Computer-aided Subassembly Generation

Arkady N. Bozhko[1], Anatoly P. Karpenko [2]

[1] *CAD, BMSTU,*
5/1, Baumanskaya 2-ya
Moscow, Russia
E-mail: abozhko@inbox.ru

[2] *CAD, BMSTU,*
5/1, Baumanskaya 2-ya
Moscow, Russia
E-mail: apkarpenko@mail.ru

Abstract

The article deals with the task of product decomposition into assembly units in CAAP systems. This is an important design decision that allows to rationally organize the process of assembling complex products and improve economic indexes of production. A hypergraph model of the assembly structure of a product is proposed. It is shown that decomposition of a product can be described as cutting a hypergraph into subgraphs. The problem of rational cutting is formulated in the terms of discrete mathematical programming. The main constraints are given, and objective functions which allow to select an optimal decomposition in the given production conditions are formulated.

Keywords: Computer-aided assembly planning, hypergraph, assembly units, subassembly generation, discrete mathematical programming.

1. Introduction

Splitting a product into assembly units is an important engineering task in modern design and production. Many technical and economic characteristics of the assembly process depend on the chosen product decomposition. This design decision affects work site arrangement and floorspace layout scheme.

In the field of computer-aided assembly planning (CAAP) studies, comparatively little attention is paid to automated synthesis of the product decomposition scheme into assembly units. In the overwhelming majority of publications on CAAP, an assembly unit is considered not as a self-sufficient phenomenon, but as a state of the product during the assembly process. In the detailed review [1] dedicated to this problem product decomposition is considered only in five of 96 cited sources.

In several papers, for example in [2 - 4], an attempt was made to apply cluster analysis for splitting the product into assembly units. The authors of these studies base their work on similarity of appearance of cluster taxonomies and assembly decompositions. The authors believe that strong internal connectivity of parts in a product is sufficient to cosider such a group as an assembly unit. In assembling, as a rule, this criterion is not correct. Special rules of the assembly process cancel the general rules of cluster analysis that are valid for poorly structured systems and poorly researched subject areas. The conditions of existence of assembly units are independent assembling, closure of dimension chains, stability of parts etc.

The problem of synthesizing assembly units is considered in detail in [5]. A subset of parts is considered an assembly unit if conditions for geometric solvability there are no geometric obstacles), connectivity, and stability are satisfied for them. Necessary design information about the product is recorded in the form of Interference Matrices, Contact Matrices and Connection matrices. An algorithm for generating assembly units for these matrices is described in the paper. This method is very labor intensive. Besides, it does not consider the system of dimension chains on which rational decomposition of the product into assembly units depends.

In [6] splitting a product into assembly units is considered not as an important and independent design decision, but as a way to reduce combinatorial complexity of assembly sequence synthesis. A combinatorial decomposition algorithm based on selection and subsequent integration of weighted undirected connected graphs is proposed in the paper. The validity of the al-

gorithm is not supported by formal calculations or examples, so this approach gives rise to serious doubts.

In [7] the authors propose a method for synthesizing assembly units based on an oriented graph which specifies precedence relations among parts in the process of product assembly (precedence knowledge). The work does not consider the method of obtaining this model; it is believed that it is known a priori. The method is oriented to subassembly of car bodies and is not universal.

Splitting a product into assembly units can be thought of as cutting the connection graph or its modifications. The problem lies in the search for such a cutting whose elements are subgraphs describing the collected and stable subsets of the details. This idea is the basis of the majority of papers dedicated to synthesis of assembly decomposition. For example, in [8] information about the properties of the product is formalized in the form of a relational graph. This model is a liaison graph whose vertices are fragments of a relational database. Each such fragment stores design information about the part that is necessary for designing an assembly unit. The authors use a heuristic approach to the definition of an assembly unit. For example, this is a set of parts that forms a simple or compound loop in the relational graph all the edges of which are screw joints. Other substructures of the relational graph that can have high connectivity and stability are also considered.

In [9] algorithmic and program realization of this approach is described. In general, cyclic structures of the relational graph and the liaison graph do not guarantee assemblability. On the contrary, they are often a source of assembly problems, since they can generate unsolvable dimension chains.

The cutting method was fairly extensively investigated in [10, 11]. The main carrier of information about the product in these works is the attributed liaison graph. In fact, this graph is a semantic network, the edges and vertices of which are connected to frames that store design information about the details and mechanical connections. The slots of the edge frame record data containing the contact form, names of contact details, type of joint (separable, permanent), form of joint (pin, key, etc.) and others. Slots of the vertex frame are filled with the following information about the details: geometrical properties (types of boundary surfaces, dimensions, etc.), physical parameters (weight, accuracy, etc.), coordinates and orientation of the described rectangle (gizmo), list of contacting parts and many others. These data are used to discard the unrealizable and nonrational solutions. It should be noted that the attributed liaison graph is a very cumbersome model. Its creation requires such a lot of manual labor inputs that it may be compared with the traditional approach to splitting a product into assembly units.

The problem of synthesizing assembly units was extensively investigated in [12]. The product's structural properties are presented in a liaison graph whose vertices and edges are divided into groups, depending on the tasks they perform in the product. The vertices of the graph describe the details of the product and the assembly units. The parts are divided into functional parts and attachment parts. The former serve as carriers of executive surfaces, the latter are used for positioning and for functional parts. Joints and mating between parts are represented as edges of different types of the graph. Among all studies based on cutting of the liaison graph, the last two papers differ in profundity and elaboration of nuances.

The liaison graph and its numerous modifications are binary mathematical objects. These means can not adequately describe spatial coordination of parts in a product and an assembly unit which, in general, is a *k*-dimensional relation, $k \geq 2$. This is the main shortcoming of the papers where the structure of a product is presented in the form of graphs or networks.

In our papers [13, 14] we propose an apparatus of *s*-hypergraphs, and it is shown that *s*-hypergraphs adequately describe the behavior of technical systems during assembly and disassembly.

The article deals with application of this model for the computer-aided planning of rational product decompositions into assembly units.

2. Hypergraphic product model

We associate the product X with the hypergraph $WS = (X, R)$ in which the set of vertices $X = \{x_i\}_{i=1}^{n}$ represents the parts, and the set of hyperedges $R = \{r_j\}_{j=1}^{m}$ represents minimal coordinated groupings of parts obtained by realizing internal mechanical connections [13,14].

Figure 1 shows an example of the product and its hypergraph.

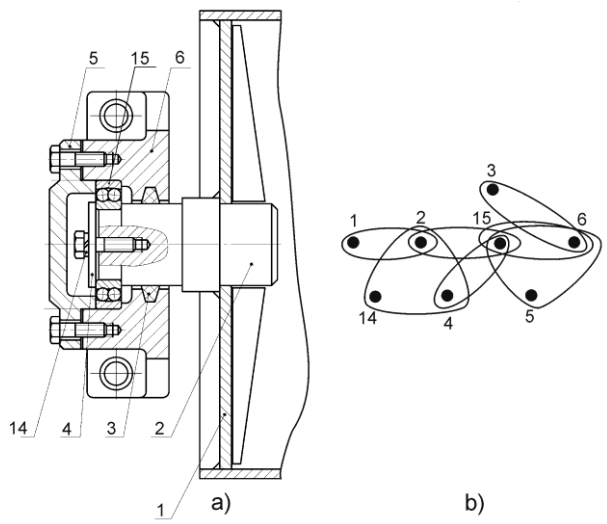

Fig. 1. Example of the product (a), hypergraph of the product (b).

Definition 1. An assembly operation is called *n*-handed, if it requires simultaneous and independent movement of *n* operating devices (hands) [15].

Figures 2 shows an example of an assembly operation that requires three hands (3-handed).

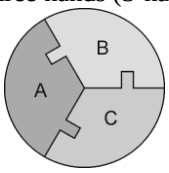

Fig. 2. An example a 3-handed assembly operation.

Definition 2. An assembly operation is called coherent if it implements at least one mechanical connection [15].

Figures 3 shows an example an incoherent assembly operation.

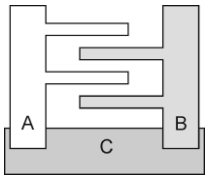

Fig. 3. An example of an incoherent assembly operation.

Obviously, for the installation of parts A and B on C, preliminary coordination of A and B is required, which is performed without contact between them.

The paper considers only coherent and 2-handed assembly operations that prevail in modern industrial production.

Definition 3. Normal contraction is merging of two vertices of a hypergraph which are connected by a hyperedge of the second degree (edge) and removal of this hyperedge.

Definition 4. A hypergraph *WS* = (*X*, *R*) is called an *s*-hypergraph if it can be transformed into a single-vertex hypergraph without loops by way of some sequence of normal contraction [13, 14].

Theorem 1. *If the hypergraph WS = (X, R) is a s-hypergraph, then it is connected, has at least one edge and* $|X| = |R| + 1$ [13,14].

Theorem 2. *Let WS = (X, R) be a s-hypergraph. Any subgraph* $H = (X_H, R_H)$ *of a hypergraph WS that includes at least one edge and for which the linear constraint* $|X_H| = |R_H| + 1$ *is valid, is connected and contractible (s-hypergraph) [14].*

3. Mathematical model of rational decomposition of the product into assembly units

The assembly unit (AU) must be independently assemblable, so its mathematical description is the *s*-subgraph of the contractible hypergraph *WS* associated with the product. It is obvious that any decomposition can be represented as cutting a *s*-hypergraph into *s*-subgraphs. The problem of synthesis of rational decomposition is put in the terms of discrete mathematical programming.

3.1. Used variables

Let us assume that the assembly structure of some product $X = \{x_i\}_{i=1}^{n}$ is represented as a *s*-hypergraph *WS* = (*X*, *R*). We denote $\overline{A} = \| a_{ij} \|_{n}^{n-1}$ – the incidence matrix *WS* in which $a_{ij} = 1$ only if the vertex x_i is connected to the hyperedge r_j. Let l be the number of assembly units.

A hyperedge of the hypergraph *WS* that is incident to two vertices will be called edge.

We introduce the variables:

$$x_{ij} = \begin{cases} 1, \text{ if } i\text{-part enters } j\text{-AU}; \\ 0, \text{ if not.} \end{cases}$$

$$y_{kj} = \begin{cases} 1, \text{ if } k\text{-hyperedge enters } j\text{-AU}; \\ 0, \text{ if not.} \end{cases}$$

$$z_{mj} = \begin{cases} 1, \text{ if } m\text{-edge enters } j\text{-AU}; \\ 0, \text{ if not.} \end{cases}$$

$$s_j = \begin{cases} 1, \text{ if } j\text{-AU is not degenerated}; \\ 0, \text{ if not.} \end{cases}$$

$$i = \overline{1,n},\ k = \overline{1,K},\ m = \overline{1,M},\ K + M = n - 1,\ j = \overline{1,S}.$$

The assembly unit is considered as nondegenerate if it includes at least two parts.

3.2. Basic constraints

Using the variables indicated in 3.1, we write a set of basic constraints (1) - (11) that formalize the rules for cutting the *s*-hypergraph into *s*-subgraphs. Following each of these constraints, we produce its meaning.

$$\sum_{j=1}^{S} x_{ij} \le 1,\ i = \overline{1,n} \qquad (1)$$

Each part enters only one AU or goes to a general assembly, bypassing inclusion in intermediate assembly units.

$$\sum_{j=1}^{S}\sum_{i=1}^{n} x_{ij} \ge 2 \qquad (2)$$

Decomposition into assembly units is not trivial. This means that it contains at least one nonempty assembly unit.

$$\sum_{j=1}^{S} y_{kj} \le 1,\ k = \overline{1,K}. \qquad (3)$$

Each hyperedge enters one AU, or connects different AUs.

$$\sum_{j=1}^{S} z_{mj} \le 1,\ m = \overline{1,M}. \qquad (4)$$

Each edge enters one assembly unit or connects different AUs.

$$\sum_{m=1}^{M} z_{mj} \ge 1,\ j = \overline{1,S}. \qquad (5)$$

Each AU contains at least one edge (the second condition of Theorem 1) (5).

$$\sum_{i=1}^{n} x_{ij} = \sum_{k=1}^{K} y_{kj} + \sum_{m=1}^{M} z_{mj} + 1,\ j = \overline{1,S}. \qquad (6)$$

For each AU condition 3 of Theorem 1 must be satisfied.

Connectivity of the subgraphs that describe the assembly units is guaranteed by Theorem 2, so for this feature there is no need to introduce a special constraint.

$$y_{kj} = 1 \Rightarrow \sum_{i=1}^{n} x_{ij} \times a_{ik} = \sum_{i=1}^{n} a_{ik};\ k = \overline{1,K},\ j = \overline{1,S}. \qquad (7)$$

The expression (7) formalizes the topological condition: if the hyperedge *k* enters the subgraph *j*, then all the vertices incident to this hyperedge enter the given subgraph.

The constraint (7) is written in a logical form using the «if … then» operation. We introduce auxiliary variables $t_{kj} \in \{0,1\}$, $k = \overline{1,K}$, $j = \overline{1,S}$. Then in the algebraic form this constraint takes the form

$$y_{kj} = t_{kj}, \sum_{i=1}^{n} x_{ij} a_{ik} \ge \sum_{i=1}^{n} a_{ik} t_{kj};\ k = \overline{1,K},\ j = \overline{1,S}.$$

This entry is correct. Indeed, for t_{kj} = 1 the original constraints are transformed into a system of inequalities $\sum_{i=1}^{n} x_{ij} a_{ik} \ge \sum_{i=1}^{n} a_{ik}$, which can be fulfilled only as equalities, which produces the initial subsystem of constraints. For t_{kj} = 0 the inequalities $\sum_{i=1}^{n} x_{ij} a_{ik} \ge 0$ are trivial.

$$z_{mj} = 1 \Rightarrow \sum_{i=1}^{n} x_{ij} \times a_{im} = \sum_{i=1}^{n} a_{im};\ m = \overline{1,M},\ j = \overline{1,S}. \qquad (8)$$

If the edge *m* enters the subgraph *j*, then all the incident vertices of the edge enter this subgraph.

By analogy with (7), the constraint (8) in the algebraic form has the form

$$z_{mj} = \tau_{mj}, \sum_{i=1}^{n} x_{ij} a_{im} \ge \sum_{i=1}^{n} a_{im} \tau_{mj};\ m = \overline{1,M},\ j = \overline{1,S}.$$

Here $\tau_{kj} \in \{0,1\}$, $m = \overline{1,M}$, $j = \overline{1,S}$ are auxiliary variables.

$$\sum_{i=1}^{n} x_{ij} \times a_{ik} = \sum_{i=1}^{n} a_{ik} \Rightarrow y_{kj} = 1 \quad k = \overline{1,K},\ j = \overline{1,S}. \qquad (9)$$

If all the incident vertices of the hyperedge *k* belong to the subgraph *j*, then the hyperedge itself must belong to this subgraph. The logical constraint (9) in the algebraic form has the form

$$\left(\sum_{i=1}^{n} x_{ij} \times a_{ik} - \sum_{i=1}^{n} a_{ik}\right) \times y_{kj} \ge 0,\ \sum_{i=1}^{n} x_{ij} \times a_{ik} - \sum_{i=1}^{n} a_{ik} + 1 \le y_{kj},$$
$$k = \overline{1,K},\ j = \overline{1,S}.$$

It is easy to verify that this information is correct.

$$\sum_{i=1}^{n} x_{ij} \times a_{im} = \sum_{i=1}^{n} a_{im} \Rightarrow z_{mj} = 1;\ m = \overline{1,M},\ j = \overline{1,S}\ . \qquad (10)$$

If all the incident vertexes of the edge *m* belong to the subgraph *j*, then the edge itself must belong to this subgraph (10). This logical constraint in the algebraic form has the form

$$\left(\sum_{i=1}^{n} x_{ij} \times a_{im} - \sum_{i=1}^{n} a_{im}\right) \times z_{mj} \ge 0,\ \sum_{i=1}^{n} x_{ij} \times a_{im} - \sum_{i=1}^{n} a_{im} + 1 \le z_{mj},$$
$$m = \overline{1,M},\ j = \overline{1,S}.$$

$$\sum_{i=1}^{n} x_{ij} \ge 2 \rightarrow s_j = 1 \text{ и } \sum_{i=1}^{n} x_{ij} < 2 \rightarrow s_j = 0,\ j = \overline{1,S}. \qquad (11)$$

If the AU with the number j is not degenerate, then the indicator variable s_j is one. The logical constraint (11) in the algebraic form can be written in the form

$$\left(\sum_{i=1}^{n} x_{ij} - 2\right) \times s_j \geq 0,\ \sum_{i=1}^{n} x_{ij} - 2 < n \times s_j.$$

3.3. Additional constraints

The set of constraints (1) – (11) is open. It can be expanded by means of additional conditions (12) – (16) that into account the specifics of decision-making in a given production situation.

$$x_{pj} = x_{rj} = 1 \tag{12}$$

The equation (12) requires that the parts with numbers p and r belong to one assembly unit. This condition is mandatory if these details enter into one design dimension chain.

$$x_{pj} + x_{rj} < 2. \tag{13}$$

Inequality (13) forbids the details with numbers p and r to enter into one assembly unit.

$$\sum_{j=1}^{S} x_{ij} = 0. \tag{14}$$

Condition (14) means that the part with the number i does not enter into any assembly unit.

$$x_{rj} \geq x_{pj}. \tag{15}$$

If the part with the number p enters the assembly unit j, then the part with the number r must be included in it.

$$N_j^{\min} \leq \sum_{i=1}^{n} x_{ij} \leq N_j^{\max}. \tag{16}$$

Inequality (16) sets constraints on the capacity of the assembly unit. Here $N_j^{\min}$ ($N_j^{\max}$) is the minimum (maximum) number of parts in AU_i.

3.4. Objective functions

To synthesize rational product decompositions into assembly units, the following objective functions (17) – (23) can be used.

$$\sum_{j=1}^{S} \sum_{i=1}^{n} x_{ij} \to \max. \tag{17}$$

The objective function (17) selects such a decomposition in which the maximum number of parts is included in assembly units.

$$\sum_{j=1}^{S} s_j \to \max. \tag{18}$$

The function (18) provides decomposition with the maximum number of assembly units.

$$\sum_{k=1}^{K} \sum_{j=1}^{S} y_{kj} \to \max. \tag{19}$$

The function (19) is aimed at creating a decomposition in which the maximum number of hyperedges enters into subgraphs. This means that connection of high multiplicity should be realized earlier when complexity of AUs is not yet very high.

$$\sum_{k=1}^{K} \sum_{j=1}^{S} y_{kj} + \sum_{m=1}^{M} \sum_{j=1}^{S} z_{mj} \to \max. \tag{20}$$

The function (20) formalizes the fact that the number of edges and hyperedges included in assembly units must be maximal. Since the total number of links in the hypergraph does not change, the objective function (20) minimizes the number of connections between individual AUs. This function works in a similar way to the criteria of the cluster analysis where the set of objects is divided into subsets (clusters) in which the «strength» of internal connections exceeds the external one.

$$\sum_{k=1}^{K} \sum_{j=1}^{S} c_k y_{kj} + \sum_{m=1}^{M} \sum_{j=1}^{S} c_m z_{mj} \to \max. \tag{21}$$

Here c_k, (c_m) is the number of different surfaces that must be brought into contact to implement a constraint represented by the k-hyperedge (m-edge). These numbers are indirect indicators of complexity of the connection. For the final assembly it is better to leave such connections that do not require complex spatial coordination over several surfaces simultaneously. The criterion (21) maximizes the total number of conjugated surfaces in assembly units. This automatically reduces the number of contacts in the general assembly phase.

$$\sum_{j=1}^{S-1} \sum_{k=j+1}^{S} \left(\left| \sum_{i=1}^{n} x_{ij} - \sum_{i=1}^{n} x_{ik} \right| \right) \to \min. \tag{22}$$

The criterion (22) formalizes a well-known rule, according to which the number of parts in different AUs should not be very different

$$\max_j \left(\sum_{k=1}^{K} t_k y_{kj} + \sum_{m=1}^{M} t_m z_{mj} \right) \to \min. \tag{23}$$

Here t_i, $i = \overline{1, K+M}$ is the time of realization of the connection represented by the edge (hyperedge) with the number i. The criterion (23) serves to select a decomposition with the minimal assembling cycle.

To solve the task of discrete mathematical programming, it is possible to apply both classical methods

(Lagrange's method of undetermined multipliers, branch and bound method, etc.) and discrete versions of modern optimization algorithms inspired by nature [16].

References

1. Wang Y., Liu J., Subassembly identification for assembly sequence planning, *The International Journal of Advanced Manufacturing Technology*. Volume 68, Issue 1–4. (2013) 781 – 793. DOI:10.1007/s00170-013-4799-y.
2. Deshmukh A., Yung P., Wang H-P., Automated generation of assembly sequence based on geometric and functional reasoning, *Journal of Intelligent Manufacturing*, Vol 4, Issue 4. (1993) 269–284. DOI:10.1007/BF00124140.
3. Hemmskerk C., Van Luttervelt C., The use of heuristics in assembly sequence planning, *CIRP Annals - Manufacturing Technology*. Volume 38, Issue 1. (1989) 37–40. DOI:10.1016/s0007-8506(07)62647-x.
4. O'Shea B., Kaebernick H., Grewal S., Using a cluster graph representation of products for application in the disassembly process planning , *Concurrent Engineering*, Volume 8, Issue 3. (2000) 158–170. DOI:10.1177/1063293x0000800301.
5. Dini G., Santochi M., Automated sequencing subassembly detection in assembly planning, *CIRP Annals – Manufacturing Technology*, Volume 41, Issue 1. (1992) 1–4. DOI:10.1016/s0007-8506(07)61140-8.
6. Cao Y., Kou X., Cao S., A sub-assembly identification algorithm for assembly sequence planning, in *Proc. International Industrial Informatics and Computer Engineering Conference*. 2015. DOI:10.2991/iiicec-15.2015.127.
7. Zhang Y., Ni J., Lai X., Automated sequencing and subassembly detection in automobile bod assembly planning, *Journal of Materials Processing Technology*, Volume 129, Issue 1 – 3. (2002) 490–494.
8. Zussman E., Lenz E., Shpitalni M., An Approach to the Automatic Assembly Planning Problem, *CIRP Annals – Manufacturing Technology*, Volume 39, Issue 1. (1990) 33–36. DOI:10.1016/s0007-8506(07)60997-4.
9. Ong N., Wong Y., Automatic subassembly detection from product model for disassembly sequence generation, *The International Journal of Advanced Manufacturing Technology*. Volume 15, Issue 6. (1999) 425–431. DOI:10.1007/s001700050086.
10. Lee S. Subassembly identification and evaluation for assembly planning, I*EEE Transactions on Systems, Man and Cybernetics*, Volume 24, Issue 3. (1994) 493–503. DOI:10.1109/21.278997.
11. Lee S., Shin Y., Assembly planning based on subassembly extraction, in *Proceedings of IEEE International Conference "Robotics and Automation"*. 1990. V. 3. Pp. 1606–1611. DOI:10.1109/ROBOT.1990.126239.
12. Marian R., Luong L., Abhary K., A genetic algorithm for optimization of assembly sequences, *Computers & Industrial Engineering*, Volume 50, Issue 4. (2006) 503–527. DOI:10.1016/j.cie.2005.07.007.
13. Bozhko A.N., Algebraic models of assembly of products, *Nauka i obrazovaniye. MGTU im. N.E. Baumana.* Elektron. zhurn. № 12. (2016). DOI: 10.7463/1216.0852565 (in Russian).
14. Bozhko A. N. Strukturnyye modeli sobirayemosti izdeliy [Structural models of assembly of products] // Nauka i obrazovaniye. MGTU im. N.E. Baumana. Elektron. zhurn. 2013. №10. DOI:107463/1013.0622946 (in Russian).
15. Ghandi S., Masehian El., Review and taxonomies of assembly and disassembly path planning problems and approaches, *Computer-Aided Design*, Vol. 67 – 68. (2015) 58–86. DOI:10.1016/j.cad.2015.05.001.
16. Karpenko A.P., *Modern algorithms of search optimization. Algorithms inspired by nature: a tutorial* (Moscow, BMSTU Publ., 2014). 446 p. (in Russian).