Optimum disassembly sequence generation

Fred J.D. Lambert
Optimum disassembly sequence generation
Fred J.D. Lambert
Techn. Univ. Eindhoven, P.O. Box 513, NL-5600 MB Eindhoven

ABSTRACT
Optimal disassembly sequences can be obtained on the basis of linear and mixed-integer programming methods. These are applied to models that are based on disassembly graphs. These graphs represent the possible subassemblies and the transitions (actions) between them. They represent the structure of the assembly in a more abstract manner than assembly drawings do. The translation from assembly drawings into graphs presupposes the selection of the feasible subassemblies and actions. In this contribution a method is discussed that supports the automatic determination of all possible subassemblies, starting from the assembly structure such as represented by drawings etc. It uses the connection diagram and a set of Boolean expressions that represent precedence relations. These are based on part removal in stead of on the disconnection of liaisons. Consequently, actions are transitions between subassemblies. Based on this method, feasible subassemblies and actions can be selected automatically. These represent the full set of data for the establishment of a disassembly graph. The method is demonstrated by the discussion of a number of assemblies that are taken from the literature. It appears that automatic selection of subassemblies is possible. Based on this selection, the set of actions can be defined and the disassembly graph can be established automatically.

Keywords: Disassembly, modelling, complex products, operations research, precedence relations

1. INTRODUCTION

During the past decades, the challenge for sustainable production systems has become increasingly important. Closing of material cycles has always been a crucial aspect in this field. Closing of cycles presupposes the avoidance or at least the adequate reuse of waste streams. Although a lot of waste streams can indeed be properly reused as parts or recycled as materials, there remains an important stream of mixed light fraction. Actually, recycling of this stream is hardly possible. An important constituent of this stream is shredder light fraction. This remains after shredding of discarded complex products and the subsequent separation of ferrous and non-ferrous metals from the resulting shredder output. Shredder light fraction usually consists of a rather undefined mixture of light materials such as plastics, rubber, glass, textile, paper, and wood. These materials might be contaminated with working fluids and other undesired materials. As most of the constituents of shredder light fraction have a low specific value, further separation is usually economically infeasible. The usual method of treating this kind of waste still is by incineration. The resulting slag will bear the contamination, which makes it unsuitable for utilisation, even in low-grade appliances such as in building materials or for road construction. Separation at the beforehand of complex product is thus required. Consequently, the quality of slag from incineration of combustible waste is safeguarded. This is the reason why legislation in industrialised countries is aimed at separate collection and subsequent processing of discarded complex products. Besides that, legislation is complemented by a ban on landfill of shredder residue. As a matter of fact, the resulting shredder residue should be incinerated under special conditions, apart from household waste.

Because treatment is becoming increasingly expensive, there is a strong economic incentive towards a quantitative reduction of shredder residue. Besides that, the quality of the shredder residue should be enhanced. This is obtained by a conscious product design (design for recycling) including a conscious selection of materials. Apart from that, selective disassembly, prior to shredding, is indispensable for the removal of strongly hazardous parts and materials, and also for the recovery of valuable parts and materials. However, bulk light materials can also be separated at beforehand, aimed at reduction of the amount of shredder residue. As an example of the latter, the inner lining and the seats of a car can be mentioned.

1 Correspondence: Email: a.j.d.lambert@tm.tue.nl; Postal address: P.O. Box 513, 5600 MB Eindhoven, The Netherlands, TM/AW Pav. H.03; Telephone: +31-40-2474634; Fax: +31-40-2444602
Starting from a given product design, as is the case by the majority of the recollected discarded products, it is becoming increasingly relevant, to investigate the optimal disassembly depth. With this in mind, the present paper aims to contribute to the development of a methodology for automatic generation of optimal disassembly sequences.

2. DISASSEMBLY SEQUENCING

As has been pointed out in the introduction, selective disassembly is aimed at multiple purposes, some of which are only understood in relation to a more extended view on the recycling chain. Consequently, it is usually studied as a part of the extended domain of sustainable manufacturing. A review of this field is given in (1). One of the prerequisites of sustainable manufacturing is the design of an appropriate reuse/recycling system. The use of recovered parts and subassemblies is usually called reuse. The use of recovered materials is usually called recycling. The reuse/recycling chain embraces all processes that are connected with reuse and recycling. Krikke et al. (2) discuss the modelling of the complete reuse/recycling chain. In their paper, a simplified partial model, which involves a fixed disassembly tree, is applied to cover selective disassembly. Such a model is useful for both design and operation of recycling chains. Within these chains, particular processes can be studied into more detail. A crucial process is selective disassembly. A lot of literature is available on this field. Originally, this emerged from work that has been principally based on assembly sequencing. One of the basic papers on assembly sequencing, which is also at the origin of recent work on disassembly sequencing, is due to Homem de Mello and Sanderson. It introduces AND/OR graphs (3), that turns out to be a valuable tool in describing disassembly processes.

Starting from this, much work has been performed till present, resulting in many papers on disassembly sequencing. These are usually based on tools such as graphical methods (4), Petri net representation (5,6), mathematical modelling (5,7,8), and heuristic methods such as genetic algorithms and neural networks (9). More publications are listed in the cited papers. Two levels of detail can be discerned here: (1) An aggregated level, aimed at determining the order of disassembly operations and disassembly depth and (2) a detailed level, e.g. for design and operation of robotised disassembly lines. The present paper is focused on the aggregated level.

2.1. Optimal disassembly sequence generation

Many authors refer to travelling salesman problems or dynamic programming for solving optimisation problems on disassembly sequencing. However, the generation of the optimal disassembly sequence can be carried out by using a linear programming (LP) approach. A rudimentary description of this method has been described in (5). This has been rediscovered and elaborated in (8). Here it has been derived based on a formerly developed graphical method (4). The LP method will be concisely described here, starting from the simple example that is presented in figure 1a and that represents the assembly ABC, consisting of three parts. From this, a connection diagram is derived, see figure 1b.

Next, a set I of all feasible subassemblies is derived. A subassembly is a combination of one or more parts. An action is a transition of a (parent) subassembly in two or more (child) subassemblies. A subassembly that has no child subassemblies is called a final subassembly or part. A complete engine can be a part if we consider it as such. In this simple example, no precedence relations are assumed. Such relations indicate whether there are subassemblies that should be removed prior to the removal of others. For instance: A > B means that subassembly A has to be removed in order to enable the removal of B.

Apart from the set I of subassemblies, a set J of actions is defined. A so-called initial action is added as an element of this set. This action represents the setting available of one unit of the original subassembly. It facilitates straightforward modeling. A disassembly graph is derived, in which the subassemblies represent nodes and the actions represent directed arcs. The disassembly graph of the example is depicted in figure 2.
In this disassembly graph, the actions are represented by numbers and the parts by letters. The parts A, B, and C are not depicted in the graph, because this is not required for unambiguous representation. Subassemblies with equal number of parts are depicted in the same column. Note that action 7 represents the decomposition of ABC straight into A, B, and C. The restriction that an action decomposes a parent subassembly into two and only two child subassemblies is unnecessary and can be avoided in this model.

Starting from the disassembly graph, a transition matrix $T$ of size $I \times J$ is defined. For the example of figure 2, the elements $T_{ij}$ of this matrix are given in table 1.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td></td>
<td></td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>AB</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td>-1</td>
<td></td>
<td></td>
<td>-1</td>
</tr>
<tr>
<td>AC</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BC</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td>-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Transition matrix of the example. The rows represent subassemblies, the columns represent actions.

The elements of this matrix should be interpreted as follows. In action 3, e.g., the subassembly ABC is decomposed in BC and A. Consequently, the values $-1$ (destruction), and $1$ (creation) respectively, are assigned to the corresponding elements. Other elements remain zero.
The parameters that represent the revenues of the subassemblies are arranged in the revenue vector \( r \) of size \( I \) with elements \( r_i \); the costs of the actions are represented by the cost vector \( c \) of size \( J \) with elements \( r_j \). The variables are represented by a flow vector \( x \) of size \( J \) and elements \( x_j \).

The problem is formulated as follows:

Maximise the net revenue:

\[
\sum_i \sum_j \left( r_i - c_j \right) x_j
\]

subject to:

\[
\sum_j T_{ij} x_j \geq 0
\]  

and the initial condition:

\[
x_0 = 1
\]  

This model enables us to find the optimal disassembly sequence.

2.2. Extensions

2.2.1. Constraints on subassemblies and actions

If the disassembly process should guarantee the availability of one or more specific subassemblies, say subassembly \( i_1 \), the optimal sequence is found after addition of the constraint:

\[
\sum_j T_{ij} x_j = 1
\]  

This means that in every solution subassembly \( i_1 \) is created but not destroyed in a subsequent action. Similarly, actions can be prescribed by setting the respective components of vector \( x_j \) equal to 1.

2.2.2. Multiple criteria analysis

The method enables the generation of a ranking of (sub) optimal sequences, by disabling the already found solutions if there are multiple solutions with the same revenue, or by setting upper boundaries on the revenue. This ranking can be evaluated with respect to other criteria. By doing so, sequences can be selected which are reasonable with respect to multiple criteria, for instance economic, environmental, energy use, etc. See e.g. \(^{10}\).

2.2.3. Sensitivity analysis

Because the data on cost and revenue are not exactly known, it is often preferable to investigate the sensitivity of the found optimal solutions with respect to these data. Each specific solution is optimal within a definite parameter domain. The stability of such a solution with respect to changes in parameter values is of interest, e.g. for reduction of the complexity of the selected solution.

2.2.4. Minimising the complexity

The flow variables \( x_j \) can be defined as positive variables. Nevertheless, due to the structure of the LP-problem, only the values 0 or 1 are returned in the solution. If one, however, wants to reduce the complexity of the solution, penalties can be assigned to actions and the objective is extended. Then, each \( x_j \) should be linked to an element \( y_j \) of a vector, which consists of Boolean \( \{0,1\} \)-variables. This does not considerably impair the size of the problem.
2.2.5. Restricted capacity

Constraints to capacities are added by imposing an upper limit to the relevant elements of $x_j$. Consequently, a combined sequence might be returned by the solver. This is characterised by values of the flow variables that are between 0 and 1. Here, minimisation of complexity might be important.

2.2.6. Clustering

The problem of clustering is addressed in 11 and involves determination of the optimal mix of materially compatible detached parts in order to arrive at the maximum revenue at minimum cost, see 8.

2.3. Number of sequences

An algorithm for finding the number of possible sequences is presented in 1, a graphical method in 4. If separation in multiple subassemblies is permitted too, the number of sequences increases considerably. The maximum number of possible sequences is established as follows.

One starts with studying a one-part assembly A, which has one possible disassembly sequence, namely: doing nothing, see figure 3a. Next, one considers a two-part assembly AB, which can be separated in A AND B. This is indicated by the filled dot in figure 3b. The number of possible sequences is: $1 \times 1 + 1 = 2$. Subsequently, the three-part assembly ABC, depicted in figure 3c, is analysed. It can be transformed into (AB AND C) OR (AC AND B) OR (BC AND A) or (A AND B AND C). This results in $(2 \times 1) + (2 \times 1) + (2 \times 1) + (1 \times 1 \times 1) + 1 = 8$ possible sequences. From figure 2 it follows that these are: 0, 01, 014, 02, 025, 03, 036, and 7. The sequels indicate the order of the actions.

The 4-part assembly ABCD, which is presented in figure 3d, has already 58 possible sequences. This follows from the rule:

$$\begin{align*}
\binom{3}{4} \cdot 8 + \frac{1}{2} \cdot \binom{2}{4} \cdot (2 \cdot 2) + \binom{2}{4} \cdot 2 + 1 + 1 = 58
\end{align*}$$

Similarly, for a 5-parts assembly already 552 sequences are possible, according to:

$$\begin{align*}
\binom{4}{5} \cdot 58 + \binom{3}{5} \cdot (2 \cdot 8) + \binom{3}{5} \cdot 8 + \binom{2}{5} \cdot 2 + 1 + 1 = 552
\end{align*}$$

Repeated application of this procedure enables the investigation of more complex assemblies in a similar way. The results are summarised in table 2, in which they are compared with those of 4, where in each action only two child subassemblies were allowed.

<table>
<thead>
<tr>
<th>Number of parts</th>
<th>Complete disassembly sequences (2 child)</th>
<th>Number of sequences (2 child)</th>
<th>Number of sequences (unrestricted)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>41</td>
<td>58</td>
</tr>
<tr>
<td>5</td>
<td>105</td>
<td>346</td>
<td>552</td>
</tr>
</tbody>
</table>

Table 2. Amount of different disassembly sequences.

With an increasing number of parts, the number of possible sequences increases dramatically. For a 10-parts assembly, this number already surpasses one billion ($10^9$).
3. DATA COLLECTION

3.1. Reduction of the problem’s size

Although the maximum number of sequences thus rapidly increases with the number of parts, this is not the principal restriction to the calculation of the optimal disassembly sequence, for linear programming does not largely diverge when the size of the problem increases. It is rather data collection that becomes restrictive. This involves the determination of the cost and revenue vectors, and the evaluation of the transition matrix. Automatic generation of this matrix requires the formal description of the structure of the assembly. Starting from assembly drawings, bills of material etc., this seems a difficult task, given the substantial diversity in and complexity of assemblies. A preliminary discussion of the problems related to a systematic approach of reduction of the amount of data, is presented in 12.

Let us remind that the maximum number of subassemblies, according to Homem de Mello et al.3, equals \(2^N - 1\) for an assembly that consists of \(N\) parts. For instance, assembly ABC has 7 possible subassemblies, such as indicated by the rows of the transition matrix of table 1.

If in an action only a separation into two child subassemblies is permitted, the maximum number of actions equals:

\[
\sum_{i=1}^{N-1} \frac{N!}{(N-i)!(i-1)!} - \binom{2}{N} + 1
\]

For \(N = 3\), this results in 6 possible actions (remind that action 7 of figure 2 is excluded here).

However, the number of possible subassemblies is considerably reduced by the following reasons:

Figure 3. AND/OR graphs of assemblies. 3a. One part; 3b. Two parts; 3c. Three parts; 3d. Four parts.
1. Some are disjunct.
2. Some are unstable.
3. Some are not detachable.
4. Some are included in modules that are considered as a final part.

This paper focuses on the determination of disjunct and undetachable subassemblies. Examples of such subassemblies can be found in figure 4a and 4b.

In the simple 4-part case of figure 4a, the subassemblies AC and BD are disjunct. Consequently, the number of possible subassemblies is reduced with about 13% from 15 to 13, and the number of possible actions is reduced with 27% from 33 to 24. Consequently, the number of possible disassembly sequences is reduced with 27% from 58 to 42.

In figure 4b, the subassembly AC, although it can physically exist, it can not be detached, because this requires removal of B at beforehand. This is expressed in a precedence relation: B can only be detached if A or C are detached. This results in an OR relation: A or C should be removed prior to detachment of C or, formally: $(R_A \lor R_C) > R_B$.

![Diagram of 4-part assembly and 3-parts assembly](image)

Figure 4. 4a. 4-part assembly with some disjunct subassemblies; 4b. 3-parts assembly with undismountable subassembly.

The need for systematic study of precedence relations has arisen from research on robotised assembly and appeared obligatory for systematic generation of assembly sequences. In these papers, disassembly analysis has been used for catching the problems that are related to assembly sequence generation. This method starts with the derivation of the full set of coherent subassemblies from the connection diagram. Next a set of $2^l$ Boolean expressions, based on establishing of connections, should be introduced. Here $l$ is the number of connections, i.e. 4 for the assembly of figure 4a. The expressions indicate: 1) what connections have to be established prior to establishing a specific connection and 2) what connections can only be established after the establishment of a specific connection.

In this paper, aimed at disassembly rather than assembly, another method is elaborated. This is based on the removal of parts in stead of on the disestablishment of connections. The connection diagram and a simple set of $N$ precedence rules (i.e. a line of Boolean relations for each part) are derived manually from the assembly drawings and entered in the computer. Starting from this, all coherent subassemblies can be derived. Each of them can be tested on detachability. However, logical rules following straight from the precedence relations, can be used to restrict the number of candidates at the beforehand. This will be illustrated by some examples from practice.
3.2. Simple case from practice

Connection diagrams, such as that in figure 5b, are straightforwardly derived from the corresponding assembly drawing, see figure 5a.

![Connection Diagram](image)

Figure 5. 5a. Assembly example; 5b. Connection diagram.

From the connection diagram, or rather a listing of the connections, all coherent subassemblies can be derived automatically. It is clear from this analysis that connection between C and E is the only failing one. Consequently, subassembly CE is the only incoherent one. This means that still 30 possible subassemblies are remaining. These are: ABCDE, ABCD, ABCE, ABDE, ACDE, BCDE, ABC, ABD, ABE, ACD, ACE, ADE, BCD, BCE, BDE, CDE, AB, AC, AD, AE, BC, BD, BE, CD, DE, A, B, C, D, E.

At a first glance, one observes that additional subassemblies, such as AE, can also not appear in the disassembly process, because these are not detachable. One is interested in a systematic method for reduction of the disassembly graph. One method, based on the decomposition in modules is presented in 15. Here, cut-vertices have to be found, i.e. parts whose removal disconnects the remaining graph. However, because all parts in the example are tightly connected with others, no cut-vertices are present.

In practice, it frequently occurs that disassembly can only start with a restricted number of actions. In this case: removal of A or removal of E. This excludes 7 more subassemblies, namely those which contain A and E simultaneously, i.e. ABCE, ABDE, ACDE, ABE, ACE, ADE, AE. This is due to precedence relations.

For this assembly, they are listed as follows, by considering the blocking of the different parts:

- \( R_A \)
- \( R_A \) or \((R_D \text{ and } R_E) > R_B \)
- \( R_A \) or \( R_D > R_C \)
- \( (R_B \text{ and } R_C) \text{ or } R_E > R_D \)
- \( R_E \)

According to this notation, \( R_A \) means: removal of part A, etc. The first line means that removal of A is not subjected to any precedence relation. The second line expresses that removal of B can only take place if either A is removed, or D and E are both removed. Other lines are explained similarly.

This approach avoids the ambiguity of establishing precedence relations for disconnections. In the assembly of figure 5a, e.g., removal of A means the simultaneous disestablishment of four connections, and the formulation of precedence relations becomes questionable.

Detachability of ABCD is tested by investigating the detachability of E without removal of A through D. This appears possible. On the other hand, ABCE is not detachable because D can not be removed prior to removal of E or (B and C). B, C, and E are in the subassembly and these are, consequently, not removed. This is found by inserting \( R_A, R_B, R_C, \text{and } R_E = \text{false} \) in the expression of the l.h.s. of the precedence relation for \( R_D \). The expression on the left is false then, thus D can not be removed. Consequently, the subassembly ABCE is not detachable.
It indeed appears possible to determine the undetachable subassemblies straight from the Boolean expressions that represent the precedence relations, without any additional information on assembly structure. One obtains: ABD, ACD, BCE, AD, and BE. Apart from 5 parts and one original assembly, we end up with 12 non-trivial subassemblies.

![Disassembly graph](https://www.spiedigitallibrary.org/conference-proceedings-of-spie)

Figure 6. Disassembly graph of the assembly depicted in figure 5.

This is indicated in figure 6. Here, each action is indicated by one arc only, because this does not introduce ambiguities. ABCD > AC means: ABCD > AC + BD, etc.

The graph is restricted to transformations into 2 child subassemblies. The number of actions is decreased by the omission of disjunct subassemblies from 346 to 43, corresponding to a reduction of 88% of the original amount. A further reduction is obtained by omission of the remaining undetachable subassemblies. Their omission reduces the number of possible actions with another 41% to 27. The actions and subassemblies thus disappearing are indicated by hairlines.

### 3.3. More complex case from practice

The method should also be justified by applying it to more complex assemblies. We have applied it to the assembly that is discussed in [13] and [14], and that is frequently referred to in the literature. It is presented in figure 7a. Its connection diagram is given in figure 7b. From this, a listing of disjunct subassemblies is easily retrieved. The precedence relations can be derived from studying figure 7a and are listed below. For each part, one lists the parts that are directly blocking this part. Rather than by analysing disconnections, this set of precedence relations is used to derive the set of feasible subassemblies.

The listing of precedence relations reads:

\[
\begin{align*}
(R_G \text{ and } R_K \text{ and } R_L) & \text{ or } (R_B \text{ and } R_C \text{ and } R_D) > R_A \\
(R_A \text{ and } R_G \text{ and } R_H \text{ and } R_I) & \text{ or } (R_C \text{ and } R_E) > R_B \\
(R_A \text{ and } R_B) & \text{ or } (R_D \text{ and } R_J) > R_C \\
(R_A \text{ and } R_C) & \text{ or } R_E > R_D \\
(R_B \text{ and } R_C) & \text{ or } R_F > R_E \\
R_F & \\
(R_A \text{ and } R_B) & \text{ or } R_H > R_G \\
R_G & \text{ or } (R_I \text{ and } R_K) > R_H \\
(R_B \text{ and } R_H) & \text{ or } R_I > R_I \\
R_I \text{ or } & R_L > R_K \\
R_L & 
\end{align*}
\]
Figure 7. 7a. The example assembly used by De Fazio and Whitney\textsuperscript{13,14}. 7b. Connection diagram.

It should be stressed that only blocking of removal paths is considered here. Additional blocking can be present. For instance, when D is connected to A by bolts, A and D can never be disconnected before E is disconnected from the assembly. Such connections have been included in the example of De Fazio et al., and the resulting additional restrictions can easily be included in the set of precedence relations that has been listed already in this paper. Consistency of the relations might be checked. A necessary condition is, e.g., that all parts should be detachable. This is indeed true, e.g. by following the sequence: $R_A, R_K, R_J, R_H, R_G, R_F, R_E, R_D, R_C, R_A$, and $R_B$.

AB appears a detachable subassembly, because all parts that do not belong to AB can be removed. Let us investigate detachability of the subassembly AC. Therefore, one inserts $R_A = \text{false}$ and $R_C = \text{false}$ in the l.h.s. of all precedence relations, apart from those for $R_A$ and $R_C$. Inevitably, the l.h.s. of the precedence relation for $R_B$ is false, thus B can not be removed and AC is not detachable.

Similarly, when BJ is considered, one ends up with:

\begin{align*}
R_H & > R_G \\
R_G & > R_H
\end{align*}

which is also not executable.

A subassembly such as EF can be obtained by subsequent removal of all other parts, e.g. by the sequence: $R_L, R_K, R_J, R_H, R_G, R_A, R_B, R_C, R_D$. However, this is not a necessary condition, although it is possible here. Suppose that detachment of the remaining parts is not possible. In this case, one has to investigate whether EF can be detached as a whole. From the connection diagram, one observes that E is a cut-vertex or, in other words\textsuperscript{15}, E is a carrier of F.

$R_F > R_E$  
$R_F$

These relations do not exclude the simultaneous detachment of EF.

Theoretically, there exist maximally \(2^{11} - 1 = 2047\) subassemblies. For testing the reduction procedure, we considered the subassemblies consisting of 2 parts. Theoretically, 55 are possible. Only 19 of these are coherent, corresponding to the number of connections. From these, 10 detachable subassemblies remain, viz. AB, AG, AK, BC, CD, EF, GH, HJ, HK, JL.

There are, theoretically, 165 combinations of 3 parts out of 11. Only 46 of these are coherent. 11 of them are detachable, according to the Boolean expressions, namely: ABC, ABG, ABK, AGH, AGK, BCD, CDE, GHJ, GHK, HJK, HJL.

A more radical reduction appears when subassemblies of many parts are considered. Theoretically, there are 462 subassemblies that are composed of 5-parts. Only 10 of them appear both coherent and detachable, namely: ABCDE, ABCDG, ABCDK, ABCGH, ABCGK, ABGHJ, ABGHK, AGHJK, BCDEF, and GHJKL.
After this substantial reduction, all actions that decompose any 5-part subassembly into a 2-part subassembly and a 3-part subassembly can be derived. There are 19 possible actions that decompose a 5-part subassembly into one 2-part subassembly and one 3-part subassembly, out of a theoretical amount of 4620. These are listed in Table 3:

<table>
<thead>
<tr>
<th>Action</th>
<th>Action</th>
<th>Action</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABCDE &gt; CDE + AB</td>
<td>ABCGH &gt; ABC + GH</td>
<td>ABGHK &gt; ABG + HK</td>
<td>AGHJK &gt; HJK + AG</td>
</tr>
<tr>
<td>ABCDG &gt; ABG + CD</td>
<td>ABCGH &gt; AGH + BC</td>
<td>ABGHK &gt; ABK + GH</td>
<td>BCDEF &gt; BCD + EF</td>
</tr>
<tr>
<td>ABCDG &gt; BCD + AG</td>
<td>ABCGK &gt; AGK + BC</td>
<td>ABGHK &gt; GHK + AB</td>
<td>BCDEF &gt; DEF + BC</td>
</tr>
<tr>
<td>ABCDK &gt; ABK + CD</td>
<td>ABGHJ &gt; ABG + HJ</td>
<td>AGHJK &gt; AGK + HJ</td>
<td>GHJKL &gt; GHK + JL</td>
</tr>
<tr>
<td>ABCDK &gt; BCD + AK</td>
<td>ABGHJ &gt; GHJ + AB</td>
<td>AGHJK &gt; GHJ + AK</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Possible decompositions of 5-part subassemblies in one 2-part and one 3-part subassembly.

The test turns out that, by the application of a modest set of Boolean expressions, rather complicated problems can be easily reduced. An adequate reduction of the number of actions and subassemblies in turn considerably reduces the size of the transition matrix. It turns out that most of the required steps can be formalised. Straightforward combinatorics and Boolean algebra is sufficient for automatic selection of the possible subassemblies and actions. This suffices for automatic generation of the T-matrix.

CONCLUSIONS

In this paper, a method for determining the optimum disassembly sequence has been reviewed, and some applications and extensions have been listed. By using this method, the calculation is straightforward. However, the determination of the disassembly graph from the assembly drawing, prior to calculation of the model, remains elaborate. A disassembly graph, however, is a prerequisite for establishing the transition matrix, which contains information on structure and precedence relations of the original assembly. In the literature, transitions are often defined as sequences of disconnections. In more complicated cases this approach implies ambiguities, because parts are subjected to many connections that have to be disconnected simultaneously. Disassembly graphs consist of subassemblies and actions. Once the possible subassemblies are listed, such graphs can be constructed. A subassembly appears in a disassembly graph when it is both coherent and detachable. Coherence is investigated using the connection diagram. Detachability is investigated using the precedence relations. This proceeds straightforwardly if actions are defined by the separation into two or more subassemblies rather than by disestablishment of connections. Even in the case of complicated assemblies, it appears possible to express the precedence relations in a line of Boolean expressions for each part. Although some manual testing might be required, in a number of various practical examples that we have investigated, the listing of subassemblies and actions and, consequently, the generation of the disassembly graph can be automatically performed up to a large extent. Only the generation and testing of precedence relations in the terms of Boolean expressions from the assembly drawings should be performed manually.

REFERENCES


