ISSN 2616-5775 Vol. 4, Issue 7: 39-45, DOI: 10.25236/AJCIS.2021.040706

Component Matching Based on Function Edit Distance in Conceptual Design

Qiantong Guo

Human Resource Department China Electronics Corporation, Beijing, China
guoqt2006@sina.com

Abstract: *This paper proposed formal descriptions of base vector, feature vector and matching function in the component semantic matching. The query input is discussed which is converted into the function descriptors of solving the query input. Components functions descripted by function edit distance model are illustrated in knowledge base. By providing semantic matching in this way, users' intention can be accurately grasped, their design ideas can be inspired, and innovative design can be supported.*

Keywords: *Conceptual design, knowledge engineering, edit distance*

1. Introduction

The requirement of component matching in conceptual design includes two parts, one is the requirement from the designer and the other is the requirement from the support system of innovate design[1].

Designers' requirements for component matching in the conceptual design stage are shown in two aspects. At the initial stage of conceptual design, designers want to have a preliminary understanding of the relevant component knowledge, such as the name, domain, category, description information and source of the component and other basic information[2]. After the preliminary judgment and overall analysis of the design, the designer needs to make a detailed inspection of the component functions listed in the design scheme, or make a comparison and screening of the component sets that realize key functions. These two different stages correspond to the two components of component knowledge[3]. One is matching query to the basic information of the component, matching to the ontology pattern information level of knowledge representation. Designers can judge whether it is necessary to obtain detailed knowledge of the component through the basic information obtained by querying the component name. Designers can also carry out matching queries on basic information such as categories, and have a global preliminary understanding of the corresponding knowledge. The second is the matching query of the functional semantics of the component corresponding to the semantic matching of the ontology pattern of knowledge representation. Designers can obtain the detailed functional semantics of matching components by querying the functional semantics of components, including functions, input and output streams and constraint conditions[4]. By providing semantic matching, users' intention can be accurately grasped, their design ideas can be inspired, and innovative design can be supported.

Component knowledge can be represented by ontology model with detailed functional semantics. Using this advantage, component matching can be extended from the basic information matching based on keywords to the functional semantics matching of components, which is more in line with the knowledge matching requirements of designers in the conceptual design stage. In this paper, the method of component functional semantic matching is studied in depth, including the model building of functional semantic matching and the algorithm building of functional semantic index.

2. The Multi-level Knowledge Representation Ontology

This paper presents a new component representation model in the conceptual design stage of mechanical products, which is shown in Figure 1. Six important elements have been proposed, i.e. Component, Function, Supplier, Feature, Constraint, Evaluation. There is a clear logical relationship between these elements. Associated with them, six key concepts have been specified.

ISSN 2616-5775 Vol. 4, Issue 7: 39-45, DOI: 10.25236/AJCIS.2021.040706

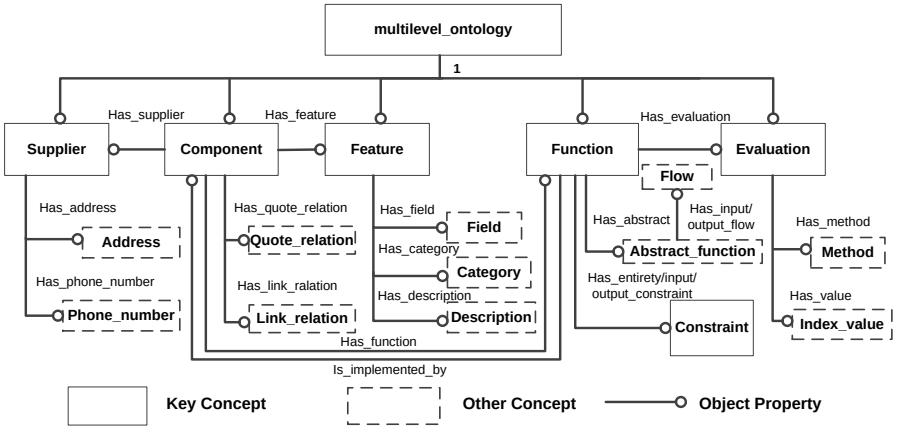

Figure 1: The schema of logical structure.

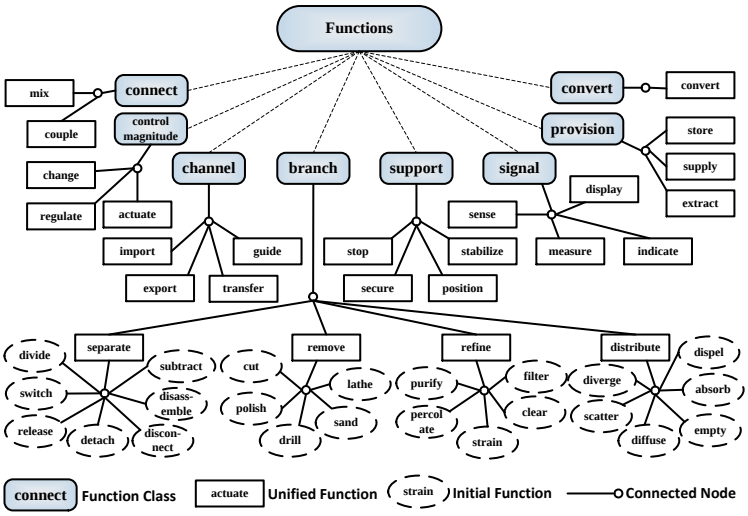

Figure 2: Structural tree of functions.

The concept Component is the subject, which specifies the name of component. The concept Function is the most important concept, As shown in Figure 2.

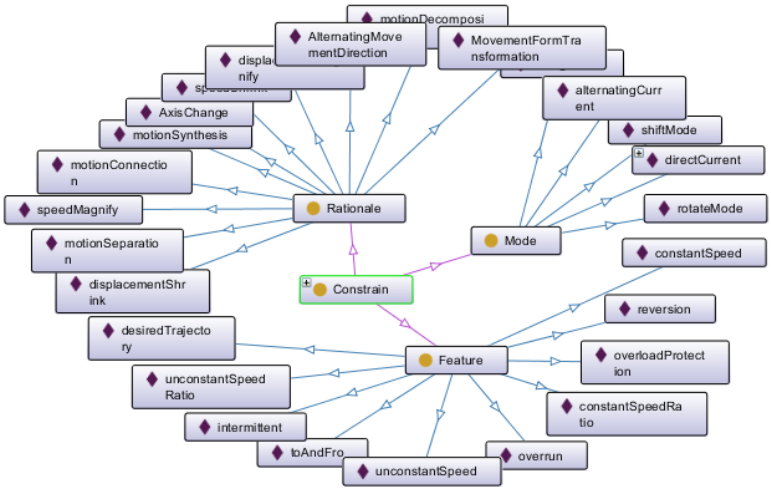

Figure 3: Individuals of constraint (partial).

The concept Constrain is the detail definition adjustment, as shown in Figure 3. By defining the constraint relationship in the process of component function realization, the index clarifies the necessary conditions and objective environment for function realization, and matches the actual constraint requirements of the required function accurately.

ISSN 2616-5775 Vol. 4, Issue 7: 39-45, DOI: 10.25236/AJCIS.2021.040706

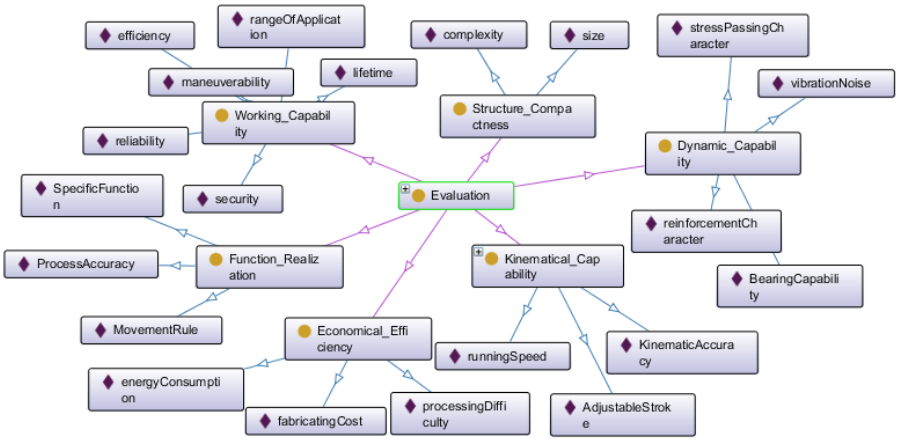

Figure 4: Individuals of evaluation (partial).

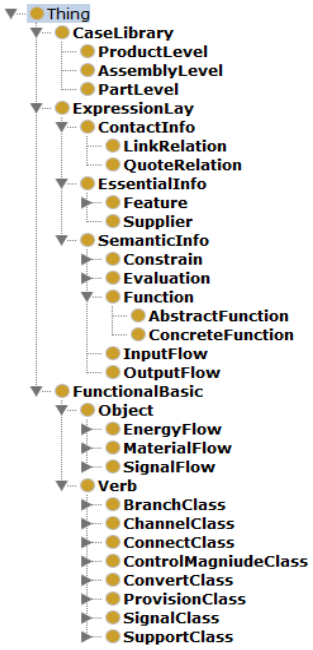

Figure 5: Organizational form modeling of the knowledge base.

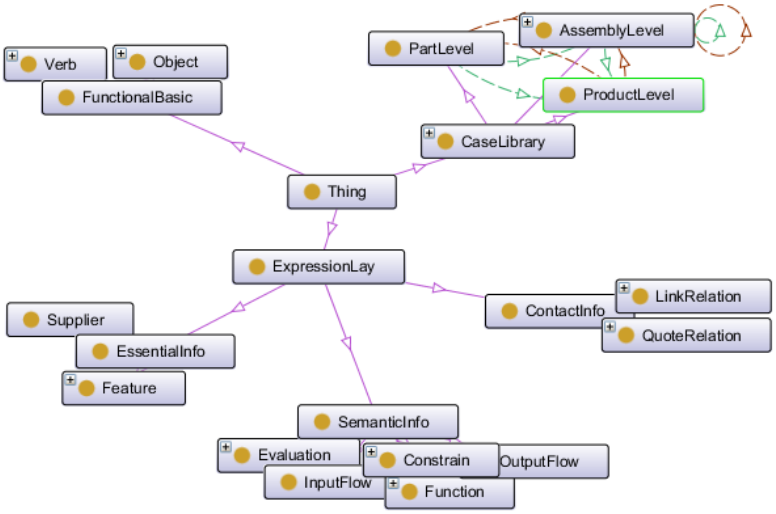

Figure 6: Conceptual relationships of the domain ontology.

The concept Evaluation is the complementary element, as shown in Figure 4. This index evaluates the design scheme from the aspects of system function, kinematic performance, dynamic performance,

ISSN 2616-5775 Vol. 4, Issue 7: 39-45, DOI: 10.25236/AJCIS.2021.040706

economy and compact structure, and provides support for the optimization analysis of the scheme generated by conceptual design.

Figure 5 shows the ontology organization structure tree established by the ontology modeling tool Protege.

As shown in Figure 6., a conceptual diagram is established, including the information layer, semantic layer, and relational layer.The domain ontology-based representation

Under the array model, most of the design knowledge can be represented based on ontologies, making the implicit knowledge explicit. Figure 7 shows a schema for representing knowledge of known structural components, together with representations of slider-crank mechanism and the frictional wheel as examples.

```
<NamedIndividual rdf:about="Name">
    <rdf:type rdf:resource="Corresponding_Level"/>
     /* A list of contact infomation */
    <KRO:hasQuoteRelation rdf:resource="values"/>
    <KRO:hasLinkRelation rdf:resource="values"/> ...
     /* A list of features of the component */
    <KRO:hasField rdf:resource="values"/>
    <KRO:hasCategory rdf:resource="values"/> ...
     /* A list of descriptions for the function */
    <KRO:hasAbstractFunction rdf:resource="values"/>
    <KRO:hasInputFlow rdf:resource="values"/>
    <KRO:hasOutputFlow rdf:resource="values"/>
    <KRO:hasEntiretyConstrain rdf:resource="values"/>
    <KRO:hasInputConstrain rdf:resource="values"/>
    <KRO:hasOutputConstrain rdf:resource="values"/> ...
     /* A list of evaluations for the function */
    <KRO:hasevaluation rdf:resource="values"/> ...
</NamedIndividual>
```

(a) The general representation scheme of a known component

```
<NamedIndividual rdf:about="&KRO;Slider_Crank_Mechanism">
    <rdf:type rdf:resource="&KRO;AssemblyLevel"/>
    <KRO:hasQuoteRelation rdf:resource="&KRO;Slider"/>
    <KRO:hasQuoteRelation rdf:resource="&KRO;Crank"/>
    <KRO:hasField rdf:resource="&KRO;Mechanical"/>
    <KRO:hasCategory rdf:resource="&KRO;Mechanical_Drive_System"/>
    <KRO:hasAbstractFunction rdf:resource="&KRO;Convert_AngularVelocity_to_LinearVelocity"/>
    <KRO:hasInputFlow rdf:resource="&KRO;Angular_Velocity"/>
    <KRO:hasOutputFlow rdf:resource="&KRO;Linear_Velocity"/>
    <KRO:hasEntiretyConstrain rdf:resource="&KRO;Motion_Axode_Change"/>
    <KRO:hasEntiretyConstrain rdf:resource="&KRO;Reversibility"/>
    <KRO:hasOutputConstrain rdf:resource="&KRO;To_And_Fro"/>
    <KRO:hasOutputConstrain rdf:resource="&KRO;Unconstant_Speed"/>
    <KRO:hasOutputConstrain rdf:resource="&KRO;Consistent_Intermittence"/>
    <KRO:hasevaluation rdf:resource="&KRO;Security=4"/>
</NamedIndividual>
```

(b) The representation case of slider-crank mechanism

```
<NamedIndividual rdf:about="&KRO;Frictional_Wheel">
    <rdf:type rdf:resource="&KRO;AssemblyLevel"/>
    <KRO:hasLinkRelation rdf:resource="&KRO;Frictional_Wheel"/>
    <KRO:hasField rdf:resource="&KRO;Mechanical"/>
    <KRO:hasCategory rdf:resource="&KRO;Mechanical_Drive_System"/>
    <KRO:hasAbstractFunction rdf:resource="&KRO;Convert_AngularVelocity_to_LinearVelocity"/>
    <KRO:hasInputFlow rdf:resource="&KRO;Angular_Velocity"/>
    <KRO:hasOutputFlow rdf:resource="&KRO;Linear_Velocity"/>
    <KRO:hasEntiretyConstrain rdf:resource="&KRO;Motion_Axode_Change"/>
    <KRO:hasEntiretyConstrain rdf:resource="&KRO;Reversibility"/>
    <KRO:hasEntiretyConstrain rdf:resource="&KRO;Movement_Form_Transformation"/>
    <KRO:hasOutputConstrain rdf:resource="&KRO;Consistent_Intermittence"/>
    <KRO:hasOutputConstrain rdf:resource="&KRO;Consistent_Stability"/>
</NamedIndividual>
```

(c) The representation case of frictional wheel

Figure 7: A schema for representation of known components.

3. Function Edit Distance Matching Model

Similar to the matching with the basic information, the semantic matching of component function need to set a set of semantic feature set as the base vector function first. The function of each component semantics can be represented by the feature vectorthat the base vector is corresponding to. All the feature vectors of components functional semantic in the component knowledge base form the feature vector space. Based on this, a matching function can be defined as follows:

$$f : Q \times V \to R \tag{1}$$

For any query input $q \in Q$, we can calculate the correlation between q and the feature vector v corresponding to any component.

In this paper, the feature vector v corresponding to the functional semantics of a component is called the functional descriptor of the component. Functional semantic matching means that the functional descriptor input by query is matched with the functional descriptor of the component knowledge base through the matching function. The process of functional semantic matching is shown

ISSN 2616-5775 Vol. 4, Issue 7: 39-45, DOI: 10.25236/AJCIS.2021.040706

in Figure 8, which is divided into online and offline operation. Offline operation refers to that function descriptors are obtained after feature extraction from the component function semantics in the component knowledge base. All function descriptors form a matching vector space, and the descriptors are indexed in a certain way for high-performance retrieval. Online operation refers to the function descriptor of the query input obtained by feature extraction, matching the function descriptor of the query input according to the index structure and the vector space, and outputting the corresponding results in the knowledge base to finish the query.

Functional descriptor is the feature vector corresponding to the spatial base vector obtained from feature extraction of component functional semantics. Before the important definition of base vector describing vector space, the definition of functional semantic feature is proposed first, as shown in Figure 9.

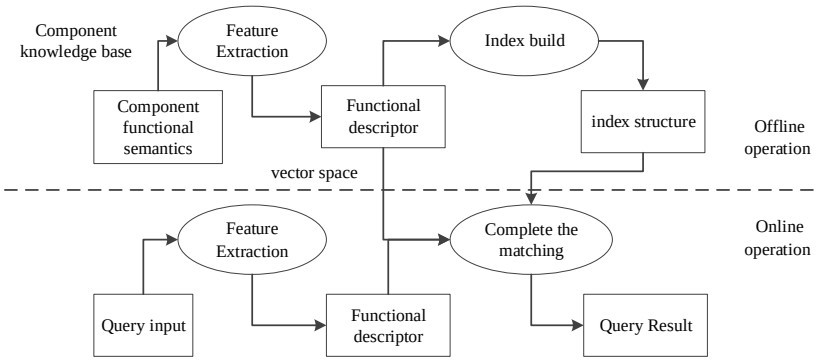

Figure 8: Functional semantic matching process.

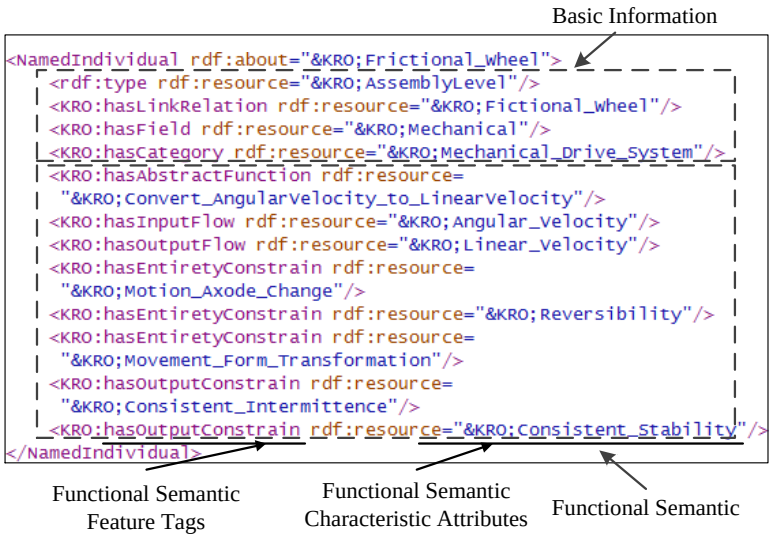

Figure 9: Functional semantic feature definition.

4. Formal Description of Base Vector, Feature Vector and Matching Function

Based on the design knowledge representation method, it can be seen that the ontology based component functional semantic representation is a set of functional semantic features, namely

$$K(c) = \{s_1, s_2, \cdots, s_k \,|\, s_i = < la_i, va_i >, la_i \in L, va_i \in V\} \tag{2}$$

Where, c is the defined component, $K(c)$ is the functional semantic representation of the component, L is the set composed of tags, V is the attribute set, and s_i is the functional semantic features containing tags la and attributes va. Equation (2) indicates that the functional semantics of the component c is composed of k functional semantic features.

The base vectors in vector space are the set of the input-output streams of all component functional semantics in the component knowledge base and the non-repeated functional semantic features

ISSN 2616-5775 Vol. 4, Issue 7: 39-45, DOI: 10.25236/AJCIS.2021.040706

contained in the constraints. The function descriptors of each component are the feature vectors corresponding to the base vectors. As the functional gerunds that designers define components are subjective, the corresponding functional properties are not listed in the base vectors. At the same time, the definition of base vector requires when the new component knowledge is stored in knowledge base, it must first check whether the component function semantics containing all the features of semantic features have already existed in the existing base vector. If exists, deposit it in the component library directly. If there is a new functional semantic features, it need to update the existing base vector, and then deposit the component in the component library.

For example, the functional semantics of c_1, c_2 and c_3 constitute the component knowledge base, which are expressed as follows:

$$K(c_1)=\{s_1,s_2,s_3\};K(c_2)=\{s_2,s_4\};K(c_3)=\{s_1,s_5,s_6\} \tag{3}$$

Then, the functional semantic feature set of the component knowledge base, namely the base vector, is:

$$e=(s_1,s_2,s_3,s_4,s_5,s_6)^T \tag{4}$$

The function descriptor of component c_1, c_2 and c_3, that is, the eigenvector of the corresponding to the base vector e, is:

$$u_1=(1,1,1,0,0,0);u_2=(0,1,0,1,0,0);u_3=(1,0,0,0,1,1) \tag{5}$$

u_1 is the function descriptor of the component c_1 in the knowledge base vector space.

The functional semantic feature sets of vector space and the functional descriptors of components are defined above. The following sections discuss the matching functions in the functional semantic matching. According to Equation (1), the matching function is used to calculate the matching number between the function descriptor of query inputand the descriptor corresponding to any component in the knowledge base. In this paper, the matching function and function edit distance in function descriptors are defined as follows:

Function edit distance function d is a function that calculates the function edit distance from the function descriptor u_q of query input q to the function descriptor u_i of the (to be matched) component c_i.

The function editing distance from u_q to u_i is the 1-norm of the vector that obtained after by subtracting the feature vector input u_i from the feature vector u_q, which is formulated as follows:

$$d(u_q,u_i)=\left\|u_q-u_i\right\|_1=\left\|u_d\right\|_1=\sum_{j=1}^{n}\left|u_{d,j}\right| \tag{6}$$

For example, the function descriptors of c_1 and c_2 are $u_1=(1,1,1,0,0,0)$ and $u_2=(0,1,0,1,0,0)$. The Function edit distance can be calculated as follows:

$$d(u_1,u_2)=\left\|u_1-u_2\right\|_1=3 \tag{7}$$

The function editing distance of function descriptor c_1 and c_2 is 3. What needs to be pointed out is that the user query intention only pays attention to the function of semantic features that have appeared in the query input and ignores the other functional semantic features of base vector in vector space. For the function descriptor obtained by extracting the feature of query input, the user concerned feature semantic is 1 and the ignored feature semantic is the default (denoted as w). The result subtracting 1 or 0 from w is 0. By setting the default bit of the function descriptor of user's query input, the query intention can be processed correctly in the retrieval process without being affected by the change of the base vector in the vector space.

ISSN 2616-5775 Vol. 4, Issue 7: 39-45, DOI: 10.25236/AJCIS.2021.040706

5. Conclusions

This paper has conducted the formal descriptions of base vector, feature vector and matching function in the component semantic matching. The query input is converted into the function descriptors of solving the query input and the model of components functions descriptor function edit distance model in knowledge base. The match results is to push the component knowledge conformed to the reserved functional edit distance back to user after sorted by distance. In the future, we can study how to quickly retrieve the function descriptors of the qualified components in the vector space of the knowledge base, that is, the establishment method of components matching index with high performance.

References

[1] Pahl G, Beitz W. Engineering design. London: Springier, 1994:1-20.
[2] Suh N P. The principles of design. New York: Oxford university press, 1990.
[3] Altšuller G S. The innovation algorithm, TRIZ, systematic innovation and technical creativity. Technical Innovation Center Inc., 2000.
[4] Deng Y M, Zhu Y W. Function to structure/material mappings for conceptual design synthesis and their supportive strategies. International Journal of Advanced Manufacturing Technology. 2009, 44(11): 1063-1072.