimization of the number of changes of direction of the product. Ding et al. (2010) developed an ant colony algorithm that minimizes three objectives: the number of workstations, a measure of balance, and the demand rate. Kalayci and Gupta (2013) proposed a particle swarm algorithm that balances the disassembly line, while removing from the product the hazardous and high-demand components, as soon as possible. Finally, Pistolesi and Lazzerini (2019) proposed a tensorial memetic algorithm for many-objective disassembly in product refurbishment, aimed at maximizing the degree of parallelism of the tasks, the level of ergonomics, and how the workers' workload is balanced, while minimizing the disassembly time and the number of times the product has to be rotated.

To the best of the authors' knowledge, most of the contributions model the precedence relationships among the disassembly tasks (*sequence feasibility*) as a constraint. A few works—e.g., those concerning disassembly sequence planning (DSP) (Zhang et al. 2014)—proposed the maximization of the number of precedence relationships as an objective. Most of these contributions use scalarized approaches. For example, Rickli and Camelio (2013) described a GA that maximizes the profit and feasibility, while minimizing the environmental impact. The feasibility of a disassembly sequence is guaranteed by a penalty value in the fitness function. However, the use of penalty techniques may result in the reduction of the decision space, thereby leading to suboptimal solutions (Marler and Arora 2004).

The minimization of the risk to which workers are exposed in industrial scenarios is another objective to take into account. This risk can be minimized by exploiting the workers' sensitivity to risk (Lazzerini and Pistolesi 2014, 2018a, b), or by increasing the level of safety of the process. When disassembling dangerous products, the hazardous parts must be removed as soon as possible. This reduces the hazards (i.e., the potential sources of harm to workers), the exposure to hazardous agents and conditions, and then the risks associated with those hazards. This objective was considered by many researchers (McGovern and Gupta 2006a, b; Kalayci and Gupta 2013; Tuncel et al. 2014; Kalayci et al. 2016). All these contributions use formulations derived from that proposed by McGovern and Gupta (2006a, b), and consider the dangerous parts without any distinction. However, some parts may be far away more dangerous than others, and should thus be given the highest extraction priority. Removing first these highly dangerous parts from the product, then continuing with those that are gradually less dangerous, can lead to better safety conditions compared to trying to extract all the dangerous parts with the same priority. By using different priorities, the most dangerous parts remain inside the product for the shortest possible time, and the workers' safety is considerably increased because the most significant components of the global hazard are eliminated as first. The risk that remains has an increasingly lower potential to cause serious effects.

This paper proposes a new formulation of partial DLBP that maximizes four objectives: (1) the *level of balancing of the idle times*; (2) the *profit*; (3) the *level of feasibility*; (4) the *level* of *safety*. The level of safety of a disassembly sequence is a novel objective. It is calculated by assigning each task that removes a dangerous part from the product a level of danger that depends on the hazardous material that the part contains, or is made of. The level of danger is obtained by using the *UN Recommendations on the transport of dangerous goods* (United Nations 2009). The second contribution is a GA with genetic operators that were designed to increase the level of feasibility of the disassembly sequences, thereby giving the algorithm the ability to explore a wider region of the decision space. Table 1 summarizes the contributions and differences of this work against the cited references. For each reference (row), the table shows the type of approach (SO or MO), the objectives, and the resolution method in the second, third and fourth columns, respectively. The remaining two columns contain the type of approach of the proposed work (MO) and the difference w.r.t. the objectives taken into account, where '+' ('–') indicates that the proposed algorithm considers (neglects) an objective that is neglected (considered) in the referenced work.

The paper is organized as follows. Section "Multi-objective optimization" gives the mathematical background. Section "Genetic algorithms" introduces the basic concepts of these algorithms. Section "Problem" outlines the formulation of the problem. Section "Algorithm" describes the algorithm. Section "Case studies" discusses the experiments. Section "Performance evaluation and comparisons" compares the results to those obtained by other algorithms. The conclusions are in section "Conclusions".

Background

Multi-objective optimization

A multi-objective optimization (MOO) problem entails minimizing/maximizing multiple objectives, also satisfying a set of constraints (Al_Janabi et al. 2019; Alkaim and Al_Janabi 2019; Deb 2014; Marler and Arora 2004).

An MOO problem can be formulated as $\min_{x \in \mathcal{X}} \mathbf{f}(\mathbf{x})$, where $\mathcal{X} = \{x \in \mathbb{R}^n : g_i(\mathbf{x}) \leq 0, h_j(\mathbf{x}) = 0, \forall i = 1, \dots, G, \forall j = 1, \dots, H\}$ is the feasible region, and G and H are the number of inequality and equality constraints that define the feasible region, respectively. Vector function $\mathbf{f}(\mathbf{x}) = (\mathbf{f}_1(\mathbf{x}), \dots, \mathbf{f}_k(\mathbf{x}))$ contains k objective functions to minimize, where $\mathbf{x} \in \mathcal{X}$ is a feasible solution. For maximization problems, it holds that $\max_{x \in \mathcal{X}} \mathbf{f}(\mathbf{x}) = \min_{x \in \mathcal{X}} -\mathbf{f}(\mathbf{x})$.

In general, in MOO problems no solution exists that minimizes all objectives. The concept of Pareto optimality is thus introduced. Given two feasible solutions $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$, solution $\mathbf{x}_1$ (Pareto-)dominates solution $\mathbf{x}_2$ if $\mathbf{x}_1$ is better than $\mathbf{x}_2$ with respect to at least one objective, without being worse than $\mathbf{x}_2$ with respect to the remaining objectives. A solution is Pareto-optimal if there is no solution that dominates it. The set of all Pareto optimal solutions forms the so-called *Pareto set*. The image of this set in the objective space is called *Pareto front*.

Genetic algorithms

Genetic algorithms (GAs) are based on biological evolution (Holland 1975). GAs can solve optimization problems where analytic methods fail, due to the high complexity. GAs encode solutions as *chromosomes* (also *individuals*), which contain *genes* that take real numbers, integers or bits as values.

GAs start generating a population of individuals (candidate solutions), and a *fitness function* measures their goodness. In maximization problems, the higher the fitness of an individual, the better the solution the individual represents. The individuals with high fitness are more likely to be selected for reproduction. Those selected form the mating pool and are then chosen (typically in pairs) to be recombined by using a *crossover* operator. Recombination generates new individuals that may undergo *mutation*. A part or all of the individuals of the population are replaced with the new individuals (*offspring*). The process iterates until a stop condition is satisfied (BoussaïD et al. 2013; Dalle Mura and Dini 2017; Dokeroglu et al. 2019).

Problem

Hypotheses

The assumptions considered in this work are as follows:

- paced disassembly lines with fixed cycle time;
- deterministic execution time of the tasks;
- serial line layout, one-sided stations;
- workstations with similar equipment;
- resell all recovered parts and recyclable materials.

Problem formulation

Consider a product made up of M parts, and consider N *disassembly tasks*. Each task removes one or more parts from the product, or splits the product into subassemblies. Each task can only be performed if certain tasks have already been performed. These tasks form a set of *precedence constraints*. A series of tasks that satisfies the precedence constraints is a *disassembly sequence*. A sequence performs a complete (partial) disassembly of the product if it executes all (part of) the tasks.

The DLBP assumes that the disassembly process is performed in a disassembly line with W workstations. Let t_i be the time a worker takes to perform task i, called *task*

Table 1 Comparison of the proposed work to the cited works of the literature

Reference	SO/MO	Objectives	Method	Proposed work	
				SO/MO	Objectives ('+' means added; '–' means neglected)
Bentaha et al. (2014)	SO	Max (Profit)	Integer programming and Monte Carlo sampling	MO	+ Max (Smoothness) + Max (Safety) + Max (Feasibility)
Mete et al. (2016)	SO	Min (number of workstations)	Beam search	MO	+ Max (Profit) + Max (Smoothness) + Max (Safety) + Max (Feasibility)
Riggs et al. (2015)	SO	Min (difference in execution time at the various workstations)	Joint precedence graph	MO	+ Max (Profit) + Max (Safety) + Max (Feasibility)
Tuncel et al. (2014)	MO	Min (number of workstations) Anticipate extraction of hazardous components Anticipate extraction of high-demand components	Scalarized Monte-Carlo based reinforcement learning	MO	+ Max (Profit) + Max (Smoothness) + Max (Safety based on level of danger of parts) + Max (Feasibility)
Kalayci et al. (2016)	MO	Min (number of workstations) Min (difference in execution time at the various workstations) Anticipate extraction of hazardous components Anticipate extraction of high-demand components	GA with variable neighborhood search	MO	+ Max (Profit) + Max (Safety based on level of danger of parts) + Max (Feasibility)
McGovern and Gupta (2006a, b)	MO	Min (number of workstations) Min (difference in execution time at the various workstations) Anticipate extraction of hazardous components Anticipate extraction of high-demand components Min (irection changes number)	Ant colony	MO	+ Max (Profit) + Max (Safety based on level of danger of parts) + Max (Feasibility) – Min (no. of direction changes)
Ding et al. (2010)	MO	Min (number of workstations) Min (measure of balance) Min (demand rate)	Ant colony	MO	+ Max (Profit which includes Min (demand rate)) + Max (Safety) + Max (Feasibility)
Kalayci and Gupta (2013)	MO	Min (difference in execution time at the various workstations) Anticipate extraction of hazardous components Anticipate extraction of high-demand components	Particle swarm	MO	+ Max (Profit) + Max (Safety based on level of danger of parts) + Max (Feasibility)
Rickli and Camelio (2013)	MO	Max (Profit) Max (Feasibility) Min (environmental impact)	Scalarized GA	MO	+ Max (Smoothness) + Max (Safety) – Min (Environ. impact)
Pistolesi et al. (2018)	MO	Max (Profit) Min (no. of workstations) Max (disassembly depth)	GA + Extremal Optimization	MO	+ Max (Smoothness) + Max (Feasibility) + Max (Safety)

Table 1 (continued)

Reference	SO/MO	Objectives	Method	Proposed work	
				SO/MO	Objectives ('+' means added; '–' means neglected)
Pistolesi and Lazzerini (2019)	MO	Max (degree of parallelism) Max (ergonomics) Max (workload balancing) Min (disassembly time) Min (rotations of the product)	Tensorial Memetic Algorithm	MO	+ Max (Profit) + Max (Feasibility) + Max (Safety) – Max (Degree of parallelism) – Max (Ergonomics) – Min (rotations of the product)

duration. The tasks of a disassembly sequence are assigned to the workstations so that the total duration of the tasks at each workstation does not exceed a maximum time T, called *cycle time*, which is the same for all workstations.

Let vector $\mathbf{x} \in \{0,1\}^{N\times W\times N}$, where $i,k \in \{1,\dots,N\}$ and $j \in \{1,\dots,W\}$, be a disassembly sequence whose elements x_{ijk}, in lexicographic order, are such that

$$x_{ijk} = \begin{cases} 1 & \begin{array}{l}\text{if task } i \text{ is the } k\text{-th in the sequence}\\ \text{and is assigned to station } j\end{array} \\ 0 & \text{otherwise.}\end{cases} \tag{1}$$

Let $\mathbf{Y} = [y_{hi}]$ be an $N \times N$ matrix where

$$y_{hi} = \begin{cases} 1 & \text{if task } h \text{ must precede task } i \\ 0 & \text{otherwise.}\end{cases} \tag{2}$$

The optimization problem can be formulated as follows:

$$\underset{\mathbf{x}}{\text{Maximize}}\ \mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), -f_3(\mathbf{x}), f_4(\mathbf{x})] \tag{3a}$$

subject to

$$\sum_{i=1}^{N}\sum_{k=1}^{N} x_{ijk} t_i \le T \quad \forall j = 1,\dots,W \tag{3b}$$

$$\left\lceil \frac{\sum_{i=1}^{N} t_i}{T} \right\rceil \le WS \le N \tag{3c}$$

$$\sum_{j=1}^{W}\sum_{k=1}^{N} x_{ijk} \le 1 \quad \forall i = 1,\dots,N \tag{3d}$$

$$x_{ijk} \le \sum_{u=1}^{j}\sum_{k=1}^{N} x_{huk} \quad \forall i = \{1,\dots,N\},\ \forall h : y_{hi} = 1 \tag{3e}$$

$$\sum_{i=1}^{N}\sum_{j=1}^{W} x_{ijk} \le 1 \quad \forall k = 1,\dots,N \tag{3f}$$

$$\sum_{i=1}^{N}\sum_{j'\neq j} x_{ij'k} = 0 \quad \forall j = 1,\dots,W,\ \forall k = 1,\dots,N \tag{3g}$$

$$x_{ijk} \in \{0,1\} \quad \forall i,k = 1,\dots,N,\ \forall j = 1,\dots,W. \tag{3h}$$

Equation (3a) is a vector function, whose elements are the level of feasibility f_1, the profit f_2, the level of balancing of the idle times of the workstations f_3, and the level of safety f_4.

The *level of feasibility* is modeled as follows:

$$f_1(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{W}\sum_{k=1}^{N} x_{ijk} \tag{4}$$

i.e., as the ratio of the number of feasible tasks to the total number of tasks N.

The *profit* is obtained as the maximum profit achieved after performing the last dangerous task of sequence $\mathbf{x}$, i.e.,

$$f_2(\mathbf{x}) = \max_{k'=K}^{N} \sum_{i=1}^{N}\sum_{j=1}^{W}\sum_{k\le k'} p_i x_{ijk} \tag{5}$$

where $p_i \in \mathbb{R}$ is the profit achieved by performing task i, and K is the position in the sequence where the last dangerous task is performed.

The *level of balancing of the idle times* (also *smoothness*) of the workstations is calculated as follows:

$$f_3(\mathbf{x}) = \sum_{j=1}^{W}\Big(T - \sum_{i=1}^{N}\sum_{k=1}^{N} t_i x_{ijk}\Big)^2. \tag{6}$$

This objective function also minimizes the number of workstations required (McGovern and Gupta 2006a, b). Objective function (6) has to be minimized, it is thus inverted in sign in (3a).

The last objective is the *level of safety* of sequence **x**. In the formulation proposed in this paper, the level of safety of a disassembly sequence is calculated by first splitting the tasks into *dangerous* and *non-dangerous*. The dangerous tasks are those that remove from the product the parts with hazardous materials, i.e., substances, solids, liquids, or gases that can harm people, living organisms, or the environment. When transported or moved, these materials are a risk to health and safety. The *UN Recommendations on the Transport of Dangerous Goods* (United Nations 2009) classifies the hazardous materials into three degrees of danger, i.e.,

- *Packing Group 1*: high danger
- *Packing Group 2*: medium danger
- *Packing Group 3*: low danger

and determines the degree of protective packaging required for dangerous goods during transportation.

In fact, a product to disassemble that contains hazardous materials is a dangerous good that is continuously moved, pushed, pulled and rotated by the workers, as it happens in transportation. The concept of packing group is used to classify the dangerous tasks based on the risk that stems from the exposure to the hazardous materials contained in the parts. The dangerous tasks are thus diversified based on their level of danger. This novelty can help prioritize the execution of the most dangerous tasks, thereby increasing the safety of the line. The new objective is modeled as follows:

$$f_4(\mathbf{x}) = -\sum_{\substack{i=1,\dots,N \\ d_i \neq 4}} \sum_{j=1}^{W} \sum_{k=1}^{N} \left(k x_{ijk}\right)^{d_i} \tag{7}$$

where $d_i \in \{1, 2, 3, 4\}$ is the *level of danger* of task *i*. Each dangerous task of **x** is associated with a level of danger in {1, 2, 3} that corresponds to the packing group of the material of the part that the task removes. Non-dangerous tasks are assigned a level of danger of 4. According to (7), a sequence **x** is safer, the sooner (low values of *k*) the dangerous tasks are performed, and the higher the number of dangerous tasks that are executed in decreasing order of level of danger.

The constraints are as follows: (3b) prevents the cycle time from being exceeded at each workstation; (3c) forces the number of workstations (*WS*) to take a value between the lowest number and the maximum number of workstations, where the latter coincides with the number of tasks; (3d) ensures that each task is assigned to at most one workstation (in the case of partial disassembly, a task may not be assigned to any workstation); (3e) prevents each task *i* from being assigned to station *j* if all the tasks that must be performed before *i* (according to the precedence constraints) are not assigned to either station *j*, or to one of the previous stations; (3g) makes each task be performed only once; (3h) forces binary decision variables.

Algorithm

This section describes the proposed GA, by giving the details of the encoding, fitness evaluation and genetic operators.

Encoding

A disassembly sequence is represented as a chromosome $\mathbf{z} \in \mathrm{Sym}(N)$, where $\mathrm{Sym}(N)$ denotes the symmetric group of degree *N*, whose elements are all the possible permutations of the tasks in $\{1, \dots, N\}$. Each gene of **z** is a disassembly task, and the length of the chromosome corresponds to the total number of tasks *N*. A gene is *feasible* if the corresponding task is preceded by all the tasks that must be performed before it (according to the precedence constraints), otherwise the gene is *infeasible*. Figure 1 shows an example of chromosome.

Fitness function

The fitness $\mathbf{f} : Sym(N) \rightarrow [0, 1] \times \mathbb{R}^3$ of a chromosome **z** is:

$$\mathbf{f}(\mathbf{z}) = [F(\mathbf{z}),\ P(\mathbf{z}),\ -B(\mathbf{z}),\ S(\mathbf{z})], \tag{8}$$

where $F(\mathbf{z})$ is the level of feasibility, $P(\mathbf{z})$ is the profit, $B(\mathbf{z})$ is the level of balancing of the idle times, and $S(\mathbf{z})$ is the level of safety. These four objective functions calculate the same values as functions (4)–(7), by using the encoding introduced in the previous section.

The level of feasibility $F(\mathbf{z})$ is expressed by the ratio of the number N_F of feasible tasks of **z** to the total number of tasks *N*:

$$F(\mathbf{z}) = \frac{N_F}{N}. \tag{9}$$

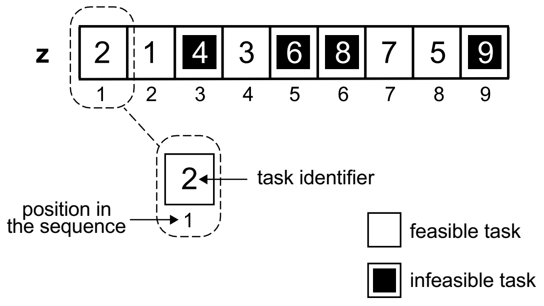


Fig. 1 Chromosome that represents a disassembly sequence for a product that requires nine tasks to be entirely disassembled

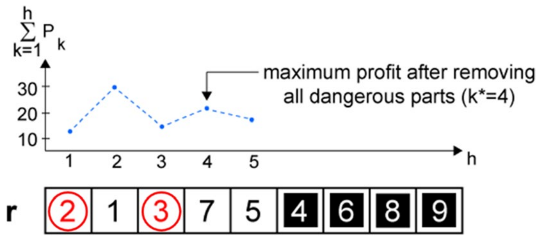


Fig. 2 Rearrangement (**r**) of the chromosome (**z**) of Fig. 1, where tasks 2 and 3 are dangerous. The plot above **r** shows the profit achieved by performing each subsequence of adjacent feasible tasks (white background) that is made up of the first h tasks of **r**, e.g., subsequence $\langle 2 \rangle$ if $h=1$, subsequence $\langle 2,1 \rangle$ if $h=2$, and so on. The disassembly stops at $h=4$ because the next task would diminish the profit

The profit $P(\mathbf{z})$ is calculated by reorganizing the genes of $\mathbf{z}$ so that all the feasible genes then come before those infeasible, in the same order as they are in $\mathbf{z}$. Let $\mathbf{r} \in \text{Sym}(N)$ be the reorganized chromosome. The profit of each subsequence of tasks that includes all the dangerous tasks is first calculated, i.e., the profits of subsequences $\langle r_1, \ldots, r_K \rangle$, $\langle r_1, \ldots, r_{K+1} \rangle$, and so on until subsequence $\langle r_1, \ldots, r_{N_F} \rangle$. The highest of these profits determines the position k^* of the last task to perform. Formally, the profit of $\mathbf{z}$ is obtained as follows

$$P(\mathbf{z}) = \max_{h=K}^{N^F} \sum_{k=1}^{h} P_k, \tag{10}$$

where P_k is the profit of task r_k, i.e., the task in position k. Figure 2 shows an example of how the profit is calculated.

Consider task r_k, from now on referred to as k. The profit P_k of task k is defined as:

$$P_k = R_k + SP_k - DC_k - D_k - D_k^{REST}. \tag{11}$$

Term R_k [€] in Eq. (11) is the revenue that comes from reselling the recyclable materials, calculated as

$$R_k = \sum_{i \in \mathcal{M}_k} \rho_i \, w_{k,i}, \tag{12}$$

where $\mathcal{M}_k$ contains the recyclable materials that are recovered by task k, whereas ρ_i and $w_{k,i}$ are the unitary revenue [€/kg] and weight [kg] of the i-th material recovered, respectively.

Term SP_k in Eq. (11) is the value [€] of the recoverable components removed from the product by task k that are resold as spare parts, and is calculated as

$$SP_k = \sum_{i \in \mathcal{S}_k} v_i \tag{13}$$

where $\mathcal{S}_k$ is the set of the spare parts recovered by task k, and v_i is the market value [€] of spare part i.

Term DC_k of Eq. (11) is the disassembly cost [€], i.e.,

$$DC_k = \frac{L}{60} t_k \tag{14}$$

where t_k is the time, in minutes, an operator takes to perform task k, and L is the hourly labor cost [€/h].

Term D_k in Eq. (11) is the disposal cost [€] due to the non-recoverable parts that are extracted by task k, i.e.,

$$D_k = \sum_{i \in \mathcal{Z}_k} c_i \tag{15}$$

where set $\mathcal{Z}_k$ contains the parts disposed, and c_i is the cost [€] to dispose part i.

Finally, D_k^{REST} [€] in Eq. (11) is the disposal cost for the rest of the product (the part that is not disassembled), calculated as

$$D_k^{REST} = \sum_{h=k+1}^{N} D_h. \tag{16}$$

The level of balancing $B(\mathbf{z})$ of the idle times of the workstations is calculated by first assigning the feasible tasks of $\mathbf{z}$ (those in $\langle r_1, \ldots, r_{k^*} \rangle$) to the workstations so that the total duration of the tasks assigned to each workstation does not exceed the cycle time T. The procedure is shown in Fig. 3, where the feasible tasks are represented as rectangles whose lengths are in proportion to the duration of the tasks. The workstations are labeled with roman numerals. From an operational point of view, the first task of the sequence (task 2) is assigned to the first workstation. Then, the second task is assigned to the second workstation because the task takes more time than the idle time that remains in workstation 1 after allocating task 2. This procedure continues until each

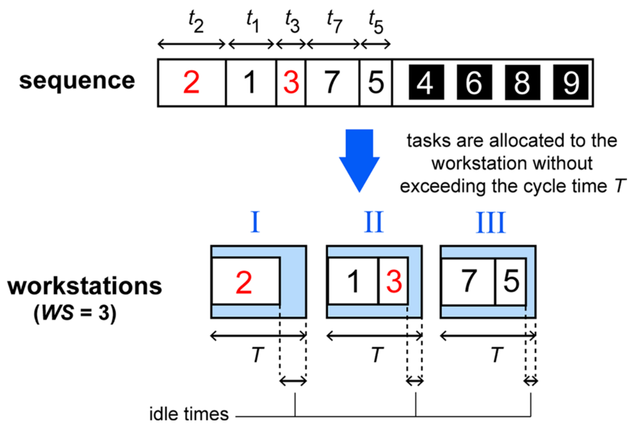

t_2 t_1 t_3 t_7 t_5
sequence
2 1 3 7 5 4 6 8 9
tasks are allocated to the workstation without exceeding the cycle time T
I
II
III
workstations
(WS = 3)
2
1 3
7 5
T
T
T
idle times

Fig. 3 Assignment of the tasks of a disassembly sequence to the workstations of the line

feasible task is assigned to a workstation. This determines the number of workstations required which is equal to 3 in this case. Formally, let *WS* be the number of workstations required by **z**, and let **w** be the assignment of the feasible tasks of **z** to the workstations, where each element w_k of **w** contains, for each feasible task k (where $k=1,\dots,k^*$), the identifier of the workstation to which task k is assigned. The level of balancing is calculated as follows

$$B(\mathbf{z}) = \sum_{j=1}^{WS} \Big(T - \sum_{h\,:\,w_k=j} t_h\Big)^2, \tag{17}$$

where the term in brackets is the idle time of workstation j.

Finally, the level of safety $S(\mathbf{z})$ is modeled in accordance with (7) as follows:

$$S(\mathbf{z}) = -\sum_{\substack{k=1,\dots,k* \\ d_k \neq 4}} k^{d_k}. \tag{18}$$

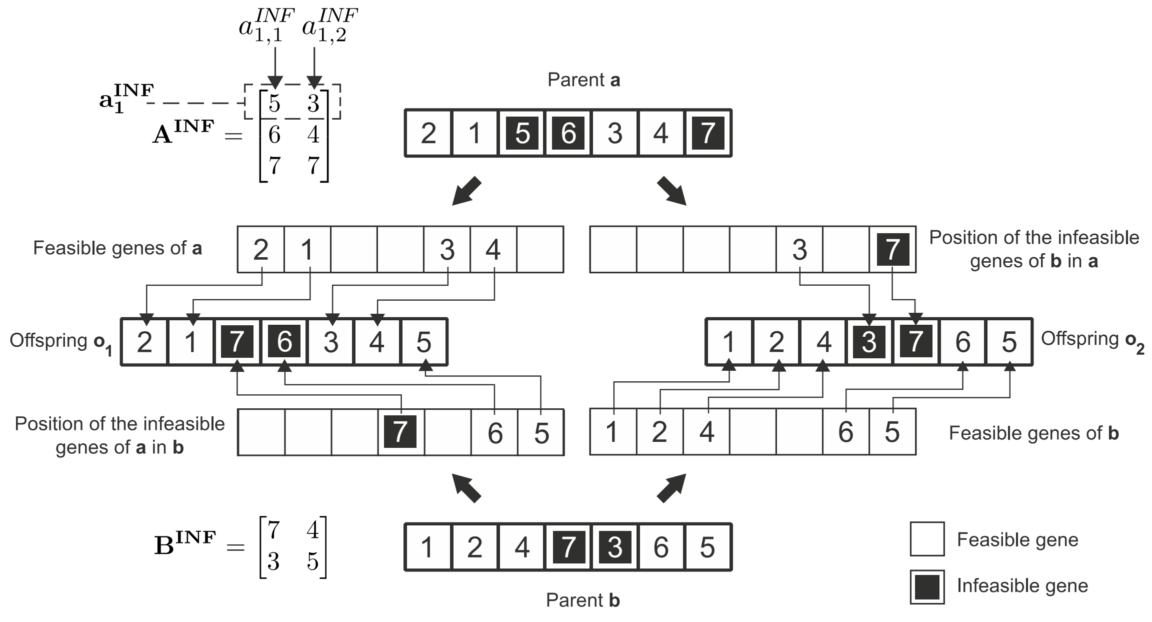
Feasibility improving crossover

The *feasibility improving crossover* (*FIX*) was designed for the proposed problem to promote an increase in the level of feasibility throughout the evolution. As the problem is NP-hard, and is also subject to constraints and extraction priorities, FIX was designed to avoid the use of penalty functions and further parameters. This helps the algorithm find solutions with diversified levels of feasibility, thereby promoting a deep exploration of the decision space to find more/better solutions.

Figure 4 considers two parents **a** and **b** from the mating pool (top and bottom of the figure), and shows the steps of FIX, by moving to the middle of the figure. Let U be the number of infeasible genes of **a**, and let $\mathbf{A^{INF}} \in \{1,\dots,N\}^{U\times 2}$ be a matrix with as many rows as the infeasible genes of **a**. In the i-th row $\mathbf{a_i^{INF}}$ of $\mathbf{A^{INF}}$, element $a_{i,1}^{\mathrm{INF}}$ contains the value of the i-th infeasible gene of **a**; element $a_{i,2}^{\mathrm{INF}}$ contains the position of that gene in **a**.

For example, looking at Fig. 4, the first row of $\mathbf{A^{INF}}$ is [5 3] because the value of the first infeasible gene of **a** corresponds to task 5, and this task is the third in the sequence. Starting from parents **a** and **b**, FIX generates two offspring $\mathbf{o}^1$ and $\mathbf{o}^2$. The left-hand side of Fig. 4 shows how FIX generates $\mathbf{o}^1$. First, the feasible genes of **a**—white background—are copied to $\mathbf{o}^1$. FIX then considers the position in **b** of the infeasible genes of **a**. In Fig. 4, genes 5, 6 and 7 are infeasible in **a** and come as seventh, sixth and fourth in **b**, respectively. FIX generates the *filling sequence*, i.e., ⟨7, 6, 5⟩, shown in Fig. 4 just under $\mathbf{o}^1$, where gene 7 precedes gene 6, and gene 6 precedes gene 5. The remaining genes of $\mathbf{o}^1$ take the values that are in the filling sequence, in the order.

Offspring $\mathbf{o}^2$ is generated by exchanging the roles of parents **a** and **b**, as shown on the right-hand side of Fig. 4. The pseudocode of FIX to generate $\mathbf{o}^1$ is in Algorithm 1.



$a_{1,1}^{INF}$ $a_{1,2}^{INF}$
$\mathbf{a_1^{INF}}$
$\mathbf{A^{INF}} = \begin{bmatrix} 5 & 3 \\ 6 & 4 \\ 7 & 7 \end{bmatrix}$
Parent **a**
2 1 5 6 3 4 7
Feasible genes of **a**
2 1 3 4
3 7
Position of the infeasible genes of **b** in **a**
Offspring $\mathbf{o_1}$
2 1 7 6 3 4 5
1 2 4 3 7 6 5
Offspring $\mathbf{o_2}$
Position of the infeasible genes of **a** in **b**
7 6 5
1 2 4 6 5
Feasible genes of **b**
$\mathbf{B^{INF}} = \begin{bmatrix} 7 & 4 \\ 3 & 5 \end{bmatrix}$
1 2 4 7 3 6 5
Parent **b**
Feasible gene
Infeasible gene

Fig. 4 Feasibility improving crossover applied to two parents, **a** (top of the figure) and **b** (bottom of the figure)

Algorithm 1: `fix_crossover`

```
1:  INPUT: a, b, $A^{INF}$
2:  OUTPUT: $o_1$
3:  begin
4:      put the feasible genes of a into $o_1$ position-wise
5:      $h \leftarrow 0$
6:      for each row $a_i^{INF}$ of $A^{INF}$
7:          find position $h$ of gene $b_h$ of b s.t. $b_h = a_{i,1}^{INF}$
8:          $k \leftarrow 0$
9:          for each row $a_{j \neq i}^{INF}$ of $A^{INF}$
10:             find position $k$ of gene $b_q$ of b s.t. $b_q = a_{j,1}^{INF}$
11:             if $q < h$ then
12:                 $k \leftarrow k + 1$
13:             end if
14:         end for
15:         $pos \leftarrow a_{k+1,2}^{INF}$
16:         $o_{pos}^1 \leftarrow a_{i,1}^{INF}$
17:     end for
18:     return $o_1$
19: end
```

Mutation

Depending on a mutation probability, the offspring generated by FIX undergo mutation. The algorithm uses the swap mutation, which selects genes in pairs and then exchanges them with each other. Swap mutation preserves most of the adjacent information, and is widely used in combinatorial problems (Sivanandam and Deepa 2007). As FIX, swap mutation does not generate duplicates in the chromosome, thus meeting constraint (3g).

Pseudocode and flowchart

The pseudocode of the algorithm is summarized in Algorithms 2, 3 and 4. In particular, Algorithms 2 and 3 describe the recombination and the fitness evaluation, whereas Algorithm 4 contains the main loop. The flowchart of the algorithm is shown in Fig. 5.

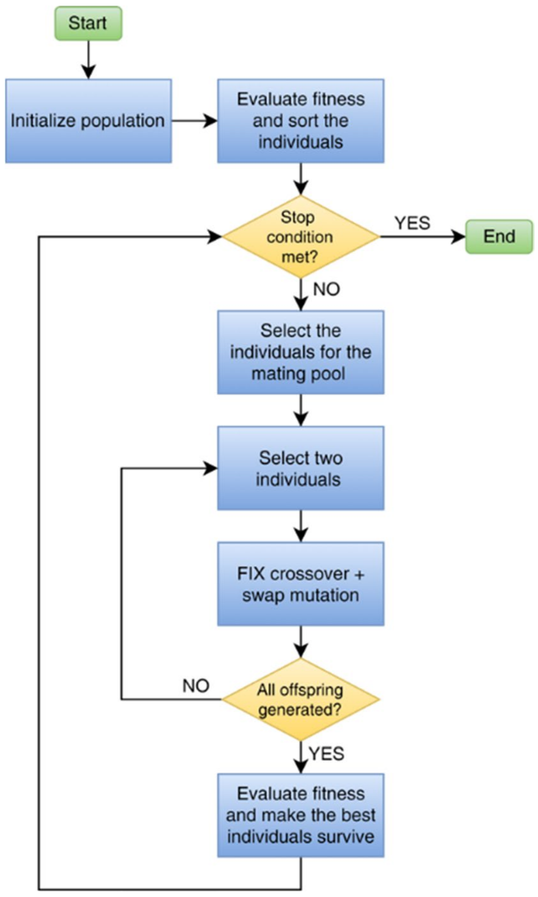

Start
Initialize population
Evaluate fitness and sort the individuals
Stop condition met?
YES
End
NO
Select the individuals for the mating pool
Select two individuals
FIX crossover + swap mutation
NO
All offspring generated?
YES
Evaluate fitness and make the best individuals survive

Fig. 5 Flowchart of the proposed algorithm

Algorithm 2: `recombine`

```
1:  INPUT: $z^1, z^2$
2:  OUTPUT: $o^1, o^2$
3:  begin
4:          $o^1 \leftarrow$ fix_crossover($z^i, z^j$)
5:          $o^2 \leftarrow$ fix_crossover($z^j, z^i$)
6:          if mutation occurs for $o^1$ then
7:              $o^1 \leftarrow$ mutate($o^1$)
8:          else if mutation occurs for $o^2$ then
9:              $o^2 \leftarrow$ mutate($o^2$)
10:         end if
11:         return $o^1, o^2$
12: end
```

Algorithm 3: `evaluate_fitness`

```
1:  INPUT: z, Y
2:  OUTPUT: f
3:  begin
4:      Determine the feasible genes, according to the
        precedence constraints in Y
5:      Rearrange the genes to get chromosome r
6:      Find the task $k^*$ that achieves the max profit
7:      Assign sequence $<r_1,\ldots,r_{k^*}>$ to the workstations
        without exceeding the cycle time $T$
8:          $f_{i1} \leftarrow$ level of feasibility of z
9:          $f_{i2} \leftarrow$ profit of z
10:         $f_{i3} \leftarrow$ level of smoothness of z
11:         $f_{i4} \leftarrow$ level of safety of z
12:     return f
13: end
```

Algorithm 4: Main loop

```
1:  INPUT: $N$, $T$, Y, $R_X$, $P_M$, $n$
2:  OUTPUT: $\mathcal{Z}$
3:  begin
4:      Generate a population $\mathcal{Z} = \{z^1,\ldots,z^n\}$ of $n$
        individuals by computing $n$ permutations of the $N$
        tasks
5:      Initialize the fitness matrix $F^{n\times4} = [f_{ij}]$ to zero
6:    repeat
7:      for each chromosome $z^i$
8:          $f^i \leftarrow$ evaluate_fitness($z^i$)
9:      end for
10:     Sort $z^1,\ldots,z^n$ according to Pareto dominance
11:     Select the fittest individuals in pairs
12:     for each pair $z^i$, $z^j$
13:         $\{o^1, o^2\} \leftarrow$ recombine($z^i$, $z^j$)
14:         $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{o^1, o^2\}$
15:     end for
16:     Divide the individuals of $\mathcal{Z}$ into subsets $g_d$, each
        containing individuals dominated by $d$ individuals
        in $\mathcal{Z}$, where $d \in \{1,\ldots, d^{MAX}\}$
17:     $d \leftarrow d^{MAX}$
18:     repeat
19:         $\mathcal{Z} \leftarrow \mathcal{Z} \setminus g_d$
20:         $d \leftarrow d - 1$
21:     until $\mathcal{Z} \setminus g_d \le N$
22:   until max_number_iterations is achieved
23:   return Z
24: end
```

Case studies

The algorithm was developed in MATLAB® on a virtual machine running Linux Debian OS, with 64 GB of RAM and four quad-core CPUs at 2.4 GHz.

The algorithm was tested on two case studies that entailed the disassembly of a TV monitor and of an air conditioner, respectively. These products were chosen as they are in high demand in recent years. Also, they contain dangerous parts that must be extracted as soon as possible in order to limit the exposure to hazardous materials, and then handled according to the regulations on hazardous waste disposal.

In both case studies, it was made the assumption of disassembling, in a disassembly line, a large quantity of EOL products taken from a collection facility.

Case study 1

This case study considers the disassembly of a TV monitor whose exploded view is in Fig. 6.

Dataset and configuration

The TV monitor requires the execution of 25 tasks to be entirely disassembled. Table 2 summarizes the information about these tasks. Each row i of the table relates to task i, whose identifier is in the first column. The second column contains the tasks that must be executed before task i. The third column contains the time an operator takes to perform task i, and the fourth column contains the identifier of the part(s) removed by task i. In this column, the parts removed all together are within brackets separated by commas. For example, task 4 removes a block made up of parts 24 and 25.

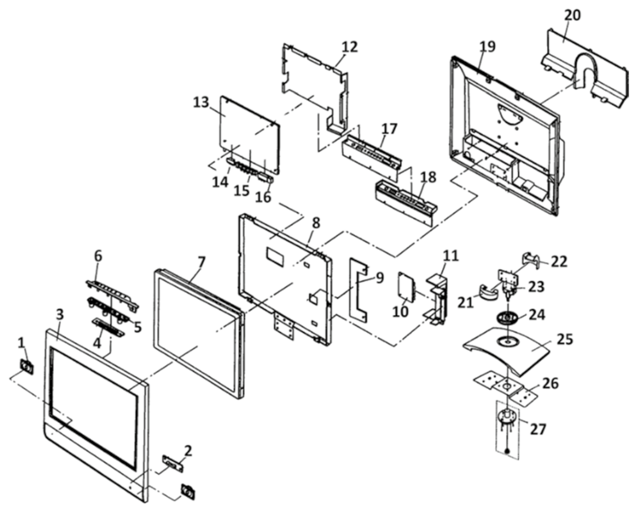


Fig. 6 Exploded view of the TV monitor

Table 2 Dataset of case study 1

Task i	Precedence constraints	Execution time t_i (min)	Removed parts	Safety level of removed parts	DC_i (€)	D_i (€)	R_i (€)	SP_i (€)
1	8	0.80	20	4	0.800	0.008	0.000	0.000
2	–	1.60	27	4	1.600	0.000	0.000	7.000
3	2	0.30	26	4	0.300	0.003	0.004	0.000
4	3	0.40	(24,25)	4	0.400	0.100	0.000	0.000
5	4	0.60	24–25	4	0.600	0.100	0.000	0.000
6	8	0.70	(21,22,23)	4	0.700	0.100	0.000	5.000
7	4,6	1.00	21–22–23	4	1.000	0.100	0.000	5.000
8	9,11,12,14	1.20	19	4	1.200	0.000	0.000	15.000
9	13	0.90	(17,18)	4	0.900	0.000	0.000	20.000
10	9	0.50	17–18	4	0.500	0.000	0.000	20.000
11	13	0.80	12	4	0.800	0.050	0.057	0.000
12	13	1.50	14–15–16	4	1.500	0.075	0.000	0.000
13	17	0.30	13	4	0.300	0.075	0.086	0.000
14	15	0.30	11	4	0.300	0.025	0.000	0.000
15	16	0.30	10	4	0.300	0.025	0.000	0.000
16	17	0.40	9	4	0.400	0.335	1.211	0.000
17	18	0.60	8	4	0.600	0.125	0.143	0.000
18	19,20,21	0.50	7	3	0.500	0.000	0.000	20.000
19	25	0.70	6	4	0.700	0.100	0.000	0.000
20	25	0.70	5	4	0.700	0.025	0.000	0.000
21	25	0.40	4	4	0.400	0.025	0.000	0.000
22	25	0.50	1	4	0.500	0.000	0.000	16.000
23	25	0.40	2	4	0.400	0.035	0.025	0.000
24	25	0.50	1	4	0.500	0.000	0.000	16.000
25	–	0.20	3	4	0.200	0.000	0.000	15.000

Instead, the parts that a task either removes from the product or separates from each other are separated by hyphen. For example, task 5 separates parts 24 and 25. The remaining columns contain the profit items DC_i, D_i, R_i, and SP_i explained in Sect. "Fitness function".

The TV monitor has one hazardous part, the LCD panel, denoted with 7, in Fig. 4. According to the classification based on the packing groups, the level of danger of the LCD is *low* (United Nations 2009). The task that removes the LCD panel, i.e., task 18, is thus assigned a level of danger equal to 3 (see Sect. "Problem").

The cycle time of the disassembly line—i.e., the maximum time that the product spends at each workstation—is determined on the basis of both the production rate and the efficiency of the line when disassembling units with similar wear conditions. The production rate was set to 32 products/h. This value was obtained by considering an existing industrial context where 64,000 EOL products per year are disassembled, in a line that works 50 weeks/year, with 5 shifts/week and 8 h/shift. An efficiency of 96% was finally considered for the disassembly line. The efficiency is the ratio of the time during which the line really operates to the total time. The total time considers set up times, downtimes due to maintenance, faults, and so on. The resulting cycle time was thus 1.90 min/product.

Parameters

The best parameter values were determined by using a two-step approach. A trial-and-error procedure was first carried out on the basis of heuristic considerations about the problem, and the fact that the mutation probability is generally set to values that are one/two orders of magnitude lower than the crossover probability. This guarantees a good compromise between exploration and exploitation (Holland 1975). The values that produced the best performance were:

- 200, 250 and 300 individuals;
- 0.6, 0.7 and 0.8 as crossover rates;
- 0.06, 0.07 and 0.08 as mutation probabilities.

All possible combinations of these values were tested as follows. Consider a combination. The algorithm was run 30 times and then the hypervolumes (Lebesgue measures) of the resulting Pareto fronts were measured. The average of these 30 hypervolumes (mean hypervolume) was finally calculated.

The mean hypervolume of each combination was compared to those obtained by the other combinations. The results were validated by using Student's *t* test with 95% confidence, assuming that the difference is due to chance, as the null hypothesis. The combination of values that obtained the highest number of rejections of the null hypothesis was chosen.

Results and discussion

A scatter plot of the Pareto front of case study 1 is shown in Fig. 7. Level of feasibility, profit, and smoothness are reported on Cartesian axes, whereas the fourth objective (level of safety) is represented by color: the cooler, the better.

The front in Fig. 7 was obtained by using 250 individuals, a crossover rate of 0.8 and a mutation probability of 0.08. These values were chosen as explained in the previous section.

An expert in the field typically chooses the solution to implement based on the product to disassemble, the current condition of the manufacturing industry, and the situation of the market of the recycled materials and spare parts. For example, consider a company whose equipment includes a disassembly line with 10 workstations, where managers look for solutions that maximize the profit, with the highest possible level of safety. One solution that may be selected is that enclosed in the red square in Fig. 7. The solution is an interesting compromise, as it requires 10 workstations and achieves the maximum profit (59.16 Euros) among the solutions characterized by the highest level of safety—the cooler the color, the safer the solution. For the sake of simplicity, the cost items such as transport, storage, and general expenses, were neglected. The high level of safety of the solution stems from performing the dangerous task (task 18) in the initial part of the sequence. This task is the first in station III, and is executed in a bit more than 3 min after the beginning, thereby making safe more than two-thirds of the disassembly process. Note that this dangerous task is performed as soon as possible according to the precedence constraints, as tasks 19, 20, and 21 require the preliminary execution of task 25 (see Table 2).

As explained in Sect. "Fitness function", the disassembly sequence only considers the feasible part of the chromosome, up to the task that achieves the maximum profit. Note that the level of feasibility of a sequence is generally higher than the number of tasks to perform in order to achieve the maximum profit. As Fig. 8 shows, the disassembly sequence performs 22 tasks out of 25 as the maximum profit—after removing the dangerous part (LCD panel)—is achieved at this point of the sequence. The optimal solution thus allows to quickly remove the dangerous part and to stop the disassembly process after achieving the maximum profit, without removing parts 2, 20, 24 and 25 (carried out by tasks 23, 1 and 5 respectively). These tasks remove buttons or small plastic ferrules and brackets, and mainly generate costs for disassembly and disposal (see Table 2).

The partial process carried out by the solution is coherent with EOL disassembly, where companies are typically not interested in removing all the parts from the product—these parts may also be deteriorated—but aim to recover as many parts as possible to reuse, and valuable materials.

Case study 2

The second case study concerns the disassembly of the air conditioner shown in Fig. 9.

Dataset and configuration

The complete disassembly of the air conditioner requires the execution of 64 tasks, whose details are in Table 3.

As can be seen from the table, the product has three hazardous parts, i.e., parts 3, 40 and 58, which correspond to a dividing wall, the cabinet, and the compressor,

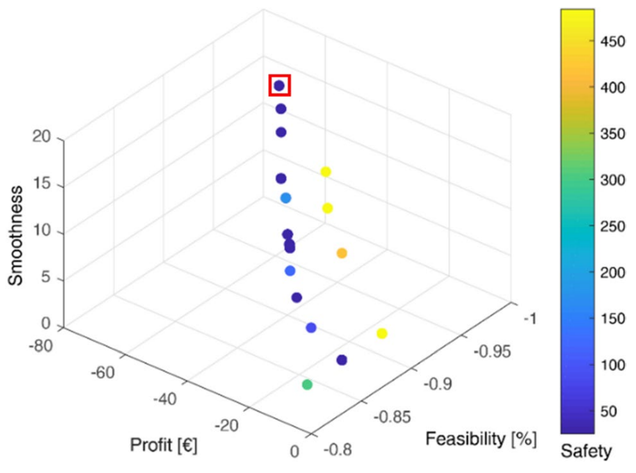


Fig. 7 Pareto front obtained in case study 1. The solution inside the square is (−0.88, −59.16, 16.32, 22), whose elements are level of feasibility, profit, smoothness, and safety level, in the order

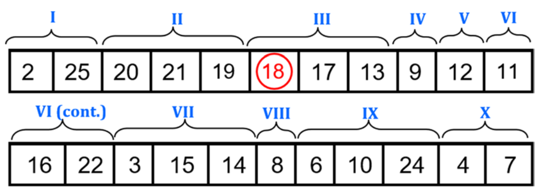


Fig. 8 Disassembly sequence, and assignment of the tasks to the workstations required by the solution inside the square, in Fig. 7

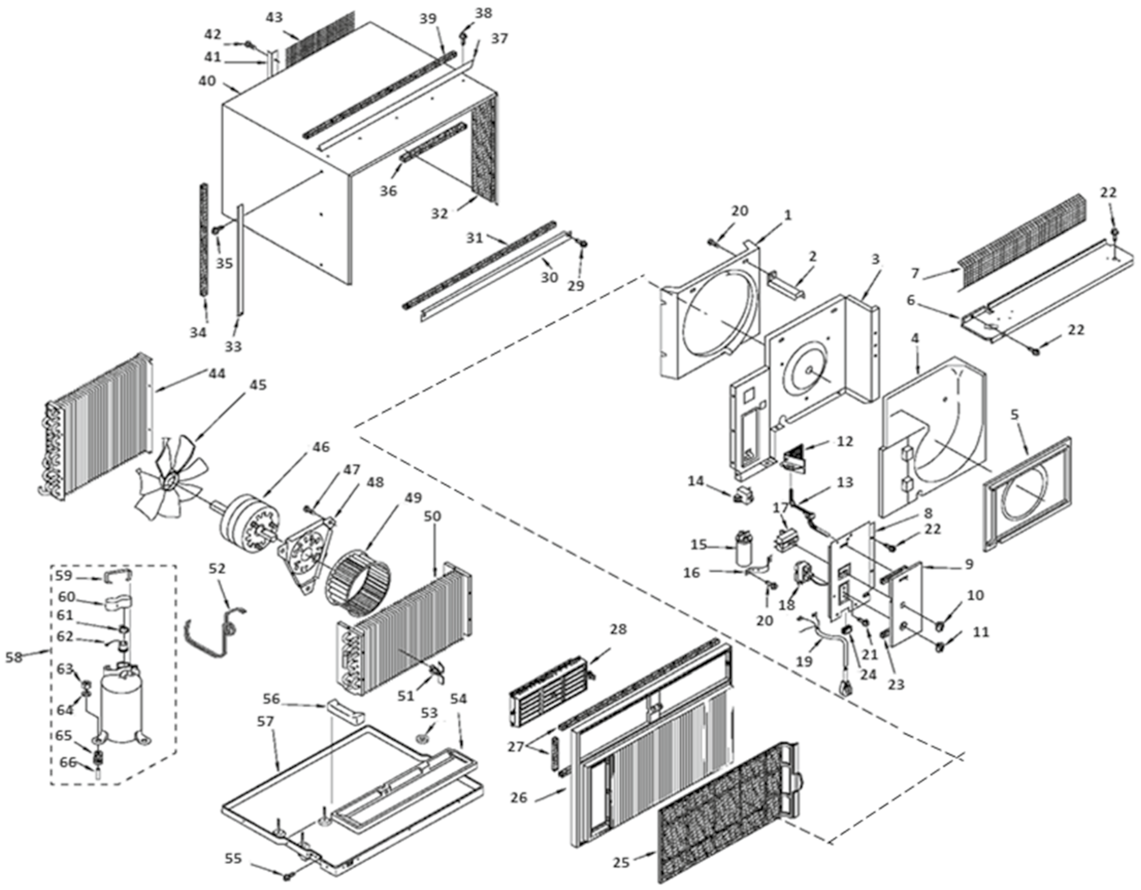


Fig. 9 Exploded view of the air conditioner

respectively. Both the dividing wall and the cabinet contain asbestos for thermal insulation, whereas the compressor contains hydrofluorocarbon (HFC), a refrigerant gas with perchloroethylene.

According to the classification based on the packing groups (United Nations 2009), the level of danger of asbestos is *medium*, so the tasks that remove the dividing wall and the cabinet have a level of danger of 2. Instead, the level of danger of perchloroethylene is *low*. The task that extracts the compressor has thus a level of danger of 3. The cycle time is set to 2.18 min. This value was established by assuming an annual collection of 53,000 products that are disassembled in a line characterized by the same parameters as those of case study 1.

Results and discussion

The Pareto front obtained in this case study is shown in Fig. 10. The algorithm was set up with 300 individuals, a crossover rate equal to 0.8, and a mutation rate of 0.08.

Let us consider a company with a 16-workstation disassembly line already installed. In this case, decision makers look for a solution that maximizes profit and safety, requiring at most 16 workstations. The solution to implement may be that in the red square in Fig. 10 and shown in Fig. 11 along with the assignment of its tasks to the workstations. As Fig. 11 shows, this solution requires 16 workstations, is characterized by the highest level of safety in the Pareto front, and achieves the best level of balancing and the highest profit (224.89 Euros).

Figure 11 also shows that the solution removes the dangerous parts of the product without performing a complete disassembly.

The level of feasibility is 77.08%: more than three quarters of the product is thus disassembled. As said earlier, in EOL dismantling it is generally not worth performing a complete disassembly, as the key goal is to achieve the maximum profit from reselling the valuable parts and materials recovered, while ensuring the highest possible level of safety.

As in the previous case study, the solution found by the algorithm stops the disassembly process when achieving the maximum profit after removing all the dangerous parts. These parts are dismounted by tasks 22, 35 and 51, which remove the dividing wall, the cabinet, and the compressor,

Table 3 Dataset of case study 2

Task i	Precedence constraints	Execution time t_i (min)	Removed parts	Safety level of removed parts	DC_i(€)	D_i (€)	R_i (€)	SP_i (€)
1	–	0.90	22	4	0.600	0.020	0.034	0.000
2	–	0.45	20	4	0.300	0.020	0.034	0.000
3	–	0.30	10–11	4	0.200	0.025	0.000	0.000
4	3	0.15	9	4	0.100	0.038	0.056	0.000
5	4	0.30	21	4	0.200	0.003	0.004	0.000
6	4	1.05	23	4	0.700	0.030	0.000	0.000
7	4	0.60	25	4	0.400	0.013	0.014	0.000
8	1,5,6	1.00	8	4	0.667	0.065	0.007	0.000
9	8	0.20	17	4	0.133	0.018	0.029	0.000
10	8	0.30	18	4	0.200	0.018	0.029	0.000
11	2	0.20	16	4	0.133	0.000	0.000	0.000
12	11,13	0.40	15	4	0.267	0.025	0.055	0.000
13	–	0.40	14	4	0.267	0.023	0.038	0.000
14	8	0.80	13	4	0.533	0.000	0.000	7.000
15	14	0.15	12	4	0.100	0.013	0.021	0.000
16	–	1.20	19	4	0.800	0.030	0.018	0.000
17	1	0.60	6–7	4	0.400	0.475	0.798	0.000
18	–	0.30	5	4	0.200	0.000	0.000	18.000
19	18	0.15	4	4	0.100	0.625	0.983	0.000
20	2	0.15	1	4	0.100	0.000	0.000	50.000
21	20	0.20	2	4	0.133	0.015	0.021	0.000
22	12,15,16,19,21	0.15	3	2	0.100	0.625	0.983	0.000
23	–	1.30	42	4	0.867	0.020	0.034	0.000
24	23	0.30	41	4	0.200	0.005	0.006	0.000
25	23	0.15	43	4	0.100	0.003	0.004	0.000
26	–	1.30	38	4	0.867	0.003	0.004	0.000
27	26	0.15	39	4	0.100	0.125	0.000	0.000
28	26	0.30	37	4	0.200	0.005	0.006	0.000
29	–	1.30	35	4	0.867	0.003	0.004	0.000
30	29	0.15	34	4	0.100	0.125	0.000	0.000
31	29	0.30	33	4	0.200	0.005	0.006	0.000
32	–	1.30	29	4	0.867	0.003	0.004	0.000
33	32	0.30	30	4	0.200	0.005	0.006	0.000
34	32	0.15	31	4	0.100	0.125	0.000	0.000
35	23,26,29,32,60	0.25	40	2	0.167	0.200	0.336	0.000
36	35	0.15	36	4	0.100	0.063	0.000	0.000
37	20,33,34	0.40	25	4	0.267	0.013	0.014	0.000
38	37	0.15	26	4	0.100	0.225	0.351	0.000
39	38	0.15	27	4	0.100	0.375	0.000	0.000
40	37	0.15	28	4	0.100	0.000	0.000	20.000
41	35	0.20	51	4	0.133	0.004	0.006	0.000
42	41,50	1.00	50	4	0.667	0.000	0.000	65.000
43	35	1.40	(45,46,47,48,49)	4	0.933	0.308	0.004	65.000
44	50	1.40	44	4	0.933	0.000	0.000	80.000
45	43	0.20	49	4	0.133	0.005	0.000	0.000
46	43	0.30	47	4	0.200	0.003	0.004	0.000
47	45,46	0.25	48	4	0.167	0.000	0.000	15.000
48	46	0.20	46	4	0.133	0.000	0.000	50.000
49	43	0.10	45	4	0.067	0.300	0.000	0.000

Table 3 (continued)

Task i	Precedence constraints	Execution time t_i (min)	Removed parts	Safety level of removed parts	DC_i(€)	D_i (€)	R_i (€)	SP_i (€)
50	35	1.30	52	4	0.867	0.000	0.000	10.000
51	50	1.00	58	3	0.667	0.253	0.438	0.000
52	51	0.15	59	4	0.100	0.000	0.000	0.000
53	52	0.15	60	4	0.100	0.000	0.000	0.000
54	53	0.15	61	4	0.100	0.000	0.000	0.000
55	54	0.15	62	4	0.100	0.000	0.000	0.000
56	51	0.15	63	4	0.100	0.000	0.000	0.000
57	56	0.15	64	4	0.100	0.000	0.000	0.000
58	51	0.15	66	4	0.100	0.000	0.000	0.000
59	58	0.15	65	4	0.100	0.000	0.000	0.000
60	–	1.30	55	4	0.867	0.020	0.034	0.000
61	42,43,44,51	0.15	56	4	0.100	0.005	0.006	0.000
62	42,43,44,51	0.40	53	4	0.267	0.003	0.004	0.000
63	62	0.25	54	4	0.167	0.063	0.105	0.000
64	60,63	0.10	57	4	0.067	0.150	0.252	0.000

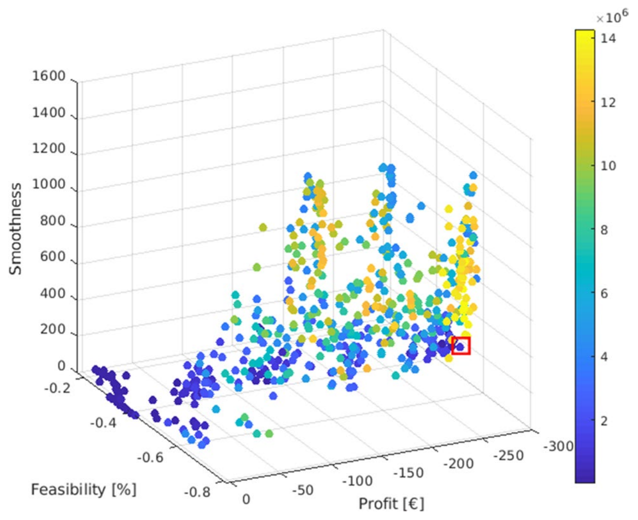


Fig. 10 Pareto front for case study 2. The solution in the red square is (−0.7708, −224.89, 382.98, 436752), whose elements are level of feasibility, profit, smoothness, and level of safety, in the order

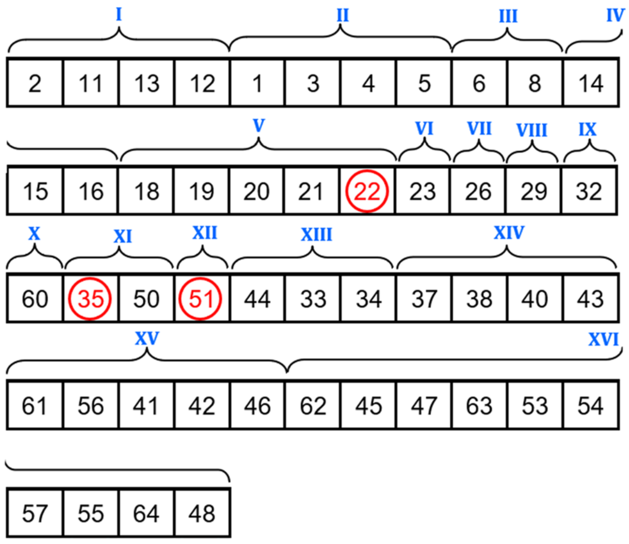


Fig. 11 Disassembly sequence and assignment of the tasks to the workstations required by the solution circled in Fig. 8

respectively. As can be seen in Fig. 11 with the help of Table 3, the algorithm does not consider the tasks that generate more costs than revenues. For example, tasks 27, 30, 36 and 58, are not part of the solution. These tasks remove low-value parts and take time, thus mainly generating costs. More in detail, tasks 27, 30 and 36 simply remove the top, side and front bars from the cabinet (parts 39, 34 and 36 in Fig. 9) respectively, whereas task 58 only unscrew a compressor fixing screw (part 66 in Fig. 9).

The solution found by the algorithm also achieves a high level of safety. As shown in Fig. 11, the first two dangerous tasks that are performed (tasks 22 and 35) remove the most hazardous parts: the dividing wall and the cabinet (see Table 3). The exposure to asbestos dust is thus more than halved. After removing the parts with asbestos, the time that remains before completing the disassembly is more than half of the total disassembly time. The product no longer contains the two parts in asbestos, which are the ones that generate the highest risk. During more than half of the disassembly process, workers are thus exposed to a low risk, due to part 58 whose Packing Group is equal to 3, i.e., *low danger* (see Table 3). Moreover, the last dangerous part is extracted just one task after removing the dividing wall: the rest of the process (the tasks in workstations XIII, XIV, XV, and XVI in Fig. 11) is thus completely safe. Finally, the smoothness is low (382.98), the disassembly times are thus balanced.

The solution obtained has thus a high quality as it achieves the maximum profit and quickly extracts the dangerous parts in decreasing order of level of danger. Note that this solution is the safest way to disassemble the product. The extraction of the dangerous parts cannot be anticipated because of the precedence constraints (see Table 3).

Performance evaluation and comparisons

The proposed algorithm was compared to MOPSO (Coello Coello and Lechuga 2002), MOEA/D (Zhang and Li 2007), SPEA2 (Zitzler et al. 2002), and NSGA-III (Deb and Jain 2014). These algorithms were chosen because they are among the most recent and efficient algorithms for complex MOO problems in the literature.

The performance evaluation was carried out in two steps. The results were first compared to those obtained by the other algorithms by considering the disassembly of the TV monitor. A further evaluation was carried out by considering the disassembly of the air conditioner. As this second product is made up of almost three times more parts than the monitor, and is characterized by a larger set of precedence constraints. The second part of the performance evaluation thus proved the efficiency of the algorithm in complex scenarios.

From an operational point of view, a total of 30 executions of the proposed algorithm were run. The hypervolume of the Pareto front obtained by each execution was measured by normalizing the objective functions in [0,1]. The average of these hypervolumes was finally calculated. The performance of the other algorithms was measured according to the same procedure. As problem (3a)–(3h) entails maximizing the objectives, the wider the hypervolume, the better.

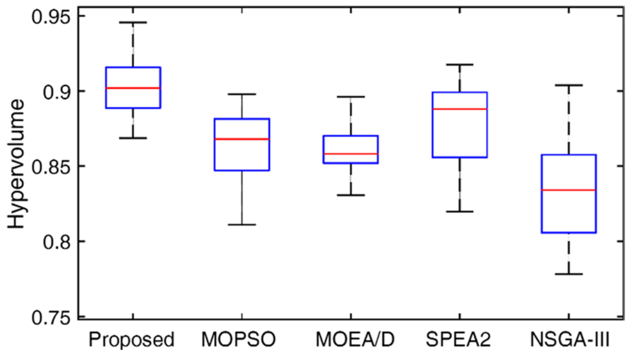


Fig. 12 Box plot of the hypervolumes obtained by the proposed algorithm and by the algorithms used for comparison in case study 1

Figures 12 and 13 show the box plots of the hypervolumes obtained by the proposed algorithm, and by MOPSO, MOEA/D, SPEA2, and NSGA-III for both case studies. As the figures show, the proposed algorithm achieved better performance than the others, in terms of hypervolume.

On average, the proposed algorithm obtained Pareto fronts 25.2% wider than those obtained by the compared algorithms in the more complex case study. The algorithm thus better explores the solution space and obtains more solutions and/or solutions with higher values of the fitness function.

A statistical validation of the results was carried out by using Student's *t* test. Finally, the execution times for both case studies are shown in Figs. 14 and 15. As can be seen from the figures, the proposed algorithm is characterized by an execution time that is similar to those of the algorithms used for comparison. As a concluding remark, it is worth noticing that, on average, in the more complex case study, the proposed algorithm is faster. The algorithm thus maintains good performance when increasing the complexity of the problem.

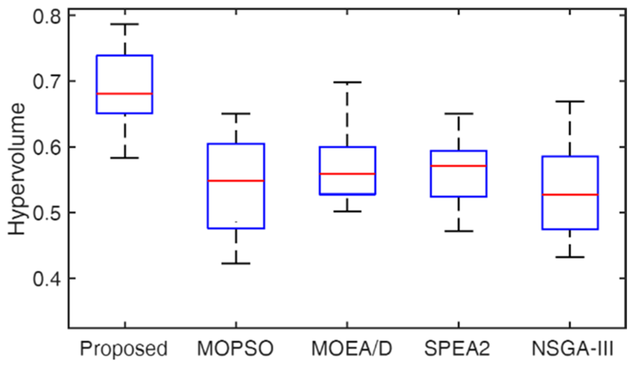


Fig. 13 Box plot of the hypervolumes obtained by the proposed algorithm and by the algorithms used for comparison in case study 2

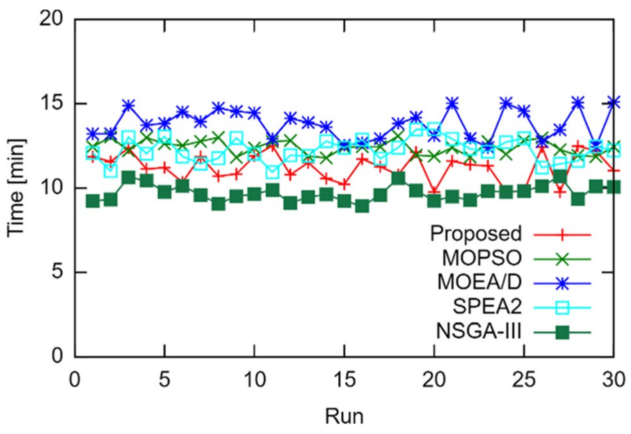


Fig. 14 Execution times of the 30 runs of all the algorithms executed throughout the experiments of the first case study

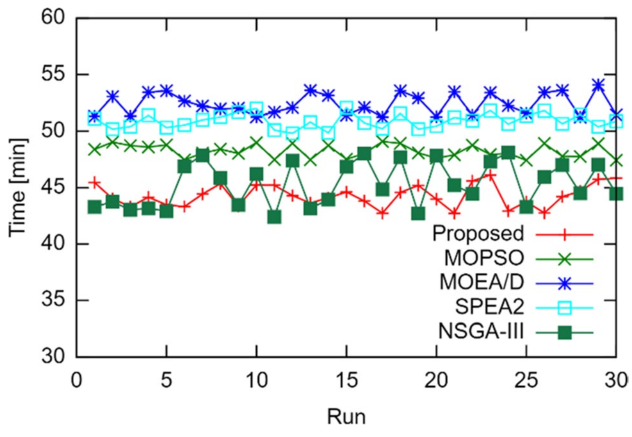


Fig. 15 Execution times of the 30 runs of all the algorithms throughout the experiments of the second case study

Conclusions

This paper has presented a new formulation of DLBP aimed at efficiently disassembling EOL products in safety, along with a GA designed for the proposed optimization problem.

The efficiency stems from minimizing the number of workstations, balancing the idle times, and maximizing the profit. The safety is maximized by introducing different priorities based on the level of danger of each dangerous part, contrary to the previous formulations that only execute the dangerous tasks before the non-dangerous. The proposed formulation thus makes it possible to consider the actual risk to which workers are exposed while disassembling products, task after task, as the extraction priority of each dangerous part is increased the longer the part remains in the product during the disassembly process, and the more dangerous the material of the part is.

The algorithm was tested by considering two case studies where a TV monitor and an air conditioner were disassembled, respectively. The results showed that the system generates efficient and safe configurations to set up a disassembly line. The experiments also showed that considering the precedence relationships as an objective—instead of as a constraint—allows a wider exploration of the solution space.

The Pareto front returned by the algorithm, with its multiple optimal solutions, makes it possible for the experts in the field to choose the best solution to the problem, by considering the current situation of the manufacturing industry.

Possible improvements of the proposed algorithm are considering non-deterministic disassembly times and aspects that deal with current open issues, such as the possible presence of destructive operations during the disassembly process.

Acknowledgements This work was supported by the Italian Ministry of Education and Research (MIUR) in the framework of the CrossLab project (Departments of Excellence).

References

Al_Janabi, S., Alhashmi, S., & Adel, Z. (2019). Design (More-G) model based on renewable energy & knowledge constraint. In *International conference on big data and networks technologies* (pp. 271–295). Springer, Cham.

Alkaim, A. F., & Al_Janabi, S. (2019). Multi objectives optimization to gas flaring reduction from oil production. In *International conference on big data and networks technologies* (pp. 117–139). Springer, Cham.

Bentaha, M. L., Battaïa, O., Dolgui, A., & Hu, S. J. (2014). Dealing with uncertainty in disassembly line design. *CIRP Annals-Manufacturing Technology, 63*(1), 21–24.

BoussaïD, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences, 237,* 82–117.

Chern, C. C., Wang, H. M., & Huang, K. L. (2015). A heuristic master planning algorithm for recycling supply chain management. *Journal of Intelligent Manufacturing, 28,* 1–19.

Coello Coello C. A., & Lechuga M. S. (2002). MOPSO: A proposal for multiple objective particle swarm optimization. In *Proceedings of the 2002 congress on evolutionary computation. CEC'02* (Vol. 2, pp. 1051–1056) Honolulu, USA.

Colledani, M., Copani, G., & Tolio, T. (2014). De-manufacturing systems. *Procedia CIRP, 17,* 14–19.

Colledani, M., & Tolio, T. (2013). Integrated process and system modelling for the design of material recycling systems. *CIRP Annals-Manufacturing Technology, 62*(1), 447–452.

Dalle Mura, M., & Dini, G. (2017). A multi-objective software tool for manual assembly line balancing using a genetic algorithm. *CIRP Journal of Manufacturing Science and Technology, 19,* 72–83.

Deb, K. (2014). Multi-objective optimization. In E. Burke, G. Kendal (Eds.), *Search methodologies*. Boston, MA: Springer. https://doi.org/10.1007/978-1-4614-6940-7_15.

Deb, K., & Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation, 18*(4), 577–601.

Ding, L. P., Feng, Y. X., Tan, J. R., & Gao, Y. C. (2010). A new multi-objective ant colony algorithm for solving the disassembly line balancing problem. *The International Journal of Advanced Manufacturing Technology, 48*(5–8), 761–771.

Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., & Cosar, A. (2019). A survey on new generation metaheuristic algorithms. *Computers & Industrial Engineering, 137,* 106040.

Habibi, M. K. K., Battaïa, O., Cung, V. D., & Dolgui, A. (2014). Integrated procurement–disassembly problem. In *IFIP international conference on advances in production management systems* (pp. 482–490). Springer, Berlin.

Holland, J. H. (1975). *Adaptation in natural and artificial systems. An introductory analysis with application to biology, control, and artificial intelligence*. Ann Arbor: University of Michigan Press.

Huang, C. C., Liang, W. Y., & Yi, S. R. (2015). Cloud-based design for disassembly to create environmentally friendly products. *Journal of Intelligent Manufacturing, 28,* 1–16.

Kalayci, C. B., & Gupta, S. M. (2013). A particle swarm optimization algorithm with neighborhood-based mutation for sequence-dependent disassembly line balancing problem. *The International Journal of Advanced Manufacturing Technology, 69*(1–4), 197–209.

Kalayci, C. B., Polat, O., & Gupta, S. M. (2016). A hybrid genetic algorithm for sequence-dependent disassembly line balancing problem. *Annals of Operations Research, 242*(2), 321–354.

Lazzerini, B., & Pistolesi, F. (2014). Classifying workers into risk sensibility profiles: A neural network approach. In: *2014 European modelling symposium*, Pisa, pp. 33–38. https://doi.org/10.1109/EMS.2014.24.

Lazzerini, B., & Pistolesi, F. (2018a). Multiobjective personnel assignment exploiting workers' sensitivity to risk. *IEEE Transactions on Systems, Man, and Cybernetics: Systems, 48*(8), 1267–1282.

Lazzerini, B., & Pistolesi, F. (2018b). An integrated optimization system for safe job assignment based on human factors and behavior. *IEEE Systems Journal, 12*(2), 1158–1169.

Liu, J., Zhou, Z., Pham, D. T., Xu, W., Yan, J., Liu, A., et al. (2018). An improved multi-objective discrete bees algorithm for robotic disassembly line balancing problem in remanufacturing. *The International Journal of Advanced Manufacturing Technology, 97,* 1–26.

Marler, R. T., & Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization, 26*(6), 369–395.

McGovern, S. M., & Gupta, S. M. (2006a). Ant colony optimization for disassembly sequencing with multiple objectives. *The International Journal of Advanced Manufacturing Technology, 30*(5), 481–496.

McGovern, S. M., & Gupta, S. M. (2006b). *The disassembly line: Balancing and modeling*. New York: McGraw-Hill. **ISBN 9780071626057**.

Mete, S., Çil, Z. A., Ağpak, K., Özceylan, E., & Dolgui, A. (2016). A solution approach based on beam search algorithm for disassembly line balancing problem. *Journal of Manufacturing Systems, 41,* 188–200.

Özceylan, E., Kalayci, C. B., Güngör, A., & Gupta, S. M. (2018). Disassembly line balancing problem: A review of the state of the art and future directions. *International Journal of Production Research, 57,* 1–23.

Pistolesi, F., & Lazzerini, B. (2019). TeMA: A tensorial memetic algorithm for many-objective parallel disassembly sequence planning in product refurbishment. *IEEE Transactions on Industrial Informatics, 15*(6), 3743–3753.

Pistolesi, F., Lazzerini, B., Dalle, Mura M., & Dini, G. (2018). EMOGA: A hybrid genetic algorithm with extremal optimization core for multiobjective disassembly line balancing. *IEEE Transactions on Industrial Informatics, 14*(3), 1089–1098.

Rickli, J. L., & Camelio, J. A. (2013). Multi-objective partial disassembly optimization based on sequence feasibility. *Journal of Manufacturing Systems, 32*(1), 281–293.

Riggs, R. J., Battaïa, O., & Hu, S. J. (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.

Sivanandam, S. N., & Deepa, S. N. (2007). *Introduction to genetic algorithms*. Berlin: Springer. **ISBN 9783540731894**.

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

United Nations. Committee of Experts on the Transport of Dangerous Goods (2009). *Recommendations on the transport of dangerous goods: Model regulations*. Vol. 1, 16th revised version. United Nations Publications, ISBN 978-92-1-139136-7.

Vongbunyong, S., Kara, S., & Pagnucco, M. (2013). Application of cognitive robotics in disassembly of products. *CIRP Annals-Manufacturing Technology, 62*(1), 31–34.

Zhang, Q., & Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation, 11*(6), 712–731.

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

Zhang, X. F., Yu, G., Hu, Z. Y., Pei, C. H., & Ma, G. Q. (2014). Parallel disassembly sequence planning for complex products based on fuzzy-rough sets. *The International Journal of Advanced Manufacturing Technology, 72*(1–4), 231–239.

Zitzler, E., Laumanns, M., & Thiele, L. (2002). *SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization, from evolutionary methods for design, Optimisation and Control*. Barcelona: CIMNE.