Int J Adv Manuf Technol (2009) 45:808–820
DOI 10.1007/s00170-009-1993-z

ORIGINAL ARTICLE

Assembly sequences merging based on assembly unit partitioning

Y. Wang · J. H. Liu · L. S. Li

Received: 11 November 2008 / Accepted: 24 February 2009 / Published online: 14 March 2009

Abstract Assembly sequence planning is a typical of combinatorial optimization problem which is difficult to be tackled when the number of parts of assembly becomes large. To reduce the searching space of assembly sequence planning of complex products, assembly sequences merging based on assembly unit partitioning is suggested. Assembly unit partitioning is presented to decompose the complex products into a group of assembly units containing a reduced number of parts or components, and the assembly design constraints and the assembly process constraints are comprehensively taken into account. The global optimal assembly sequences can be acquired through three steps. Firstly, the assembly units and decision graph of assembly unit are generated utilizing fuzzy analytical hierarchy process approach. Secondly, the optimal or near-optimal subsequences of assembly units can be obtained with current efficient methods of assembly sequence planning. Thirdly, under the assembly interference of assembly relations (geometrical constraints) of the whole products and the assembly precedence concluded by subsequences of assembly units, the assembly sequence merging is implemented to generate the global assembly sequences, and the optimal sequence is obtained through assembly sequences evaluation. The assembly constraints considered at the two previous steps is represented by the evaluation function. The effectiveness of the method is verified by an illustrative example and the results show that the searching space of assembly sequence merging of complex products is reduced remarkably and the optimal assembly sequence of the whole produces is obtained.

Keywords Assembly sequence planning · Assembly sequences merging · Assembly unit partitioning · Decision graph of assembly unit · Assembly interference of assembly relations

1 Introduction

Assembly sequence planning of complex products is apt to get into a hobble of combinatorial explosion because the number of assembly sequences is exponentially proportional to the number of parts or components of the product. Collaborative assembly planning [1, 2] is advocated to decompose the assembly planning task of complex products into several simpler planning tasks and the combinatorial explosion problem of assembly sequence planning is skipped. To reduce the number of independent parts or components for assembly sequence planning, the subassembly identification and extraction is widely studied by many researchers.

The methods of subassembly identification and extraction can be classified into two types: top-down approach and bottom-up approach. The top-down approach is that the subassemblies are identified and extracted referring to the structural integrity of products. The components or subassemblies are often chosen in view of the assembly design demands [3]. Most of these methods depend on domain-specific knowledge and the subassemblies are mainly defined by product designers.

Y. Wang · J. H. Liu (✉) · L. S. Li
School of Mechanical Engineering and Automation,
Beihang University,
No.37, XueYuan Road, HaiDian District,
Beijing 100083, People's Republic of China
e-mail: ryukeiko@buaa.edu.cn

Y. Wang
e-mail: wangyong@me.buaa.edu.cn

L. S. Li
e-mail: liliansheng1981@163.com

The subassemblies can also be generated with respect to assembly connections. Chakrabarty et al. [4] groups the parts assembled with fasteners into subassemblies and the subassemblies are reused to identify the other subassemblies with the similar assembly structure. Ong et al. [5] identifies the subassemblies using the concepts and property of "cut-node" of assembly connection diagram, and the subassemblies are generated when the "cut-node" is removed from the connection diagram. With top-down method, the assembly design constraints and the structural integrity of products are taken into consideration while the assembly process constraints are not taken into account substantially.

As a reverse, the subassemblies are identified according to the parts and the assembly relations between them with the bottom-up method, and the assembly process constraints and the local assembly constraints play an important role. Lambert [6] suggests that a subassembly is composed of the parts that are connected directly and achieve a particular function. Lee [7] quantifies the connections strength classified by them and the degree of freedom of the assembled parts is also quantified. The subassembly is obtained by evaluating the assembly cost weight between parts. Mejbri [8] assigns the parts assembled with assembly tolerance to the same subassembly. Therefore, the local assembly structure and the assembly process constraints are taken into account with the bottom-up methods, but the assembly design constrains are usually ignored.

After the subassembly identification process, the simpler assembly sequence planning tasks can be completed using the efficient approaches of assembly sequence planning and the corresponding optimal or near-optimal assembly sequences are detected. In particular, complex products are often partitioned into a group of subassemblies and the subsequences of subassemblies are obtained first by assembly sequence planning. The global assembly sequences of complex products are generated by assembly sequences merging methods. The assembly constraints considered in the previous assembly sequence planning process should be regarded in assembly sequences merging process. Most assembly sequences merging methods need a number of human-machine interactions. Chakrabarty et al. [4] and Swaminathan et al. [9] presume that the ordering of the identical parts in two subsequences is consistent, and the optimal or near-optimal assembly sequences are acquired by adjusting the assembly directions of parts in different subsequences. These methods are also applied for collaborative assembly planning by Wang et al. [1] and Dong et al. [2].

The assembly sequence merging based on assembly unit partitioning is proposed to generate the optimal or near-optimal assembly sequences of complex products. Assembly unit partitioning is used to decompose the complex products into assembly units containing smaller number of parts or subassemblies for assembly sequence planning. The assembly constraints including assembly design constraints and assembly process constraints are comprehensively taken into account and attached to the assembly relations. The fuzzy analytical hierarchy process method (FAHP) is used to compute the decision values of assembly relations according to the above assembly constraints and the decision graph of assembly unit is generated. The decision values of assembly relations are viewed as one of the crucial indices to extract assembly units. After the optimal or near-optimal subsequences of assembly units are acquired, the assembly sequences merging are implemented to produce the global assembly sequences of the whole products. The decision values of assembly relations are deemed as one type of heuristic information to detect the optimal assembly sequences. Moreover, the longest continuous fragments of optimal or near-optimal subsequences are also maintained in the global optimal assembly sequences. The assembly constraints considered in the assembly unit partitioning process and previous assembly sequence planning process have a great effect on the assembly sequences merging, and the consistency of the assembly units, assembly subsequences and global assembly sequences of product are maintained.

2 Related concept

2.1 Definition of assembly unit partitioning and assembly sequences merging

Assembly unit partitioning It is the approaches or technologies to decompose the complex assembly into a group of assembly units under appointed assembly constraints, and each assembly unit is composed of less parts or components than the assembly.

Assembly sequences merging It is the approaches or technologies to merge the subsequences of assembly units to generate the global assembly sequences of the whole products conforming to the geometrical constraints of assembly and precedence constraints of assembly subsequences of assembly units.

The assembly constraints considered in assembly unit partitioning process includes assembly design constraints and assembly process constraints besides the geometrical constraints and assembly precedence constraints. These assembly constraints are classified and presented in Fig. 1. These assembly constraints are quantified and attached to the assembly relations. The FAHP method is used to compute the decision values of assembly relations and the

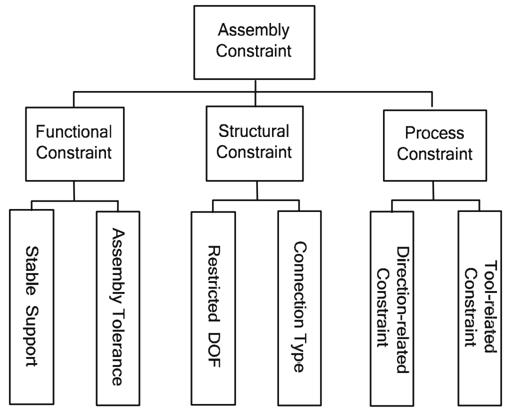


Fig. 1 Constraints for assembly unit partitioning

decision graph of assembly unit is generated to aid to seek the assembly units.

Many assembly design constraints and process constraints are considered in the assembly unit partitioning process and assembly sequence planning process [10], these constraints should be considered in the assembly sequences merging process to preserve the consistency of assembly units, assembly subsequences and global assembly sequences.

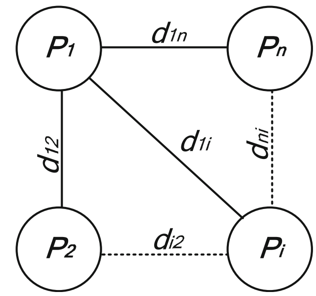
Therefore, the global assembly sequences should satisfy the two following conditions. (1) The global assembly sequences conform to the assembly constraints considered in the assembly unit partitioning process. (2) The assembly precedence of parts in subsequences is maintained in the global assembly sequences.

The assembly constraints considered in the assembly unit partitioning process is represented by the decision values of assembly relations. The decision values of assembly relations are deemed as the heuristic information to seek the global optimal assembly sequences and the first condition is satisfied. The second condition confirms that the process constraints and geometrical constraints considered in the assembly subsequence planning process are merged into the global assembly sequences.

2.2 The decision graph of assembly unit

The decision graph of assembly unit is illustrated in Fig. 2. The parts or components are represented as the nodes and the assembly relations between them are represented as the edges. The weights d_{ij} on the edges are the decision values of assembly relations between two parts P_i and P_j. The decision values are concluded by the functional constraints, structural constraints and process constraints presented in Fig. 1 and calculated through the FAHP method. Because some type of assembly constraints has directions, such as support between two parts, the decision values of assembly

Fig. 2 Decision graph of assembly unit



relations are not unique. In general, the bigger decision value of assembly relations is conserved and the smaller is neglected. The larger is the decision value between two parts, the more feasible is the probability of the two parts merged into the same assembly unit. For example, in Fig. 2, if $d_{12}>d_{1n}$, then, part P_1 and P_2 will be merged into one same assembly unit and part P_1 and P_n will belong to different assembly units. The functional constraints and structural constraints are put on directly on the edges represented by assembly relations and the process constraints are only put on edges linking two immediate parts or components to be assembled in the assembly processes.

3 The flowchart of assembly sequences merging based on assembly unit partitioning

The flowchart of assembly sequences merging based on assembly unit partitioning is illustrated in Fig. 3. For assembly unit partitioning process, there are four steps.

Step 1. The assembly constraints indices and assembly constraints weights are derived from CAD model or defined by designers.

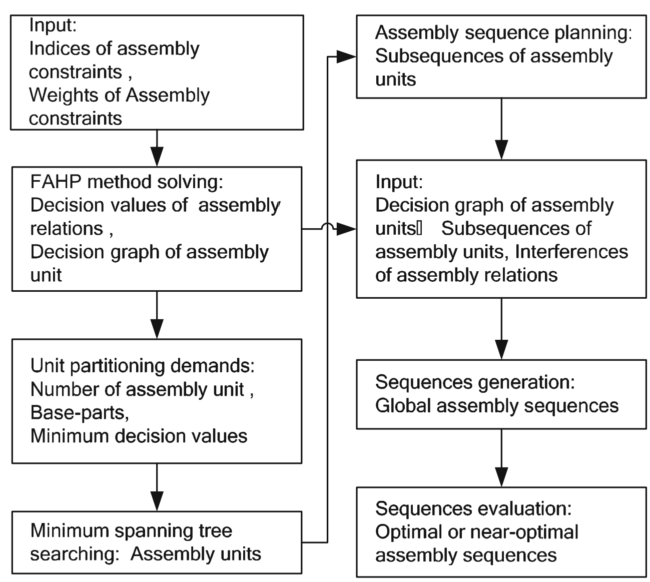


Fig. 3 Flowchart of assembly sequences merging based on assembly unit partitioning

Table 1 Indices of connection strength

Type	Attachment	Fit	Sticking	Tight fit
Strength	0.1	0.2	0.3	0.4
Type	Push	Screw	Rivet	Welding
Strength	0.5	0.6	0.7	1.0

Step 2. The decision values of assembly relations are computed by the FAHP method and the decision graph of assembly unit is generated.

Step 3. The base-parts, the number of assembly units and the minimum decision value indicating parts to be merged are specified by designers.

Step 4. In the last step of assembly unit partitioning, the minimum spanning tree algorithm [11] is implemented to seek the parts meeting the given demands and the proper part is merged to the base-parts one by one until all of the parts are searched over.

Because the part number of assembly unit is relatively less than that of the whole products, the optimal or near-optimal assembly sequences of assembly units can be generated using efficient methods of assembly sequence planning. After the subsequences of assembly units are obtained, the assembly sequences merging is implemented through the three steps.

Step 1. The decision graph of assembly unit, assembly directions of each part and assembly interference between assembly relations (global geometrical constraint) but not parts [12] are prepared.

Step 2. With respect to the interference constraints of assembly relations and assembly precedence constraints of subsequences, assembly sequences merging is actuated to generate the global assembly sequences of the whole products.

Step 3. The assembly sequences are evaluated as well as the assembly constraints used in the previous process is considered and the optimal or near-optimal assembly sequences are obtained.

4 Assembly unit partitioning method

The indices of assembly constraints are provided at the first step. The assembly constraints in Fig. 1 are quantified to obtain the indices of assembly constraints.

4.1 Definition of assembly constraints

4.1.1 Indices of functional constraints

(1) Index of stable supports

Given two parts, P_i and P_j, and assume that P_j is on P_i. When the gravitation direction of part P_j passes through the contact plane of the two parts, it can be concluded that part P_i is a stable support of part P_j. The index of stable supports can be defined as $SS_{ij}=1$. In contrast, the index of unstable supports is defined as zero.

(2) Index of assembly tolerances

If part P_i and P_j are assembled with tolerance, the index of mechanical function realized by assembly tolerance can be defined as $AT_{ij}=1$. Otherwise, $AT_{ij}=0$.

The two indices above denote the product function dependence and modularity design, and they can be combined as the Eq. 1.

$$\mathrm{FI}_{ij} = \alpha \mathrm{SS}_{ij} + (1-\alpha)\mathrm{AT}_{ij} \quad (1)$$

FI_{ij} implies the index of functional constraints between part P_i and P_j after they are assembled, α is the weight of the stable support, $0\leq\alpha\leq1$, $0\leq i, j\leq n$, and n is the parts number.

Table 2 The restricted *df* of assembly types

Direction	Type						
	Force assembly					Weak assembly	
	Screw	Rivet	Tight fit	Welding	Sticking	Surface mating	Clearance fit
$+X$	1	1	1	1	1	1	0
$-X$	1	1	1	1	1	0	0
$+Y$	1	1	1	1	1	0	1
$-Y$	1	1	1	1	1	0	1
$+Z$	1	1	1	1	1	0	1
$-Z$	1	1	1	1	1	0	1

4.1.2 Indices of structural constraints

(1) Index of connection strength

The index of connection strength proposed by Lee [7] is referred to and shown in Table 1. The connection strength of part P_i and P_j is denoted as CS_{ij}.

(2) Index of restricted degree of freedom

Each part has six degrees of freedom before being assembled, three translation degrees and three rotation degrees. The translation degrees of parts are considered here because most assembly directions are linear. A part's restricted *df*s are the reduced *df*s after it is assembled with the relative base-part whose *df* is zero in the assembly process. The index of the restricted *df*s of part P_i and P_j can be defined as:

$$LD_{ij} = DFC\left(P_i/P_j\right)/6 \tag{2}$$

The restricted *df*s are subject to the assembly types. The assembly types can be classified into forceful assembly (the weight is excluded) and weak assembly. The corresponding restricted *df*s are shown in Table 2. In the table, "1"denotes the assembled part is constrained in that direction and "0"is not.

The two indices imply the assembly stability and modularity design. They are united as Eq. 3.

$$SI_{ij} = \beta CS_{ij} + (1-\beta)LD_{ij} \tag{3}$$

SI_{ij} implies the structural constraints of part P_i and P_j after they are assembled. β is the weight of connection strength, $0 \le \beta \le 1$, $0 \le i, j \le n$.

4.1.3 Indices of assembly process constraints

If the parts with uniform assembly directions and tools can be gathered into the same assembly unit, assembly process planning will become easier. The assembly process constraints should be considered by assembly unit partitioning for the convenience of assembly process planning. Assembly direction change and tool change are two main factors to evaluate the assembly difficulty [13]. It is assumed that one part will be assembled for each assembly operation, and one specified tool is used along one direction.

When the assembly directions of two immediate assembly operations are unchangeable, the two assembled parts have the same assembly direction. In this case, the index of direction-related constraint of parts P_i and P_j can be defined as $DR_{ij}=1$. Otherwise, $DR_{ij}=0$.

Similarly, when the assembly tools used in two immediate operations are the same, the index of tool-related constraint of part P_i and P_j can be defined as $TR_{ij}=1$. Otherwise, $TR_{ij}=0$.

The two indices represent the assembly difficulty and they are united as the Eq. 4.

$$TI_{ij} = \gamma TR_{ij} + (1-\gamma)DR_{ij} \tag{4}$$

TI_{ij} implies the assembly process constraints of part P_i and P_j in assembly process, γ is the weight of tool-related constraint, $0 \le \gamma \le 1$, $0 \le i, j \le n$.

It is noted that each index of the assembly constraints will be normalized to a value within the interval [0,1] for the convenience of FAHP computation.

4.2 Calculation of decision values of assembly relations based on FAHP

The FAHP method stems from the AHP method proposed by Saaty [14], which is a systematic method for solving complicated and subjective decision-making problems. To tackle effectively the fuzziness during decision-making, Laarhoven and Pedrycz [15] evolve the original AHP method into the fuzzy AHP method by introducing the triangular fuzzy number of the fuzzy set theory into the comparison matrix of the AHP. From then on, many multiple criteria decision making (MCDM) problems are solved by the FAHP method. Assembly unit partitioning is also considered as a MCDM problem and the FAHP method is applied to deal with it in the paper. There are four steps to determine the decision values of assembly relations based on the FAHP method.

Step 1. Construction of the decision-making model for the assembly unit partitioning

The decision-making model for the assembly unit partitioning is illustrated in Fig. 4. The decision-making objective is put on the top level of the hierarchical model. On the second level are the main factors (the functional, structural and process constraints), and the sub-factors are put on the lower levels. The bottom level is the assembly relations to be evaluated.

Step 2. Construction of the fuzzy judgment matrix of assembly constraints

The indices of assembly constraints are given by designers and the judgment matrix is computed by Eqs. 1–4. The fuzzy judgment matrix is generated referring to Table 3. If there are M assembly relations in an assembly and there are N main indices (constraints) to be considered. The triangular fuzzy judgment matrix is $\widetilde{A} = \left(\widetilde{a}_{ij}\right)_{M\times N}$, $(1 \le i \le M, 1 \le j \le N,)$. $\widetilde{a}_{ij}$ is the triangular fuzzy number, $\widetilde{a}_{ij} = (l, m, u)$.

Step 3. Generation of the fuzzy weight vectors

After fuzzy judgment matrix of indices is generated, the weightiness of different assembly constraints should be

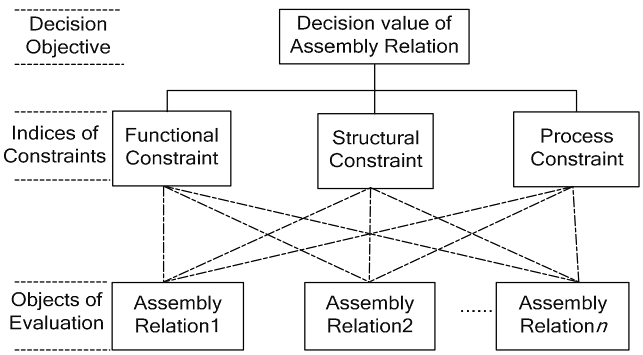


Fig. 4 Decision-making model for assembly unit partitioning

given. The weights can be appointed by experts and the fuzzy weight vector is generated referring to Table 3. If there are N assembly constraints, the fuzzy weight vector is $\widetilde{W} = (\widetilde{w}_1, \widetilde{w}_2, \ldots \widetilde{w}_N)$ and $\widetilde{w}_i (1 \leq i \leq N)$ is a triangular fuzzy number.

Step 4. Calculation of decision values of assembly relations

The fuzzy decision values of assembly relations can be computed by the following formula.

$$S_{M\times 1} = \widetilde{A}_{M\times N} \otimes \left(\widetilde{W}^T\right)_{N\times 1} \quad (5)$$

"T" denotes transposing of matrix and "$\otimes$" is the multiplicative operator of fuzzy mathematics.

The final decision values of assembly relations are the average of the triangular fuzzy numbers obtained above. If $\widetilde{S}_i = (L_i, M_i U_i)$, then the formula is as follows.

$$S_i = (L_i + M_i + U_i)/3 \quad (6)$$

The functional, structural, and process relationships between parts are represented by the decision values of assembly relations. The decision values are viewed as the heuristic information to generate the assembly units which can meet multiple demands of assembly process.

5 Assembly sequences merging

After the optimal or near-optimal subsequences of assembly units are generated, assembly sequences merging is implemented to generate the global assembly sequences of

Table 3 The mapping rules from the general numbers to the triangular fuzzy numbers($a_{ij} \in [c, d]$)

a_{ij}	0	[0,0,2)	[0,2,0,4)	[0,4,0,6)	[0,6,0,8)	[0,8,1,0)
(l,m,u)	(0,0,0)	(1,1,3)	(1,3,5)	(3,5,7)	(5,7,9)	(7,9,9)

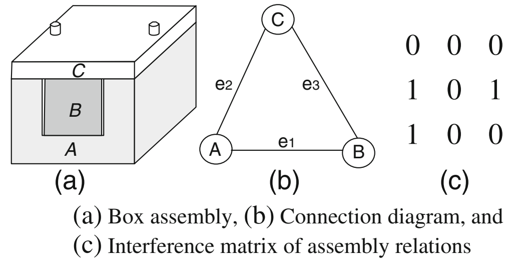


Fig. 5 **a** Box assembly, **b** connection diagram, and **c** interference matrix of assembly relations

the product. The acquisition of assembly interference of assembly relations, generation of assembly sequences and evaluation of assembly sequences are the three steps to achieve the global optimal assembly sequence.

5.1 Definition and representation of assembly interference of assembly relations

Assembly interference of assembly relations If assembly relation e_j cannot be established after e_i for assembly interference, it is called e_j is blocked by e_i. It is noted as $\overline{R} = [e_i e_j]$ or $e_i \overline{R} e_j$, commonly, $[e_i, e_j] \neq [e_j, e_i]$.

A simple box assembly illustrated in Fig. 5 (a) is used to clarify the assembly interference of assembly relations. The connection diagram of the assembly is displayed in the Fig. 5 (b). The nodes are parts and the lines between them are assembly relations. Part A and C are screwed together and the connectors are discarded here. It is obvious that assembly relation e_1 will be blocked by e_2 for geometrical constraint if e_2 is established prior to e_1. In practical assembly process, assembly process precedence constraints are also considered much. Though the geometrical constraint does not happen, some assembly operations cannot be fulfilled for the limited techniques or they are uneconomical. For example (Fig. 5 (a)), to make the assembly processes economical, one process precedence constraint is needed that assembly relation e_1 is blocked by e_3.

The Boolean variables can be used to specify the interference between assembly relations. All assembly interference can be represented by the interference matrix which is noted as $I(\overline{R})$. Suppose that the assembly relations sets is E=(e_1,e_2,…,e_n), there are n assembly relations and $\overline{R} \subseteq E \times E$, then $I(\overline{R}) = (r_{ij})_{n\times n}$, r_{ij} is the interference between assembly relations e_i and e_j. "$\subseteq$"means "belong to" of set theory.

The value of r_{ij} can be defined as:

If $e_i \overline{R} e_j (e_i \neq e_j)$, then r_{ij}=1, e_j is blocked by e_i if e_i is established prior to e_j; else r_{ij}=0.(i,j=0,1,2…n).

The assembly interference of assembly relations of an assembly can be represented by the corresponding interference matrix, e.g., the interference matrix of box assembly is shown in Fig. 5 (c).

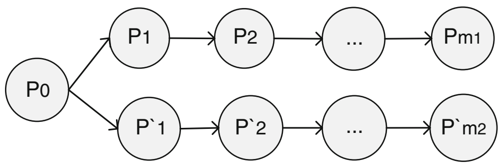


Fig. 6 Assembly precedence graph of assembly sequences merging

5.2 Generation of assembly sequences

Referring to the definition of interference of assembly relations and $I(\overline{R})$, if r_{ij}=0^r_{ji}=1, then assembly relation e_i should be established prior to e_j. Moreover, if all the elements of the *i*th row of $I(\overline{R})$ are zero, assembly relation e_i can be established prior to the other assembly relations. With assembly interference of assembly relations, the ordering of assembly relations can be concluded. To acquire assembly sequence, the proposal related to assembly structure is necessary: each assembly relation is associated with two parts and each part is at least associated with one assembly relation.

After one assembly relation is established, one or two parts will be assembled. Therefore, the assembly sequences of product have the following cases corresponding to the ordering of assembly relations.

(1) If the established relation concerns only one part, then the part is assembled, and the sequence number is equal to the order number of assembly relations.
(2) If the established relation concerns two parts, it has the following two cases.
(3) If the assembly relations produced by each part with the assembled parts are not blocked with each other, then the ordering of the two parts can be exchanged and the sequences number becomes double to the order number of assembly relations.
(4) If one assembly relation established by one part with the assembled parts is blocked by some relation(s) produced by the other part with the assembled parts, then the ordering of the two parts is constrained and the sequences number is equal to the order number of assembly relations.

5.3 Number of merged sequences

The number of assembly sequences of the assembly precedence graph is given by Lambert [6]. For an assembly precedence graph of the divergent type, containing *N* parts, all the disassembly sequences can be computed by Eq. 7.

$$S_{dis} = \frac{N!}{\prod_{i=1}^{R}(r_i + 1)} \tag{7}$$

Where, *R* is the number of nodes in the graph except for the roof node, and r_i is the number of arcs representing the precedence relationships that are leaving the *i*th node after the graph is made transient.

Suppose that there are two subsequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ to be merged at each step. The part number of subsequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ is m_1 and m_2, respectively. Obviously, r =1 for each part except the leaf node of subsequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$. A root node is added to aid to generate the global sequences of subsequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$. The merged precedence graph of subsequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ is illustrated in Fig. 6.

Only the assembly precedence constraints are considered and Eq. 7 is referred to and transformed, the number of global assembly sequences of subsequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ can be calculated by Eq. 8.

$$S_{asm} = \frac{(m_1 + m_2)!}{2^{m_1+m_2-2}} \tag{8}$$

In the assembly sequences merging process, some assembly process constraints will be taken into account except for the geometrical constraints. The actual number of global assembly sequences is less than that of computed by Eq. 8.

5.4 Evaluation of global assembly sequences

Assembly sequences merging is based upon the assembly constraints considered in the assembly unit partitioning process and previous assembly sequence planning process. The assembly constraints taken into account in assembly unit partitioning process are represented by the decision values in the decision graph of assembly unit, and the maximum continuous fragments of subsequences maintained in the global assembly sequences indicates the function of the assembly constraints considered in the assembly sequence planning process to the assembly sequences merging. Moreover, the minimum direction change [13] of assembly sequence is always viewed as one of the fundamental factors to evaluate the optimal or near-optimal assembly sequences. Therefore, the evaluation function of the global assembly sequences can be defined as follows:

$$f(s) = \frac{n_1 + n_2}{n} \frac{1}{l+1} \sum_{i=1}^{n} \mathrm{MAX}\left(\sum_{\mathrm{j=i}}^{\mathrm{n}} \mathrm{d_{ij}}\right) \tag{9}$$

$$n_1 = \mathrm{MAX}(\mathrm{SIMILARITY}(\mathrm{s}, \mathrm{s}_1)).$$

$$n_2 = \mathrm{MAX}(\mathrm{SIMILARITY}(\mathrm{s}, \mathrm{s}_2)).$$

For an assembly containing *n* parts, the global assembly sequence is represented as $\boldsymbol{s}=(P_1, P_2, \ldots, P_n)$ and P_i $(1 \leq i \leq n)$ is the part. $\boldsymbol{s}_1, \boldsymbol{s}_2$ are the subsequences to be merged. n_1, n_2 is the maximum number of parts of the longest continuous

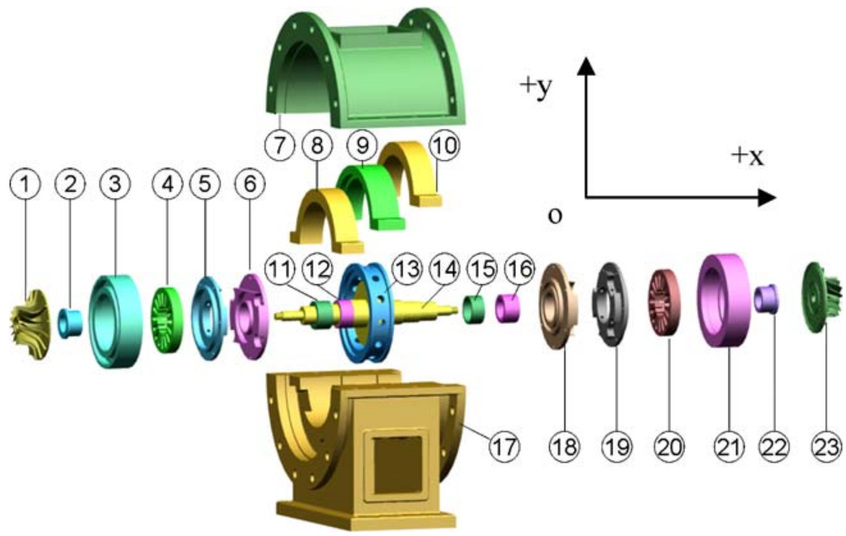


Fig. 7 The motor assembly

fragments which belong to the two subsequences s_1,s_2 and are maintained in the global assembly sequences, respectively. l is the direction change of the global assembly sequences. d_{ij} is the decision value of assembly relation between part P_i and P_j, which is the maximum decision value of assembly relations between the assembled part P_i and the next part P_j to be assembled. It is obvious that the bigger is the value of the evaluation function, the more optimal is the global assembly sequence.

6 An illustrative example

The exploded solid model of the motor shown in Fig. 7 is used as an illustrative example to verify the method. There are 23 parts in the assembly. Part 1, 23 and part 14 is assembled with screws. Part 7 and 17 are also screwed. The rest of the parts are supported by one abutting part and most of them are assembled with tight fit. The assembly directions of the parts are in the *x-o-y* plane displayed in Fig. 7. The assembly tools used to assemble the same type of connections are identical. The connection diagram of the motor assembly is given in Fig. 8. There exist 30 assembly relations in the assembly and the interference between assembly relations is presented in Table 4.

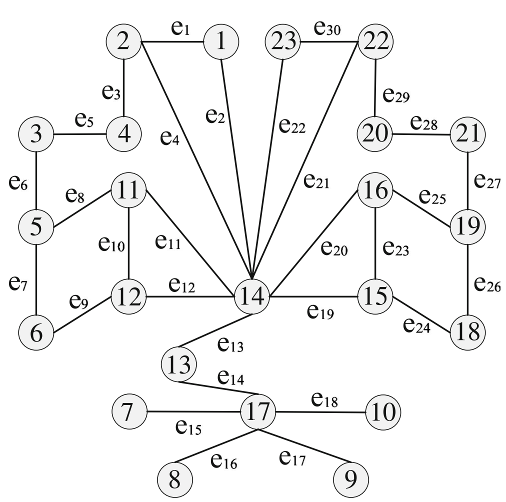


Fig. 8 The connection diagram of the motor assembly

In the first step, the assembly is partitioned into several assembly units for assembly sequence planning and assembly sequences merging, and each of which contains a reduced number of parts. In the second step, Assembly sequence planning can be applied for the assembly units and the optimal assembly subsequence of each assembly unit is acquired. Assembly sequences of assembly units can be produced with robust and efficient assembly planning approaches. In the last step, with subsequences and assembly interference of assembly relations, assembly sequence merging is implemented to generate the global assembly sequences of the product and these assembly sequences are evaluated to obtain the optimal assembly sequence of the product.

6.1 Assembly unit partitioning process

6.1.1 Computation of assembly constraints indices and weights

The indices of assembly constraints are computed by Eqs. 1–4, Table 2 and Table 3. Suppose that α=0.5, β= 0.5, γ=0.5 (the parameters can be adjusted with respect to practical applications). The judgment matrix of the functional constraints among the parts is presented in Table 5. The judgment matrices of structural and process constraints are omitted here. The weights of assembly constraints are determined by experienced designers, suppose that the weights of functional, structural and process constraints are 3, 5, 7, respectively.

Table 4 The interference between assembly relations

Assembly relations	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	e_{10}	e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}	e_{17}	e_{18}	e_{19}	e_{20}	e_{21}	e_{22}	e_{23}	e_{24}	e_{25}	e_{26}	e_{27}	e_{28}	e_{29}	e_{30}
e_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_2	1	0	0	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_4	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_5	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_6	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_7	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{10}	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{11}	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{12}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{13}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{14}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
e_{16}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{17}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{18}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{19}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{20}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
e_{21}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
e_{22}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1
e_{23}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
e_{24}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{25}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{26}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
e_{27}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
e_{28}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
e_{29}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e_{30}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5 Indices of functional constraints between parts

Part number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.5	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0
8	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0
11	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
12	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
14	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	0.5	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5
15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.5	0.0	1.0	0.0	0.0	0.0	0.0	0.0
16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
17	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
18	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
19	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0
20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.5	0.0
21	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0
22	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
23	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

6.1.2 Construction of fuzzy judgment matrix and weight vectors of assembly constraints

The fuzzy judgment matrices and the fuzzy weights can be computed based on the judgment matrix of indices and weights of assembly constraints referring to Table 3. Each general number is transformed to the triangular fuzzy number and the fuzziness of designers' judgment is integrated into the algorithm. For example, the element $FI_{1\ 14}$=0.5 shown in Table 5 is changed to the triangular fuzzy number $\widetilde{F}I_{114} = (3,5,7)$ referring to Table 3. If the weights of the functional, structural and process constraints are given 3, 5, 7 respectively, the triangular fuzzy weight vector is presented as $\widetilde{W} = ((1,3,5),(3,5,7),(5,7,9))$.

6.1.3 Generation of assembly units

The decision graph for assembly unit partitioning is generated and illustrated in Fig. 9. The decision values of assembly relations are listed beside the edge (only the integer of the decision value is shown). Suppose that there are three assembly units will be generated, the base-parts are part 14, 15, and 17, respectively. It is suggested that each assembly unit comprises no more than ten parts. With graph-search algorithm, the decision graphs of the assembly unit_1, unit_2 and unit_3 are illustrated in Fig. 10 and the parts number in different type of circles belong to different assembly units. Assembly unit_1 contains ten parts, unit_2 contains eight parts and unit_3 contains five parts.

6.2 The results of assembly sequences merging

After the assembly unit partitioning process, the assembly units are assigned to the collaborative participators for assembly sequence planning. Because the parts number of each assembly unit is small, the optimal or near-optimal subsequences of each assembly unit can be obtained quickly. Suppose that the optimal subsequences $\boldsymbol{s}_1$, $\boldsymbol{s}_2$, $\boldsymbol{s}_3$, of the three assembly units are provided as follows.

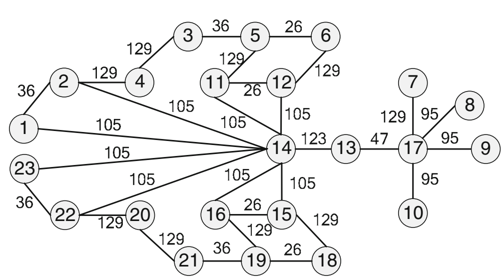


Fig. 9 The decision graph of assembly unit

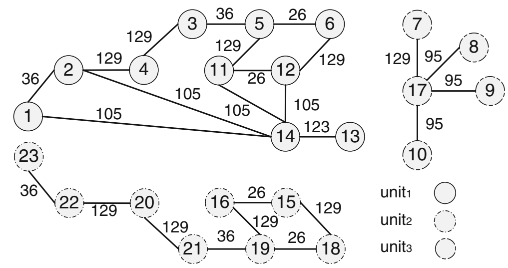


Fig. 10 The result of assembly unit partitioning

$$s_1: 14(+x)->13(+x)->12(+x)->11(+x)->6(+x)->5(+x)->2(+x)->4(+x)->3(+x)->1(+x).$$

$$s_2: 15(-x)->16(-x)->18(-x)->19(-x)->22(-x)->20(-x)->21(-x)->23(-x).$$

$$s_3: 17(+y)->8(+y)->9(+y)->10(+y)->7(+y).$$

Where, "->" points to the next part to be assembled. The symbols in brackets are the assembly directions of the corresponding parts, respectively. The global assembly sequences will be derived from the three subsequences and the assembly sequences merging program is coded with C++ language. The program has been implemented on a PC with 2.0 GHz processor and 1.0 GB memory.

The subsequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ are merged at first to obtain the global assembly sequences of unit_1 and unit_2. With the assembly interference of assembly relations and precedence constraints concluded by subsequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, there are a total of 12,871 assembly sequences generated and the computation time is 3,718 ms. After evaluation of assembly sequences, the optimal assembly subsequence $\boldsymbol{s}_4$ of unit_1 and unit_2 is given as follows.

$$s_4: 14(+x)->13(+x)->12(+x)->11(+x)->6(+x)->5(+x)->2(+x)->4(+x)->3(+x)->1(+x)->15(-x)->16(-x)->18(-x)->19(-x)->22(-x)->20(-x)->21(-x)->23(-x).$$

The maximum value of evaluation function of subsequence s_4 is 948.833374 and the direction change of subsequence s_4 is 1. Furthermore, the subsequences s_1 and s_2 are totally embedded in the subsequence s_4.

To generate the global optimal assembly sequence of the products, subsequences s_3 and s_4 are merged in the next step. There are a total of 20,350 assembly sequences generated and the computation time is 8,109 ms. The optimal assembly sequence s_5 is also obtained through assembly sequences evaluation and illustrated in the following.

$$\begin{aligned} s_5: & 14(+x)->13(+x)->12(+x)->11(+x)- \\ & >6(+x)->5(+x)->2(+x)->4(+x)->3(+x)- \\ & >1(+x)->15(-x)->16(-x)->18(-x)- \\ & >19(-x)->22(-x)->20(-x)->21(-x)- \\ & >23(-x)->17(+y)->8(+y)->9(+y)- \\ & >10(+y)->7(+y). \end{aligned}$$

The maximum value of evaluation function of s_5 is 786.333357 and the direction change is 2. Furthermore, subsequences s_1, s_2 and s_3 are totally embedded in the assembly sequence s_5. That is to say, the assembly constraints considered in assembly unit partitioning process and in previous assembly sequence planning and merging process are comprehensively taken into account in the next assembly sequences merging process.

To verify the efficiency of assembly sequence merging, the assembly sequence planning program is also implemented using the motor product under the same conditions. In fact, we have to terminate the program because the memory (3.0 GB, including physical memory and virtual memory) of the computer is crammed and the computation time is unbearable.

We have done another several experiments with different products. The results show that not all the subsequences are kept intact in the global optimal assembly sequences while the longest continuous fragments of subsequences are always maintained in the global optimal assembly sequence and the direction change of them is always the minimum. From the experiments, we also conclude that the searching space of assembly sequences merging is reduced remarkably and the efficiency of assembly sequence merging is also improved than that of assembly sequence planning of the same product under identical conditions.

7 Conclusion and future work

The two key technologies of collaborative assembly planning are assembly unit partitioning and assembly sequences merging. The assembly unit partitioning is different from subassembly identification and more assembly constraints are comprehensively considered, and the assembly units can meet multiple demands of assembly process. On the other hand, the global geometrical constraints, the precedence constraints of assembly subsequences and the assembly constraints considered in assembly unit partitioning process are taken into account in assembly sequences merging process. The consistency of assembly units, assembly subsequences and the global assembly sequences are preserved. The searching space of assembly sequences merging is compressed remarkably and the global optimal assembly sequence is obtained through assembly sequences evaluation. Future research includes: (1) the assembly sequence planning and optimization referring to decision graph of assembly unit. (2) The automatic extraction of assembly interference of assembly relations. (3) Assembly sequences merging considering more assembly processes constraints.

Acknowledgments The authors gratefully acknowledge the fund support from the National High-Tech Research and Development Program of China (863 program), Grant No.2006AA04Z138.

References

1. Wang JF, Liu JH, Zhong YF (2004) Integrated approach to assembly sequence planning of complex products. Chin J Mech Eng 17(2):181–184
2. Dong TY, Tong RF, Zhang L, Dong JX (2005) A collaborative approach to assembly sequence planning. Adv Eng Inform 19 (2):155–168. doi:10.1016/j.aei.2005.05.008
3. Baldwin CY, Clark KB (2006) Modularity in the Design of Complex Engineering Systems. Springer, Berlin
4. Chakrabarty S, Wolter J (1997) A structure-oriented approach to assembly sequence planning. IEEE Trans Robot Autom 13(1):14–29. doi:10.1109/70.554344
5. Ong NS, Wong YC (1999) Automatic subassembly detection from a product model for disassembly sequence generation. Int J Adv Manuf Technol 15(6):425–431. doi:10.1007/s001700050086
6. Lambert AJD, Gupta SM (2005) Disassembly modeling for assembly, maintenance, reuse, and recycling. CRC, Florida
7. Lee S (1994) Subassembly identification and evaluation for assembly planning. IEEE Trans Syst Man Cybern 24(3):493–503. doi:10.1109/21.278997
8. Mejbri H, Anselmetti B, Mawussi B (2003) A recursive tolerancing method with subassembly generation. Proceedings of the 5th IEEE International Symposium on Assembly and Task Planning, 10–11 July. Besançon, France, pp 235–240
9. Swaminathan A, Barber KS (1996) An experience-based assembly sequence planner for mechanical assemblies. IEEE Trans Robot Autom 12(2):252–267. doi:10.1109/70.488945

10. Halperin D, Latombe JC, Wilson RH (2000) A general framework for assembly planning: the motion space approach. Int J Adv Manuf Technol 16(3–4):577–601
11. Weiss MA (2006) Data structures and algorithm analysis in C++. Posts & Telecom, Beijing
12. Zhou XM, Du PG (2008) A model-based approach to assembly sequence planning. Int J Adv Manuf Technol 39(9–10):983–994. doi:10.1007/s00170-007-1272-9
13. Wang JF, Liu JH, Zhong YF (2005) A novel ant colony algorithm for assembly sequence planning. Int J Adv Manuf Technol 26(11–12):1137–1143. doi:10.1007/s00170-003-1952-z
14. Saaty TL (1980) The analytic hierarchy process. McGraw-Hill, New York
15. Laarhoven P, Pedrycz W (1983) A fuzzy extension of Saaty's priority theory. Fuzzy Sets Syst 11(3):229–241. doi:10.1016/S0165-0114(83)80083-9