

A graph-based computerized optimal conceptual design synthesis within the distributed multi-disciplinary resource environment

Bin Chen*, Jie Hu and Weixing Chen

* State Key Laboratory of Mechanical System and Vibration, Shanghai Jiao Tong University, Shanghai, 200240, China
Corresponding Author: chenbinsun@outlook.com

Submitted : 05/01/2019

Revised : 18/10/2020

Accepted : 31/10/2020

ABSTRACT

The trend of inter-disciplinary conceptual design synthesis requires designers to involve more and more distributed multi-disciplinary design resources. Therefore, this paper proposes a graph-based computerized optimal conceptual design synthesis to help designers explore novel design schemes within the distributed multi-disciplinary resource environment. The design resources tightly related to the design goal can be extracted from the huge resource environment by a proposed searching engine. The optimal design scheme can be generated from these related design resources by a proposed graph-based algorithm. A set of computer applications called Automatic Conceptual Design System (ACDS) is established to verify the feasibility of this proposed conceptual design synthesis, and a garbage power system's conceptual design is completed by this software prototype.

Keywords: Distributed multi-disciplinary resource environment; Optimal conceptual design synthesis; Graph-based algorithm.

INTRODUCTION

The trend of inter-disciplinary conceptual design synthesis requires designers to involve more and more distributed multi-disciplinary design resources. These design resources are distributed at different locations and institutes, and at the same time, they belong to various disciplines. They actually construct a distributed multi-disciplinary resource environment. If this resource environment can be fully introduced into the conceptual design synthesis, more innovative and creative design schemes can be obtained in a much wider exploring space. However, the physical and disciplinary barriers in this distributed multi-disciplinary resource environment block the bridge between the designers and the design resources. Generally, designers can only obtain the design resources around their locations or institutes, and meanwhile, they are only familiar with the design resources in their own disciplinary domains. These two barriers largely limit the application of the distributed multi-disciplinary resource environment.

With the advanced Internet and computer technologies, the physical and disciplinary barriers between the designers and the design resources are gradually disappearing. The communicating efficiency among distributed

locations is largely promoted, and the advanced logistic network also makes it is more and more convenient for the designers to obtain new information outside their experience range. Therefore, this paper proposes a uniform multi-input and multi-output representation for the multifarious design resources to make the distributed multi-disciplinary resource environment more accessible, and then, establishes a computerized optimal conceptual design synthesis to help the designers complete the repetitive and mechanical works of data searching and processing during the optimal conceptual design synthesis. This proposed computerized optimal conceptual design synthesis include two parts, i.e. the design resource searching engine and the optimal design scheme generating algorithm. The design resource searching engine extracts the design resources tightly related to the design goal from the numerous design resources in the distributed multi-disciplinary resource environment. As for the optimal design scheme generating algorithm, it helps find out the optimal design scheme from the related design resources offered by the design resource searching engine. The whole study consists of three parts, i.e. the three basic concepts and their representations, the graph-based computerized optimal conceptual design synthesis, and the illustrative case. In the rest of this paper, these three parts of work are shown, and finally, some discussions about this approach is made, and the whole work and the future study plan are also concluded.

LITERATURE REVIEW

The computerization of the optimal conceptual design synthesis should be established on a proper representation of the design resources. There are many inspiring former works focused on the design resource representation. Among them, the parameter theory indicates that the information abstraction plays a basic role in efficient representation. A multi-relational and interactive model was developed for the design resource representation in the conceptual design. (Camelo & Mulet, 2010) A qualitative and heuristic representing approach was proposed for the design resources by classifying their various perspectives. (Li et al., 1996) A representation was constructed for the design resources in the mechanical design based on their functions and behaviors. (Deng, 2002) Furthermore, matrix theory can also be used to build a design resource representation. A matrix-based representation was proposed as the foundation of computer-aided conceptual design. (Kota & Chiou, 1992; Chiou & Kota, 1999) A representation was established for the functional design of mechanical products by combining the data constructing methods of graph and matrix together. (Zhang et al., 2005) Some researchers tried to establish the representation based on language theory which involves the tools like syntax, semantics, pragmatics, etc. A design resource representing model was established based on the design language SysML. (Christophe et al., 2010) A language of describing the behavior of fixed-axes mechanisms was proposed for the mechanism design. (Joskowicz & Neville, 1996) Besides, the flow theory, often combined with the graph theory, is also inspiring for some researches about design resource representation. A flow-based behavior graph was developed for the conceptual design. (Welch & Dixon, 1994) A design representation called conceptual dynamics graph was proposed to describe the information of physical design components. (Wu et al., 2008) In some former works (Chen & Xie, 2017; Chen & Xie, 2017; Chen & Xie, 2018), the flow concept was developed into an input-output structure for the representation of the design resources and design goals.

With the above inspiring works on design resource representation, many approaches were proposed for the conceptual design synthesis. A framework for computer-aided conceptual design synthesis was developed for the mechatronics products. (Komoto & Tomiyama, 2012) An agent-based approach was proposed for the conceptual design in a dynamic environment (Campbell et al., 1999), and it was further developed into a design approach to managing automated design synthesis (Campbell et al., 2003). An approach was developed for the functional synthesis of solutions in mechanical conceptual design. (Chakrabarti & Bligh, 1994; Chakrabarti & Bligh, 1996; Chakrabarti & Bligh, 1996) A general graph grammar methodology was proposed for the structure synthesis of mechanisms. (Schmidt & Chase, 2000) A framework was developed for computational design synthesis based on graph-grammars and function-behavior-structure. (Helms et al., 2009) A methodology automating the function-based synthesis paradigm was established by combining various computerized methods. (Kurtoglu et al., 2010) A

computerized method was proposed to generate design solutions based on the mechanism combination and reasoning. (Tang, 2006) A structural framework was established with a method set including product specification, functional representation, artifact representation, artifact behavior, and tolerance representation. (Roy et al., 2001) These researches are mainly focus on constructing frameworks and models for the conceptual design synthesis. In order to find more alternative design schemes to promote the design innovation and creativity, a wider design resource space should be introduced, which means the design synthesis efficiency should be promoted for the increasingly growing workload of data processing. Therefore, many researchers were focused on the design data processing and synthesis algorithm improving. A data warehousing system was established to integrate and organize the design information and data for the conceptual design. (Golfarelli et al., 1998) The system-in-use data was introduced from in-service records and receiver needs regarding the use of large, capital-intensive, technical assets to generate design solutions at the conceptual design stage. (Hussain et al., 2012) A computerized tool was proposed for the conceptual design synthesis based on analytical and synthetic strategies. (Umeda et al., 1996) An automatic approach was proposed to complete the function identification in the conceptual design of mechanistic systems. (Lin, 2008)

With these above precious works, large design databases can be queried, and the design solution space is largely broadened. However, the feasibility and superiority of design solutions should be inspected based on some constraints. A constraint-based approach was established under a computational reasoning environment. (O'Sullivan, 2002) A generic constraint-based approach was proposed with the help of a comprehensive functional design model. (Deng et al., 2000) The constraint strategy was combined with the input-output model of design resources and functional requirements to establish a graph structure modelling a large amount of design resources as the exploring environment for the conceptual design synthesis. (Chen & Xie, 2017) Furthermore, the conceptual design process was concluded as generating design resource chains in this graph structure. (Chen & Xie, 2016; Chen & Xie, 2016) Based on these works, a criterion was established to test the feasibility and stability of a design scheme, so that, the designers can narrow down the alternative space and make the final decision more precisely. (Chen & Xie, 2017) However, there is still no quantitatively evaluating system for the design resources and design schemes to help improve these works for the design optimization, and the model used in these former works can only support single-input and single-output mode, which also limits the practicability. Therefore, this paper proposes a novel graph-based computerized optimal conceptual design synthesis, which includes a design resource searching engine and an optimal design scheme generating algorithm, to assist designers explore a wider design schemes efficiently within the large distributed multi-disciplinary resource environment.

OVERVIEW

As shown in Figure 1, the proposed graph-based computerized optimal conceptual design synthesis consists of two parts, i.e. the design resource searching engine, and the optimal design scheme generating algorithm. The design resource searching engine is aimed to extract the related design resources from the numerous design resources in the distributed multi-disciplinary resource environment. These searched related design resources play the roles of the raw materials for the design scheme generating. This searching process provides the optimal design scheme generating algorithm a compacter running environment, so that, the useless calculations can be avoided, the algorithm running time can be shortened, and the algorithm efficiency can be promoted. Within this compacter running environment, the optimal design scheme generating algorithm tries to generate the optimal design scheme automatically based on the evaluating standard set by the designers.

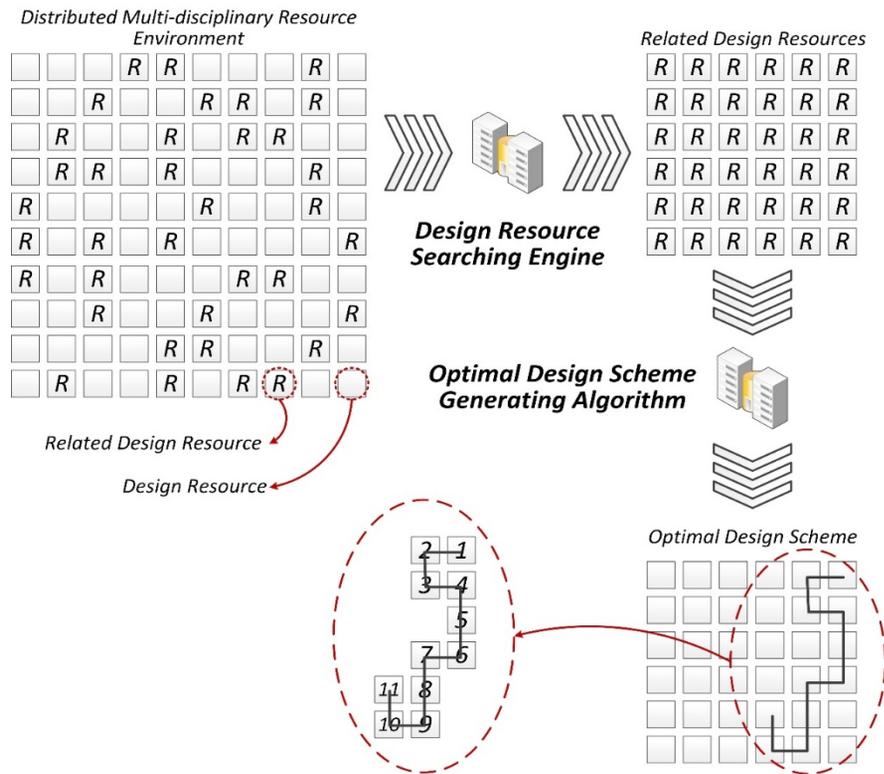


Figure 1. Overview of the graph-based computerized optimal conceptual design synthesis

THREE BASIC CONCEPTS AND THEIR REPRESENTATION

In order to realize the graph-based computerized optimal conceptual design synthesis, three basic concepts are introduced in this study, i.e. functional object, functional requirement, and functional unit. Functional object is similar to the concept “flow” in the research of Welch and Dixon [8], but the scope of its cover is not limited within material, energy and information. A functional object may be a specific material, such as steel or wood, and it may also be some physical concept, like electrical, optical, rotation, force, etc. A functional object can be transformed into another. If this transformation meets the design goal, it is defined as a functional requirement. As for an entity which can achieve some functional object transformation, it is defined as a functional unit. It may be a machine or equipment, and it may also be a computer program, an operation process or an administrative system. Functional unit is the specific form of design resource, which means the distributed multi-disciplinary resource environment is actually constructed by numerous functional units, and the aim of conceptual design synthesis is searching proper functional units in the distributed multi-disciplinary resource environment and combining them together for the functional requirement.

Presentation of a Functional Object

The representation of a functional object is made up of two parts. The first part is the name of the functional object as an exclusive identification. The second part is the set of the functional object’s features. Every feature is represented by name, value range and unit. As shown in Figure 2, this functional object’s name is “Rotary motion”, and its features include “Rotational speed”, “Torque”, “Axial orientation”, “Direction” and “Continuity”.

```

Function_Object_Name:
Funciton_Object_Features:
{
  Feature_1: {Feature_1_Name, Feature_1_ValueRange, Feature_1_Unit}
  Feature_2: {Feature_2_Name, Feature_2_ValueRange, Feature_2_Unit}
  ...
}

Function_Object_Name: Rotary_motion
Funciton_Object_Features:
{
  Feature_1: {Rotational_speed, [2000, 3000], rpm}
  Feature_2: {Torque, 1600, g·cm}
  Feature_3: {Axial_orientation, {X, Y}, Null}
  Feature_4: {Direction, {Clockwise, Anticlockwise}, Null}
  Feature_5: {Continuity, true, Null}
}

```

Figure 2. Representation of a functional object

Representation of a Functional Requirement

According to the definition of functional requirement, it is determined by the input and output functional objects jointly, which represent the union of its input and output functional objects. As shown in Figure 3, a functional requirement may have multiple input functional objects, and it may also have multiple output functional objects.

```

Input_Funciton_Objects:
{
  Function_Object_1: {Name, Features}
  Function_Object_1: {Name, Features}
  ...
}
Output_Funciton_Objects:
{
  Function_Object_1: {Name, Features}
  Function_Object_1: {Name, Features}
  ...
}

```

Figure 3. Representation of a functional requirement

Representation of a Functional Unit

A functional unit is an entity which can achieve the transformation from its input functional objects to its output functional objects, therefore, its representation contains the information of both input and output functional objects. Furthermore, the properties of the functional unit itself should also be considered in its representation, including its backgrounds and features. These two kinds of information play important roles in the functional unit searching within the distributed multi-disciplinary resource environment and the optimal design scheme generating. The backgrounds

include four kinds of information about the functional unit, including discipline, industry, region and institute. They can help the design resource searching engine better locate the related functional units for the functional requirement. As for the features, they are measurable values which characterize the functional unit's specific parameters.

Figure 4 conclude the representation of a functional unit. A functional unit is represented by five parts, i.e. name, input functional objects, output functional objects, backgrounds, and features. In this study, seven features are considered into the representation, including price, operation cost, power waste, failure rate, weight, volume, and emission. They are usually the main measurable features considered by the designers as the design constraints, and they are also the common concerns of the designers' clients for design scheme evaluation.

<pre>Name: Input_Funciton_Objects: { Function_Object_1, Function_Object_2, ... } Output_Funciton_Objects: { Function_Object_1, Function_Object_2, ... }</pre>	<pre>Backgrounds: { Discipline, Industry, Region, Institute } Features: { Price, Operating_Cost, Power_Waste, Failure_Rate, Weight, Volume, Emission }</pre>
---	--

Figure 4. Representation of a functional unit

GRAPH-BASED COMPUTERIZED OPTIMAL CONCEPTUAL DESIGN SYNTHESIS

As mentioned before, the proposed graph-based computerized optimal conceptual design synthesis consists two parts, i.e. the design resource searching engine and the optimal design scheme generating algorithm. These two parts are introduced here next.

Design Resource Searching Engine

As mentioned before, functional unit is the specific form of design resource, and the distributed multi-disciplinary resource environment is actually constructed by numerous functional units. Therefore, the design resource searching engine is aimed to search related functional units from the distributed multi-disciplinary resource environment for the functional requirement. Design resource searching engine is set by the designers who deeply understand the whole design mission, so that, a small group of proper related functional units can be searched out from the numerous functional units in the distributed multi-disciplinary resource environment. As shown in Figure 5, the designers can establish searching conditions for the engine according to the functional units' backgrounds and features. Only the functional units which meet all the searching conditions simultaneously can be picked out by the searching engine. All these picked functional units construct the related functional unit set, which is the running environment for the optimal design scheme generating algorithm.

Search Condition
(based on Backgrounds and Features)

(Discipline IN Physics OR IN Chemistry)
 AND (Industry IN Engineering OR IN Mechanical)
 AND (Region WITH Un-limited)
 AND (Institute WITH Un-limited)

AND (Price <= 2000 dollars)
 AND (Operating Cost <= 20 dollars/hour)
 AND (Power Waste <= 90 W)
 AND (Failure Rate <= 10 fit)
 AND (Weight >= 50 kg AND <=100 kg)
 AND (Volume >= 2 m³ AND <= 5 m³)
 AND (Emission <= 0.5 mg/m³(O³))

Figure 5. Searching conditions for the design resource searching engine

Optimal Design Scheme Generating Algorithm

Optimal design scheme generating algorithm is aimed to generate the optimal design scheme within the related functional unit set. It is established on graph theory, which means the related functional unit set's data structure is constructed as a graph in this study. In this graph, a design scheme is actually a path constructed by functional units connected together as a chain.

(1) Relational Graph

As shown in Figure 6, two functional units can be connected together via their input and output functional objects. In this case, if Output functional object 1 and Input functional object 4 have the same name and unit, and simultaneously, Output functional object 1's value range is contained by Input functional object 4's, the mapping from Output functional object 1 to Input functional object 2 can be determined. Based on this, if Functional unit 2's both input functional objects, Input functional object 4 and Input functional object 5, can be mapped from Functional unit 1's output functional objects, it can be determined that Functional unit 1 is connected to Functional unit 2.

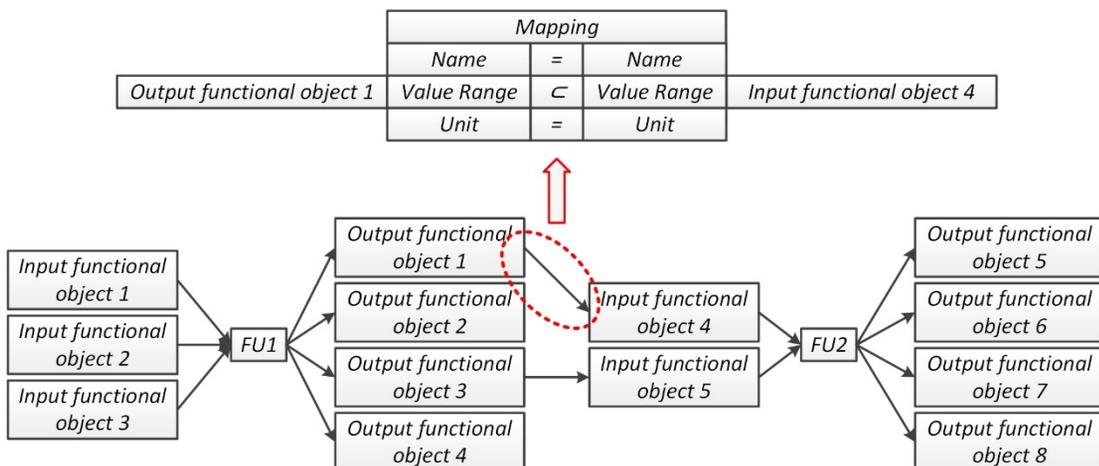


Figure 6. Connection between two functional units

Based on the functional unit connection, all the functional units in the related functional unit set can be connected together via graph vertexes as hubs, and a relational graph can be established as shown in Figure 7. In this relational graph, functional unit plays the role of graph edge. As for the graph vertexes, they can be classified into three kinds, i.e. emitting vertex, receiving vertex, and transmitting vertex. In this case, Vertex 5 is an emitting vertex. It is connected to Vertex 4 through Functional unit 14 as a graph edge, but there is no other vertex connected to it. Vertex 6 is a receiving vertex. Vertex 4 is connected to it via Functional unit 15 as a graph edge, but Vertex 6 is not connected to any other vertex. Vertexes 1, 2, 3, and 4 are transmitting vertexes, they are connected to other vertexes and there are also other vertexes connected to them.

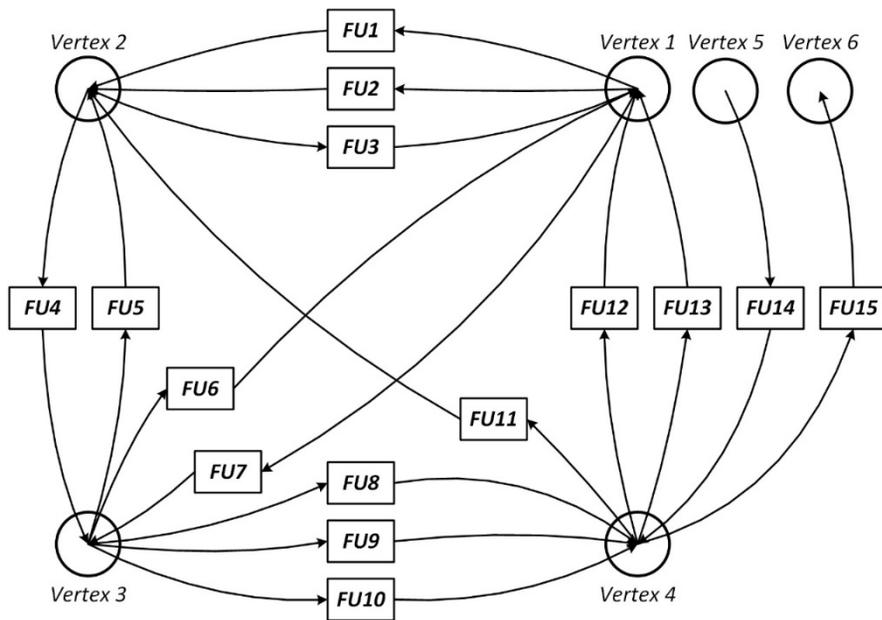


Figure 7. Relational graph

(2) Storage of the Relational Graph

A relational graph is stored in computers by an orthogonal list. Figure 8 shows an orthogonal list corresponding to the relational graph shown in Figure 7. In this list, the graph vertexes are represented by the header nodes, and the functional units (graph edges) are represented by the ordinary nodes. A header node consists of three data domains, i.e. the number domain, the backward pointer domain, and the forward pointer domain. The number domain records the serial number of the vertex. The forward pointer domain stores the pointer to the first functional unit precursor. The backward pointer domain stores a pointer to the first functional unit successor. As for an ordinary node, it consists of four data domains, i.e. the name domain, the data domain, the forward pointer domain, and the backward pointer domain. The name domain records the functional unit's name. The data domain records all kinds of information about the functional unit. The forward pointer domain stores the pointer to the next functional unit precursor. The backward pointer domain stores a pointer to the next functional unit successor.

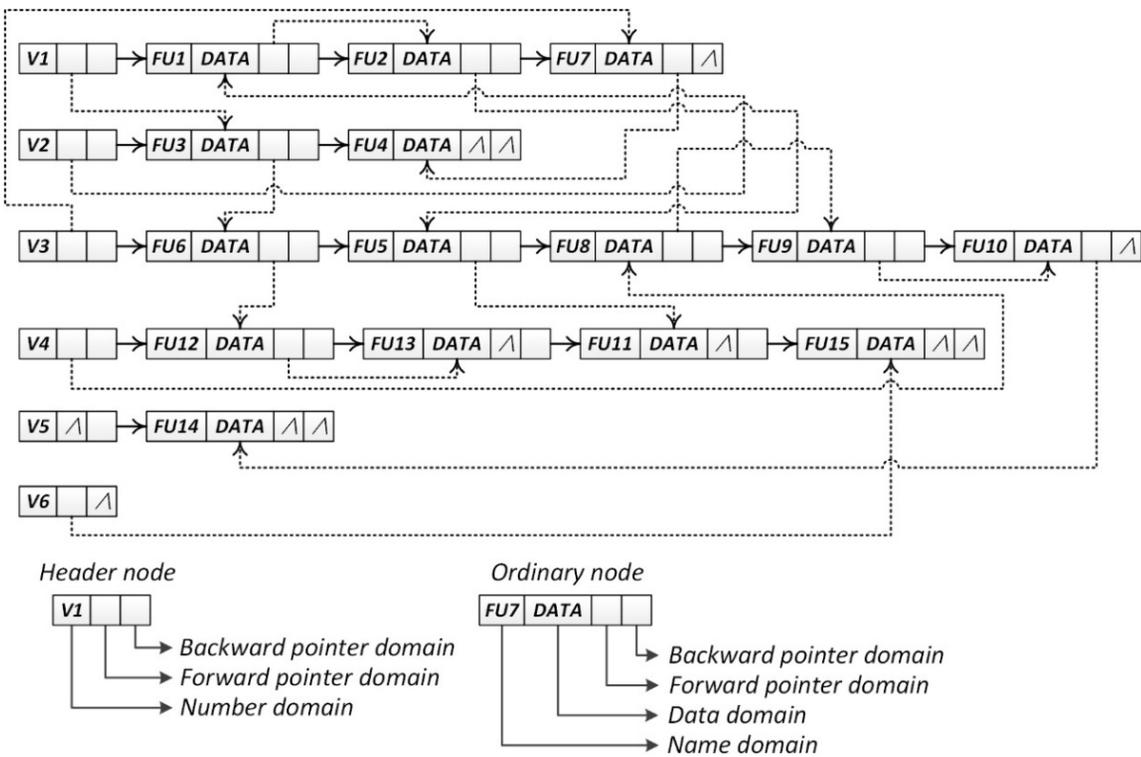


Figure 8. Orthogonal list as the storage of the relational graph

(3) Evaluating Index

As mentioned before, in the relational graph, a design scheme is a path consisting of several functional units connected together as a chain. Therefore, the evaluating method of a design scheme can be established on the information of its component functional units. In the representation of a functional unit, there are two kinds of information about its own properties, i.e. backgrounds and features. Among them, the features are measurable values which can be constructed into a weighted average as the evaluating index for the functional unit. As for the whole design scheme, its evaluating index can be established by summing up the evaluating indexes of all its component functional units.

The following equation shows the expression of a functional unit’s evaluating index. It is a weighted average of the seven features of the functional unit. The designers should set these features’ weights to determine their relative importance based on the design goal.

$$E = \sum_{i=1}^7 w_i F_i \quad (1)$$

In this equation, E represents the evaluating index of the functional unit. The meanings of $F_1 \sim F_7$ are shown in the following list:

- F_1 Cost
- F_2 Operating Cost
- F_3 Power waste
- F_4 Failure rate
- F_5 Weight
- F_6 Volume
- F_7 Emission

And $w_1 \sim w_7$ represent the seven feature weights, the sum of which is 1. As for the evaluating index of a design scheme, its expression is as follows.

$$I = \sum_{i=1}^n E_i \quad (2)$$

Here, I represents the evaluating index of this design scheme, and E_i means the evaluating index of its component functional unit No. i , and n represents the amount of its component functional units.

(4) Optimal Design Scheme Generating Algorithm

The functional unit evaluating indexes can be attached at the graph edges as their length in the relational graph as shown in Figure 9. Therefore, the evaluating index of a design scheme is represented by the length of the path corresponding to this design scheme in the relational graph. If the evaluating indexes of functional unit and design scheme are both set the smaller the better, the work of generating the optimal design scheme from the related functional unit set is turned into the work of finding the shortest path connecting one graph vertex to another in the relational graph.

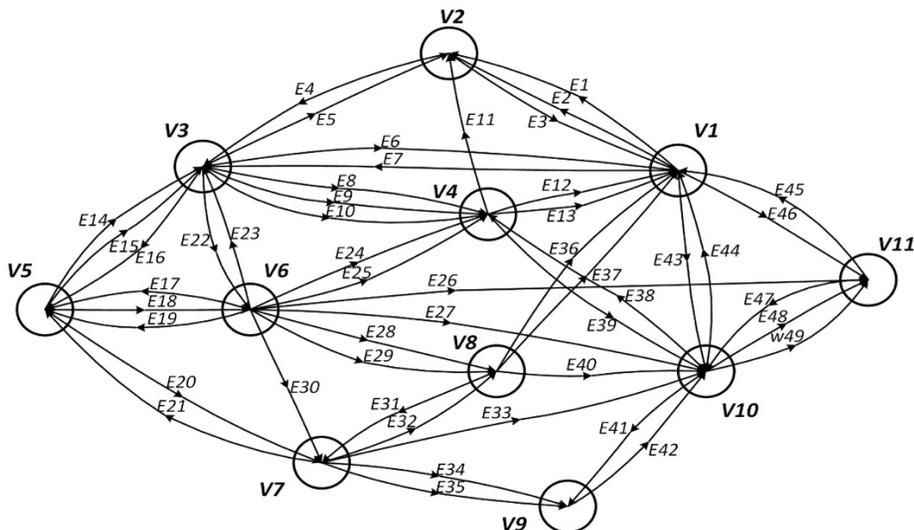


Figure 9. Relational graph attached by the functional unit evaluating indexes as the graph edge lengths

Based on the above assumption, it is obvious that among the multiple graph edges (functional units) connecting one graph vertex to another, only the shortest one (the functional unit with the smallest evaluating index) is meaningful for the shortest path finding (optimal design scheme generating). Therefore, the relational graph shown in Figure 9 can be simplified into the graph shown in Figure 10 by remaining the meaningful graph edges and removing the rest ones.

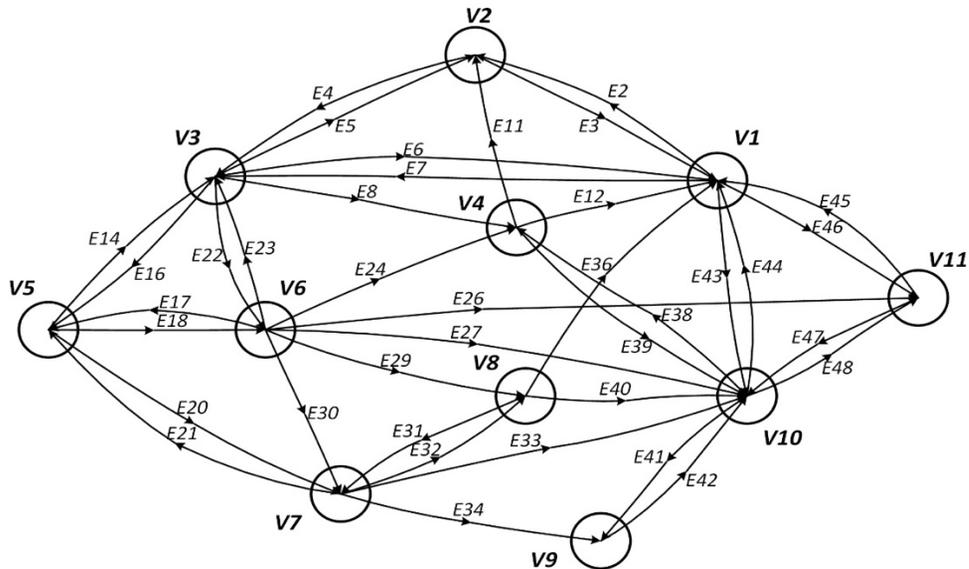


Figure 10. Simplified relational graph

The algorithm travels the simplified relational graph and tries to generate the optimal design scheme. At the beginning of this travelling, it should find a target graph vertex pair based on the input and output functional objects of the functional requirement. This target graph vertex pair consists of a starting graph vertex and a terminal graph vertex. In the simplified relational graph, there may be multiple paths from the starting graph vertex to the terminal graph vertex. All these paths represent the alternative design schemes meeting the functional requirement. Among them, the shortest one, which represents the optimal design scheme, is the one the algorithm should find.

To illustrate the detailed process of the travelling, a concept, SPSGV (Shortest Path from the Starting Graph Vertex), should be introduced. In the simplified relational graph, every graph vertex has an SPSGV. For a graph vertex, there may be multiple paths from the starting graph vertex to it, the shortest one is defined as its SPSGV. If there is no path from the starting graph vertex to this vertex, its SPSGV is defined as empty, and its SPSGV's length is defined as $+\infty$. In the next, the detailed running process of the algorithm is introduced based on SPSGV.

During the travelling, the algorithm should investigate the graph vertices in the simplified relational graph one by one, and work out their SPSGVs. If a graph vertex is investigated, it should be tabbed as "visited", so that, this graph vertex will not be visited repeatedly in the future. The algorithm keeps doing this investigation during the whole travelling until the terminal graph vertex is investigated. At this moment, the SPSGV of the terminal graph vertex is found out, and it represents the optimal design scheme. The flow chart of all this running process is shown in Figure 11.

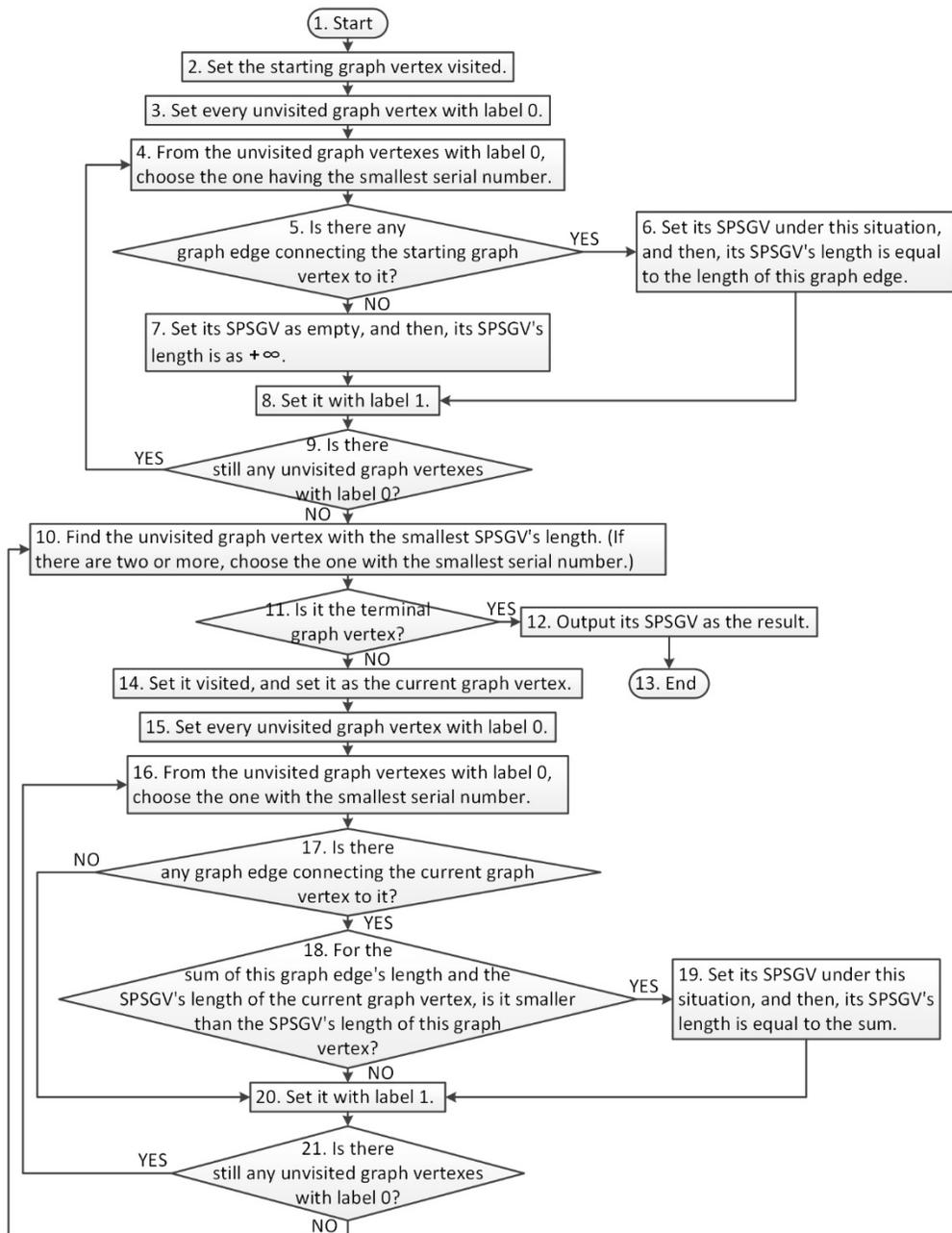


Figure 11. Flow chart of the optimal design scheme generating algorithm

The detail procedures are shown as follows.

- Step 1, start.
- Step 2, set the starting graph vertex visited.
- Step 3, set every unvisited graph vertex with label 0. This label exists for the tab of local loop.
- Step 4, from the unvisited graph vertices with label 0, choose the one having the smallest serial number. Here, the serial number is the identification of graph vertex as mentioned before.

- Step 5, is there any graph edge connecting the starting graph vertex to it? If yes, turn to Step 6, if no, turn to Step 7.
- Step 6, set its SPSGV under this situation, and then, its SPSGV's length is equal to the length of this graph edge.
- Step 7, Set its SPSGV as empty, and then, its SPSGV's length is as $+\infty$.
- Step 8, set it with label 1. So that the local loop will go on moving.
- Step 9, is there still any unvisited graph vertexes with label 0. If yes, turn to Step 4, if no, the local loop is end, turn to Step 10.
- Step 10, find the unvisited graph vertex with the smallest SPSGV's length. If there are two or more, choose the one with the smallest serial number.
- Step 11, is it the terminal graph vertex? If yes, turn to Step 12, if no, turn to Step 14.
- Step 12, output its SPSGV as the result.
- Step 13, end.
- Step 14, set it visited, and set it as the current graph vertex.
- Step 15, set every unvisited graph vertex with label 0. Here, a local loop is started.
- Step 16, from the unvisited graph vertexes with label 0, choose the one with the smallest serial number.
- Step 17, is there any graph edge connecting the current graph vertex to it? If yes, turn to Step 18, if no, turn to Step 20.
- Step 18, for the sum of this graph edge's length and the SPSGV's length of the current graph vertex, is it smaller than the SPSGV's length of this graph vertex? If yes, turn to Step 19, if no, turn to Step 20.
- Step 19, set its SPSGV under this situation, and then, its SPSGV's length is equal to the sum.
- Step 20, set it with label 1.
- Step 21, is there still any unvisited graph vertexes with label 0? If yes, turn to Step 16, if no, turn to Step 10.

ILLUSTRATIVE CASE

Using Microsoft Visual Studio 2010, the proposed graph-based computerized optimal conceptual design synthesis is packed into a set of computer applications, which is called Automatic Conceptual Design System (ACDS). Designers can apply ACDS to the product conceptual design. As an illustrative case, the conceptual design of a garbage power system is completed by ACDS.

First of all, the functional requirement should be inputted. Here, daily garbage is the only one input functional object, and municipal electricity is the only one output functional object. From the survey of daily garbage and the supply standard of municipal electricity, the features of both input and output functional objects can be obtained, and then, inputted into ACDS. As shown in Figure 12, in the column of input functional objects, click "Add" button to create a new record of input functional object, and input the name of it. There is a button named "Edit" at the right end of this record. Click it to get the editing interface shown in Figure 13, and input the features of this input functional object here in the corresponding blanks, so that, the process of editing input functional objects can be completed. In the same process, edit the output functional objects. In this design mission, the functional requirement is shown in Figure 12, and the features of input and output functional objects are respectively shown in Figure 13 and Figure 14.

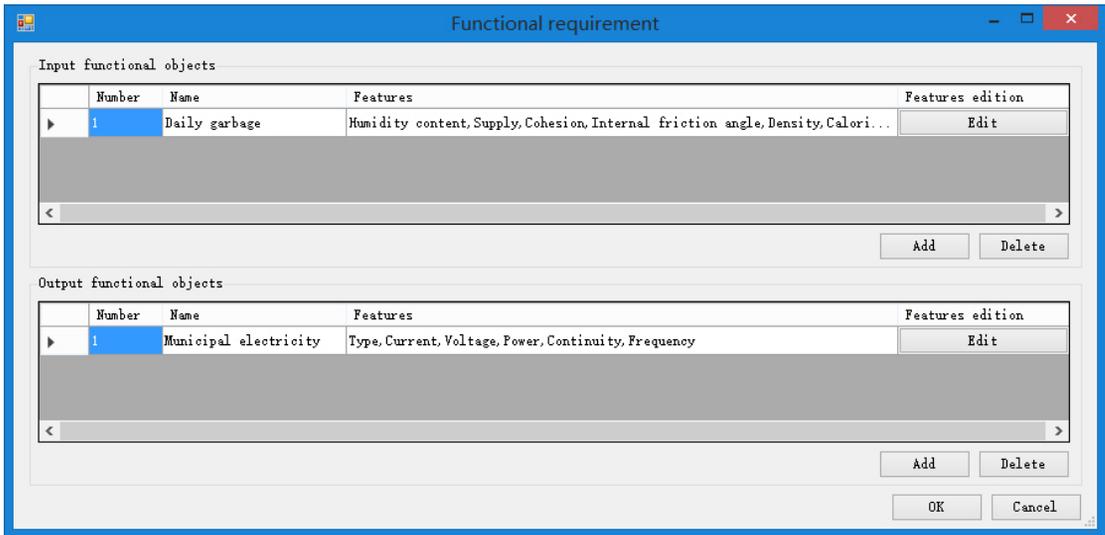


Figure 12. Interface for editing the functional requirement

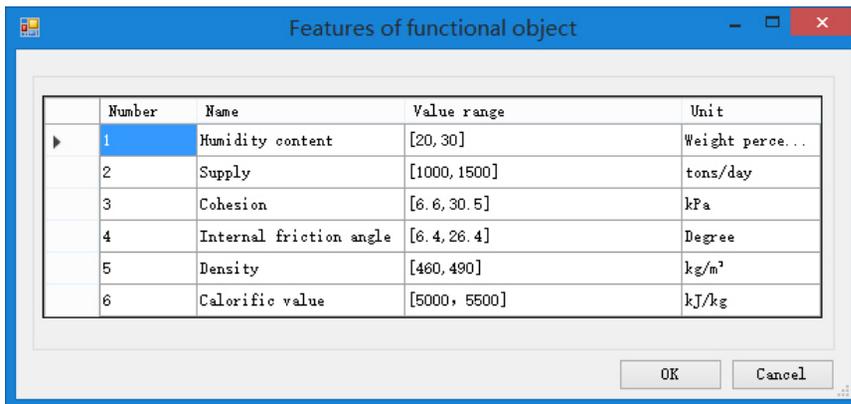


Figure 13. Interface for editing the input functional object features

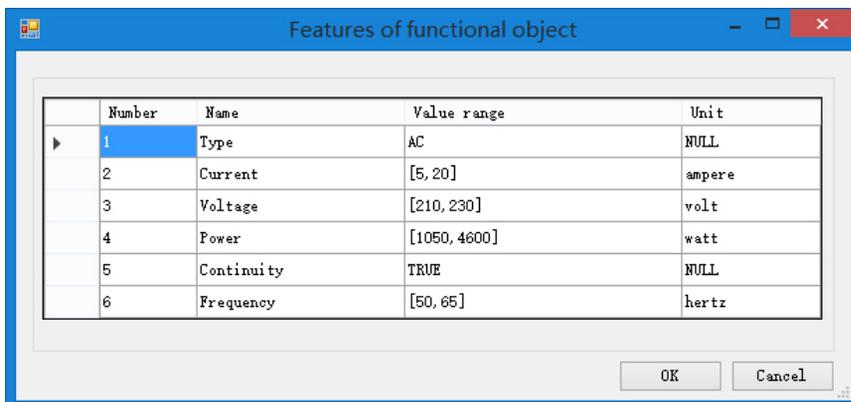


Figure 14. Interface for editing the output functional object features

In the next step, the design resource searching engine should be set. As shown in Figure 15, in the column of searching conditions in backgrounds, check the items needed in the four kinds of background information, including discipline, industry, region and institute. Then, in the neighboring column, input the searching conditions constructed by functional unit features, and then, click the "Generate" button to get the final searching condition.

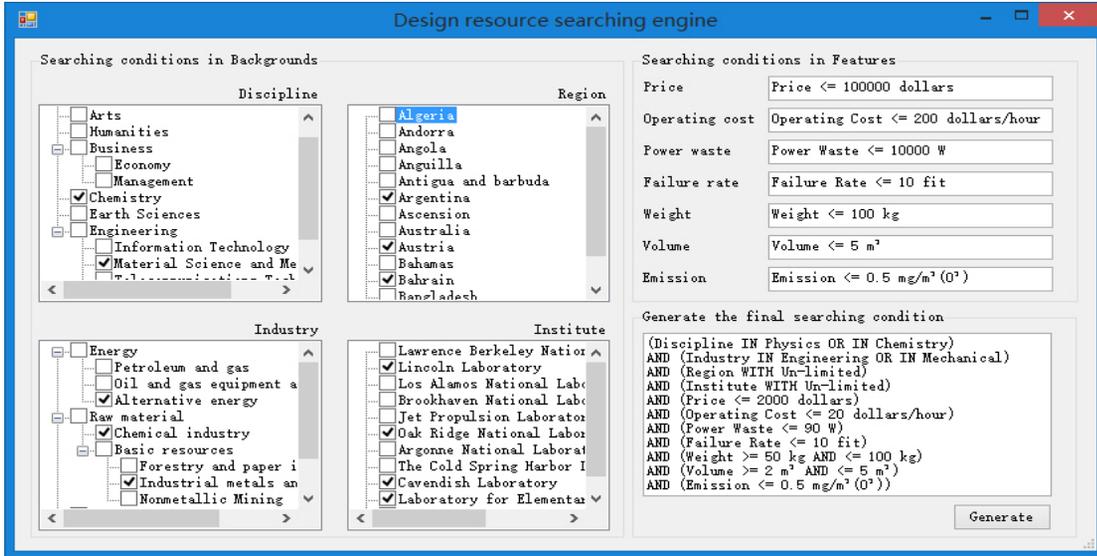


Figure 15. Interface for setting the design resource searching engine

ACDS can automatically search the related design resources meeting the searching conditions from the distributed multi-disciplinary resource environment, and then, jump into the related functional unit set browsing interface. Here, set the weight values in the expression of the functional unit evaluating index. As shown in Figure 16, in the corresponding blank, input the weight values, and click the "Calculate" button below. Then, the evaluating indexes of all the functional units in the related functional unit set can be figured out, and the result is shown in the table nearby.

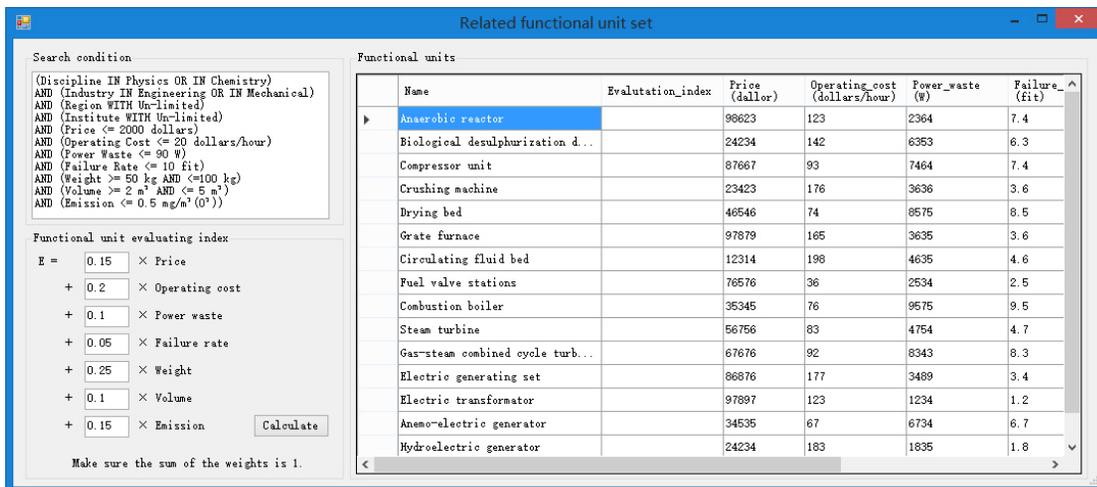


Figure 16. Interface of the related functional unit set

Finally, ACDS jumps into the optimal design scheme generating interface. As shown in Figure 17, in the relational graph, the two solid black graph vertexes constitute the target graph vertex pair based on the functional requirement. Clicking the "Generate" button, the optimal design scheme can be displayed in the text box down below the relational graph. This design scheme is made up of crushing machine, drying bed, grate furnace, compressor unit, steam turbine, electric generating set, and electric transformer. Its evaluating index is 78015.12 which is the smallest in all the alternative design schemes.

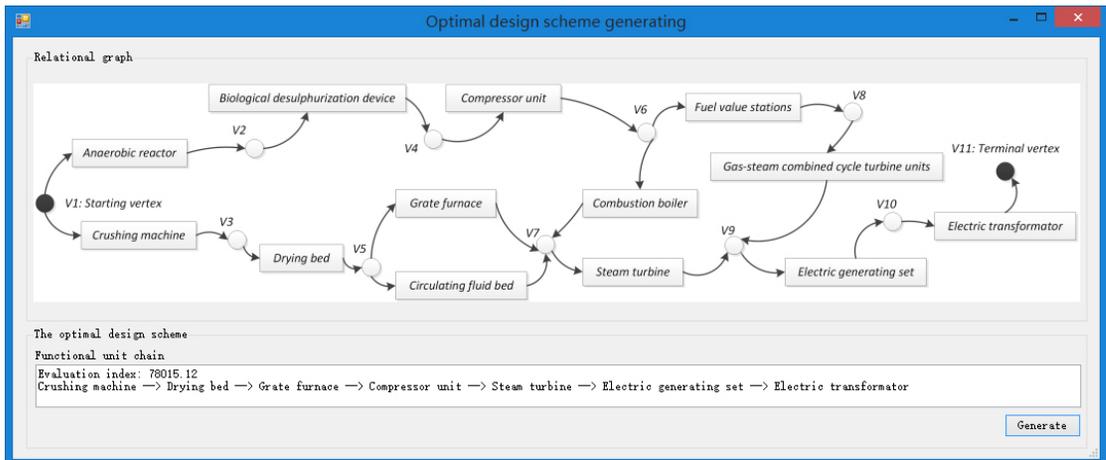


Figure 17. Interface for the optimal design scheme generating

DISCUSSION

Compared with other design synthesis approaches, this proposed approach has some unique advantages.

1. This proposed approach introduces the distributed multi-disciplinary resource environment into the conceptual design synthesis. The multifarious design resources in the resource environment are modeled under the uniform input-output structure, so that, their connections can be detected, and the entire resource environment can be modeled as a graph-structured running environment for the conceptual design synthesis. Within this running environment, the proposed design resource searching engine can search out the proper design resources based on the design goal, and the optimal design scheme can be obtained by the proposed optimal design scheme generating algorithm.
2. With this proposed approach, the repetitive and mechanical work of data searching and processing are done by computers. The main workload of optimal conceptual design synthesis exists in the design resource searching and optimal design scheme generating. These two parts of the work are repetitive and mechanical operations which is not suitable and economical for the human designers' labor. In this study, these two parts of the work are performed by the computers via the proposed design resource searching engine and optimal design scheme generating algorithm.
3. The algorithm used in this proposed approach has a high efficiency. The running environment of the synthesis algorithm is constructed only by the design resources tightly related to the design goal, which means a much smaller searching and exhausting scope for the algorithm to run in. Meanwhile, the selection of the optimal design scheme is integrated into the conceptual design synthesis. Therefore, the algorithm only focuses on the generation of the optimal design scheme, which means a large

amount of workload for the generation of the rest design scheme alternatives can be avoided. Both these two features make a contribution to the high efficiency of this proposed method.

4. The design representation of this proposed approach supports the multi-input and multi-output structure for design resources. In the former works about the input-output model of design resources, the representation only supports the single-input and single-output structure. This improved design resource representation makes the design model more precise and suitable for a wide application background.
5. This proposed approach is established on an evaluating system for the design resources and design schemes. In this study, the data structure of the design resources records the measurable features of the design resources themselves. These features can not only be used for the design resource searching, but also the construction of the evaluating indexes of the design resources and the design schemes. The evaluating index of a design scheme is the sum of the evaluating indexes of its component design resources, and this mechanism is also the foundation of the proposed computerized optimal conceptual design synthesis.

This proposed computerized optimal conceptual design synthesis also has some defects need to be overcome, therefore, more efforts should be made in the future works. First, the design scheme should be developed into a reticular structure of related functional unit set to adjust the large industrial design background. Second, the extensibility of the constraint information's calculation could be promoted to adapt the approaches for various designers and design requirements. Third, the algorithm should be further improved with the strategy of collaboration in distributed calculating source environment. All these efforts will make the research more efficient, precise and practical.

CONCLUSION

This paper proposes a graph-based computerized optimal conceptual design synthesis consisting of two parts, i.e. a design resource searching engine and an optimal design scheme generating algorithm. The design resource searching engine can extract the design resources tightly related to the design goal from the numerous design resources in the distributed multi-disciplinary resource environment, based on the searching conditions set by the designers. As for the optimal design scheme generating algorithm, it is established on the graph model and theory, and it can find out the optimal design scheme from the related design resources offered by the design resource searching engine. This proposed computerized conceptual design synthesis is packed into a set of computer applications called Automatic Conceptual Design System (ACDS), and the conceptual design of a garbage power system is completed by it as an illustrative case.

ACKNOWLEDGEMENT

This research is supported by the National Natural Science Foundation of China (Grant No. 51805322, 51775332, 51805319 and 51675329), the China Postdoctoral Science Foundation (Grant No. 2018M642008), the Major Project of National Social Science Fund, China (Grant No. 17ZDA020), the Shanghai Committee of Science and Technology (Grant No. 16XD1425000), and the Special Program for Innovation Method of the Ministry of Science and Technology, China (Grant No. 2018IM020100).

REFERENCES

- Camelo, D. M., & Mulet, E. 2010.** A multi-relational and interactive model for supporting the design process in the conceptual phase. *Automation in Construction* 19(7): 964-974.
- Campbell, M. I., Cagan, J., & Kotovsky, K. 1999.** A-design: an agent-based approach to conceptual design in a dynamic environment. *Research in Engineering Design* 11(3): 172-192.
- Campbell, M. I., Cagan, J., & Kotovsky, K. 2003.** The A-Design approach to managing automated design synthesis. *Research in Engineering Design* 14(1): 12-24.
- Chakrabarti, A., & Bligh, T. P. 1994.** An approach to functional synthesis of solutions in mechanical conceptual design. Part I: Introduction and knowledge representation. *Research in Engineering Design* 6(3): 127-141.
- Chakrabarti, A., & Bligh, T. P. 1996.** An approach to functional synthesis of solutions in mechanical conceptual design. Part II: kind synthesis. *Research in Engineering Design* 8(1): 52-62.
- Chakrabarti, A., & Bligh, T. P. 1996.** An approach to functional synthesis of solutions in mechanical conceptual design. Part III: Spatial configuration. *Research in Engineering Design* 8(2): 116-124.
- Chen, B. 2018.** Conceptual design synthesis based on series-parallel functional unit structure. *Journal of Engineering Design* 29(3), 87-130.
- Chen, B., Liu, J., & Xie, Y. 2016.** An improved axiomatic design approach in distributed resource environment, Part 2: Algorithm for functional unit chain set generation. *Procedia CIRP* 53: 44-49.
- Chen, B., & Xie, Y. B. 2016.** A computer-assisted automatic conceptual design system for the distributed multi-disciplinary resource environment. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 231(6): 1094-1112.
- Chen, B., & Xie, Y. B. 2016.** A complex-number-domain-based conceptual design synthesis for multidisciplinary products. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 231(7): 1292-1307.
- Chen, B., & Xie, Y. 2016.** Functional knowledge integration of the design process. *Science China Technological Sciences* 60(2): 209-218.
- Chen, B., & Xie, Y. B. 2017.** A function unit integrating approach for the conceptual design synthesis in the distributed resource environment. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 232(5): 759-774.
- Chiou, S. J., & Kota, S. 1999.** Automated conceptual design of mechanisms. *Mechanism and Machine Theory* 34(3): 467-495.
- Christophe, F., Bernard, A., & Coatanéa, É. 2010.** RFBS: a model for knowledge representation of conceptual design. *CIRP Annals-Manufacturing Technology* 59(1): 155-158.
- Deng, Y. M. 2002.** Function and behavior representation in conceptual mechanical design. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing* 16(5): 343-36.
- Deng, Y. M., Britton, G. A., & Tor, S. B. 2000.** Constraint-based functional design verification for conceptual design. *Computer-Aided Design* 32(14): 889-899.
- Golfarelli, M., Maio, D., & Rizzi, S. 1998.** Conceptual design of data warehouses from E/R schemes. In: 31st Hawaii International Conference on System Sciences. Kona, Hawaii, USA: IEEE, Vol. 7, pp. 334-343.
- Helms, B., Shea, K., & Hoisl, F. 2009.** A framework for computational design synthesis based on graph-grammars and function-behavior-structure. In: ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. San Diego, California, USA: American Society of Mechanical Engineers, pp. 841-851.

- Hussain, R., & Lockett, H. 2012.** A framework to inform PSS conceptual design by using system-in-use data. *Computers in Industry* 63(4): 319-327.
- Joskowicz, L., & Neville, D. 1996.** A representation language for mechanical behavior. *Artificial Intelligence in Engineering* 10(2): 109-116.
- Komoto, H., & Tomiyama, T. 2012.** A framework for computer-aided conceptual design and its application to system architecting of mechatronics products. *Computer-Aided Design* 44(10): 931-946.
- Kota, S., & Chiou, S. J. 1992.** Conceptual design of mechanisms based on computational synthesis and simulation of kinematic building blocks. *Research in Engineering Design* 4(2): 75-87.
- Kurtoglu, T., Swantner, A., & Campbell, M. I. 2010.** Automating the conceptual design process: "From black box to component selection". *Artificial Intelligence for Engineering Design Analysis and Manufacturing* 24(1): 49-62.
- Li, C. L., Tan, S. T., & Chan, K. W. 1996.** A qualitative and heuristic approach to the conceptual design of mechanisms. *Engineering Applications of Artificial Intelligence* 9(1): 17-32.
- Lin, T. 2008.** An approach to function identification in automated conceptual design of mechanism systems. *Researches in Engineering Design* 19: 151-159.
- Liu, J., Chen, B., & Xie, Y. 2016.** An improved axiomatic design approach in distributed resource environment, Part 1: Toward functional requirements to design parameters transformation. *Procedia CIRP* 53: 35-43.
- O'Sullivan, B. 2002.** Interactive constraint-aided conceptual design. *Artificial Intelligence for Engineering Design Analysis and Manufacturing* 16(4): 303-328.
- Roy, U., Pramanik, N., Sudarsan, R., Sriram, R. D., & Lyons, K. W. 2001.** Function-to-form mapping: Model, representation and applications in design synthesis. *Computer-Aided Design* 33(10): 699-719.
- Schmidt, L. C., Shetty, H., & Chase, S. C. 2000.** A graph grammar approach for structure synthesis of mechanisms. *Journal of Mechanical Design* 122(4): 371-376.
- Tang, L. 2006.** Many-degree-of-freedom mechanism combination in automated conceptual design of mechanism systems. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 220(5): 727-740.
- Umeda, Y., Ishii, M., and Yoshioka, M. 1996.** Supporting conceptual design based on the function-behavior-state modeler. *Artificial Intelligence for Engineering Design Analysis and Manufacturing* 10: 275-288.
- Welch, R. V., & Dixon, J. R. 1994.** Guiding conceptual design through behavioral reasoning. *Research in Engineering Design* 6(3): 169-188.
- Wu, Z., Campbell, M. I., & Fernández, B. R. 2008.** Bond graph based automated modeling for computer-aided design of dynamic systems. *Journal of Mechanical Design* 130(4): 137-139.
- Zhang, W. Y., Tor, S. B., & Britton, G. A. 2005.** A graph and matrix representation scheme for functional design of mechanical products. *International Journal of Advanced Manufacturing Technology* 25(3-4): 221-232.