ARTIFACT 2008 | Volume II, Issue 1 | Pages 27–35

A Solution to the Back and Forth Problem in the Design Space Forming Process: A Method to Convert Time Issue to Space Issue

Toshiharu Taura,
Kobe University, Japan

This paper discusses a method for converting time issue into space issue in the design process. During the design process, we often determine something that can be evaluated only after the design process has proceeded for a while. However, in some cases, this kind of a problem (the back and forth problem) can be converted into a spatial problem. In this paper, the author approaches the design space forming process in which the function is decomposed, as an example of the back and forth problem, by extending our previous mathematical discussion and using computer simulation. The author shows that conserving the similarity between the space for the required function description and the space for the decomposed function description is a key to solve the back and forth problem. This result indicates that forming an appropriate space for the decomposed functions for searching the design solution in an efficient manner is replaced by the criterion of similarity conservation. In other words, it is possible to analyse the back and forth problem in the design process by converting it into a spatial problem.

Keywords: back-and-forth problem, spatial problem, similarity, simulation

INTRODUCTION

This paper discusses, particularly by focusing on *the back and forth problem*, a method for converting time issue into space issue in the design process. During the design process, we often determine something that can be evaluated only after the design process has proceeded for a while.

For example, let us consider the process of synthesizing two concepts. This process is regarded as the simplest and most essential process in formulating a new concept from the existing ones. Here, the term ‘‘concept’’ is used to represent the image regarding an object held in mind that existed in the past, exists now, and will exist in the future.

From an empirical viewpoint, the invention of the art knife the first snap-off blade cutter is a good example (Figure 1). The inspiration for this incredible idea came from the synthesis of two concepts chocolate segments that can be broken off and sharp edges of broken glass (Taura et al., 2005).

Although this invention is rather attractive, the problem of focusing on the chocolate remains unsolved. In other words, why is the chocolate focused on? Generally, the chocolate is unconcerned with the knife. It is extremely difficult to select the concepts to be synthesized before synthesizing them because the appropriateness of selecting the concepts can be evaluated only after they have been synthesized and the newly created concept has been judged.

We can consider the second example of the back and forth problem in the reasoning type of synthesis. Many studies adopt the view that the abductive pattern of reasoning shown below is characteristic for the reasoning in the design process, such as from function to form.

Premise $p \longrightarrow q$
Premise q
—
Conclusion p

Further, some discussions have clarified that the pattern of reasoning in the design process is modelled as shown below (Roozenburg, 2002).

Premise q
—
Conclusion $p \longrightarrow q$
Conclusion p

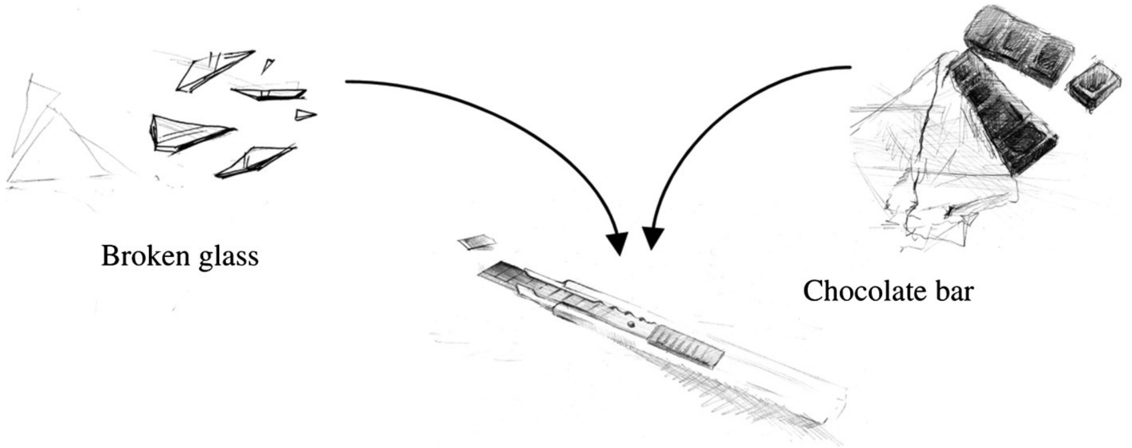

Figure 1. Design idea for an art knife by combining two concepts – glass and chocolate.

In this case, there exists only one premise, and the rule $p \longrightarrow q$ becomes part of the conclusion. In other words, at the time of reasoning (creating) a new concept (design solution), the rule must be part of the conclusion and must be inferred together with the antecedent. This discussion also suggests the existence of the back and forth problem. It is extremely difficult to select or form the rules to be used for reasoning the design solution before inferring it because the appropriateness of selecting or forming the rules can be evaluated only after they have been applied and the newly created concept has been judged.

The third example of the back and forth problem in the design stage can be recognized in the process of forming the design space (hereafter referred to as *design space forming process*). A design process is often defined as a solution-searching process in a given design space. In this case, the manner in which to select or form an appropriate design space is an essential problem. In other words, selecting or forming the appropriate design space that is to be used for describing and searching the design solution before the search process is extremely difficult. This is because the appropriateness of selecting or forming design space can be evaluated only after the new design solution has been searched for and judged.

As mentioned above, the back and forth problem from the viewpoint of time is an essential and unresolved problem in the design.

On the other hand, the back and forth problem is directly or indirectly involved in other fields such as manufacturing scheduling (for example, Giffler & Thompson, 1960) or packing problem (for example, Sweeney & Paternoster, 1992). Furthermore, the back and forth problem is regarded as one of the fundamental issues in the solution search problem that have recently been aggressively approached, such as the lazy evaluation problem (Launchbury, 1993).

AN APPROACH FOR THE BACK AND FORTH PROBLEM

In some cases, the back and forth problem can be converted into a spatial problem. For example, let us consider the path of the light beam shown in Figure 2, wherein we attempt to discover the light beam that passes from A to B by reflecting on the mirror.

If we attempt to predict the path of the beam based on the knowledge that ‘‘a light beam travels along the path which takes the shortest time’’, we are unable to evaluate whether or not the path takes the shortest time before the beam has travelled. However, if we apply the knowledge that ‘‘the angle of incidence is equal to the angle of reflection’’, then it is possible to calculate the path of the light beam before observing the travelling beam. In this case, the back and forth problem from the viewpoint of time is converted into a spatial problem.

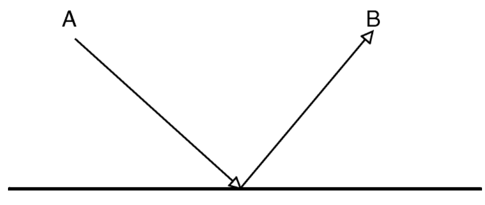

Figure 2. An example of converting time issue to space issue – a light beam travelling path.

Astudy of the General Design Theory (GDT) (Yoshikawa, 1981) provides a hint in this direction. The GDT defines design knowledge and the design process using mathematical topological space. It defines the design process as a mapping from function space, where the design requirement is described, to attribute space, where the design solution is searched. Previous studies on GDT have derived many theorems that can explain the characteristics of design knowledge and the design process. In this paper, the author approaches the design space forming process as an example of the back and forth problem, by extending our previous study based on GDT.

In the early stage of the design process, the required functions are generally decomposed into some partial functions. Although this decomposition process is not always necessary in finding design solutions, it is well known that it is useful in the design process. Not only has its importance been pointed out in an empirical study (Pahl & Beitz, 1988) but also its rationale has been analysed in a theoretical study (Taura, 1995).

Although the importance and necessity of the function decomposition process is accepted in both industry and academe, its methodology has not been thoroughly clarified. Considering that the function decomposition process is divided into (1) the process of forming a function space for describing the decomposed functions and (2) the process of establishing a decomposed function structure for searching the design solution, the former process, in particular, has not been clarified. Although Suh (Suh, 1990) pointed out that the required function should be decomposed in an independent and brief manner, the mechanism of forming the space for the decomposed function has not been analysed. In this study, the design space forming process for the decomposed functions is analysed as an example of the back and forth problem; this analysis is particularly carried out by focusing on the problem of determining the classes (defined later) that are used to describe the decomposed functions.

METHOD FOR STUDYING THE BACK AND FORTH PROBLEM IN DESIGN

In general, it is extremely difficult to study the design process because this phenomenon occurs in the human brain. We cannot observe this phenomenon directly. Thus far, the protocol analysis method has been used to observe the phenomenon. However, its accuracy is limited. Modelling design process is another method that is used to analyse the characteristics of the design process. There exist two methods for modelling the design process: mathematical modelling and computer simulation. Both these methods are effective with regard to studying the design process. In the author's previous study, the function decomposition process was determined in a precise manner, and its nature was discussed mathematically using GDT (Taura & Yoshikawa, 1992). In this paper, the author approaches the back and forth problem by extending our previous mathematical discussion and using computer simulation.

METHOD

Basic idea

In our previous study, we proposed the following hypothesis (Taura & Yoshikawa, 1992).

> **Hypothesis 1**: ''In function decomposition, the elements that are near each other in the space for the required functions are mapped onto elements that are near each other in the space for the decomposed functions in some cases.''

This hypothesis implies that between two machines, if the function structures that are described using the decomposed functions are near each other, they manifest similar functions as a whole, and the inverse is also valid. This hypothesis is explained using the example in Figure 3. First, let us define the following terms.

Def. 1 *Total function* is defined as a function that is manifested by an object (machine) as a whole.
Def. 2 *Partial function* is defined as a function that is manifested by a component of an object (machine).
Def. 3 *Partial function structure* of an object (machine) is defined as a subset of the partial functions manifested by the object (machine).

In this case, it is assumed that the total function cannot be determined from the partial function. The total function becomes clear by calculating the attributes of all the components which compose the object (machine).

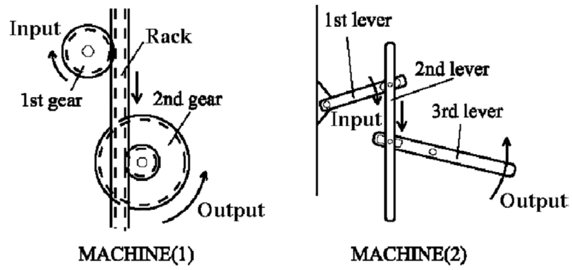

Figure 3. An example of function similarity between machines.

Let us consider the relation between a total function and a partial function structure based on the example provided in Figure 3. In this example, let the arrows in this figure be regarded as the behaviours and the function be described as the categories of the behaviours ({Rotational, Straight-Line} of {Input, Output}, Speed is {Amplified, Equal} between Input and Output). First, let us examine the total function of these machines. It is found that both these machines manifest the same total function, such as ''Input is Rotational Behaviour'', ''Output is Rotational Behaviour'' and ''The Speed of Behaviour is Amplified''. Next, let us examine the partial function structure of these two machines. It is understood that MACHINE(1) comprises some components a first gear, a rack and a second gear. The first gear has functions such as ''Input is Rotational Behaviour'', ''Output is Straight-Line Behaviour'' and ''The Speed of Behaviour is Equal''. The rack has functions such as ''Input is Straight-Line Behaviour'', ''Output is Straight-Line Behaviour'', and ''The Speed of Behaviour is Equal''. The second gear has functions such as ''Input is Straight-Line Behaviour'', ''Output is Rotational Behaviour'', and ''The Speed of Behaviour is Amplified''. In the same manner, the partial function structure of MACHINE(2) is as follows. The first lever has functions such as ''Input is Rotational Behaviour'', ''Output is Straight-Line Behaviour'', and ''The Speed of Behaviour is Equal''. The second lever has functions such as ''Input is Straight-Line Behaviour'', ''Output is Straight-Line Behaviour'', and ''The Speed of Behaviour is Equal''. The third lever has functions such as ''Input is Straight-Line Behaviour'', ''Output is Rotational Behaviour'', and ''The Speed of Behaviour is Amplified''. In this case, one will notice that both the partial function structures are the same, although they comprise different mechanical components. As a result, similarity in the partial function structures is obtained between two machines that manifest the same total function, but comprise different mechanical components.

In our previous study, we simulated the function decomposition process and showed that design solutions can be found in an efficient manner by adopting this hypothesis. However, this simulation was operated in the spaces defined in an ad hoc manner. In particular, the classes that are used to describe the partial functions (viewpoints for recognizing the partial functions) are usually defined in an ad hoc manner. In the example of Figure 3, the classes for the partial functions are determined in the same manner as those for the total functions. Therefore, the following question is raised: ''How is the most appropriate space for the partial function formed?''

In the design process, the process of forming the space for the partial functions precedes the process of establishing a partial function structure for searching the design solution. Although the space for the total function can be formed on the basis of the fact that the design specification is described, the space for the partial function cannot be fixed because we cannot find a rational criterion to evaluate the space.

One can assume that the space for the partial function is formed in order for the design solution to be found in an efficient manner. However, the following question is raised: ''How can the designer find the space for searching the design solution efficiently, before beginning to search?''

In order to discuss this problem in a scientific manner, the author extends our previous discussion. The similarity conservation between the space for the total function and the space for the partial function structures was the key to form the function space. Let us consider this idea by using the examples in Figure 3 again. If we focus on the partial functions from another viewpoint, for example, colour, we cannot find any similarity between these two machines, and it may be easily assumed that this viewpoint is not useful for finding a design solution. This discussion suggests that the similarity conservation between two spaces is strongly related to the design space forming process. In other words, Hypothesis 1 is valid only when the design solution can be found in an efficient manner. Therefore, we proposed the following hypothesis (Taura & Tobita, 2003).

> **Hypothesis 2:** ''In order to find a design solution using the space for the partial functions efficiently, the space for the partial function should be formed so that the elements that are near each other in the space for the total functions are mapped onto elements that are near each other in the space for the partial functions.''

Mapping criterion

In this study, the author defines the criterion for evaluating the degree of similarity conservation between the space for the total functions and the space for the partial functions (Figure 4). By applying this criterion, we can find a more appropriate space for the partial functions.

Methods of searching

A computer system is implemented and the above method is simulated in order to investigate the manner in which the degree of similarity conservation between the Total Function Space (TFS) and the Partial Function Space (PFS) is related to the efficiency of searching for a required machine. In this simulation, we use a method to search for a required machine, which is referred to as a ''Gradually Approaching Search'' in this study.

Definition of the function space.

In this study, the function spaces are defined as follows in set theory.

Def. 4 The TFS is defined as a space whose elements are objects (machines), and its classes (subsets of elements) are the objects' total functions.

Def. 5 The PFS is defined as a space whose elements are objects (machines), and its classes (subsets of elements) are generated by classifying the objects from the viewpoints of their components' functions.

Def. 6 The *distance between two objects* (machines) that expresses the distance between two elements in one space is defined as follows. Consider two elements in the total function space s1 and s2 that express the total function of the two objects (machines). They share A number of classes, and they do not share B number of classes. The following formula can be obtained as the distance between the two objects (machines).

$$d = \frac{B}{A+B} \tag{1}$$

In the PFS, the order of the classes (partial function) is also considered.

Def. 7 The *degree of similarity conservation* between the TFS and PFS is defined as follows. When the distances between the *standard element* and all other elements in the PFS are dp_n (n=1,2,3,...,N); and those in the TFS after mapping are dt_n (n=1,2,3,...,N); the degree of similarity conservation between the TFS and PFS (S) is calculated by using the following formula. The maximum of S is 1.0, and the minimum of S is 0.0.

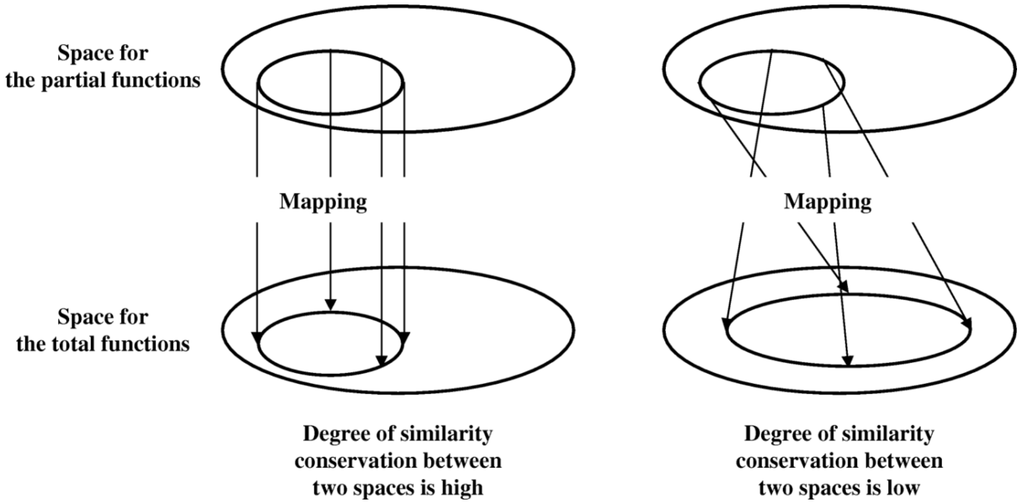

Figure 4. Difference in the degree of similarity conservation between two spaces

$$S=1.0-\frac{\sum_{n=1}^{N}|dt_n\ dp_n|}{N} \quad (2)$$

The Gradually Approaching Search.

In the Gradually Approaching Search, initially, the neighbourhood of the standard element is selected in the PFS. Following this, the components that correspond to the partial function are searched, and the total function is determined. Next, the searched elements in the PFS are mapped onto the TFS. One machine closest to the required machine is selected. If it satisfies the required total function completely, the search ends. If it does not, the selected machine becomes a new standard element and the loop is repeated until the required machine is found (Figure 5). This algorithm is shown in Figure 6. This searching models the situation in which a designer finds a required function after changing the composition of the partial function in a gradual manner.

SIMULATION USING THE GRADUALLY APPROACHING SEARCH

We conducted computer simulation as follows.

Setting up this simulation

A required machine is searched using randomly generated PFSs in order to observe the relationship between the similarity conservation of spaces

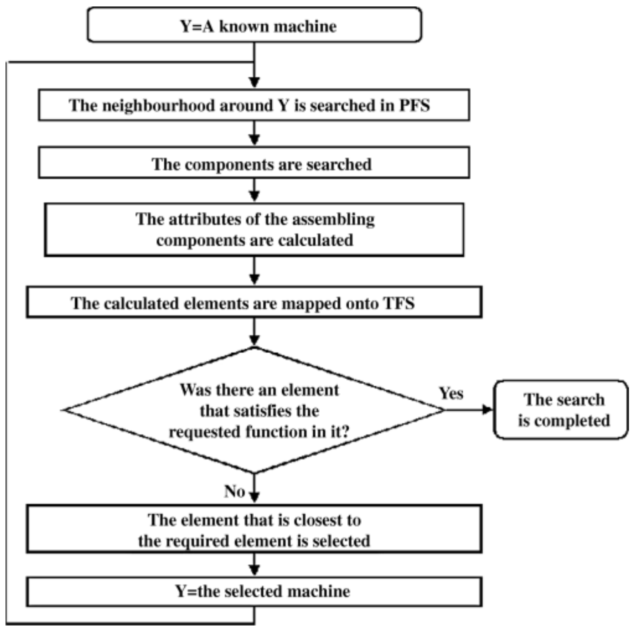

Figure 6. Algorithm of the 'Gradually Approaching Search'.

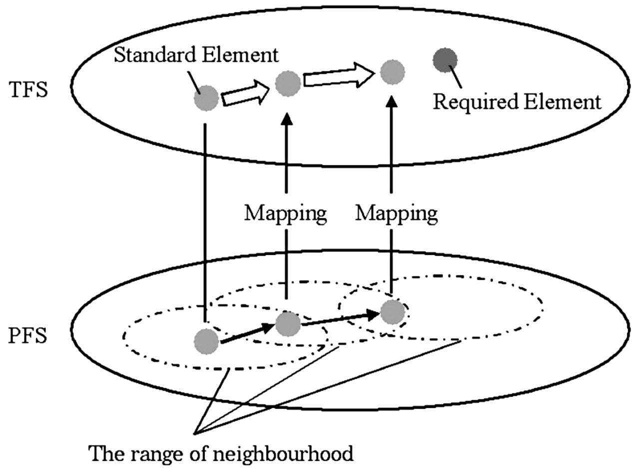

Figure 5. Mechanism of the 'Gradually Approaching Search'.

and search efficiency. In this computer system, the data pertaining to 30 components are stored, and one known machine and the total function of a required machine are inputted; then, a combination of components that fills the required total function is outputted. The setup of this simulation is as follows.

1. We prepare 30 components, the mechanisms of which are simple (gear, belt mechanism, cam, link mechanism, spring and so on).
2. All the machines comprise three components. The total number of component combinations is 718.
3. Each component has five attributes. Two attributes are behavioural ones: ''Input Behaviour'' and ''Output Behaviour''. The other three attributes are numerical ones: a change of speed between Input and Output, the direction of ''Output'' from ''Input'' on the X-axis, and the weight of the component. These three attributes have numerical values (−9∼9).
4. Six types (value of attribute) of behaviour are prepared for Input Behaviour and Output Behaviour: ''Horizontal Straight-Line Movement'', ''Vertical Straight-Line Movement'', ''Horizontal Reciprocating Movement'', ''Vertical Reciprocating Movement'', ''Rocking Movement'', and ''Rotational Movement''.
5. The three numerical attributes of a machine are calculated by summing the values of the attributes of the three components composing the machine, while the values of the two behavioural attributes of the machine are determined as the value of the attribute of Output of the third

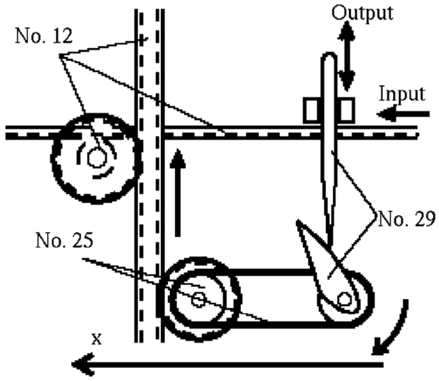

Figure 7. Structure of a standard machine.

component. Each function is represented using a function name and its value, which is assigned by dividing the numerical attributes into three groups or by describing the attributes of Output of the third component. For example, if a machine's numeral value for weight is under 7, the machine has the function of its whole weight being light. If the value is under 7 and beyond 7, the machine has the function of its whole weight being medium. If the value is beyond 7, the machine has the function of its whole weight being heavy.

6. One known machine is prepared in advance. It is the standard machine in the simulation. Its Input Behaviour is Horizontal Straight-Line Movement, and its Output Behaviour is Vertical Reciprocating Movement. The change of speed between Input and Output is ''Equal'' throughout the machine; the direction of Output from Input on the X-axis is ''Almost Zero'' and the machine's weight is ''Medium''. This machine comprises component numbers 12, 25 and 29 (Figures 7 and 8).

7. With regard to the required function, its ''Input Behaviour'' is ''Horizontal Reciprocating Movement'', and its ''Output Behaviour'' is ''Vertical Straight-Line Movement''; the change of speed is ''Decrease'' throughout the machine; the direction of Output from Input on the X-axis is ''Plus''; and the weight of the machine is ''Heavy''. The distance between this machine and the standard machine is 0.7 in the TFS.

8. The simulation is completed when the required machine is found. The search efficiency is defined as the reciprocal number of times that the loop is repeated before finding the required machine.

9. In the PFS, the functions of 30 components are classified into six classes which are determined as partial function.

10. The PFS is generated at random and for each PFS, 50 simulations are tried and their average is shown.

11. During the search, the degree of similarity conservation between the TFS and PFS can be changed since the standard machine changes. Therefore, the average degree of similarity conservation between the TFS and the PFS throughout the entire search is shown.

12. The maximum number of searching loops is 300. If a required machine is not found in 300 loops, then it is considered that the required machine cannot be found.

Result of the simulation and discussion

The relationship between the average degrees of similarity conservation of spaces and search efficiency is shown in Figure 9,which shows that the conservation of the similarity between the PFS and TFS throughout the search is related to the improvement in the search efficiency.

	Function Structure			
	Input			Output
Behaviour	**Horizontal Straight-Line Movement**	**Vertical Straight-Line Movement**	**Rotational Movement**	**Vertical Reciprocating Movement**
	No.12			
	Speed Change	**Increase**		
	Direction(X-axis)	**Plus**		
	Weight	**Heavy**		
		No.25		
		Speed Change	**Constant**	
		Direction(X-axis)	**Minus**	
		Weight	**Light**	
			No.29	
			Speed Change	**Constant**
			Direction(X-axis)	**Almost Zero**
			Weight	**Medium**

Figure 8. Function structure of a standard machine.

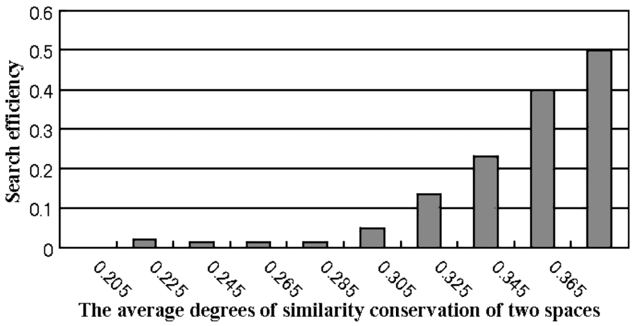

Figure 9. Relation between the average degree of similarity conservation of spaces and search efficiency.

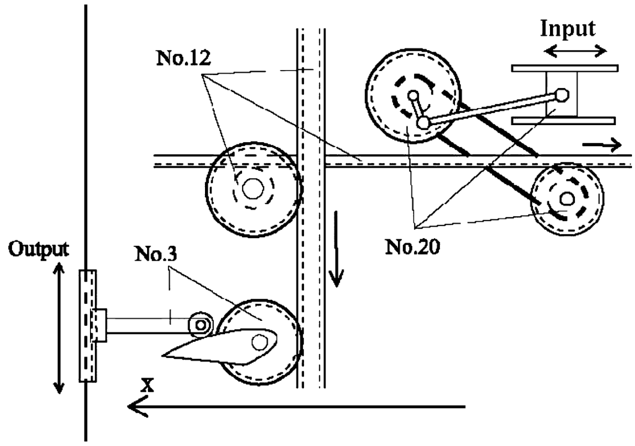

Figure 10. The searched machine after the first search.

When the average degree of similarity conservation of the spaces was the highest, the search was completed in only two loops. After the first search (loop) was finished, the machine shown in Figures 10 and 11 was created. The distance between this machine and the standard machine was 0.4 in the TFS.

On completion of the second search (loop), the machine shown in Figures 12 and 13 was created. This machine satisfied the required function.

This result indicates that forming an appropriate decomposed function space to search for the design solution in an efficient manner is replaced by the criterion of similarity conservation. In other words, it is possible to analyse the back and forth problem in the design process by converting it into a spatial problem. However, in this simulation a type of neighbourhood searching method, in which the notion of "similarity" is implied, is adopted. Therefore, this conversion is valid only when the design process involves the notion of neighbourhood searching process. However, the notion of neighbourhood or similarity is considered to be involved in nearly all the effective searching methods, such as genetic algorithms. Therefore, the method proposed in this paper may be general and applied to other fields.

CONCLUSION

It is concluded that the conservation of similarity between the space for the required function description and the space for the decomposed function description is the key to the design space forming process. This result indicates that forming an appropriate space in order to find the design solution in an efficient manner is replaced by the criterion of similarity conservation. In other words, the back and forth problem in the design process may be analysed by the spatial nature.

Function Structure

	Input			Output
Behaviour	Horizontal Reciprocating Movement	Horizontal Straight-Line Movement	Vertical Straight-Line Movement	Vertical Reciprocating Movement
	No.20			
	Speed Change	Constant		
	Direction(X-axis)	Almost Zero		
	Weight	Heavy		
		No.12		
		Speed Change	Increase	
		Direction(X-axis)	Plus	
		Weight	Heavy	
			No.3	
			Speed Change	Increase
			Direction(X-axis)	Plus
			Weight	Medium

Figure 11. Function structure of the searched machine after the first search.

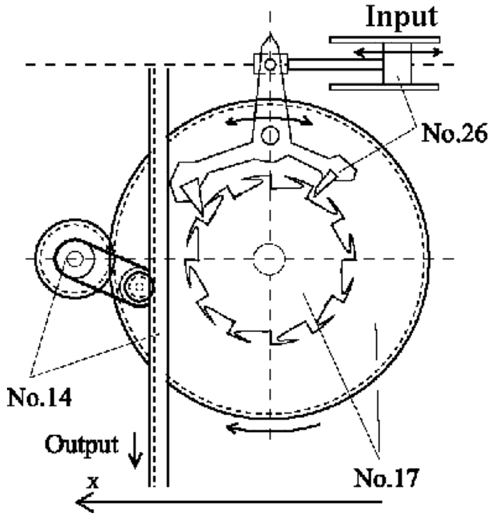

Figure 12. The searched machine after the second search.

ACKNOWLEDGEMENTS

The computer simulation in this paper was performed by Shusuke Tobita. The author is grateful for his work.

REFERENCES

Giffler, B., & Thompson, G. J. (1960). Algorithms for solving production-scheduling problems. *Operations Research, 8*(4), 487503.

Launchbury, J. (1993). A Natural Semantics for Lazy Evaluation. In *Proceedings of the 20th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (pp. 144154). Charleston, SC: ACM Press.

Pahl, G., & Beitz, W. (1988). *Engineering design: A systematic approach*. New York: Springer Verlag.

Roozenburg, F. M. N. (2002). Defining synthesis: On the sense and the logic of design synthesis. In A. Chakrabarti (Ed.), *Engineering Design Synthesis* (pp. 318). New York etc.: Springer Verlag.

Sweeney, P. E., & Paternoster, E. R. (1992). Cutting and packing problems: A categorized, application-oriented research bibliography. *Journal of the Operational Research Society, 43*(7), 691706.

Suh, N. (1990). *The Principle of Design*. Oxford and New York: Oxford University Press.

Taura, T. (1995) Design science for functional design process modeling. In *Proceedings of the 10th International Conference on Engineering Design ICED95* (pp. 456464). Prague: WDK.

Taura T., Nagai, Y., & Tanaka, S. (2005). Design space blending. In *Proceedings of ICED 2005: 14th International Conference on Engineering, CD-ROM*. Melbourne: The Design Society.

Taura, T., & Tobita S. (2003). Similarity convervation-A key to forming the function space. In A. Folkeson, K. Gralen, M. Norell and U. Sellgren (Eds.), Proceedings of ICED 2003: 14th International Conference on Engineering (CD-ROM). Stockholm: The Design Society.

Taura, T. & Yoshikawa, H. (1992). A metric space for intelligent CAD. In Intelligent Computer Aided Design. Proceedings of the IFIP WG5.2 Working Conference on Int CAD 91 (pp. 133 157). Amsterdam: North Holland.

Yoshikawa, H. (1981). General design theory and a CAD system. In T. Sata & E. Warman (eds.), ManMachine Communication in CAD/CAM, Proceedings of the IFIP WG5.2-5.3 Working Conference 1980 (Tokyo) (pp. 3557). Amsterdam: North-Holland.

Function Structure

	Input			Output
Behaviour	**Horizontal Reciprocating Movement**	**Rocking Movement**	**Rotational Movement**	**Vertical Straight-Line Movement**
	No.26			
	Speed Change	**Constant**		
	Direction(X-axis)	**Almost Zero**		
	Weight	**Medium**		
		No.17		
		Speed Change	**Constant**	
		Direction(X-axis)	**Plus**	
		Weight	**Medium**	
			No.14	
			Speed Change	**Decrease**
			Direction(X-axis)	**Almost Zero**
			Weight	**Medium**

Figure 13. Function structure of the searched machine after the second search.