Operations of Functionality

Tomohiko Sakao and Yasushi Umeda and Tetsuo Tomiyama
Department of Precision Machinery Engineering
Graduate School of Engineering, The University of Tokyo
Hongo 7-3-1, Bunkyo-ku, Tokyo 113, Japan
e-mail: {sakao,umeda,tomiyama}@zzz.pe.u-tokyo.ac.jp

Abstract

A framework to represent and operate functions is needed, because functions are crucial in various engineering activities. This paper formalizes combining multiple functions. The presented method is based on the Function-Behavior-State (FBS) modeling. In the FBS modeling, a functional world consisting of subjective descriptions and a physical world described based on Qualitative Process Theory are linked with each other. A computer tool named the FBS modeler is extended to handle combination of function operations. To represent a function, introduced are parametric conditions for a function to start or terminate, and subjects of a function. To represent relations among functions, introduced are temporal and structural/spatial relations among functions. To reason about a combination of functions, representations in the functional world are interpreted in the physical world and the behavior simulation is executed. From the results which may include additional physical phenomena, a function is extracted. An example of combining two functions is demonstrated. Importantly, the proposed method guarantees physical soundness.

Introduction

Functions[1] play a crucial role in various engineering activities, such as design and diagnosis. Therefore, a framework to represent functions is critical for supporting such engineering activities, and research on functional representation and reasoning is becoming active(Chakrabarti & Blessing 1996). However, there exists no research on operations of functions, such as combination of two functions. This paper aims at establishing a methodology to manipulate and operate functions which are described symbolically. It formalizes combination of functions as the most fundamental operation of functions.

In this paper, functions to be combined are called *input functions* and a function derived from a combination is called a *sum function*. For instance, consider the combination of to "rotate a coil" and to "generate magnetic field," each of which is an input function. The sum function may be to "generate an electric current." Here, a combination describes a sum function performed in the behavior of the embodiments of the input functions under certain relations among the input functions.

We adopt the Function-Behavior-State (FBS) modeling (Umeda *et al.* 1990) as a basis to deal with function, behavior, and state. We extended the FBS modeler (Umeda *et al.* 1996) and formalized combination of functions on it.

The rest of the paper consists of the followings: Section 2 describes the FBS modeling and a computer tool named the FBS modeler, and Section 3 describes the representations of the extended FBS modeler which allows combination of functions on it. Section 4 discusses how to reason about combination of functions, and in Section 5 an example of combining two functions is illustrated. Section 6 describes discussions including related work, and Section 7 concludes the paper.

Function-Behavior-State Modeling

Definition

The FBS modeling defines a function, a behavior, and a state as follows:

State: A state s is defined by the following triplet:

$$s = < E, A, R >,$$

where E, A, and R denote a set of *entities* included in the state, a set of *attributes* of the entities, and a set of *relations* among the entities, respectively. An *entity* is a component such as a gear, a spring, and a

[1] In this paper, a function does not mean a mathematical function, but a functionality. While the paper is entitled "Operations of Functionality" in order to avoid confusions, function is used throughout the paper.

shaft. An *attribute* of an entity is a physical, chemical, mechanical, geometrical, or other property such as "velocity," "position," and "weight." A *relation* represents a structural or spatial relation among entities such as "on," "above," and "connected."

Behavior: A behavior b is defined by temporal state transitions over time and represented by the following ordered list, where S and T denote a set of states and an ordered set of time, respectively:

$$b = (s_0, t_0), (s_1, t_1), \ldots, (s_n, t_n) \ (n \geq 0, s_i \in S, t_i \in T)$$

Function: A function is "a description of behavior recognized by a human through abstraction in order to utilize it." Therefore, a function f is represented as a tuple

$$< f_{symbol}, \ b >. \qquad (1)$$

where f_{symbol} is a symbol described in the form of "*verb objects*," and b denote a behavior that can realize this function.

Therefore, the relations between functions and behaviors are subjective and many-to-many correspondent. We call them *F-B relationships*. On the other hand, behavior of an entity can be determined by *physical phenomena* that are called *B-S relationships*. Figure 1 illustrates the relationships among functions, behaviors, and states.

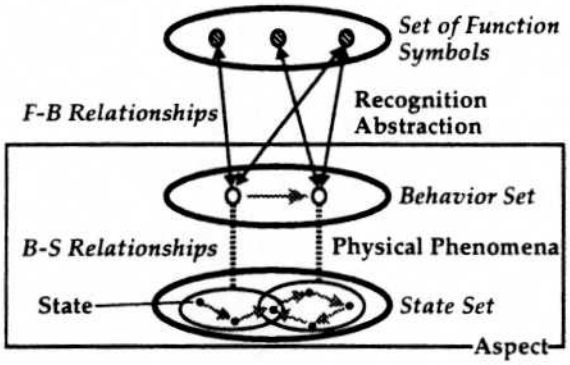

Figure 1: Relationships among Functions, Behaviors, and States (Umeda *et al.* 1990)

Function-Behavior-State Modeler

A computer tool called an FBS modeler has been developed based on the idea described in Section . Figure 2 shows the architecture of the FBS modeler.

The FBS modeler incorporates a qualitative reasoning (QR) system (Kiriyama, Tomiyama, & Yoshikawa 1990). The QR system gives a representational scheme of behaviors and states, which is based on Qualitative Process Theory (QPT) (Forbus 1984), and reasoning facilities in the level of behaviors and states. This section explains the representational scheme of function

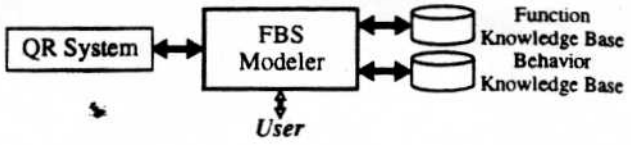

Figure 2: The Architecture of the FBS modeler

Table 1: Definition of Function Prototype (Umeda *et al.* 1990)

Item	Contents
Name	*verb + objects*
Decomposition	networks of subfunctions
F-B relationship	physical features

and behavior knowledge, and the reasoning method of the FBS modeler.

Table 1 shows the scheme of a function prototype. *Name* is a symbol for representing human's intention of a behavior in the form of "*verb objects*." Here, *objects* represent entities and stuff related to the function such as "gear" and "electric current." *Decomposition* describes feasible candidates for decomposing the function in the form of networks of subfunctions. This super-sub hierarchy is composed of either abstract-concrete relations or whole-part relations. The *F-B Relationship* describes candidates of embodiments of the function in the form of *physical features*. A physical feature is a building block consisting of entities, relations, and physical phenomena occurring on the entities (Kiriyama, Tomiyama, & Yoshikawa 1990). It is represented as a network of three kinds of elements; namely, entities, relations and *physical phenomena*. A physical feature P_f is defined by the following triplet:

$$P_f = < E, R, P >,$$

where E, R, and P denote a set of the entities, a set of relations, and a set of physical phenomena, respectively. A physical phenomenon is the same concept as a *process* in QPT and defined by Table 2. An entity also plays the same role as an *individual* in QPT. While the parameters in *q-conditions* are called *quantity parameters*, those in *s-conditions* are called *state parameters*. *Quantity parameters* and *state parameters* describe attributes of an entity. If a phenomenon is activated by satisfying its *conditions*, it adds parameters and qualitative equations defined in the *influences*.

In the FBS modeler, a model of an object consists of a *functional world* and a *physical world*. The *functional world* consists of functions with hierarchical relationships among them. On the other hand, the *physical world* consists of physical phenomena, entities, and relations with physical dependencies among them. The QR system maintains consistency in the physical world by performing behavioral simulation based on QPT.

Table 2: Definition of Physical Phenomenon (Kiriyama, Tomiyama, & Yoshikawa 1990)

Item	Contents
Name	name of the phenomenon
Supers	super classes
Conditions	
prerequisites	needed entities
references	needed causal dependencies among prerequisites
relations	needed relations
q-conditions	parametric conditions
s-conditions	parametric conditions given outside of QPT
Influences	
quantities	definition of parameters
q-relations	proportional equations
influences	differential equations

Namely, it reasons about the behavior as the state transition, which is caused by physical phenomena, according to time from an initial state.

As an example, Figure 3 shows an FBS model of a bell. The top function is to "make sound," which is decomposed into two functions, to "oscillate oscillator" and to "hit bell." The physical feature adopted for to "oscillate oscillator" consists of one physical phenomenon of "Oscillating," three entities of "Mass," "Spring," and "Wall," and two relations of "Fixed."

A physical phenomenon, "Oscillating," is defined as in Table 3. *Quantities* defines parameters and their quantity space. In the definition of a quantity space, a value starting with "@" is a landmark, and one starting with "~" is an interspace. In *q-relations*, while "direct" expresses a proportional equation, "inverse" expresses an inverse proportional one. The description following "%" denotes a relation between values of two parameters. In this case, if the parameter "`(spring length)`" is equal to "`natural`," then the value of the parameter "`(mass position)`" should be "`zero`," and vice versa.

Extension of the Representations in the Functional World

This section explains the extension of the representations of the FBS modeler needed to execute combination of functions. Here, we assume that a function prototype defined in Table 1 does not have information concrete enough to be combined with another function. Therefore, we combine *instantiated* functions, whose physical embodiment are selected from their physical features. Also, a sum function of two input functions expresses human's subjective recognition of how the object behaves when the physical embodiments of these two functions are combined. Therefore, the relations among the physical embodiments of the input functions should be given.

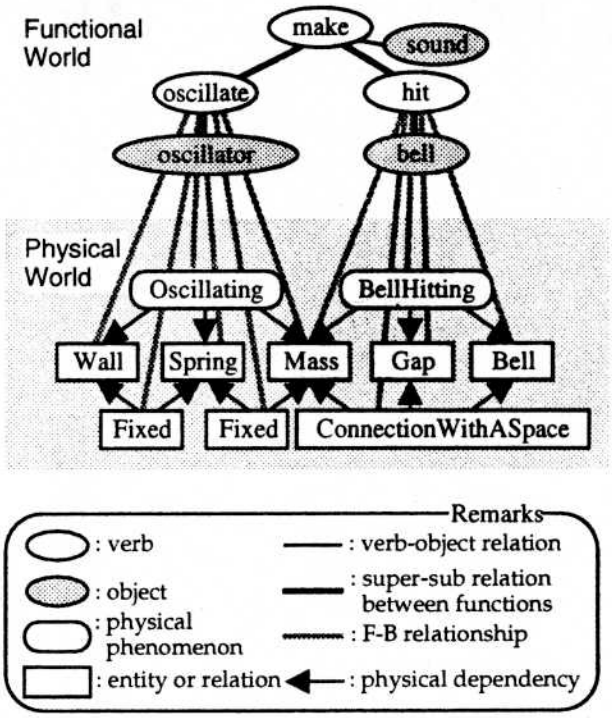

Figure 3: An FBS model of a Bell

First, we describe extensions of the representation of an instantiated function. Second, based on the extended representation, we describe representations of relations among functions.

Representation of a Function

We represent an instantiated function by the following:

$$f = < b, f_{symbol}, S, O, C_s, C_t >,$$

where:

$S \subseteq S_0 \cup \{\phi\}$. S_0 is a universal set of subject S_b.
$S_b = < S_{bsymbol}, e >$.
$O \subseteq O_0$. O_0 is a universal set of object O_b.
$O_b = < O_{bsymbol}, e >$.
C_s : *starting conditions* of the function.
C_t : *terminating conditions* of the function.

Each item is described in Sections and .

Objects and Subjects of a Function A symbolic representation of a function should be associated with the stuff related to the function. Some of them are related to the entities which are effected by the represented function, such as "water" in the function "to boil water." Others are related to the entities which exhibit the function, such as "flame" in "to boil water."

Here, we introduce *subjects* of a function to the function symbol in combination to *objects*. While an *object* is something which, a designer recognizes, is effected

Table 3: Definition of Oscillating

Name	`Oscillating`
Supers	`Dynamics`
Conditions	
prerequisites	`mass=Mass. wall=Wall. spring=Spring`
references	
relations	`fixed(mass spring). fixed(wall spring)`
q-conditions	
s-conditions	
Influences	

quantities	`(spring length)=(~less @natural ~more).`
	`(spring intnlForce)=(~minus @zero ~plus)`
	`(mass position)=(~minus @zero ~plus)`

q-relations	`(spring length)inverse(spring intnlForce)%(natural = zero)` `(spring length)direct(mass position)%(natural = zero)`
influences	

by the function, a *subject* is something which, a designer recognizes, plays a fundamental role to exhibit the function. An *subject* and a *object* are represented by a tuple of a symbol and the symbolized entity if such an entity exists in the physical world. Namely, they are represented in the following form:

$$< symbol,\ e >,$$

where e is the entity that is described as a *symbol* if such an entity exists.

No function can be represented without its object. However, some functions cannot have their subjects, because their embodiment is not constructed yet or not focused on. Thus, discriminating a subject from an object helps classifying functions with the same verb in terms of the decomposition level of the function. Also, it depends on how a human recognizes a physical process which entities are selected and abstracted to describe a subject. For instance, consider a function of transporting water which is performed by a physical process that water flows in the system of a pump and a pipe connected to each other. The symbol of this function's subject may be *pipe*, *pump*, or *pump-pipe-system*.

Parametric Conditions of a Function It is often required that a function has some conditions which start or terminate the performance of the function. For instance, to exhibit a function to "boil water," first one has to pour water in a can to its *full point* level and heat the can. Then, after the water is heated to its *boiling point* temperature, he/she has to turn off the heater.

Therefore, introduced to the representation of a function are parametric conditions which trigger the function to be started or terminated.

Elements of C_s and C_t are represented as $< equation, f_{symbol}, \texttt{started} >$ and $< equation, f_{symbol}, \texttt{terminated} >$, respectively. An *equation* is in the form of either an equality or an inequality of a parameter of an entity and a landmark.

Representations of the Relations among Functions

This section describes the relations among the input functions which are given before combination is executed. Here, we represent the relations among input functions $f_1, \cdots, f_n$ $(n \geq 2)$ as follows:

$$R\ (\ f_1, \cdots, f_n\ )\ =\ R_t\ \cup\ R_{ss},$$

where R_t and R_{ss} are a set of temporal relations described in the functional world and a set of structural/spatial relations described in the physical world, respectively.

Temporal Relations among Functions It is obvious that a sum function depends on the temporal relations of performing multiple input functions. For instance, a sum function from performing to "heat a can" before performing to "pour water" is different from one from performing "to pour water" before performing to "heat a can." Thus, it is indispensable to represent temporal relations.

We introduce the 13 relations[2] of Allen's temporal logic (Allen 1983) to describe the temporal relationships between two functions. A temporal relation between two functions, f_1 and f_2, is represented as follows:

$$relation\ (f_1,\ f_2),$$

where one of the 13 relations is substituted to *relation*.

Since the physical meaning for each relation is clearly defined, they can be interpreted in the physical world. Note that a function is supposed to exhibit for a *time interval* in Allen's temporal logic.

[2]They consist of *meets*, *before*, *finishes*, *starts*, *overlaps*, *during*, the inverse relation for each of them, and *equal*.

Structural/Spatial Relations among Functions Obviously, it is necessary to represent structural or spatial relations among subjects and objects to specify the relations among the input functions. For instance, to carry out the combination of to "rotate a coil" and to "generate magnetic field," the spatial relation that the coil's rotational axis is perpendicular to the direction of the magnetic field should be specified.

We represent a set of this kind of relations as R_{ss} using Relation among Entities belonging to the physical embodiments of the input functions. Each element is represented as follows, if it links $Entity_1$ and $Entity_2$:

$$Relation\ (Entity_1, Entity_2).$$

Combining Functions on the FBS Modeler

This section explains how a sum function is recognized as a result of combining multiple input functions. To do so, first the representations in the functional world must be interpreted. Second, the behavior simulation should be executed and finally a function should be extracted from the result.

Interpretation of the Representations in the Functional World

Behavior simulation is executed in the physical world and it must incorporate parametric conditions and temporal relations. Since they are represented in the functional world, they must be interpreted in the physical world before the simulation.

The temporal relations among functions and the parametric conditions of functions are integrated into a temporal order of activated physical phenomena and parametric conditions. Note that a set of all the parametric conditions given to the input functions is represented as C, each element of which is represented in the same way shown in Section . The algorithm to generate a temporal order is as follows:

1. A function required to exhibit for a time interval is replaced by all the physical phenomena included in the function's physical feature. Here, it is assumed that a physical feature represents that all of its physical phenomena occur simultaneously.
2. All the time intervals on one time axis are ordered using the definition of each vocabulary in Allen's temporal logic.

As a result, a temporal order is represented by an ordered set:

$$\{C_0,\ P_1,\ C_1,\ \cdots,\ P_n,\ C_n\}. \qquad (2)$$

where P_i and C_i are a set of physical phenomena and a set of parametric conditions, respectively.

Figure 4 shows an example in which the following conditions are given to f_1 and f_2. Note that in Figure 4, Pf_i and pp_i mean a physical feature and physical phenomenon, respectively.

- $R_t = \{meets(f_1, f_2)\}$
- $C = \{ < \mathtt{a} = \mathtt{a_1},\ f_{1_{symbol}},\ \mathtt{started} >, < \mathtt{b} = \mathtt{b_1},\ f_{2_{symbol}},\ \mathtt{terminated} > \}$

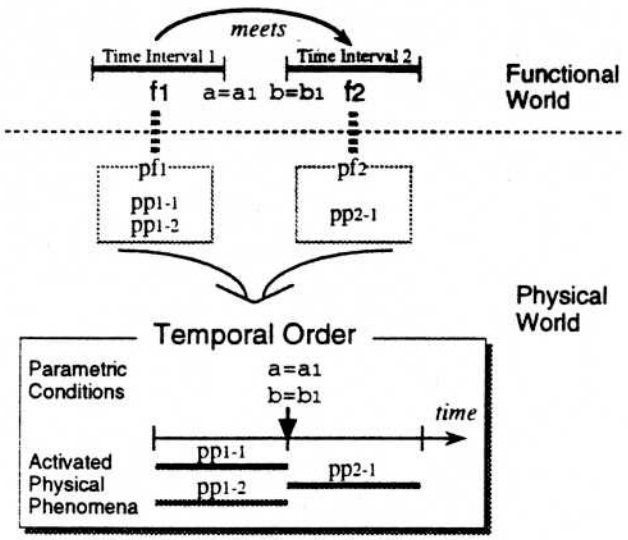

Figure 4: Generation of a Temporal Order

Behavior Simulation

In this simulation, the values of state parameters may be changed to satisfy the temporal order given in (2), although only the values of quantity parameters are changed through the physical causalities in the envisioning of QPT (see Table 2 for the information on quantity parameters and state parameters). The system reasons about all the possible physical phenomena, and the physical phenomena which do not occur to the physical embodiment of each input function but occur to the combined physical embodiments are also incorporated in the simulation.

As a result, the simulation generates an ordered set of s'_i, where $i \in \{1, \cdots, n\}$. The steps in the simulation after s'_i is determined are explained below (see also Figure 5). Note that we also define s' by the following:

$$s' = < E,\ A,\ R,\ P >,$$

where P is a set of the physical phenomena which occur in $s = < E,\ A,\ R >$.

Before executing the simulation, a value of each parameter in the physical world at the initial state is given, which determines s_1 of the object.

1. *Giving actions*: Suppose the system currently reasons about the object to which P_j in (2) are supposed to occur. If all the parametric conditions in C_j are satisfied, the values of the state parameters are changed so that each physical phenomenon included in P_{j+1} and not included in P_j occurs, and each physical phenomenon included in P_j and not

included in P_{j+1} does not occur. This generates s_{i+1}.

2. *Finding the occurring physical phenomena*: The QR system reasons about and finds all the physical phenomena which actually occur in s_{i+1}. This determines s'_{i+1}.
3. *Checking a conflict*: If the system currently reasons about the step at which P_{j+1} are supposed to occur, it compares P_{j+1} with s'_{i+1}.
   If $P_{j+1} \subseteq s'_{i+1}$ is not satisfied, it is a conflict.
4. *Limit analysis in QPT*: The limit analysis generates $\{s'_{i+1}, \cdots, s'_{i+l}\}$ by changing the parameters' values based on their differential values. Return to 1 with each s'.

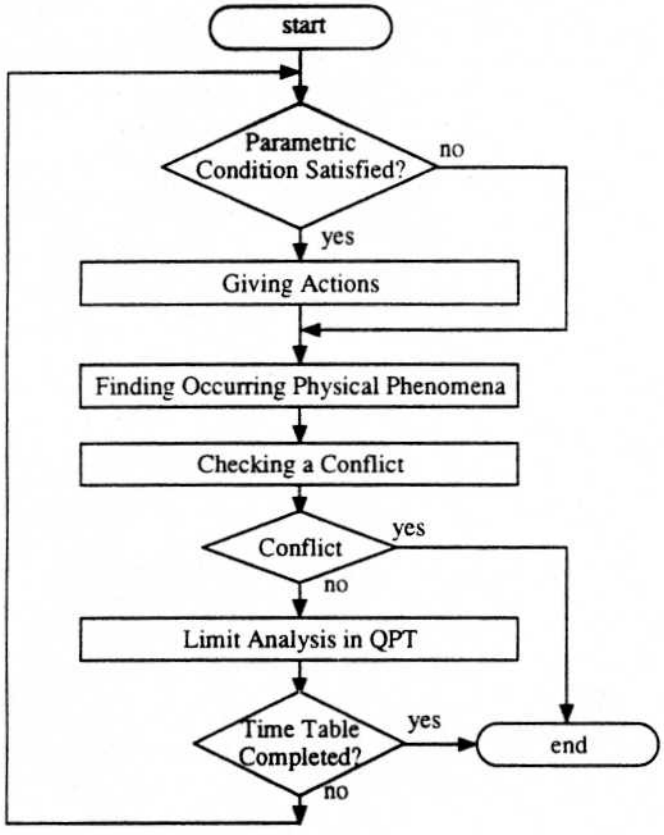

Figure 5: The Algorithm of the Simulation

Extraction of a Function

Finally, a sum function is recognized in the results of the behavior simulation. Each physical feature, $P_{f_k} = < E_k, R_k, P_k >$ in the Function Knowledge Base, is used to examine whether or not function f_k, of which P_{f_k} is a physical feature, is performed in the results of the simulation. Namely, if the following condition is satisfied, f_k is recognized to exhibit.

$$\exists i \ : \ included\ (P_{f_k},\ s'_i),$$

where *included* $(P_{f_k},\ s'_i)$ means that the network of P_{f_k} is found in s'_i. Note that an entity, a relation, and a physical phenomenon, found in s'_i can be an instance of the sub-classes of the matched entity in the physical feature, one of those of the relation, and one of those of the physical phenomenon, respectively.

If f_k is extracted, f_k is recognized to be a sum function. A sum function f_s is described by the following formula, if input functions are f_1 and f_2.

$$f_s \ = \ sum(\ f_1,\ f_2,\ R(f_1,\ f_2),\ C\ ) \qquad (3)$$

Suppose f_1 and f_2 are to "rotate a coil" and to "generate magnetic field." To "generate an electric current" can be f_s, if the followings are given:

$$R(f_1,\ f_2) \ = \ \{\ equal(\text{"rotate a coil"},\ \text{"generate magnetic field"})\} \cup \{\texttt{Perpendicular(Coil, MagneticField} \qquad (4)$$

$$C \ = \ \{\phi\} \qquad (5)$$

Example

This section demonstrates an example to combine functions on the extended FBS modeler based on the method described in Section 4.

Object Descriptions

This section describes the object to be used in Section 5 and the related knowledge in the Function and Behavior Knowledge Base. The example is a model of the process of transferring an image in a photocopier. In the photocopier shown in Figure 6, the halogen lamp lights the original paper to get an image on the drum.

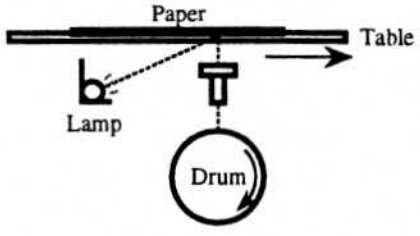

Figure 6: Image Transferring in a Photocopier

Table 4 lists the knowledge about three functions found in Figure 6. Note that possible relationships among the three functions are not recognized. The three physical features in Table 4 are defined as in Figures 7, 8, and 9. Figure 7 shows that "ReflectionWithMovement" is composed of two physical phenomena, "Moving" and "Reflecting," three entities, "Motor," "Paper," and "Lamp," and two relations, "On" and "Facing." "Moving" occurs to "Motor" and "Paper" which are linked by "On." "Reflecting" occurs to "Paper" and "Lamp" linked by "Facing."

The entities for the subjects and objects whose symbols are "lamp," "paper," "drum_motor," and "drum," in Table 4, are "Lamp," "Paper," "Motor," and "Drum" in Figures 7, 8 and 9, respectively.

Table 4: Knowledge of Functions

Verb	Objects	Subjects	F-B relationship
create	optical_image	lamp, paper	ReflectionWith-Movement
rotate	drum	drum_motor	DrumRotation
transfer	image, drum	lamp	ImageTransfer-ingToDrum

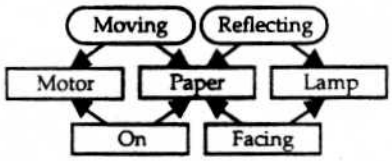

Figure 7: ReflectionWithMovement

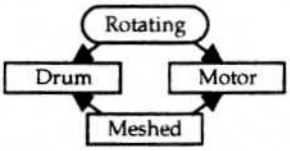

Figure 8: DrumRotation

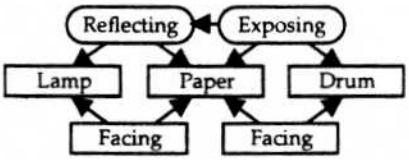

Figure 9: ImageTransferingToDrum

In the rest of Section 5, let us illustrate reasoning about the combination of the two functions, to "create optical image" and to "rotate drum."

Representations of the Relations among the Functions

Suppose the following conditions are given:

$$C = \{ < \texttt{(Paper position)} = \texttt{start}, \text{"create optical image"}, \texttt{started} >, < \texttt{(Paper position)} = \texttt{end}, \text{"create optical image"}, \texttt{terminated} > \} \quad (6)$$

$$R_t = \{ equal(\text{"create optical image"}, \text{"rotate drum"}) \} \quad (7)$$

$$R_{ss} = \{ \texttt{Facing(Paper,Drum)} \} \quad (8)$$

Figure 10 shows the input functions with their physical embodiments on the FBS modeler.

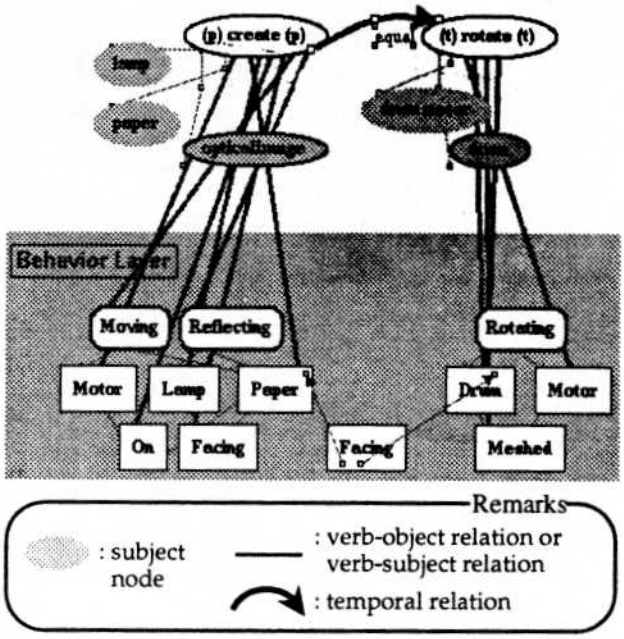

Figure 10: Relations between Functions

A sign "(p)" on a verb node is used to show the function has parametric conditions. "(p)" located in the verb's left (right) expresses the function has the parametric conditions to be started (terminated).

Interpretation of the Representations in the Functional World

The generated temporal order is an ordered set $\{ C_0, P_1, C_1 \}$, where:

- $C_0 = \{ < \texttt{(Paper position)} = \texttt{start}, \text{"create optical image"}, \texttt{started} > \}$
- $P_1 = \{$ Reflecting, Moving, Rotating $\}$
- $C_1 = \{ < \texttt{(Paper position)} = \texttt{end}, \text{"create optical image"}, \texttt{terminated} > \}$

Behavior Simulation and Extraction of a Function

Before the simulation, the value of each parameter in the behavior layer at the initial state is given. Figure 11 is a hard copy showing the results of the behavior simulation.

For our example, three functions, f_1, f_2, and f_3, are extracted from the state transition shown in Figure 11. The symbols of f_1, f_2, and f_3 are listed below.

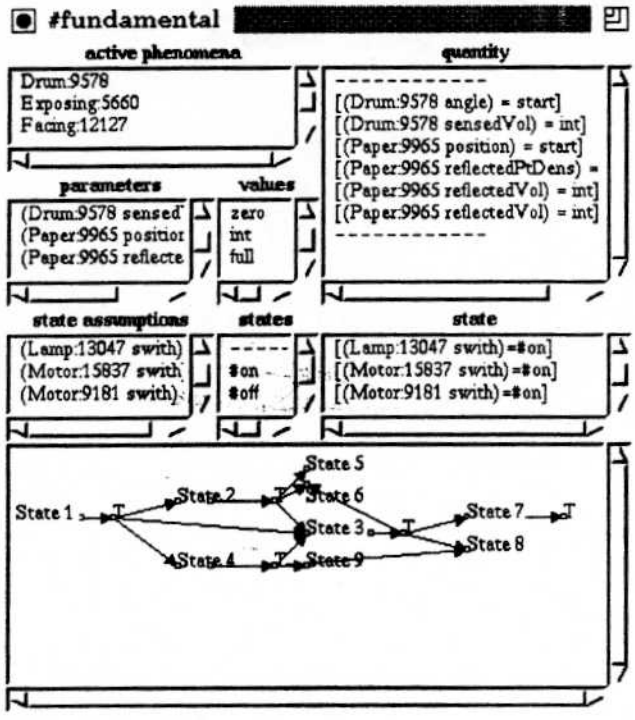

Figure 11: The Results of the Behavior Simulation of the System

- $f_{1\,symbol}$ = "create optical image"
- $f_{2\,symbol}$ = "rotate drum"
- $f_{3\,symbol}$ = "transfer image to drum"

This results in the following formula:

$$\forall i \in \{1.2.3\} : f_i = sum(\ f_1.\ f_2.\ R(f_1.f_2).\ C\ ).$$

where $< R(f_1.f_2).C >$ is given by (6), (7) and (8).

Discussions and Related Work

We proposed a method for combining functions by extending the FBS modeler. The formula (3) represents the combination. It should be emphasized that the relationship among the functions represented by (3) is grounded on physical reasoning. However, the method has a limitation in the process of function extraction. We used a physical feature to extract a function from the temporally ordered set of states. This makes it impossible to extract a function which is too abstract to have a physical feature as its F-B Relationship and has only a network of subfunctions as its Decomposition. Extracting this kind of abstract functions is our future work.

The formula (3) represents a relationship among functions. Thus, the combination can contribute for consistently constructing vocabulary of function. Using such vocabulary of function, ***subtraction*** can also be executed. ***Subtraction*** is deriving a sub-function with $< R, C >$, if the other sub-function is given in order to perform a function. Executing subtraction is powerful for supporting a synthetic stage of functional design, namely, functional development. On the other hand, if a functional hierarchy in functional development constructed by a designer is available, the combination is useful for verifying its physical soundness (see Table 5).

Table 5: Operations of Functions

f_s	f_1	f_2	$< R, C >$	operation
?	o	o	o	*combination*
o	o	?	?	*subtraction*
o	?	?	?	*functional development*

Note that "?" and "o" mean unknown and determined, respectively.

To construct such a vocabulary of function, it is also required to limit the number of functions. Having guidelines on symbolizing them may contribute to do so. However, the present FBS modeling does not have such guidelines. Keuneke's classification (Keuneke 1991) can approach this kind of problem. She classifies functions into four types; namely, *ToMake*, *ToMaintain*, *ToPrevent*, and *ToControl*. We consider those are not enough to represent all kinds of functions.

The process described in Section 4 is complicated for the following reasons. First, a function is associated with a designer's recognition based on their viewpoint. Therefore, a function should be extracted from the behavior of the entire object. Second, the relations given among the functions may cause the combinational physical phenomena which do not happen without the combination. Therefore, the behavior of the entire object should be derived from the result of the behavior simulation. These two points also make the result of the combination more than just the combination of the physical embodiment of each function.

The relations given among functions include actions and spatial or structural relations which a designer wants to be realized in the physical world. They can be referred to as *intentions*, which are indispensable to represent an object especially in design. Intentions are represented also in Causal Functional Representation Language (CFRL) (Iwasaki *et al.* 1993). CFRL represents behavioral functions based on the causality among state transitions. Although their approach is similar to ours, they do not represent functions as symbols.

Bracewell *et al.* (Bracewell & Sharpe 1996) has proposed a conceptual design tool, where operations of functions are formalized to some extent. However, it is limited to the decomposition of primary functions in Bond graphs, such as decomposing "Source of Effort" to "Source of Flow" and "Conflict Resolver." On the other hand, our method does not have a limitation to

the functions to be operated. No research deals with an operation of general functions like ours.

Conclusion

This paper formalized combination of multiple functions. We adopted the FBS modeling as a basic idea to deal with functions, behaviors, and states. We extended the FBS modeler to reason about a combination of functions, mainly on the following four points: (1) The representation of parametric conditions for a function to be started or terminated, and subjects of a function. (2) The representation of relations among functions. (3) The behavior simulation incorporating the representations in the functional world by interpreting them in the physical world. (4) The extraction of a function from the results of the behavior simulation. The method to combine functions on the extended FBS modeler is explained with an example. Two points should be emphasized: (1) our operation guarantees the physical soundness, while functions are represented symbolically. (2) our method does not have a limitation to the functions to be operated. Also, a combination can be transformed to a subtraction, which is found in functional development.

Future work includes the following:

- Applying the formalized operations to the process of conceptual mechanical design, especially that of functional development.
- Collecting vocabulary of function in a real engineering field on the extended FBS modeler.
- Developing a method to extract a more abstract function from the results of behavior simulation.

References

Allen, J. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM* 26(11):832–843.

Bracewell, R. H., and Sharpe, J. E. E. 1996. Functional descriptions used in computer support for qualitative scheme generation - 'Shemebuilder'. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 10(4):333 - 345.

Chakrabarti, A., and Blessing, L., eds. 1996. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing. Special Issue: Representing Functionality in Design.* Cambridge University Press.

Forbus, K. 1984. Qualitative process theory. *Artificial Intelligence* 24(3):85–168.

Iwasaki, Y.; Fikes, R.; Vescovi, M.; and Chandrasekaran, B. 1993. How things are intended to work: Capturing functional knowledge in device design. In *Proceedings of IJCAI-93*, 1516–1522.

Keuneke, A. M. 1991. Device representation the significance of functional knowledge. *IEEE Expert* 6(2):22–25.

Kiriyama, T.; Tomiyama, T.; and Yoshikawa, H. 1990. Qualitative reasoning and conceptual design with physical features. In *Proceedings of the 4th International Workshop on Qualitative Physics*, 153–160.

Umeda, Y.; Takeda, H.; Tomiyama, T.; and Yoshikawa, H. 1990. Function, behaviour, and structure. In *AIENG'90 Applications of AI in Engineering*, 177–193. Computational Mechanics Publications and Springer-Verlag.

Umeda, Y.; Ishii, M.; Yoshioka, M.; Shimomura, Y.; and Tomiyama, T. 1996. Supporting conceptual design based on the Function-Behavior-State modeler. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 10(4):275 - 288.