A Method for Dealing with Multi-Objective Optimization Problem of Disassembly Processes

L. Duta[†] Florin Gh. Filip[*] J. M. Henrioud[+]

[†]Computer Science Department
Valahia State University
18-24, Unirii Av., Targoviste,
Romania 40200
duta@valahia.ro

[*]Research Institute for Informatics
8-10, Maresal Averescu, Bucharest
Romania, 47000
ffilip@acad.ro

[+]Laboratoire d'Automatique de Besançon
Rue Alain Savary, 25
Besançon, France 25000
henrioud@ens2m.fr

Abstract

Disassembly of manufactured goods induces both disassembly costs and revenues from the parts saved by the process. Thus, a good trade-off has to be found that depends, both on the depth of the disassembly, and on the sequence of operations. This optimization problem depends upon the structure of the disassembly system: if it is made of a single workstation, the costs depend mainly upon the process duration. If the system is a line, the costs depend mainly upon the line balancing, all the more if it is highly manual. In this paper the authors consider the line structure and propose an algorithm which will allow to find a disassembly sequence that optimizes a very simple function integrating the income from the parts, the material produced by the process and the cycle time of the disassembly line. An example is given to illustrate the proposed algorithm.

1 Introduction

The disassembly process is the main stage in the recycling of manufactured products at the end of their life. The aim of this process is to extract the reusable parts of the product, materials to recycle, as well as the dangerous materials. Products submitted to the disassembly process are out of use so we have to take into account their physical state.

Disassembly is a non-destructive technique: it implies the extraction of the desired components and/or materials. If parts are not reusable after reconditioning, partial or total destructive operations are applied: drilling, cutting, wrenching, and shearing. These techniques are used in view of material or energy recovery.

The aim of the valorization process is to save the value of parts and materials by repairing and recovering operations.

The major problem in a disassembly process is the occurrence of uncertainty in the possibility of components separations [1]. Deteriorations and deformations of some elements, absence of one or more components, presence of corrosion and rust are perturbations often encountered in the disassembly process. Therefore, some operations cannot be carried out due to the physical degradations of the components and other operations are not performed if they are not profitable. So, a choice has to be made, between applying an alternative disassembly destructive operation (dismantling), and abandoning the disassembly procedure [2].

In this context one must decide how deep the disassembly process can go so as to maximize the total profit obtained by the valorization of the components.

In the recycling industry, manufacturers are also interested in gaining time and money using an optimal distribution of the disassembly tasks on workstations that provides a maximal value for this total profit in less time. So, the manufacturers from the disassembly industry are looking to accomplishing three major objectives: lower costs, greater outcomes and well-balanced lines [3].

2 The Optimization Problem

In this paper the authors address disassembly lines where the cycle-time is not simply the sum of all operative and logistic times but it also depends strongly upon the line balancing. Disassembly systems made of a single workstation are an easier problem and have already been addressed, to among others, by the authors of this paper.

The objective is to find the most profitable disassembly sequence taking into account, on one hand - the end-of-life options for each part or subassembly of a given product [4], and on the other hand - the operational times for a given assignment of the tasks on the disassembly workstations.

For each disassembly sequence experts can establish the revenue associated to each component or subassembly according to its end-of-life destination [5]. The final revenue obtained after performing a given disassembly sequence is the sum of the partial revenues:

$$r = \sum_i r_i \qquad i = 1..m \tag{1}$$

where m is the number of final components or subassemblies obtained after the disassembly process.

Usually, the assignment of tasks to the disassembly workstations is made in such a way that it minimizes the difference between operational times on each of these stations. This means that the disassembly line is well-balanced [6].

Having equal working times on stations means equal costs of the work done on each of them. That is why the operational costs are usually considered proportional to the operational times.

One function that gives the imbalance between stations is given by the following formula:

$$F = \sum_{i=1}^{n} \left(\sum_{j \in \{tasks\, W_i\}} t_j - t_{cy} \right)^2 \qquad (2)$$

Where n is the number of workstations, t_{cy} is the cycle time and W_i is the workstation i, with $i = 1..n$.

Minimizing the value of this function leads to a well-balanced disassembly line [6].

The cycle time can be defined like the operational time of the slowest workstation on the line [3], [7].

$$t_{cy} = \max_{W_i} \sum_{j \in (tasks\ on\ W_i)} t_j \qquad (3)$$

To minimize the cycle time allows minimizing the operational costs.

So, one has to find one cost function mixing both disassembly costs and revenues. This is typically a multicriteria problem that can be addressed by minimizing (r-λF), λ being a weight. The problem of computing λ has replaced the initial one.

To avoid this problem, the authors propose the following objective function:

$$f = \frac{r}{t_{cy}} \qquad (4)$$

Where r is taken from (1) and t_{cy} from (3).

One can argue that this function uses implicitly some hidden weighting between its two arguments. This is true, but this weighting has a meaning that cannot be found in the classical formulation (r-λF), the objective function f given by (4) being the **income flow of the disassembly system**.

A real drawback of function f is that there are situations when a poor balancing according to (2) yields a lower cycle time. That's why, one should consider the best tasks assignment that balances the line and then calculate the maximum value of f taking into account the value of the cycle time obtained for a well-balanced line. However, this is obviously the good objective function either when there is only one workstation or when the disassembly system is automated. Even in the case of a manual disassembly line, f does not ignore the balancing problem since it tends to minimize the cycle time.

In the following section we shall consider only paced, linear, mono-product disassembly lines, with an infinite supply of products, so that the line can never be starved.

3 Dealing with the Optimization Problem

The workstations are already placed on the line in an increasing order of their destroying effect. The disassembly or the dismantling times are assumed to be deterministic and known.

The disassembly sequences are supposed to be known and memorized as a single Petri Net.

Notations are given hereafter:

- m - number of tasks;
- n - number of stations;
- d - number of possible disassembly sequences;
- $t_{cy(k)}$ - cycle time for disassembly sequence k;
- t_j - duration of task j;
- T_k - vector of the tasks' durations t_j for disassembly sequence k;
- T - the matrix of the T_k vectors;
- R - the vector of the global revenues R^k for disassembly sequence k;
- $M = \{m_{ij}\}, i = 1..n, j = 1..m$ is the matrix of possible tasks' assignments with:

$$m_{ij} = \begin{cases} 1 & if\ \ task\ \ j\ \ may\ be\ assigned\ to\ station\ i \\ 0 & otherwise \end{cases}$$

To deal with the optimization problem formulated before we have to find the assignment matrices of the tasks to satisfy the assessment formulated in section 2**.**

Let $S=\left\{s_{ij}\right\}, i=1..n, j=1..m$ be the matrix which gives the effective assignment, that is:

$$s_{ij}=\begin{cases}1 & \text{if task } j \text{ is assigned to workstation } i\\ 0 & \text{otherwise}\end{cases}$$

This matrix is subjected to the following constraints:

$$s_{ij}\in\{0,1\} \tag{5}$$

$$\sum_i s_{ij}=1 \tag{6}$$

$$\sum_{k=1}^{n} k\cdot s_{ki}-\sum_{k=1}^{n} k\cdot s_{kj}\le 0 \tag{7}$$

Constraint (5) is known as the **non-divisibility constraint** that does not allow a task to be assigned to more than one station. Constraint (6) is the **assignment constraint** and it requires that each task be assigned to *exactly* one station. Constraint (7) is the **precedence constraint** that invokes technological order so that if task *i* is to be done before task *j* (*i*<*j*), then *i* cannot be assigned to a station downstream from task *j* [7].

The objective is then to determine the matrices S that meet all these three constraints and which maximize the objective function *f* from (4).

4 The Proposed Algorithm

The matrices S that verify equations (5), (6) and (7) are determined by a classic procedure that calculates all the possible assignments of the tasks on workstations using the notion of cartesian product and the backtracking programming method. The *algorithm* is:

Step_1
Read the input data *m, n, d, M, T,* and *R.*

Step_2
Generate the cartesian product

$$A_1\times A_2\times\ldots A_m=\left\{(e_1,\ldots,e_m)_1,\ldots,(e_1,\ldots,e_m)_p \,\middle|\, e_i\in A_i, i=1..m\right\}$$

Where A_i = {is the set of the workstation indexes that can perform task *i* } and *p* the number of elements;

Step_3
For each $v=1..p$
Begin
Generate_ S_v (n, m, e_i);
For S_v so generated
Begin
Verify the relations (5) and (6) and (7);
If *all* these three constraints are TRUE then
Memorize the matrix S_v ;
End;
End;

Step_4
For each disassembly sequence *k*
For each matrix S_v determined at *Step_3*
Begin

$$Q_{vk}=S_v\cdot T_k=\left\{q_i\right\}_{vk}\overset{not}{=}q_{i(vk)}$$

$$t_{cy(k)}=\max_{W_i}\sum_{j\in(tasksW_i)} t_{j(k)}$$

$$F_k=\sum_{i=1}^{n}\left(q_{i(vk)}-t_{cy(k)}\right)^2$$

End.

Step_5
Calculate $\min_k\left\{F_k\right\}$ and find the matrix S_{opt} that minimize this function.

Step_6
For S_{opt} calculated at *Step_5*
For each disassembly sequence *k* calculate

$$\max_k f_k=\frac{R^k}{t_{cy(k)}} \quad \text{where k=1.. d}$$

Step_7
Write *k* (the index of the most profitable disassembly sequence) that maximizes the objective function *f* at the *Step_6*;
Write the values of f_k in a descending order;
STOP.

Notes:

1. The cartesian product at the **Step_2** is generated using the backtracking programming method; Its number of elements is given by the relation (8):

$$p = \prod_{j=1}^{m} \left(\sum_{i=1}^{n} m_{ij} \right) \tag{8}$$

2. At the **Step_3** the procedure **Generate_S$_v$** uses the results of the **Step_2** and gives *all* the matrices *S* defined in section 3**.** The procedure needs as input data the values of n, m, and e_i from **Step_1** and **Step_2**:

Generate_Sv(n, m, e_i)
Begin
For each $i = 1..m$
For each $j = 1..n$
$s_{ij} = 0$;
For each $i = 1..m \quad s_{e_i,i} = 1$;
End;

3. In the program the valid matrices S_v that meet the constraints (5) and (6) and (7) are memorized in a vector of matrices with *s* components ($s < p$);

4. At the **Step_4** we see that $Q_{vk} = \{q_i\}_{vk}$ is a vector with $i = 1..n$ components, and $q_{i(vk)} = \sum_{j \in (tasks W_i)} t_j$, $v = 1..s,\ k = 1..d$

5. The representation of the disassembly sequences in the program by the matrix $T = \{T_k\}$ allows taking into account the failure of the disassembly process so that some sequences are not completed.

The result of the algorithm is the maximum value of the objective function, the best disassembly sequence *k* that maximizes the profit, and a classification of the other disassembly sequences in a decreasing order of the achieved profit.

5 Example

We take as a reference the disassembly sequences given in [8] by a Disassembly Petri Net [9].

E. Zussman and M. Zhou represented the possible sequences for a telephone handset. The corresponding Disassembly Petri Net is given in figure 1. An income r_i is associated to each place: *the end-of-life value of the component*. Also, a cost τ_j is associated to each transition. We supposed that this cost is proportional to the operative duration for the corresponding transition.

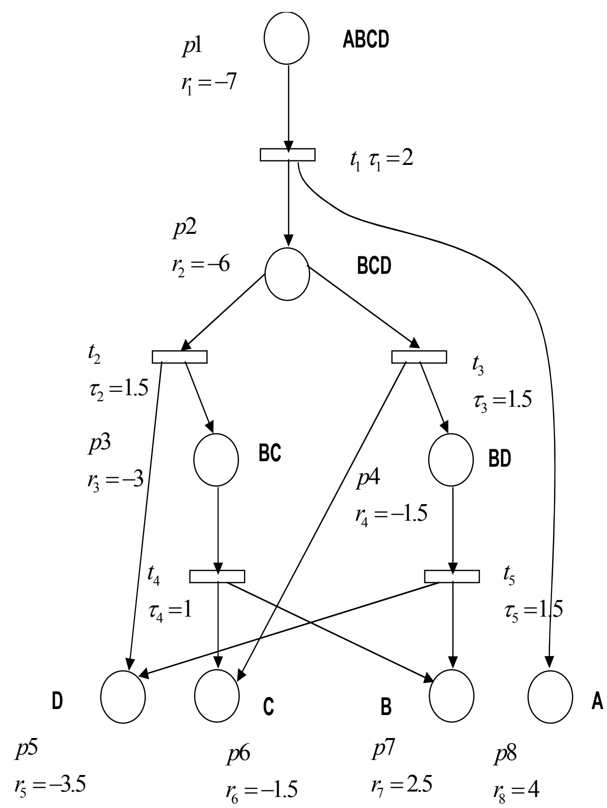

Figure 1: DPN of the handset after [8].

The complete disassembly sequences are:

$$t_1 \to t_3 \to t_5 \qquad (k = 1)$$

$$t_1 \to t_2 \to t_4 \qquad (k = 2)$$

We have considered that the disassembly line had two workstations, one that performs disassembly tasks and the other – dismantling tasks. We took into consideration some alternative dismantling operations in case of the failure of the disassembly ones. For example $t_2^{'}$ and $t_5^{'}$ with the corresponding operative durations $\tau_2^{'} = 1$ and $\tau_5^{'} = 1$, are two extracting operations that could affect the structure of the component D.

So, we can also have the alternative disassembly sequences:

$$t_1 \to t_3 \to t_5^{'} \qquad (k = 3)$$

$$t_1 \to t_2^{'} \to t_4 \qquad (k = 4)$$

Therefore, in case of a disassembly failure there are several combinations from these four disassembly sequences that are not complete, like

$$t_1 \to t_3 \quad (k = 5);\ t_1 \to t_2 \quad (k = 6);\ t_1 \quad (k = 7).$$

The global revenue for each sequence is calculated from (1):

$$R^1 = r_5 + r_6 + r_7 + r_8 = 1.5$$

$$R^2 = r_5 + r_6 + r_7 + r_8 = 1.5$$

$$R^3 = r_5^{'} + r_6 + r_7 + r_8 = 1$$

$$R^4 = r_5^{'} + r_6 + r_7 + r_8 = 1$$

$$R^5 = r_4 + r_6 + r_8 = 1$$

$$R^6 = r_3 + r_5 + r_8 = -2.5$$

$$R^7 = r_2 + r_8 = -2$$

where $r_5^{'} < r_5$ is the revenue when the component D is alterated due to the destructive dismantling operations $t_2^{'}$ or $t_5^{'}$.

Input data is: m=5, n=2;

$$M = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

d=7; $R = \{R^k\} \qquad k = 1..7$

$$T = \begin{pmatrix} 2 & 0 & 1.5 & 0 & 1.5 \\ 2 & 1.5 & 0 & 1 & 0 \\ 2 & 0 & 1.5 & 0 & 1 \\ 2 & 1 & 0 & 1 & 0 \\ 2 & 0 & 1.5 & 0 & 0 \\ 2 & 1.5 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The results of the proposed algorithm, implemented in C++ are:

$k=2;\ S_2 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix}$ and

$\max_k f_k = f_2 = 0.6$ where

$$f_k = \{0.5; 0.6; 0.4; 0.5; 0.5; -1.25; -1\}$$

Thus, the disassembly sequence that maximizes the objective function *f* from (4) is

$$t_1 \to t_2 \to t_4 \qquad (k = 2)$$

So, the sequencing of transitions is t_1, t_2, t_4 with the following assignments: t_1 to W_1, t_2 and t_4 to workstation W_2. One can observe that the line is well balanced the value of the function F from (2) being minimized.

Analyzing the results we can conclude that if the total disassembly is not possible due to different component degradations, the partial disassembly given by the fifth sequence $t_1 \rightarrow t_3$ is better than the one given by the sixth partial disassembly sequence $t_1 \rightarrow t_2$. So, it is more profitable to separate C from BD than D from BC. Moreover, the values for the objective function show that destructive sequences three and four are preferred to the partial nondestructive disassembly sequence six. Even a disassembly process with one sequence (separating only A from BCD) is more advantageous in this case like the partial disassembly sequence six (separating D from BC).

6 Conclusions

The method proposed in this paper has the advantage that it takes into account the operational durations, as well as the profit achieved after a disassembly process from the valorization of the obtained components or subassemblies. In a balanced disassembly line, the cycle time has the lowest value so the operational costs are minimized. The algorithm does not optimize the balance of the disassembly line, but gives a solution that improves this balance.

The proposed algorithm gives the most profitable disassembly sequence for a given product and permits, in the meantime, a complete analysis of the whole set of disassembly sequences in respect of obtained costs and incomes.

The main weakness of this method obviously lies in the exponential complexity of the proposed algorithm. It has only been proposed in order to test quickly the efficiency of the objective function presented in section 2.

Future work will be concentrated on the introduction of stochastic algorithms, the only ones able to deal with complex products.

References

[1] K. D. Penev and A. J. de Ron, "Determination of a disassembly strategy", *International Journal of Production Research,* vol. 34, pp. 495-506, 1996.

[2] N. Salomonski and E. Zussman, "On-line predictive model for disassembly process planning adaptation", *Robotics and Computer Integrated Manufacturing,* vol. 15, pp. 211-220, 1999.

[3] A. Gungor and S. Gupta, "A Systematic Solution Approach to the Disassembly Line Balancing Problem", *Proceedings of the 25th International Conference on Computers and Industrial Engineering*, pp. 70-73, 1999.

[4] L. Duta, F. Gh. Filip and J.M. Henrioud (2002), "Automated Disassembly: Main Stage in Manufactured Product Recycling", *Proceedings of the 4th International Workshop on Computer Science and Information Technologies CSIT 2002*, on CD, Patras, Greece, September 2002.

[5] S. Addouche, C. Perrard and J.M. Henrioud (2002), "Linear Programming Model To Find The Optimal Disassembly Sequence", *The 3rd CIRP International Seminar on Intelligent Computation in Manufacturing Engineering* , (ICME2002), Italy.

[6] B. Rekiek, *Assembly Line Design*, Ph.D. thesis, Free University of Bruxelles, 2001.

[7] S. Nof, *Industrial Assembly*, Chapman&Hall, 1997.

[8] E. Zussman and M. Zhou, "A Methodology for Modeling an Adaptive Planning of Disassembly Processes", *IEEE Transactions on Robotics and Automation*, vol. 15, no. 1, February, 1999.

[9] K. Moore and A. Gungor, "Disassembly Process Planning Using Petri Nets", *Proceedings of the IEEE International Conference on System, Man and Cybernetics*, San Diego, California, 1998.