APPLYING EQUAL PILES APPROACH TO DISASSEMBLY LINE BALANCING PROBLEM

L. Duta [(1)], F. Gh. Filip [(2)], J. M. Henrioud [(3)]

[(1)] *Computer Science Department*
Valahia State University
18-24, Unirii Av. Targoviste, ROMANIA
Fax: +40 (0) 245.217.683, E-mail: duta@valahia.ro
[(2)] *Research Institute for Informatics*
8-10, Maresal Averescu, Bucharest, ROMANIA
E-mail: ffilip@acad.ro
[(3)] *Automation Laboratory*
Institute of Production Engineering
25, Alain Savary, Besançon, FRANCE
E-mail: henrioud@ens2m.fr

Abstract: Disassembly process decomposes a product into parts or subassemblies. Due to the uncertainties that occur during this process, designing and balancing a disassembly line is a very challenging problem. To deal with the disassembly line balancing problem a new method which relies on the equal piles approach is proposed. *Copyright © 2005 IFAC*

Keywords: environmental engineering, process control, optimal load flow, time schedule control

1. INTRODUCTION

The disassembly process is the main stage in the recycling of manufactured products at the end of their life. Disassembly promotes reuse, recycling, material and energy recovery.

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 for material or energy recovery.

Disassembly lines are usually manual and hard work. Products subjected to disassembly have an uncertain structure due to the components conditions. Therefore, designing and optimizing a disassembly system are important problems for the manufacturers whose aim is to gain money and time by using such a system (Touzanne, 2002).
The objective of the Disassembly Line Balancing Problem (DLBP) is to use the resources of the disassembly line as efficiently as possible while meeting the demand.

Very few authors have addressed the problem of Disassembly Line Balancing.

Gungor and Gupta treated for the first time the problem of Simply Disassembly Line Balance (DLBP-S). The formulation of this problem was as follows: *a number of n tasks have to be assigned on m machines so as to respect the precedence constraints between the parts of the product* (Gungor, 1999). In their approach, deterministic operative times are used. Moreover, the disassembly process is considered as the reverse of assembly. The authors didn't take into account the uncertainties of the disassembly process. Recently, the same two authors studied the DLBP in the presence of failures (Gungor, 2001). Disassembly sequences which minimized the idle time were chosen. A failure cost had been assigned to each task with a certain probability. The new approach gives an optimal algorithm for balancing the line in respect of the operative costs. The authors were interested in the architecture of the disassembly line. Their method didn't take into account the end-of-life values of the used components and thus the fact that in real systems destructive operations are sometimes preferred to the un-destructive ones so as to maximize the recuperated value. To this day, these have been the only works that treat the DLBP clearly.

Although the problem of DLBP seems similar to Assembly Line Balancing (ALBP) there are many important differences between assembly and disassembly processes (Duta, *et al*, 2003).

One difference is that in the assembly case operations follow each other considering the physical and functional constraints, while in the disassembly case the precedence between operations is given mainly by the physical constraints, the functional ones being relaxed. Moreover, disassembly is not the reverse of assembly, because some operations cannot be carried out due to the physical degradations of the components and other operations are omitted if they are not profitable. These uncertainties influence the operative times that become stochastic variables.

So, the most difficult problem in a disassembly system is that a disassembly operation can fail any time because of the product or component degradation. In this case we have to choose between applying an alternative destructive disassembly operation (dismantling), and abandoning the disassembly procedure. This decision must be taken in real time because in a used product the components states are not known from the beginning of the process.

In order to balance the operations in the disassembly line we have developed an algorithm based on the Equal Piles Approach.

2. EQUAL PILES APPROACH

Installing a disassembly system requires large investments and thus needs a long term decision. Depending of the line layout the investor has to take into consideration different objectives. The objective of the simple DLBP is to assign a number of disassembly tasks on a product with individual disassembly times to a fixed number of workstations without violating the existing precedence relationships among the tasks.

In the Equal Piles Approach (EPA) the objective is to equalize the stations loads. This method seeks to assign tasks to a fixed number of stations in such a way that the working time of each station is nearly equal (Rekiek, 2001). Having equal loads on stations means equal working times and therefore equal costs of the work done on each workstation.
The simple EPA for the Assembly Line Balancing Problem (ALBP), which does not take precedence constraints into account, is the following: given a set of N objects of various sizes we have to distribute the objects into K ‘piles’ in such a way that the ‘heights’ of the piles are as equal as possible (Rekiek, 2001).

In the case of the assembly lines, the problem is to equalize stations workloads for a fixed cycle time and number of stations. However, in real industrial systems the EPA must provide a solution which respects all precedence constraints of the product (Lambert, 2003). If these constraints are violated for a given order of stations along the line, the product has to move against the sense of the line conveyor, visiting at least one station several times. In the case of the disassembly lines we have to equalize the station loads by taking into account the appropriate disassembly task (destructive or not), or even in the situation of process failure. Another difference between the ALBP and DLBP is that in the last case the cycle time is not the ratio between the planning period and the demand of assembled products, but the ratio between the duration of the planning period and the maximum number of products that need to be disassembled to meet a certain demand of components (Duta, *et al*, 2003), (Gungor and Gupta, 1999).

In this context, the DLBP can be defined as follows: *a number of given disassembly tasks of a product with individual disassembly times must be assigned to a fixed number of workstations in order to satisfy the EPA and the existing precedence relationships among the disassembly tasks, taking into account all the uncertainties that can occur due to the state of the product.*

3. DEALING WITH DLBP

For simplicity we suppose that we have a paced, linear, mono-product disassembly line, and also an infinite supply of products. The workstations are placed on the line in an increasing order of their destructing effect. The disassembly or the dismantling times are assumed to be deterministic and known. The DLBP needs the following input data:
- the desired number of workstation;
- the cycle time;
- the precedence relationships between disassembly tasks;
- the duration of each task;
- the possible assignment of the tasks to the workstations

With these data and the previous presumptions we can illustrate the data flow for DLBP:

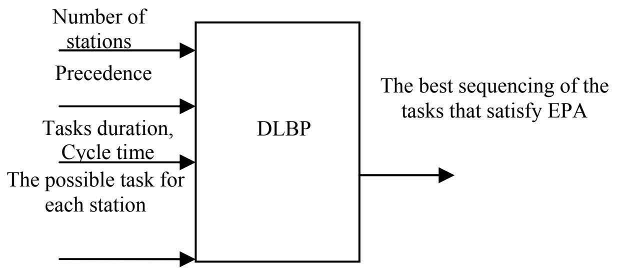
Number of stations
Precedence
Tasks duration, Cycle time
The possible task for each station
DLBP
The best sequencing of the tasks that satisfy EPA

Fig. 1. Data flow for DLBP

Used notation:

m number of tasks
n number of stations
d number of possible disassembly sequences
t_{cy} cycle time
t_j the operative time of task j

$T = \left\{ t_j^k \right\}$ k=1..d and j=1..m the matrix of the operative times for each disassembly sequence k

$M = \left\{ m_{ij} \right\}$ the matrix of possible assignement of the tasks, where

$$m_{ij} = \begin{cases} 1 & \textit{if the task j is possible to be assigned to station i} \\ 0 & \textit{if the task j can be done on station i} \end{cases}$$
$i = 1..n, j = 1..m$

To deal with the DLBP formulated before we have to minimize the imbalance between stations by finding the minimum of the function

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

For each assignable disassembly sequence k, and then to calculate $\min_k \left\{ f_k(t) \right\}$

First, we have to find the matrices of the assignement of the tasks to satisfy the DLBP formulated in 2.

Let $S = \left\{ s_{ij} \right\}$ be the researched matrix where

$$s_{ij} = \begin{cases} 1 & \textit{if task j is assigned to workstation i} \\ 0 & \textit{if task j isn't assigned to station i} \end{cases}$$

and $i = 1..n, j = 1..m$

This matrix is subjected to the following constraints:

$$s_{ij} \in \{0,1\} \qquad (2)$$

$$\sum_i s_{ij} = 1 \qquad (3)$$

$$\sum_{i=1}^{n} i \cdot s_{ik} - \sum_{i=1}^{n} i \cdot s_{ij} \le 0 \qquad (4)$$

and $i = 1..n$, $j = 1..m$, $k = 1..m$

Constraint (2) is known as the **non-divisibility constraint**, that does not allow a task to be assigned to more than one station. Constraint (3) is the **assignement constraint** and it requires that each task is assigned to *exactly* one station. Constraint (4) is the **precedence constraint** that invokes technological order so that if task *k* is to done before task *j* (k<*j*), then *k* cannot be assigned to a station downstream from task *j* (Nof, 1997).

If we determine the matrices S, we can calculate the sum of the tasks duration on each workstation, which is the term $\sum_{j \in \{tasks\, W_i\}} t_j$ from the equation (1). Then, the objective function *f(t)* can be minimised.

4. THE PROPOSED ALGORITHM

To find the matrices S that verify equations (2), (3) and (4) we have used a classic procedure that calculates all the possible assignements of the tasks to the workstation using the notion of cartesian product and the backtracking programming method. Let v be the vector which memorise the valid matrices S. The *algorithm* is given below:

Step 1 Read the input data *m, n, M,* t_{cy} and *T*

Step 2 Generate the cartesian product $A_1 \times A_2 \times ... A_m = \left\{ (e_1,...,e_m)_1, ..., (e_1,...,e_m)_p \middle| e_i \in A_i, i = 1..m \right\}$ where A_i = {is the set of the workstation indexes that can perform task i } and *p* the number of the elements;

Step 3
```
s=0;
for each $k = 1..p$
begin
Generate_S(n, m, $e_i$);
if the relations (2) and (3) and (4) are TRUE
        begin
        s=s+1; v[s]=S;
        end;end;
```
Step 4
```
k=0; for each $j = 1..d$ for each $i = 1..s$
begin
k=k+1; //the assignable disassembly sequences k
$Q_k = S_i \cdot T_{kj}$
```

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

end;

Step 5

Calculate $\min_k \{f_k(t)\}$ for each assignable disassembly sequence $k = 1..(s \times d)$

Write $f_k(t)$, *k*, S_k ;

STOP

Notes:

1) The cartesian product at the **Step_2** is generated using the backtraking programming method; The number of the elements of the cartesian product is

$$p = \prod_{j=1}^{m} \left(\sum_{i=1}^{n} m_{ij} \right) \qquad (5)$$

2) At the **Step_3** the procedure **Generate_S** uses the results of the **Step_2**:

Generate_S(n, m, e_i)

```
    begin
    for each $i = 1..m$
    for each $j = 1..n$   $s_{ij} = 0$;
    for each $i = 1..m$   $s_{e_i,i} = 1$;
    end;
```

3) At the **Step_4** we see that

$Q_k = \{q_i\}_k$ and $q_i = \sum_{j \in (tasks W_i)} t_j$, $k = 1..s \times d$

4) When we give the disassembly sequences by the matrix **T** we can also take into account the failure of the disassembly process so that the sequence is not completed.

The result of the algorithm is the minimum value of the objective function, the best disassembly sequence *k* that minimise the imbalance of the line, and the assignement of the tasks to the stations for this sequence in the matrix S_k.

5. EXAMPLES

For the first implementation, we took as reference the disassembly sequences given in (Zussman and Zhou, 1999) by a Disassembly Petri Net. The order of the tasks for two different disassembly sequences is:

$task_1 \rightarrow task_3 \rightarrow task_5$

and $task_1 \rightarrow task_2 \rightarrow task_4$

We have considered that at the beginning we had two workstations, one that performs disassembly tasks and the other – dismantling tasks. So we must take into consideration some alternative operative times, for example $t_2^{'}$ and $t_5^{'}$ for the destructive tasks.

In this situation, the following order of the tasks can form two different disassembly sequences:

$task_1 \rightarrow task_2^{'} \rightarrow task_4$

and $task_1 \rightarrow task_3 \rightarrow task_5^{'}$

Therefore, in case of disassembly failure we can have many combinations from these four disassembly sequences that are not complete, like

$task_1 \rightarrow task_2^{'} \rightarrow none$ *or*

$task_1 \rightarrow none \rightarrow none$

The proposed algorithm was implemented in C++.

Input data used is:

m=5; n=2; $t_{cy} = 2.5$; $M = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix}$

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

where the lines three and four corespond to alternative dismantling tasks and the last two lines correspond to incomplete disassembly tasks.

The results of the proposed algorithm are:

k=2; $S_2 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix}$ and $f_2(t) = 0.25$

We have reached the conclusion that the best disassembly sequence that minimizes the imbalance of the line is the second one. The sequencing of tasks is $task_1, task_2, task_4$ with the following assignments: $task_1$ to W_1, $task_2$ and $task_4$ at workstation W_2. We notice that the line is well balanced, the value of the objective function being under 1.

For the second example we validated our algorithm in the case of a radio disassembly (Salomonski and Zussman, 1999). The disassembly sequences are given in a Disassembly Petri Net (Moore *et al*, 1998).

The input data is given below:

m=8; n=4; $t_{cy} = 1$

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

and $T=\begin{pmatrix} 0.12 & 0.07 & 0.07 & 0.12 & 0.95 & 0.75 & 0.75 & 0.95 \\ 0.15 & 0.07 & 0.07 & 0.12 & 0.95 & 0.75 & 0.75 & 0.95 \\ 0.15 & 0.07 & 0.07 & 0.15 & 0.95 & 0.75 & 0.75 & 0.95 \\ 0.15 & 0.07 & 0.07 & 0.12 & 0.95 & 0.9 & 0.75 & 0.95 \end{pmatrix}$

The results are:

$k = 3; f_3(t) = 0.57$

$$S_3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

In this case the best disassembly sequence that gives a good balance of the line is the 3rd one, that is given in the matrix T. The order of the tasks is $task_{1,1}, task_2, task_3, task_{4,1}, task_5, task_6, task_7, task_8$ where the first and the fourth tasks are dismantling ones. The assignement is: $task_{1,1}$, $task_2$, $task_3$, $task_{4,1}$ to W_1, $task_5$ to W_2, $task_6$ and $task_7$ to W_3 and $task_8$ to W_4.

6. DISASSEMBLY SYSTEM CONTROL

The control of disassembly process involves two essential decision variables: disassembly level and disassembly mode: clean or destructive. A third decision variable may be added: the task to station assignment.

When we speak of controlling a disassembly process we must take into consideration the use in real time of the Decision Support Systems (DSS). Usually, DSS in real time are used in "crisis" situations like: the stop of the production flow, the machines out of order, appearance of unpredictable situations, value changing of one or more production parameters (Filip, 2002). The DSS must be in permanent dialog with the control system. This increases the reactivity of the control system at the appearance of difficult situations. In fact, there are situations in which the decision must be taken very quickly so as to give optimal command to the process.

The same situation can occur in a disassembly process: a disassembly operation cannot be carried out due to the bad state of the part or of the product itself. A decision must be taken so that the disassembly process can continue without failures.

In the case of manual disassembly, the perturbation is analyzed by a human operator. In the case of an automatic process, specialized detection and analysis tools are needed. Moreover, in the case of disassembly lines the control system has to decide the flow of tasks on workstations. This means that the systems control has to make a good balancing of the disassembly line. In this case a DSS has to be integrated in the architecture of the system control. A Data Base and a Knowledge Base are created with all the characteristics of the product (DB+KB).

This information is sent to the planner that analyzes the data together with the given manufactured problem. The result of the planner is a model of the product or of the process itself, model that can be next simulated on a computer. The integrated DSS gives the best decision-solution of the manufacturing problem in respect to the chosen criteria. It also generates some data files that are used by the control system to provide the necessary command to the process. The reactivity loop is closed by the visual information analysis system that acquires information in real time from the process and sends it on the serial bus to be analysed by the DSS. So the dialogue between the two systems is permanent .

In the Figure 1, the architecture for controlling a disassembly process is proposed (Duta, 2004).

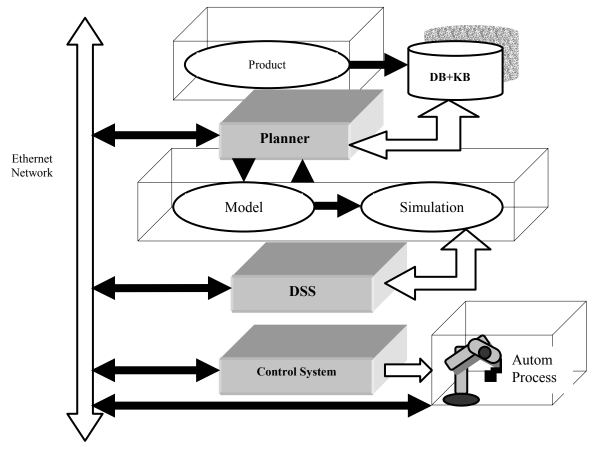

Fig. 2. Integrating DSS in the process control

We particulized this architecture in the case of the disassembly systems. As a planner we used the software LEGA, entirely conceived in the Automatic Laboratory of Besançon, France, (Addouche *et al*, 2003) and the DSS is the software that integrates the program that assures a good balance of the line. LEGA gives all the possible disassembly sequences of the product using modeling by Petri Nets. DSS analyzes these sequences and it gives the best decision for disassembling the product and the flow of products on the line. This is a good feature of the system because manufactures are interested in reducing waste and difficulties in disassembling end-of-life products and reducing time in revalorization of the subassemblies (Addouche *et al.*, 2002). The disassembly planner gives the sequence of the components that must be separated to achieve the target. It can be computed for the total disassembly of the product or for its partial disassembly (Wiendahl et al, 1999). Output data from LEGA (given in the output files) is the input data for DSS. The Decision Support System (DSS) integrates the model and performs the simulation. As a result, tasks are assigned to workstations. The algorithm presented in paragraph 4 is executed and the Decision System provides the best disassembly sequence that minimizes the imbalance of the line. At a given moment, this is an off-line solution.

It is the role of the control system to use this information to change the states of the variables that control the movements of the line and robots. The control system merges the information from the artificial vision system (linked to the Ethernet Network) with that contained in the database for each component or subassembly. There is an information feedback that allows the adjustment of the model while the product is disassembled, since there are components that can be visible only after removing other ones. The values obtained as a result of the proposed algorithm are saved in files. The control software uses these files to change the variables in the system. A good balance of the line is assured. The software is entirely written in the C++ language. The system needs a host computer that is responsible for the coordination of the control task, a personal computer for planning and simulation and an artificial vision system with mobile cameras.

7. CONCLUSIONS

The validity of the algorithm was proved in two situations: one in which the number of stations is small and the value of the time cycle is rather high, and the other when the number of stations is bigger but the cycle time t_{cy} is smaller. By increasing the number of stations one could observe that the value of the objective function *f (t)* rises, so the line is less balanced. Indeed, *f(t)* is a cost function. The bigger the number of stations the smaller the number of tasks by station, so the line cost increases and the balancing becomes more complicated. There is a similar situation in the case of the disassembly failure: the imbalance is greater because of the incomplete sequences. Thus, one has to change the design of the line (decreasing the number of workstations) or, better, to choose another disassembly sequence. A new system control architecture that provides a normal flow of the product on the disassembly line was proposed. A Decision Support System (DSS) was integrated in the control architecture. The DSS analyzes the model and performs the simulation. As a result a good balance of the line is given.

However, the proposed method has its limitations. The complexity of the algorithm is exponential and is given mainly by the structure of the line (the number of workstations) and the number of disassembly tasks. In the objective function *f(t)* we did not take into account the profit that can be obtained by the valorisation of the disassembled components. Future work will focus on the problem of maximising this income while minimising the cost of the disassembly line using multi-criteria optimisation methods.

REFERENCES

Addouche, S., C. Perrard and J.M. Henrioud (2002) Linear Programming Model to Find the Optimal Disassembly Sequence.In: *Proceedings of The 3rd CIRP International Seminar on Intelligent Computation in Manufacturing Engineering*, Italy.

Addouche, S., and Perrard, C., and Henrioud, J. M. (2003) Integration of Reassembly Tasks in Disassembly Process Planning, *Preprints of IFAC International Workshop on Intelligent Assembly and Disassembly*, *Bucharest, Romania*

Chevron D. (1999), *Contribution to study the supervision of a disassembly cell for end-of-life products*, Doctoral Thesis, University of Grenoble, November 1999

Duta L. (2004), Automatic Disassembly Systems Control, *Proceedings of the Annual Romanian Academy Congress, ARA29, Bochum, Germany*

Duta, L., F.Gh. Filip and J.M. Henrioud (2003). A Method for Dealing with the Multi-Objective Optimisation Problem of Disassembly Processes, In: *Proceedings of the IEEE International Symposium on Assembly and Task Planning ISATP 2003,* Besançon, France

Filip, F.Gh. (2002). *Computer Aided Decision*, Technical Publishing House, Bucharest, Romania.

Gungor, A., S. M. Gupta (1999). A Systematic Solution Approach to the Disassembly Line Balancing Problem, *Proceedings of the 25th International Conference on Computers and Industrial Engineering*

Gungor A., Gupta S., A solution approach to the disassembly line balancing problem in the presence of task failures, *International Journal of Production and Research* , **Vol. 39**, No. 7, 2001, pp 1427-1467

Lambert, A. J. D. (2003), Disassembly Sequencing: a survey, *International Journal of Production and Research*, **Vol. 41**, No. 16, Pg. 3721-3759,

Moore, K., A. Gungor and S. Gupta (1998). Disassembly Process Planning Using Petri Nets. In: *Proceedings of The IEEE International Conference on System, Man and Cybernetics,* San-Diego, California.

Nof, S., *Industrial assembly*, Chapman&Hall, 1997

Rekiek, B. *Assembly Line design*, Doctoral Thesis, University of Bruxelles, 2001

Touzanne, F. (2002). *Contribution to a disassembly system design for the end-of-life products,* Doctoral Thesis, Franche-Comté University, Besançon, France

Salomonski, N. and E. Zussman (1999), On-line Predictive Model for Disassembly Process Planning Adaptation. *Robotics and Computer Integrated Manufacturing,* **vol. 15**, pp. 211-220

Wiendahl, H. P., and Selinger, G., and Burkner, S. (1999) A General Approach to Disassembly Planning and Control, *Production Planning and Control*, **10**, 718, 1999

Zussman, E. and M. Zhou (1999), A Methodology for Modeling an Adaptive Planning of Disassembly Processes. *IEEE Transactions on Robotics and Automations,* **vol. 15**, No. 1