Intelligent Analysis Method of Assembly Process Based on Semantic Elements

Zenghui Du¹, Jianhua Liu¹* and Yongxi He¹
¹School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China
*Corresponding author’s e-mail: jeffliu@bit.edu.cn

Abstract. The assembly process file is a direct expression of the product assembly design intent. In order to improve the quality and efficiency of the assembly, the 3D model of the product and resources need to be introduced into the virtual assembly environment. However, the assembly process is verified and optimized by assembly simulation. In order to realize the intelligent analysis of assembly process files and the automation of assembly simulation, an intelligent analysis method of assembly process based on semantic elements is proposed. This article takes the assembly process documentation of complex products as the research object. An improved TF-IDF algorithm is proposed to automatically extract key semantic elements in the assembly process file. We quickly resolve previous assembly process data by building assembly key semantic corpora. An assembly semantic dependency tree algorithm based on maximum entropy is proposed to automatically reason the logical relationship of each assembly phrase in the assembly process statement. According to the above algorithm, the assembly semantic intelligent analysis system is developed. The assembly process of a certain type of ground weapon system is taken as an example to verify the system. And the intelligent analysis of the product process file is realized which lays a foundation for realizing the semantic driving of product assembly simulation.

1. Introduction

Assembly is one of the key factors affecting the quality and cost of the final product. In order to improve the quality and efficiency of the assembly, 3D models such as products and resources need to be introduced into the virtual assembly environment. The assembly process is verified and optimized through interactive simulation. However, the traditional assembly process design and simulation are two parallel lines. The latter is always used as a verification tool for key assembly processes. The verification results can only be optimized, modified and reconstructed by the indirect feedback of the process personnel.

There are many problems in the above process. First, the assembly process design and simulation must be carried out in series. There is a large amount of repeated information that needs to be input twice during the assembly and simulation process. The number of iterations accompanying the process design and simulation is increasing in simulation workload. The assembly efficiency is generally low. Secondly, the assembly process designer and the assembly simulator are often not the same person. The assembly process simulator often uses his personal experience to interpret the assembly process and simulate the assembly process. This process greatly affects the reliability of assembly simulation results. Because the simulation process relies on people's understanding of the process text. It is necessary to re-assemble the assembly simulation for similar assembly processes, and it is impossible
to reuse similar assembly simulation tasks. In order to solve the above problems, it is necessary to analysis and process the assembly process design information, and build an assembly semantic information model which is shared by humans and computer. More and more scholars have launched research on the above issues.

He et al. proposed the "trinity" concept of assembly semantics, and assembled the semantic association diagram to implement assembly-to-constrained mapping driver [1]. Lv et al. used the assembly semantic graph ASG to describe the assembly information model, and realized the collaborative assembly design through the reasoning of the semantic model [2]. Yang et al. introduced assembly semantics into the field of process planning and proposed a virtual assembly process planning solution based on semantic association model [3]. The researchers above mainly use association diagrams, structural trees, etc. to describe the relationship between products, process, and assembly constraints to build an assembly semantic model. Although this method can quickly carry out product assembly simulation, it cannot understand and excavate the existing assembly files, and cannot output the corresponding process documents for the assembler to understand. The methods are different, because each scholar describes the product assembly diagram in a different way and the defined assembly constraints have no certain limitations [4]. Due to the natural language processing technology, it can process a large amount of text information automatically and in batches without prior definition of key information, which can realize automatic analysis and mining of assembly process history data and solve the problem of real-time assembly process simulation. It realizes the intelligent construction of the assembly process semantic model [5]. However, since the assembly process knowledge belongs to the professional domain knowledge, the meaning of some vocabulary and text descriptions are different from the traditional dialogue or the customary way of text documents [6]. Therefore, it is necessary to improve on the basis of natural language processing algorithms to realize the assembly process text automatically [7].

In order to solve the above problems, this paper proposes an intelligent analysis method of assembly process based on semantic elements. It aims to realize the systematic expression of assembly design intent by extracting key assembly semantic elements. Firstly, the key elements in the assembly process statement are sorted out, and the assembly semantic element model is established. Then, the automatic extraction of assembly semantic elements is realized based on Improved TF-IDF algorithm [8]. The assembly semantics corpus is optimized by analysing the historical assembly process data. Finally, the assembly semantic dependency tree algorithm based on maximum entropy realizes the automatic reasoning of the logical relationship of semantic elements in the sentence.

2. Assembly process semantic element model definition

The assembly process semantics are mainly contained in the assembly step description, which is the essential abstraction of the assembly design intent. Each process statement describes which assembly operation is used to install the assembly object according to which assembly path is installed at that location. That is five assembly semantic elements including assembly objects, assembly resources, assembly operations, assembly paths, and assembly positions. The assembly process and assembly process information can be abstracted into a combination of the five elements, a default or derived assembly semantic model

\[ A_{as} = \{ A_o, A_r, A_m, A_p, A_l \}. \]

Definition: \( A_o \) indicates the assembly object, including the assembly part name and ID. \( A_r \) indicates the assembly resource, including the assembly tool name and ID. \( A_m \) indicates the assembly operation. \( A_p \) indicates the assembly path, including the direction and order of the assembly. \( A_l \) indicates the assembly position, describing the final assembly position of the assembly object. The basic properties of each semantic element are shown in the table1.

| Assembly semantic element | Basic keyword | Part of speech | Element of sentence |
|---------------------------|---------------|---------------|-------------------|
| Assembly object           | Name          | Noun          | Subject           |
| Hydraulic pumps, cylinders, manifolds, flanges, locking cylinders, landing gear, etc. |
### 3. Key assembly semantic element extraction based on improved TF-IDF algorithm

TF-IDF, a word frequency-inverse document frequency algorithm, is often used to evaluate how important a word is to a document in a document set. TF counts the frequency at which a word appears in a document. The basic idea is that the more times a word appears in a document, the stronger its ability to express it. IDF counts how many words appear in a document set [8]. The basic idea is that if a word appears in fewer documents, its ability to distinguish the document is stronger.

Through the application of natural language processing technology, the automatic participle of the assembly process description language in the assembly process file is used to determine the part of speech and sentence components of each semantic element in the process statement, because there are a large number of virtual words and interruption words in the process statement, in order to automatically extract the key assembly semantic elements such as assembly objects, assembly resources, assembly paths, assembly actions, and assembly positions. Firstly, according to the rules of the above attribute table, a certain number of assembly semantic element corpora are established, and the TF-IDF algorithm is improved for the corpus. The calculation of part-of-speech similarity and sentence component similarity.

Definition: $W_i$ is a word segmentation unit. $p$ and $n$ represent the part-of-speech and semantic dependencies. $W_{ao}$, $W_{ar}$, $W_{an}$, $W_{ap}$, $W_{al}$ and $W_{aw}$ are assembly object corpus, assembly resource corpus, assembly operation corpus, assembly path corpus, and assembly location corpus, respectively.

The steps for assembling the semantic feature extraction algorithm are as follows:

**Step 1**: Import the assembly step text content, and get the following results through word segmentation: $\{W_i(p,n), W_j(p,n), ..., W_k(p,n), ..., W_n(p,n)\}$.

**Step 2**: Extract assembly object semantics:

- The semantics of the assembly object is preferentially matched, mainly to calculate the semantic similarity between the word segmentation unit $W_i$ and the assembly object semantic library in the current text. First, the assembly object semantic library is traversed. If the same word segmentation unit exists, the $W_i$ is assigned the $ao$ tag as $W_{ao}$, and then refreshed the value of $W_i$.
- If there is no identical word segmentation unit in the semantic library, the following equation is taken to perform similarity matching. If yes, the $W_i$ is assigned the $ao$ tag as $W_{ao}$, and the value of $W_i$ is refreshed. Otherwise, the process proceeds to the next step.

$$
\begin{align*}
\left\{ p(W_i-p, W_{ao}-p) = P_{p-match} \\
n(W_i-n, W_{ao}-n) = P_{n-match}
\right.
\end{align*}
$$

$P_{p-match}$ and $P_{n-match}$ are the part-of-speech similarity matching value and the semantic-dependent similarity matching value of the matching model respectively, and the assembly semantic information needs to be adjusted for different products, thereby realizing the assembly semantics. Accurate classification.
Step 3: Extract assembly resource semantics:
- Traverse the assembly object semantic library, if the same word segmentation unit exists, assign the \( ar \) tag to \( W_i \) as \( W_{ar} \).
- If there is no identical word segmentation unit in the semantic library, the following equation is taken to perform similarity matching. If it is satisfied, the \( ar \) tag is assigned to the \( W_i \) as \( W_{ar} \). Otherwise the process proceeds to the next step.

\[
p(W_{i-p},W_{ar-p}) = P_{p-match}
\]
\[
n(W_{i-n},W_{ar-n}) = P_{n-match}
\]

Step 4: Extract assembly operation semantics:
- Traverse the assembly object semantic library, if the same word segmentation unit exists, assign the \( am \) tag to \( W_i \) as \( W_{am} \).
- If there is no identical word segmentation unit in the semantic library, the following equation is taken to perform similarity matching. If it is satisfied, the \( am \) tag is assigned to the \( W_i \) as \( W_{am} \). Otherwise the process proceeds to the next step.

\[
\begin{align*}
p(W_{i-p},W_{am-p}) &= P_{p-match} \\
n(W_{i-n},W_{am-n}) &= P_{n-match}
\end{align*}
\]

Step 5: Extract assembly path semantics
- Traverse the assembly object semantic library, if the same word segmentation unit exists, assign the \( ap \) tag to \( W_i \) as \( W_{ap} \).
- If there is no identical word segmentation unit in the semantic library, the following equation is taken to perform similarity matching. If it is satisfied, the \( ap \) tag is assigned to the \( W_i \) as \( W_{ap} \). Otherwise the process proceeds to the next step.

\[
\begin{align*}
p(W_{i-p},W_{ap-p}) &= P_{p-match} \\
n(W_{i-n},W_{ap-n}) &= P_{n-match}
\end{align*}
\]

Step 6: Extract assembly location semantics
- Traverse the assembly object semantic library, if the same word segmentation unit exists, assign the \( al \) tag to \( W_i \) as \( W_{al} \).
- If there is no identical word segmentation unit in the semantic library, the following equation is taken to perform similarity matching. If it is satisfied, the \( al \) tag is assigned to the \( W_i \) as \( W_{al} \). Otherwise the process proceeds to the next step.

\[
\begin{align*}
p(W_{i-p},W_{al-p}) &= P_{p-match} \\
n(W_{i-n},W_{al-n}) &= P_{n-match}
\end{align*}
\]

Step 7: Extract assembly location semantics
- Update the next \( W_i \) and bring the loop iteration into the above steps, until the assembly process text is sorted.

For example, the semantics of assembling a hydraulic pump station in an assembly tool file for a model of a weapon loading vehicle is described as "Install the hydraulic pump station longitudinally on the left side of the truck platform through M16 bolts." The process of extracting the above semantics to obtain the key semantic elements of the assembly process statement is shown in Figure 1.
4. Assembly Relational Reasoning Based on Maximum Assembly Sentence Tree Algorithm

By extracting the semantic elements in the assembly process text, the key assembly semantic information in the process text can be extracted and sorted, and the key assembly corpus contained in the whole process file is obtained, which realizes the quantitative analysis of the assembly process, but in the corpus Stored discrete assembly component phrases, the computer cannot directly understand the meaning of the assembly statement through discrete phrases. Because the logical relationship of these discrete phrases is lacking, only the assembly semantic elements can be correctly arranged to get the assembly intent of the assembly statement. The logical relationship between assembly semantic elements is actually the semantic dependence between phrases in a statement.

In order to clearly describe the semantic dependencies between assembly semantic elements, the assembly semantic dependencies commonly used in assembly statements are defined, and the assembly dependencies are defined by different labels. The specific assembly semantic dependencies are shown in Table 2.

| Assembly semantic relationship | Label | Instance                  |
|-------------------------------|-------|---------------------------|
| Centering relationship        | ATT   | Installed with M19 bolts   |
| Quantity relationship         | QUN   | Install four screws diagonally |
| Constellation                 | COO   | Install pipe 1 and bracket together at interface B |
| Pre-attachment                | LAD   | Longitudinal installation of hydraulic pump |

Figure 1. Assembly semantic feature extraction process.
Definition: The assembly semantic dependency tree is denoted as $Y = \{(W_i \rightarrow W_j)\}$ represents a set of directed edges, where the edge $(W_i \rightarrow W_j)$ indicates that the assembly phrase $W_i$ depends on the assembly phrase $W_j$ or virtual root ($\textit{ROOT}$), for convenience of description. Also known as $W_j$ is the parent node of $W_i$. Each edge has a certain weight, which can be expressed as: $M(W_i \rightarrow W_j) = M \cdot f(i,j)$. Where $f(i,j)$ is the feature vector extracted from the edge $(W_i \rightarrow W_j)$, and $M$ is the weight of the feature vector. This point multiplication formula gives the calculation of the probability (weight) of a directed edge. The weight of an assembly semantic dependency tree can be obtained by calculating the weight sum of all edges in this syntax tree. The specific formula is as follows:

$$P(Y) = \sum_{(W_i \rightarrow W_j) \in Y} M(W_i \rightarrow W_j)$$

If each assembly phrase is used as a vertex of the directed graph, there is a directed edge with weights between each assembly phrase, and the set of these directed edges can obtain the assembly semantic dependency directed graph $G$, which The specific expression is:

$$G = (K, D)$$
$$D = \{(i \rightarrow j): i \neq j, i \in [1 : n], j \in [0 : n]\}$$

Where $W_i$ corresponds to the $i$th assembly phrase and $D$ is the set of directed edges. Let $N$ be the set of assembly syntax trees. The maximum spanning tree algorithm is to find the largest assembly semantic dependency generation tree $Y^*$ in $N$:

$$Y^* = \text{fmax} P(Y), Y \in N$$
$$\forall W_i, \exists W_j, W_j \neq W_i, (W_i \rightarrow W_j) \in Y$$

**Step 1:** Data initialization
- Initialize assembly phrase descendant node set $\text{child} = \{W_1, W_2, ..., W_i, ..., W_n\}$.
- Initialize all assembly phrase node collections $\text{All} = \{W_1, W_2, ..., W_i, ..., W_n\}$.
- Initialize the spanning tree collection $Y = \{\text{NULL}\}$

**Step 2:** Loop until the assembly phrase descendant node set has only one node
- The largest edge should be found from the child collection to the all collection, and brought it into the loop formula.

$$\begin{cases}
(W'_i, W'_j) = \text{fmax} M(W_i \rightarrow W_j) \\
W'_i \in \text{child}, W' \in \text{All}
\end{cases}$$
- The new assembly semantic relationship edge $(W'_i \rightarrow W'_j)$ should be checked. If it produces a loop in the assembly syntax spanning tree set $Y$. Then this edge should be discarded in the tree.
- $W'_i$ should be removed from the assembly phrase descendant node collection $\text{child}$. The $\text{child}$ should be refreshed.

$$\text{child} = \text{child} - \{W'_i\}$$
- The newly generated assembly semantic relationship edge $(W'_i \rightarrow W'_j)$ should be added to the assembly syntax spanning tree.

$$Y = Y + \{W'_i \rightarrow W'_j\}$$
Step 3: The last node $W_n$ is used as the root node of the assembly semantic relationship spanning tree

$$ Y = Y + \{W_n \rightarrow \text{ROOT}\} $$

For example, the semantics of assembling a hydraulic pump station in an assembly tool file for a model of a weapon loading vehicle is described as "Install the hydraulic pump station longitudinally on the left side of the truck platform through M16 bolts." The dependency tree of the assembly process statement is shown in Figure 2.

![Figure 2. Assembly semantic dependency tree.](image)

5. Conclusion and outlook

In this work, an intelligent analysis method of assembly process based on semantic elements has been presented. Through the two algorithms proposed in this paper, the automatic reasoning of the extraction and assembly dependencies of assembly semantic elements in assembly process text can be realized. Batch analysis of the assembly process can be achieved by constructing a semantic model. Provides a way to achieve computer understanding of human assembly design intent. Future research should concentrate on the semantic drive of product assembly actions. Semantic drive for assembly simulation by building an assembly semantic action library. It simplifies the assembly process and enables real-time verification of the assembly process.

Acknowledgments

The research work of this thesis was funded by 206 of the Second Academy of Aerospace Science and Technology of China. The related algorithms in the paper solve the assembly simulation problem of the institute. Improve the preparation process of their product assembly process. The algorithm in the paper is verified in the process planning process of a certain type of missile car of the company. The algorithm processing result is consistent with the actual process.

References

[1] He H., Yixiong F., Jianrong T., & Yang, X. (2008). Self-organized semantic feature evolution for axiomatic design. Chinese Journal of Mechanical Engineering, 21(2), 108-113.

[2] Lv M., Hou W J., Li X J. The Research of Hierarchy Assembly Semantic Model Based on Intelligent Assembly Process Planning [J]. Key Engineering Materials, 2011, 467-469:1933-1939.

[3] Yang Y, Fang H, Luan H, et al. Research of the Assembly Model Based on Parts Attribute Semantic [J]. Applied Mechanics and Materials.

[4] Mei-Yu L, Wen-Jun H, Xiang-Ji, L.I. = Multilevel Semantic Model for Assembly in Intelligence Assembly Process Planning [J]. Journal of Donghua University, 2010.

[5] Sui A. Assembly Constraint Semantic Model in Distributed Virtual Environment [J]. Journal of Computer Research & Development, 2006, 43(3):542-550.

[6] Yang Y, Fang H, Luan H, et al. Research of the Assembly Model Based on Parts Attribute Semantic [J]. Applied Mechanics and Materials.

[7] Tanaka K, Abe N, Taki H. Understanding of mechanical assembly instruction manual by integrating vision and language processing and simulation[C]// IEEE International Conference on Robotics & Automation. IEEE Xplore, 1999.
[8] Tata S, Patel J M. Estimating the selectivity based cosine similarity predicates [J]. ACM SIGMOD Record, 2007, 36(4):75-80.