

On System Algebra: A Denotational Mathematical Structure for Abstract System Modeling

Yingxu Wang, University of Calgary, Canada

ABSTRACT

Systems are the most complicated entities and phenomena in abstract, physical, information, and social worlds across all science and engineering disciplines. System algebra is an abstract mathematical structure for the formal treatment of abstract and general systems as well as their algebraic relations, operations, and associative rules for composing and manipulating complex systems. This article presents a mathematical theory of system algebra and its applications in cognitive informatics, system engineering, software engineering, and cognitive informatics. A rigorous treatment of abstract systems is described, and the algebraic relations and compositional operations of abstract systems are analyzed. System algebra provides a denotational mathematical means that can be used to model, specify, and manipulate generic "to be" and "to have" type problems, particularly system architectures and high-level system designs, in computing, software engineering, system engineering, and cognitive informatics.

Keywords: abstract systems; cognitive informatics; denotational mathematics; engineering applications; mathematical models; software engineering; system algebra; system theory

INTRODUCTION

Systems are the most complicated entities and phenomena in abstract, physical, information, and social worlds across all science and engineering disciplines. Systems are needed because the physical and/or cognitive power of an individual component or a person is not enough to carry out a work or solve a problem. System philosophy intends to treat everything as a system, and it perceives that a system always belongs to other super system(s) and contains more subsystems.

The system concept can be traced back to the 17th century, when René Descartes (1596-

1650) noticed the interrelationships among scientific disciplines as a system. The general system notion was then proposed by Ludwig von Bertalanffy in the 1920s (von Bertalanffy, 1952; Ellis & Fred, 1962). The theories of system science have evolved from classic theories (Ashby, 1958, 1962; Ellis & Fred, 1962; Heylighen, 1989; G. J. Klir, 1992; R. G. Klir, 1988; Rapoport, 1962) to contemporary theories in the mid-20th century, such as I. Prigogine's dissipative structure theory (Prigogine et al., 1972), H. Haken's synergetics (Haken, 1977), and Eigen's hypercycle theory (Eigen & Schuster, 1979). Then, during the late part of the last century,

there are proposals of complex systems theories (G. J. Klir, 1992; Zadeh, 1973), fuzzy theories (Zadeh, 1965, 1973), and chaos theories (Ford, 1986; Skarda & Freeman, 1987).

System algebra is an abstract mathematical structure for the formal treatment of abstract and general systems as well as their algebraic relations, operations, and associative rules for composing and manipulating complex systems. System algebra (Wang, 2005, 2006a, 2006b, 2007a, 2007c) presented in this article is the latest attempt to provide a formal and rigorous treatment of abstract systems and their properties. This article treats systems as a mathematic entity and it studies the generic rules and theories of abstract systems. A new mathematical structure of abstract systems as the most complicated mathematical entities beyond sets, functions, and processes is presented. Properties of abstract systems are modeled and analyzed. System algebra is introduced as a set of relational and compositional operations for manipulating abstract systems and their composing rules. The relational operations of system algebra are described encompassing *independent, related, overlapped, equivalent, subsystem, and supersystem*. The compositional operations of system algebra are explored encompassing *inheritance, tailoring, extension, substitute, difference, composition, decomposition, aggregation, and specification*. A wide range of applications of system algebra are identified in cognitive informatics, system science, system engineering, computing, software engineering, and intelligent systems.

THE ABSTRACT SYSTEM THEORY

This section demonstrates that systems may be treated rigorously as a new mathematical structure beyond conventional mathematical entities. Based on this view, the concept of abstract systems and their mathematical models are introduced.

Definition 1. *An abstract system is a collection of coherent and interactive entities that has*

stable functions and a clear boundary with the external environment.

An abstract system forms the generic model of various real-world systems and represents the most common characteristics and properties of them.

Lemma 1. *The generality principle of system abstraction states that a system can be represented as a whole in a given level k of reasoning, $1 \leq k \leq n$, without knowing the details at levels below k .*

Definition 2. *Let \mathcal{C} be a finite or infinite nonempty set of components, and \mathcal{B} a finite or infinite nonempty set of behaviors, then the universal system environment U is denoted as a triple, i.e.:*

$$\begin{aligned} U &\triangleq (\mathcal{C}, \mathcal{B}, \mathcal{R}) \\ &= \mathcal{R} : \mathcal{C} \rightarrow \mathcal{C} \mid \mathcal{C} \rightarrow \mathcal{B} \mid \mathcal{B} \rightarrow \mathcal{C} \mid \mathcal{B} \rightarrow \mathcal{B} \end{aligned} \quad (1)$$

where \mathcal{R} is a set of relations between \mathcal{C} and \mathcal{B} , and \mid demotes alternative relations.

Abstract systems can be classified into two categories known as the *closed* and *open* systems. Most practical and useful systems in nature are open systems in which there are interactions between the system and its environment. However, in order to develop the theoretical framework of abstract systems, the closed systems in which there is no interaction with the external environment will be modeled first in the following subsection.

The Mathematical Model of Closed Systems

The axiom of the abstract system theory is based on the Object-Attribute-Relation (OAR) model (Wang, 2007b, 2007d; Wang & Wang, 2006c), in which the architecture of a system object O_s can be modeled by a set of attributes A and a set of binary relations R among A and O_s , i.e.:

$$O_s \triangleq (A, R) \quad (2)$$

Encompassing both architectures and behaviors of a system on the basis of Equation 2, an abstract closed system without interactions with the environment can be formally described as follows.

Definition 3. A closed system \hat{S} on \mathcal{U} is a 4-tuple, i.e.:

$$\hat{S} \triangleq (C, R, B, \Omega) \tag{3}$$

where

- C is a nonempty set of components of the system, $C = \{c_1, c_2, \dots, c_n\} \subseteq \mathcal{P}\mathcal{C} \sqsubseteq \mathcal{U}$.
- R is a nonempty set of relations between pairs of the components in the system, $R = \{r_1, r_2, \dots, r_m\} \subseteq C \times C$.
- B is a set of behaviors (or functions), $B = \{b_1, b_2, \dots, b_p\} \subseteq \mathcal{P}\mathcal{B} \sqsubseteq \mathcal{U}$.
- Ω is a set of constraints on the memberships of components, the conditions of relations, and the scopes of behaviors, $\Omega = \{\omega_1, \omega_2, \dots, \omega_q\}$.

According to Definition 3, a closed system $\hat{S} = (C, R, B, \Omega)$ on \mathcal{U} can be illustrated in Figure 1.

It is noteworthy that system behaviors B is the most broad set of system actions implemented or embodied on the given layout of the systems, including any kind of system functions, interactions, and communications. This is the

major difference that distinguishes an abstract system from other mathematical structures such as a set, lattice, group, or concept (Wang, 2008b).

Lemma 2. A closed system $\hat{S} = (C, R, B, \Omega)$ is an asymmetric (directed) and reflective system because the relations R in it are constrained by the following rules:

a. *Asymmetric:*
 $\forall a, b \in C \wedge a \neq b \wedge r \in R, r(a, b) \neq r(b, a)$ (4)

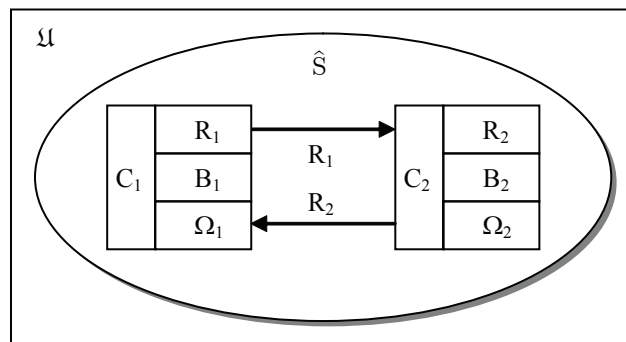
b. *Reflective:*
 $\forall c \in C, r(c, c) \in R$ (5)

Corollary 1. The maximum number of binary relations n_r between all pairs of the n_c components in a closed system $\hat{S} = (C, R, B, \Omega)$ can be determined as follows:

$$n_r = \#R = \#(C \times C) = n_c^2 \tag{6}$$

if all reflective self-relations are not considered, n_r becomes the maximum number of binary relations of fully connected systems, n' , i.e.:

Figure 1. The abstract model of a closed system



$$n'_r = n'_r - n_c = n_c(n_c - 1) \quad (7)$$

Example 1. According to Corollary 1, the creation of relations in a closed system is solely determined by the number of components possessed in the system. This property can be illustrated in Figure 2, where \uplus denotes a system composition for both closed or open systems, which will be formally explained later. The relations of the three closed systems \widehat{S}_1 , \widehat{S}_2 , and \widehat{S} are as follows, observing that all pairwise relations are asymmetric or different:

$$n'_{r_1}(\widehat{S}_1) = n_{c_1}^2 = 3^2 = 9; \quad n'_{r_1}(\widehat{S}_2) = 2^2 = 4;$$

$$n'_r(\widehat{S}) = 5^2 = 25$$

and

$$n'_{r_1}(\widehat{S}_1) = n_{r_1} - n_{c_1} = 9 - 3 = 6;$$

$$n'_{r_2}(\widehat{S}_2) = 4 - 2 = 2; \quad n'_r(\widehat{S}) = 25 - 5 = 20$$

The Mathematical Model of Open Systems

Most practical systems in the real world are not closed. That is, they need to interact with external world known as the *environment* Θ in order to exchange energy, matter, and/or information. Such systems are called “open systems.” Typical interactions between an open system and the environment are inputs and outputs.

Contrary to that the relations of a closed system are defined on the Cartesian product of internal components, the set of relations R of an open system needs to be extended to include both internal relations R^c and external (input/output) relations R^i and R^o , i.e.:

$$R = \{R^c \cup R^i \cup R^o\} \quad (8)$$

Definition 4. An open system S on U is a 7-tuple, i.e.:

$$S \triangleq (C, R, B, \Omega, \Theta), R = \{R^c, R^i, R^o\}$$

$$= (C, R^c, R^i, R^o, B, \Omega, \Theta) \quad (9)$$

where the extensions of entities beyond the closed system as given in Definition 3 are as follows:

- Θ is the environment of S with a nonempty set of components C_Θ outside C , i.e., $\Theta = C_\Theta \sqsubseteq \mathcal{U}$.
- $R^c = C \times C$ is a set of internal relations.
- $R^i \subseteq C_\Theta \times C$ is a set of external input relations.
- $R^o \subseteq C \times C_\Theta$ is a set of external output relations.

An open system $S = (C, R^c, R^i, R^o, B, \Omega, \Theta)$ can be illustrated in Figure 3.

Lemma 3. An open system $S(C, R^c, R^i, R^o, B, \Omega, \Theta)$ on \mathcal{U} is an asymmetric and reflective

Figure 2. Creation of relations in open and closed systems

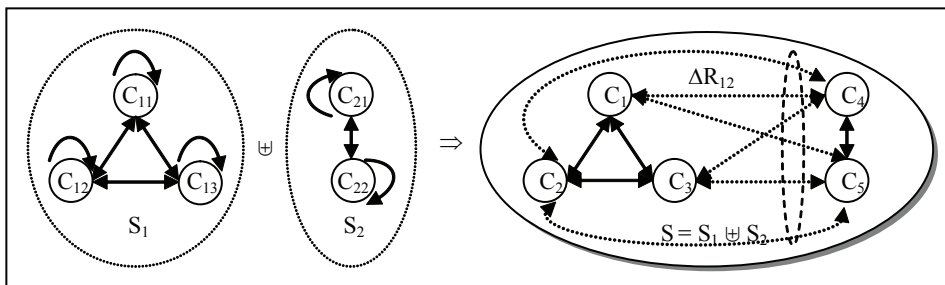
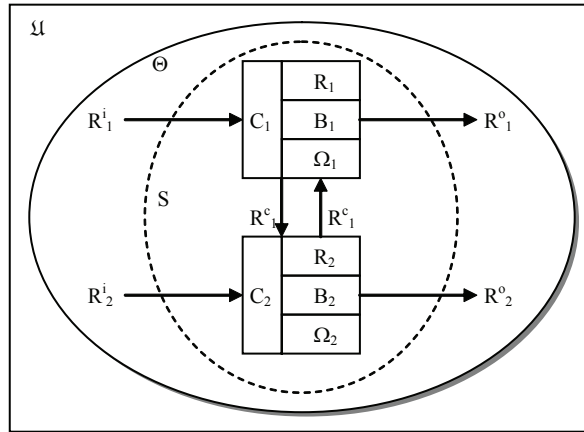


Figure 3. The abstract model of an open system



system because its relations R^c , R^i , and R^o are constrained by the following rules:

a. Internally asymmetric:

$$\forall a, b \in C \wedge a \neq b \wedge r \in R^c, r(a, b) \Rightarrow r(b, a) \tag{10}$$

b. Externally asymmetric:

$$\forall a \in C \wedge \forall x \in C_\Theta \wedge r \in R^i \vee r \in R^o, r(x, a) \Rightarrow r(a, x) \tag{11}$$

c. Reflective:

$$\forall c \in C, r(c, c) \in R^c \tag{12}$$

Corollary 2. The maximum number of binary relations n_r and n'_r in an open system $S(C, R^c, R^i, R^o, B, \Omega, \Theta)$ is determined by the numbers of internal relations R^c as well as external relations R^i and R^o , i.e.:

$$\begin{aligned} n_r(S) &= \# R^c + \# R^i + \# R^o = \\ n_c^2 + 2(n_c \cdot n_{c_\Theta}) &= n_c(n_c + 2n_{c_\Theta}) \end{aligned} \tag{13}$$

$$\begin{aligned} n'_r(S) &= n_r(S) - n_c = n_c \cdot (n_c + 2n_{c_\Theta}) - n_c = \\ n_c \cdot (n_c + 2n_{c_\Theta} - 1) \end{aligned} \tag{14}$$

Example 2. According to Corollary 2, the creation of relations in an open system is solely determined by the number of components possessed in the system and its environment. This property can also be illustrated in Figure 2, where S_1 is treated as the open system and S_2 as the environment Θ :

$$\begin{aligned} n_r(S) &= n_c(n_c + 2n_{c_\Theta}) \\ &= 3(3 + 2 \cdot 2) = 21 \end{aligned}$$

and

$$\begin{aligned} n'_r(S) &= n_r(S) - n_c \\ &= 21 - 3 = 18 \end{aligned}$$

According to Corollaries 1 and 2, as well as Examples 1 and 2, it is apparent that either a closed or an open system may result in a huge number of relations n_r and the exponential increases of complexity, when the number of components possessed in them is considerably large. Therefore, system algebra is introduced to formally and efficiently manipulate abstract and general systems.

Definition 5. A system algebra SA on a given universal system environment \mathcal{U} is a triple, i.e.:

$$SA \triangleq (S, OP, \Theta) = (\{C, R^c, R^i, R^o, B, \Omega\}, \{\bullet_r, \bullet_c\}, \Theta) \tag{15}$$

where $OP = \{\bullet_r, \bullet_c\}$ are the sets of relational and compositional operations on abstract systems.

System algebra provides a denotational mathematical means for algebraic manipulations of abstract systems. System algebra can be used to model, specify, and manipulate generic “to be” and “to have” type problems, particularly system architectures and high-level system designs, in computing, software engineering, system engineering, and cognitive informatics. The relational and compositional operations on abstract systems will be formally described in the following sections.

PROPERTIES OF ABSTRACT SYSTEMS

Taxonomy of Systems

Systems as the most complex entities in the physical and abstract worlds may be classified

into various categories according to the key characteristics of their components (C), relations (R), behaviors (B), constraints (Ω), and/or environments (Θ). A summary of the system taxonomy is shown in Table 1, according to Definitions 3 and 4.

Table 1 shows that all types of systems fit the unified framework of system taxonomy. There are hybrid systems that may fall in two or more categories, such as a dynamic nonlinear system and a discrete fuzzy social system. The types of systems may also be classified by their magnitudes as described in the following subsection.

Magnitude of Systems

Abstract and real-world systems may be very small or extremely large (Qian et al, 1990; Rosen, 1977). Therefore, a formal model of system magnitudes is needed to classify the size properties of systems and their relationship with other basic system attributes. In order to derive such a model, a set of measures on system sizes, magnitudes, and complexities is introduced next.

Table 1. Taxonomy of systems

No.	System	Key Characteristics			
		Components (C)	Relations (R)	Behaviors (B)	Environment (Θ)
1	Concrete	Natural or real entities			
2	Abstract	Mathematical or virtual entities			
3	Physical	Natural entities			
4	Social	Humans			
5	Finite	#C ≠ ∞			
6	Infinite	#C = ∞			
7	Closed		$R^i = \emptyset \wedge R^o = \emptyset$		
8	Open		$R^i \neq \emptyset \wedge R^o \neq \emptyset$		
9	Static			Invariable	
10	Dynamic			Variable	

continued on following page

Table 1. continued

11	Linear			Linear functions	
12	Nonlinear			Nonlinear functions	
13	Continuous			Continuous functions	
14	Discrete			Discrete functions	
15	Precise			Precise functions	
16	Fuzzy			Fuzzy functions	
17	Determine			Response predictable to same stimulates	
18	Indetermine			Response unpredictable to same stimulates	
19	White-box	Observable	Transparent	Fully observable	
20	Black-box	Unobservable	Non-transparent	Partially observable	
21	Intelligent			Autonomic	Adaptive
22	Non-intelligent			Imperative	Nonadaptive
23	Maintainable	Fixable		Recoverable	
24	Non-maintainable	Nonfixable		Nonrecoverable	

Definition 6. The size of a system S_s is the number of components encompassed in the system, i.e.:

$$S_s = \#C = n_c \tag{16}$$

Definition 7. The magnitude of a system M_s is the number of asymmetric binary relations among the n_c components of the system including the reflexive relations, i.e.:

$$M_s = \#R = n_r = \#(C \times C) = n_c^2 \tag{17}$$

If all self-reflective relations are ruled out in n_r , the pure number of binary relations M'_s in the given system is determined as follows:

$$M'_s = M_s - n_c = n_c^2 - n_c = n_c(n_c - 1) \tag{18}$$

Lemma 4. The pure number of binary relations M'_s equals to exactly two times of the number of pairwise combinations among n_c i.e.:

$$M'_s = n_c(n_c - 1) = 2 \cdot \frac{n_c(n_c - 1)}{2} = 2 \cdot C_{n_c}^2 \tag{19}$$

where the factor 2 represents the asymmetric binary relation r , i.e., $arb \neq bra$.

The magnitude of a system determines its complexity. The complexities of systems can be classified based on if they are fully or partially connected. The former is the theoretical upper-bound complexity of systems in which all components are potentially interconnected with each other in all n -nary ways, $1 \leq n \leq n_c = \#C$. The latter is the more typical complexity of systems where components are only pairwise connected.

Definition 8. *The complexity of a fully connected system C_{max} is a closure of all possible n -nary relations R^* , $1 \leq n \leq n_c$, among all components of the given system $n_c = \#C$, i.e.:*

$$C_{max} = R^* = 2 \sum_{k=0}^{n_r} C_{n_r}^k \approx 2 \cdot 2^{n_r} = 2^{n_r+1} = 2^{n_c^2+1} = 2^{M_s+1} \quad (20)$$

where C_{max} is also called the maximum complexity of systems.

According to Definition 8, the closure of all possible n -nary relations R^* may easily result in an extremely huge degree of complexity for a system with few components. For example, when $n_c = 10$, $C_{max} = 2^{101}$. This explains why most of the real-world systems are really too hard to be modeled and handled in conventional models and techniques.

It is noteworthy that almost all functioning systems are partially connected, because a fully connected system may not represent or provide anything meaningful. Therefore, the complexity of partially connected systems can be simplified as follows on the basis of Definition 8.

Definition 9. *The complexity of a partially connected system C_r is determined by the number of asymmetric binary relations M'_s of the system, i.e.:*

$$C_r = M'_s = 2 \cdot C_{n_c}^2 = n_c(n_c - 1) \quad (21)$$

where C_r can be referred to the relational complexity of systems.

The extent of system magnitudes (Wang, 2006d, 2007c) can be classified at seven levels known as the *empty, small, medium, large, giant, immense, and infinite* systems from the bottom up. A summary of the relationships between system magnitudes, sizes, internal relations, and complexities can be described in the *system magnitude model*, shown in Table 2.

Table 2 indicates that the complexity of a small system may easily be out of control of human cognitive manageability. This leads to the following theorem.

Theorem 1. *The holism complexity of systems states that within the 7-level scale of system magnitudes, known as the empty, small, medium,*

Table 2. The system magnitude model

Level	Category	Size of systems ($S_s = n_c$)	Magnitude of systems ($M_s = n_r = n_c^2$)	Relational complexity of systems ($C_r = n_c(n_c - 1)$)	Maximum complexity of systems ($C_{max} = 2^{n_c^2}$)
1	The empty system (\emptyset)	0	0	0	-
2	Small system	[1, 10]	[1, 10 ²]	[0, 90]	[2, 2 ¹⁰⁰]
3	Medium system	(10, 10 ²]	(10 ² , 10 ⁴]	(90, 0.99 • 10 ⁴]	(2 ¹⁰⁰ , 2 ^{10,000}]
4	Large system	(10 ² , 10 ³]	(10 ⁴ , 10 ⁶]	(0.99 • 10 ⁴ , 0.999 • 10 ⁶]	∞
5	Giant system	(10 ³ , 10 ⁴]	(10 ⁶ , 10 ⁸]	(0.999 • 10 ⁶ , 0.9999 • 10 ⁸]	∞
6	Immense system	(10 ⁴ , 10 ⁵]	(10 ⁸ , 10 ¹⁰]	(0.9999 • 10 ⁸ , 0.99999 • 10 ¹⁰]	∞
7	The infinite system (\aleph)	∞	∞	∞	∞

large, giant, immense, and infinite systems, almost all systems are too complicated to be cognitively understood or mentally handled as a whole, except small systems or those that can be decomposed into small systems.

According to Theorem 1, the basic principle for dealing with complicated systems is system decomposition or modularity, in which the complexity of a lower level subsystem must be small enough to be cognitively manageable. Details of system decomposition theories and the art of system architectures will be developed in the following sections.

RELATIONAL OPERATIONS ON SYSTEMS

The relational operations of abstract systems are static and comparative operations that do not change the systems involved. The relational operations on abstract systems are described below.

Lemma 5. *The relational operations \bullet_r in system algebra encompasses 6 comparative operators for manipulating the algebraic relations between abstract systems, i.e.:*

$$\bullet_r \triangleq \{\leftrightarrow, \leftarrow, \sqcap, =, \sqsubseteq, \sqsupseteq\} \quad (22)$$

where the relational operators stand for independent, related, overlapped, equivalent, subsystem, and supersystem, respectively.

Algebraic Relations of Closed Systems

Relationships between two closed systems can be independent, equivalent, being subsystem, and being super system. The four relational operations of closed systems are defined as follows.

Definition 10. *Two closed systems \widehat{S}_1 and \widehat{S}_2 are independent, denoted by \leftrightarrow , if both their component sets and relation sets are disjoint, i.e.:*

$$\begin{aligned} \widehat{S}_1(C_1, R_1, B_1, \Omega_1) &\leftrightarrow \widehat{S}_2(C_2, R_2, B_2, \Omega_2) \\ &\triangleq C_1 \cap C_2 = \emptyset \wedge R_1 \cap R_2 = \emptyset \end{aligned} \quad (23)$$

Definition 11. *Two closed systems \widehat{S}_1 and \widehat{S}_2 are equivalent, denoted by $=$, if all sets of components, relations, behaviors, and constraints are identical, i.e.:*

$$\begin{aligned} \widehat{S}_1(C_1, R_1, B_1, \Omega_1) &= \widehat{S}_2(C_2, R_2, B_2, \Omega_2) \\ &\triangleq C_1 = C_2 \wedge R_1 = R_2 \wedge B_1 = B_2 \wedge \Omega_1 = \Omega_2 \end{aligned} \quad (24)$$

Definition 12. *A subsystem \widehat{S}' is a system that is implicated in another system \widehat{S} , denoted by \sqsubseteq , i.e.:*

$$\begin{aligned} \widehat{S}'(C, R, B, \Omega) &\sqsubseteq \widehat{S}(C, R, B, \Omega) \\ &\triangleq C' \subseteq C \wedge R' \subseteq R \wedge B' \subseteq B \wedge \Omega' \subseteq \Omega \end{aligned} \quad (25)$$

The aforementioned definition indicates that a subsystem of a closed system is a coherent part with all integrated components, internal/ input/output relations, behaviors, constraints, and the environment.

Definition 13. *A supersystem \widehat{S} is a system that consists of one or more subsystems S' , denoted by \sqsupseteq , i.e.:*

$$\begin{aligned} \widehat{S}(C, R, B, \Omega) &\sqsupseteq \widehat{S}'(C', R', B', \Omega') \\ &\triangleq C' \subseteq C \wedge R' \subseteq R \wedge B' \subseteq B \wedge \Omega' \subseteq \Omega \end{aligned} \quad (26)$$

Algebraic Relations of Open Systems

Relationships between two open systems can be independent, overlapped, related, equivalent, being subsystem, and being supersystem. The six compositional operations of open systems are defined as follows.

Definition 14. *Two open systems S_1 and S_2 are independent, denoted by \leftrightarrow , if both their*

component sets and external relation sets are disjoint, i.e.:

$$\begin{aligned}
 S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1) &\leftrightarrow \\
 S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2) & \\
 \triangleq C_1 \cap C_2 = \emptyset \wedge R_1^i \cap R_2^i = & \\
 \emptyset \wedge R_1^o \cap R_2^o = \emptyset & \\
 (27) &
 \end{aligned}$$

Definition 15. Two open systems S_1 and S_2 are overlapped, denoted by Π , if their component sets are joint, i.e.:

$$\begin{aligned}
 S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1) &\Pi \\
 S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2) & \\
 \triangleq C_1 \cap C_2 \neq \emptyset & \\
 (28) &
 \end{aligned}$$

Definition 16. Two open systems S_1 and S_2 are related, denoted by \leftrightarrow , if either the sets of their input relations or output relations are overlapped, i.e.:

$$\begin{aligned}
 S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1) &\leftrightarrow \\
 S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2) & \\
 \triangleq R_1^i \cap (R_2^o)^{-1} \neq \emptyset \vee R_2^i \cap (R_1^o)^{-1} \neq \emptyset & \\
 (29) &
 \end{aligned}$$

where $(R^o)^{-1}$ or $(R^i)^{-1}$ denotes an inverse relation, i.e., $\forall a \in C_1 \wedge b \in C_2, r(a, b) \in R^o \Rightarrow r(b, a) \in (R^i) = (R^o)^{-1}$.

It is noteworthy that, by definition, there is no closed system that is related or overlapped.

Definition 17. Two open systems S_1 and S_2 are equivalent, denoted by $=$, if all sets of components, relations, behaviors, constraints, and environments are identical, i.e.:

$$\begin{aligned}
 S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1) &= \\
 S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2) &\triangleq \\
 C_1 = C_2 \wedge R_1^c = R_2^c \wedge R_1^i = R_2^i \wedge R_1^o = & \\
 R_2^o \wedge B_1 = B_2 \wedge \Omega_1 = \Omega_2 \wedge \Theta_1 = \Theta_2 & \\
 (30) &
 \end{aligned}$$

Definition 18. A subsystem S' is a system that is implicated in another system S , denoted by \sqsubseteq , i.e.:

$$\begin{aligned}
 S'(C', R^{c'}, R^{i'}, R^{o'}, B', \Omega', \Theta') &\sqsubseteq \\
 S(C, R^c, R^i, R^o, B, \Omega, \Theta) &\triangleq \\
 C' \subseteq C \wedge R^{c'} \subseteq R^c \wedge R^{i'} \subseteq R^i \wedge R^{o'} & \\
 \subseteq R^o \wedge B' \subseteq B \wedge \Omega' \subseteq \Omega \wedge \Theta' = \Theta & \\
 (31) &
 \end{aligned}$$

The above definition indicates that a subsystem of an open system is a coherent part of it with all integrated components, internal/input/output relations, behaviors, and constraints. However, they share the same environment.

Definition 19. A supersystem S is a system that consists of one or more subsystems S' , denoted by \sqsupseteq , i.e.:

$$\begin{aligned}
 S(C, R^c, R^i, R^o, B, \Omega, \Theta) &\sqsupseteq \\
 S'(C', R^{c'}, R^{i'}, R^{o'}, B', \Omega', \Theta') & \\
 \triangleq C' \subseteq C \wedge R^{c'} \subseteq R^c \wedge R^{i'} \subseteq R^i \wedge R^{o'} & \\
 \subseteq R^o \wedge B' \subseteq B \wedge \Omega' \subseteq \Omega \wedge \Theta' = \Theta & \\
 (32) &
 \end{aligned}$$

Relations between Open and Closed Systems

Although, the previous subsections analyze the relations of closed and open systems separately, it is noteworthy that closed and open systems are transformable, when the environment of them is treated as a supersystem as well. This notion can be described in the following theorem and corollaries on the basis of Definitions 3 and 4.

Theorem 2. The equivalence between open and closed systems states that an open system S is equivalent to a closed system \hat{S} , or vice versa, when its environment Θ_S or $\Theta_{\hat{S}}$ is conjoined, respectively, i.e.:

$$\begin{cases} \hat{S} = S \sqcup \Theta_S \\ S = \hat{S} \sqcup \Theta_{\hat{S}} \end{cases} \quad (33)$$

Theorem 2 can be proved by observing the embedded relation of close and open systems as illustrated in Figure 3. According to Theorem 2, the following properties of equivalence between closed and open systems can be derived.

Corollary 3. Any subsystem \widehat{S}_k of a closed system \widehat{S} is an open system, i.e.:

$$\forall \widehat{S}_k \subseteq \widehat{S} \Rightarrow R_k^i \neq \emptyset \wedge R_k^o \neq \emptyset \wedge \Theta_k = C_s \neq \emptyset \quad (34)$$

Corollary 4. Any supersystem S of a given set of n open systems S_k , conjoining with their environments Θ_k , $1 \leq k \leq n$, is a closed systems, i.e.:

$$\forall S_k, \forall \Theta_k, \widehat{S} = \bigoplus_{k=1}^n (S_k \sqcup \Theta_k) \Rightarrow R_S^i = \emptyset \wedge R_S^o = \emptyset \wedge \Theta_S = \emptyset \quad (35)$$

where $\bigoplus_{k=1}^n S_k$ is an operator known as the *big-R notation* (Wang, 2002, 2008a, 2008b, 2008c) that denotes a repetitive behavior or recurrent structure as defined in real-time process algebra (RTPA) (Wang, 2002, 2003, 2006a, 2006c, 2007a, 2007c, 2008a, 2008d).

COMPOSITIONAL OPERATIONS ON SYSTEMS

This section describes how abstract systems and their relations as modeled in previous sections may be manipulated by an algebraic system. The compositional operations of system algebra are dynamic and integrative operations that manipulate all systems involved in parallel. Compositional operations on abstract systems provide a set of fundamental mathematical means to construct complex systems on the basis of simple ones or to derive new systems on the basis of exiting ones.

Lemma 6. The compositional operations \bullet_c in system algebra encompasses 9 associative operators for manipulating the algebraic compositions among abstract systems, i.e.:

$$\bullet_c \triangleq \{\Rightarrow, \bar{\Rightarrow}, \overset{+}{\Rightarrow}, \overset{\sim}{\Rightarrow}, \boxplus, \boxminus, \boxdot, \boxtimes, \boxdiv, \boxplus\} \quad (36)$$

where the compositional operators stand for system inheritance, tailoring, extension, substitute, difference, composition, decomposition, aggregation, and specification, respectively.

System Inheritance

Definition 20. The inheritance of a closed system \widehat{S}_2 from a given system \widehat{S}_1 , denoted by \Rightarrow , is the creation of the new system \widehat{S}_2 by reproducing \widehat{S}_1 , i.e.:

$$\begin{aligned} \widehat{S}_1(C_1, R_1, B_1, \Omega_1) &\Rightarrow \widehat{S}_2(C_2, R_2, B_2, \Omega_2) \\ &\triangleq \widehat{S}_2(C_2, R_2, B_2, \Omega_2 \mid C_2 = C_1, R_2 = \\ &R_1, B_2 = B_1, \Omega_2 = \Omega_1) \end{aligned} \quad (37)$$

where \widehat{S}_1 is called the parent system, \widehat{S}_2 the child system.

Similarly, the inheritance of open systems can be defined as follows.

Definition 21. The inheritance of an open system S_2 from the parent system S_p , denoted by \Rightarrow , is the creation of the new system S_2 by reproducing S_p and the establishment of new associations between them, see Box 1, where \parallel denotes that an open system inheritance creates new associations between S_1 and S_2 in parallel via (R_1^o, R_2^i) and (R_2^o, R_1^i) .

Definition 22. The multiple inheritance of an open system S from n parent systems S_p, S_2, \dots, S_n , denoted by \Rightarrow , is an inheritance that creates the new system S via a set of n conjoint systems and establishes new associations among them, see Box 2.

System Tailoring

Definition 23. The tailoring of a closed system \widehat{S}_2 from the parent system \widehat{S}_1 , denoted by $\overset{\sim}{\Rightarrow}$, is

Box 1.

$$\begin{aligned}
 & S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1) \Rightarrow S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2) \\
 & \triangleq S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2 \mid C_2 = C_1, R_2^c = R_1^c, \\
 & \quad R_2^i = R_1^i \cup \{C_1 \times C_2\}, R_2^o = R_1^o \cup \{C_2 \times C_1\}, \\
 & \quad B_2 = B_1, \Omega_2 = \Omega_1, \Theta_2 = \Theta_1) \\
 & \parallel S_1(C_1, R_1^c, R_1^{i'}, R_1^{o'}, B_1, \Omega_1, \Theta_1 \mid R_1^{i'} = R_1^i \cup \{C_2 \times C_1\}, \\
 & \quad R_1^{o'} = R_1^o \cup \{C_1 \times C_2\}, \Theta_1' = \Theta_1 \cup C_2)
 \end{aligned} \tag{38}$$

Box 2.

$$\begin{aligned}
 & \overset{n}{\mathbf{R}} S_i \Rightarrow (S, C, R^c, R^i, R^o, B, \Omega, \Theta) \\
 & \triangleq S(C, R^c, R^i, R^o, B, \Omega, \Theta \mid C = \bigcup_{i=1}^n C_i, R^c = \bigcup_{i=1}^n R_i^c, R^i = \bigcup_{i=1}^n R_i^i \cup \{\overset{n}{\mathbf{R}}(C_i \times C)\}, \\
 & \quad R^o = \bigcup_{i=1}^n R_i^o \cup \{\overset{n}{\mathbf{R}}(C \times C_i)\}, B = \bigcup_{i=1}^n B_i, \Omega = \bigcup_{i=1}^n \Omega_i, \Theta = \bigcup_{i=1}^n \Theta_i) \\
 & \parallel \overset{n}{\mathbf{R}} S_i(C_i, R_i^c, R_i^i, R_i^o, B_i, \Omega_i, \Theta_i \mid R_i^{i'} = R_i^i \cup \{C \times C_i\}, \\
 & \quad R_i^{o'} = R_i^o \cup \{C_i \times C\}, \Theta_i' = \Theta_i \cup C)
 \end{aligned} \tag{39}$$

a special system inheritance that creates the new system \widehat{S}_2 based on \widehat{S}_1 with the removal of specific subsets of components C' , behaviors B' , and constraints Ω' , see Box 3.

Similarly, the tailoring of open systems can be defined as follows.

Definition 24. The tailoring of an open system S_2 from the parent system S_1 , denoted by $\overrightarrow{\Rightarrow}$, is a special system inheritance that creates the new system S_2 based on S_1 with the removal of specific subsets of components C' , behaviors B' , and constraints Ω' ; and at the same time, it establishes new associations between them, see Box 4.

System Extension

Definition 25. The extension of a closed system \widehat{S}_2 from the parent system \widehat{S}_1 , denoted by $\overset{+}{\Rightarrow}$, is a special system inheritance that creates the new system \widehat{S}_2 based on \widehat{S}_1 with additional subsets of components C' , behaviors B' , and constraints Ω' , see Box 5.

Similarly, the extension of open systems can be defined as follows.

Definition 26. The extension of an open system S_2 from the parent system S_1 , denoted by $\overset{+}{\Rightarrow}$, is a special system inheritance that creates the new system S_2 based on S_1 with additional subsets of components C' , behaviors B' , and constraints Ω' ; and

Box 3.

$$\begin{aligned}
& \widehat{S}_1(C_1, R_1, B_1, \Omega_1) \xrightarrow{-} \widehat{S}_2(C_2, R_2, B_2, \Omega_2) \\
& \triangleq \widehat{S}_2(C_2, R_2, B_2, \Omega_2 \mid C_2 = C_1 \setminus C', R_2^c = R_1^c \setminus \{C_1 \times C'\}) \\
& \quad B_2 = B_1 \setminus B', \Omega_2 = \Omega_1 \setminus \Omega'
\end{aligned}
\tag{40}$$

Box 4.

$$\begin{aligned}
& S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1) \xrightarrow{-} S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2) \\
& \triangleq S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2 \mid C_2 = C_1 \setminus C', R_2^c = R_1^c \setminus \{C_1 \times C'\}, \\
& \quad R_2^i = R_1^i \cup \{C_1 \times C_2\}, R_2^o = R_1^o \cup \{C_2 \times C_1\}, \\
& \quad B_2 = B_1 \setminus B', \Omega_2 = \Omega_1 \setminus \Omega', \Theta_2 = \Theta_1) \\
& \parallel S_1(C_1, R_1^c, R_1^{i'}, R_1^{o'}, B_1, \Omega_1, \Theta_1 \mid R_1^{i'} = R_1^i \cup \{C_2 \times C_1\}, \\
& \quad R_1^{o'} = R_1^o \cup \{C_1 \times C_2\}, \Theta_1 = \Theta_1 \cup C_2)
\end{aligned}
\tag{41}$$

Box 5.

$$\begin{aligned}
& \widehat{S}_1(C_1, R_1, B_1, \Omega_1) \xrightarrow{+} \widehat{S}_2(C_2, R_2, B_2, \Omega_2) \\
& \triangleq \widehat{S}_2(C_2, R_2, B_2, \Omega_2 \mid C_2 = C_1 \cup C', R_2^c = R_1^c \cup \{C_1 \times C'\}) \\
& \quad B_2 = B_1 \cup B', \Omega_2 = \Omega_1 \cup \Omega'
\end{aligned}
\tag{42}$$

Box 6.

$$\begin{aligned}
& S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1) \xrightarrow{+} S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2) \\
& \triangleq S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2 \mid C_2 = C_1 \cup C', R_2^c = R_1^c \cup \{C_1 \times C'\}) \\
& \quad R_2^i = R_1^i \cup \{C_1 \times C_2\}, R_2^o = R_1^o \cup \{C_2 \times C_1\}, \\
& \quad B_2 = B_1 \cup B', \Omega_2 = \Omega_1 \cup \Omega', \Theta_2 = \Theta_1) \\
& \parallel S_1(C_1, R_1^c, R_1^{i'}, R_1^{o'}, B_1, \Omega_1, \Theta_1 \mid R_1^{i'} = R_1^i \cup \{C_2 \times C_1\}, \\
& \quad R_1^{o'} = R_1^o \cup \{C_1 \times C_2\}, \Theta_1 = \Theta_1 \cup C_2)
\end{aligned}
\tag{43}$$

at the same time, it establishes new associations between the two systems, see Box 6.

System Substitution

Definition 27. The substitute of a closed system \widehat{S}_2 from the parent system \widehat{S}_1 , denoted by $\widetilde{\Rightarrow}$, is a flexible system inheritance that creates the new system \widehat{S}_2 based on \widehat{S}_1 with the new subsets of components C'_{c_2} , behaviors B'_{c_2} , and constraints Ω'_{c_2} to replace the corresponding inherited ones C'_{c_1} , B'_{c_1} , and Ω'_{c_1} that share the same identifiers, see Box 7. Where $C'_{s_1} \subset C_1 \wedge C'_{s_2} \subset C_2 \wedge \#C'_{s_1} = \#C'_{s_2}$; $B'_{s_1} \subset B_1 \wedge B'_{s_2} \subset B_2 \wedge \#B'_{s_1} = \#B'_{s_2}$; and $\Omega'_{s_1} \subset \Omega_1 \wedge \Omega'_{s_2} \subset \Omega_2 \wedge \#\Omega'_{s_1} = \#\Omega'_{s_2}$. Similarly, the substitute of open systems can be defined as follows.

Definition 28. The substitute of an open system S_2 from the parent system S_p , denoted by \Rightarrow , is a flexible system inheritance that creates the new system S_2 based on S_1 with the new subsets of

components C'_{c_2} , behaviors B'_{c_2} , and constraints attributes Ω'_{c_2} to replace the corresponding inherited ones C'_{c_1} , B'_{c_1} , and Ω'_{c_1} that share the same identifiers; and at the same time, it establishes new associations between the two concepts, see Box 8. Where $C'_{s_1} \subset C_1 \wedge C'_{s_2} \subset C_2 \wedge \#C'_{s_1} = \#C'_{s_2}$; $B'_{s_1} \subset B_1 \wedge B'_{s_2} \subset B_2 \wedge \#B'_{s_1} = \#B'_{s_2}$; and $\Omega'_{s_1} \subset \Omega_1 \wedge \Omega'_{s_2} \subset \Omega_2 \wedge \#\Omega'_{s_1} = \#\Omega'_{s_2}$.

Binary tailoring, extension, and substitution can also be extended to corresponding n -ary operations, similar to that of inheritance as given in Definitions 22.

System Composition

As a preparation to describe the important property of system relations, the mathematical calculus of incremental union between sets of relations is introduced below (Wang, 2006c, 2007c).

Box 7.

$$\begin{aligned} & \widehat{S}_1(C_1, R_1, B_1, \Omega_1) \widetilde{\Rightarrow} \widehat{S}_2(C_2, R_2, B_2, \Omega_2) \\ & \triangleq \widehat{S}_2(C_2, R_2, B_2, \Omega_2 \mid C_2 = (C_1 \setminus C'_{s_1}) \cup C'_{s_2}, R_2^c = (R_1^c \setminus (C_1 \times C'_{s_1})) \cup (C_1 \times C'_{s_2}), \\ & \quad (B_2 = B_1 \setminus B'_{s_1}) \cup B'_{s_2}, \Omega_2 = (\Omega_1 \setminus \Omega'_{s_1}) \cup \Omega'_{s_2}) \end{aligned} \quad (44)$$

Box 8.

$$\begin{aligned} & S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1) \Rightarrow S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2) \\ & \triangleq S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2 \mid C_2 = (C_1 \setminus C'_{s_1}) \cup C'_{s_2}, \\ & \quad R_2^c = (R_1^c \setminus \{C_1 \times C'_{s_1}\}) \cup \{C_1 \times C'_{s_2}\}, R_2^i = R_1^i \cup \{C_1 \times C_2\}, \\ & \quad R_2^o = R_1^o \cup \{C_2 \times C_1\}, B_2 = (B_1 \setminus B'_{s_1}) \cup B'_{s_2}, \Omega_2 = (\Omega_1 \setminus \Omega'_{s_1}) \cup \Omega'_{s_2}, \Theta_2 = \Theta_1) \\ & \quad \parallel S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1 \mid R_1^i = R_1^i \cup \{C_2 \times C_1\}, R_1^o = R_1^o \cup \{C_1 \times C_2\}, \\ & \quad \Theta_1 = \Theta_1 \cup C_2) \end{aligned} \quad (45)$$

Definition 29. An incremental union of two sets of relations R_1 and R_2 , denoted by \boxplus , are a union of R_1 and R_2 plus a newly generated incremental set of relations ΔR_{12} , i.e.:

$$R_1 \boxplus R_2 \triangleq R_1 \cup R_2 \cup \Delta R_{12} \tag{46}$$

where $\Delta R_{12} \not\subseteq R_1 \wedge \Delta R_{12} \not\subseteq R_2$, but $\Delta R_{12} \subseteq R_1 \boxplus R_2$.

The number of the incremental relations ΔR_{12} generated in the incremental union can be determined as stated in the following theorem.

Theorem 3. The maximum number of newly gained relations ΔR_{12} obtained during the incremental union of two sets of relations is a product of the numbers of elements of the two sets $\#C_1$ and $\#C_2$, i.e.:

$$\Delta R_{12} = 2(\#C_1 \bullet \#C_2) = 2n_{c_1} n_{c_2} \tag{47}$$

Proof: Because ΔR_{12} is the difference between the relations of the newly generated entire system $\#R$ and those of the independent systems $\#R_1$ and $\#R_2$, according to Corollaries 1 and 2, Theorem 3 is proved by the following inference process:

$$\begin{aligned} \Delta R_{12} &= \#R - (\#R_1 + \#R_2) \\ &= (n_{c_1} + n_{c_2})^2 - (n_{c_1}^2 + n_{c_2}^2) \\ &= (n_{c_1}^2 + 2n_{c_1} n_{c_2} + n_{c_2}^2) - (n_{c_1}^2 + n_{c_2}^2) \\ &= 2n_{c_1} n_{c_2} \end{aligned} \tag{48}$$

Box 9.

$$\begin{aligned} \boxplus_{i=1}^n R_i &\triangleq \mathbf{R}(R_i \cup R_j \cup \Delta R_{ij}), j = i + 1 \\ &= (\dots((R_1 \cup R_2 \cup \Delta R_{12}) \cup R_3 \cup \Delta R_{2,3}) \cup \dots) \cup R_n \cup \Delta R_{n-1,n} \end{aligned} \tag{49}$$

The incremental union of relations reveals an important property of systems, which indicates that the merge of two systems results in new relations, behaviors, functions, and/or constraints that are not belong to any original individual subsystems. Theorem 3 can be used to predict the maximum numbers of newly established relations, behaviors, and/or constraints in a composition of two systems. According to Theorem 3, the maximum *incremental system gain* equals to the number of by-directly interconnection between all components in both S_1 and S_2 , i.e., $2(\#C_1 \bullet \#C_2)$.

Definition 29 can be extended to n -nary incremental unions for multiple sets of relations.

Definition 30. The n -nary incremental union for multiple sets of relations,

$$\boxplus_{i=1}^n R_i,$$

is a series of cumulative binary incremental unions as in Box 9.

Based on the calculus of incremental union of sets of relations, system compositions can be defined as follows.

Definition 31. The composition of two closed systems \widehat{S}_1 and \widehat{S}_2 , denoted by \uplus , results in a super system \widehat{S} that is formed by union of sets of components and environments, as well as incremental union of sets of relations, behaviors, and constraints, respectively, see Box 10.

Theorem 4. The system fusion principle states that new relations ΔR_{12} , new behaviors (functions) ΔB_{12} , and new constraints $\Delta \Omega_{12}$ generated

Box 10.

$$\begin{aligned}
 & \widehat{S}_1(C_1, R_1, B_1, \Omega_1) \uplus \widehat{S}_2(C_2, R_2, B_2, \Omega_2) \\
 & \triangleq \widehat{S}(C, R, B, \Omega \mid C = C_1 \cup C_2, R = R_1 \boxplus R_2, B = B_1 \boxplus B_2, \Omega = \Omega_1 \boxplus \Omega_2) \\
 & = \widehat{S}(C, R, B, \Omega \mid C = C_1 \cup C_2, R = R_1 \cup R_2 \cup \Delta R_{12}, \\
 & \qquad \qquad \qquad B = B_1 \cup B_2 \cup \Delta B_{12}, \Omega = \Omega_1 \cup \Omega_2 \cup \Delta \Omega_{12})
 \end{aligned} \tag{50}$$

in system compositions are solely a property of the new super system \widehat{S} , but not belong to any of the independent subsystems, i.e.:

$$\Delta R_{12} \in \widehat{S} \wedge (\Delta R_{12} \notin \widehat{S}_1 \wedge \Delta R_{12} \notin \widehat{S}_2) \tag{51.a}$$

$$\Delta B_{12} \in \widehat{S} \wedge (\Delta B_{12} \notin \widehat{S}_1 \wedge \Delta B_{12} \notin \widehat{S}_2) \tag{51.b}$$

$$\Delta \Omega_{12} \in \widehat{S} \wedge (\Delta \Omega_{12} \notin \widehat{S}_1 \wedge \Delta \Omega_{12} \notin \widehat{S}_2) \tag{51.c}$$

where E denote a membership relation of a given set in a system.

The discovery in Theorems 3 and 4 reveal that the nature of system utilities can be rigorously explained as the newly gained relations ΔR_{12} , as well as behaviors ΔB_{12} and constraints $\Delta \Omega_{12}$, during the composition of two or more systems. The empirical awareness of this key system property has been intuitively or qualitatively described in the literature of system science (Ellis & Fred, 1962; G. J. Klir, 1992). However, Theorems 3 and 4 are the first mathematical explanation of the mechanism of system gains during system compositions (Wang, 2006b, 2007c).

More generally, Definition 31 and Theorem 4 can be extended to open systems.

Definition 32. *The composition of two open systems S_1 and S_2 , denoted by \boxplus , results in a super system S that is formed by simple conjunctions*

both of sets of components, as well as incremental unions of sets of relations, behaviors, and constraints, respectively, see Box 11.

The operation of open system composition is illustrated in Figure 4, where the generation of the new relations ΔR_{12}^c and ΔR_{22}^c in S after the composition of S_1 and S_2 can be observed.

System compositions as modeled in Definitions 31 and 32 can be extended to n -nary compositions as follows.

Definition 33. *The composition of multiple open systems, denoted by*

$$\bigoplus_{i=1}^n S_i,$$

is an iterative integration of a pair of systems, which cumulatively creates the new supersystem S , see Box 12.

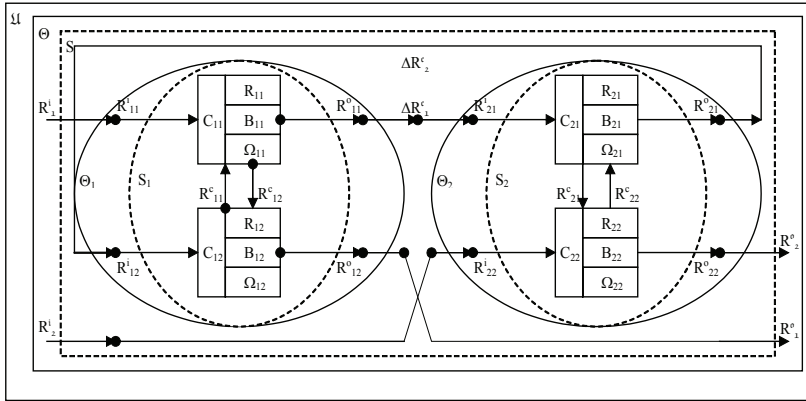
The composition of multiple closed systems is similar to Definition 33, which can be tailored from Equation 53.

System composition at the top level is a complicated algebraic operation that integrates two or more systems into a supersystem with a hierarchical architecture. There are three basic types of system structural relations in system composition known as *parallel* (\parallel), *serial* (\rightarrow), and *nested* (\dashrightarrow) as shown in Figure 5. Complex system compositions can be represented by a combination of these three meta-architectural relations between subsystems. The syntaxes and semantics of these three system relations in

Box 11.

$$\begin{aligned}
 & S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1) \sqcup S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2) \\
 & \triangleq S(C, R^c, R^i, R^o, B, \Omega, \Theta) \mid C = C_1 \cup C_2, R^c = R_1^c \boxplus R_2^c, R^i = R_1^i \boxplus R_2^i, \\
 & \quad R^o = R_1^o \boxplus R_2^o, B = B_1 \boxplus B_2, \Omega = \Omega_1 \boxplus \Omega_2, \Theta = \Theta_1 \cup \Theta_2) \\
 & \quad \parallel \prod_{i=1}^2 S_i(C_i, R_i^c, R_i^{i'}, R_i^{o'}, B_i, \Omega_i, \Theta_i \mid R_i^{i'} = R_i^i \cup \{C \times C_i\}, \\
 & \quad \quad R_i^{o'} = R_i^o \cup \{C_i \times C\}, \Theta_i' = \Theta_i \cup C) \\
 & = S(C, R^c, R^i, R^o, B, \Omega, \Theta \mid C = C_1 \cup C_2, R^c = R_1^c \cup R_2^c \cup \Delta R_{12}^c, \\
 & \quad R^i = R_1^i \cup R_2^i \cup \Delta R_{12}^i, R^o = R_1^o \cup R_2^o \cup \Delta R_{12}^o, \\
 & \quad B = B_1 \cup B_2 \cup \Delta B_{12}, \Omega = \Omega_1 \cup \Omega_2 \cup \Delta \Omega_{12}, \Theta = \Theta_1 \cup \Theta_2) \\
 & \quad \parallel \prod_{i=1}^2 S_i(C_i, R_i^c, R_i^{i'}, R_i^{o'}, B_i, \Omega_i, \Theta_i \mid R_i^{i'} = R_i^i \cup \{C \times C_i\}, \\
 & \quad \quad R_i^{o'} = R_i^o \cup \{C_i \times C\}, \Theta_i' = \Theta_i \cup C)
 \end{aligned} \tag{52}$$

Figure 4. The composition of two open systems



Box 12.

$$\begin{aligned}
 & S(C, R^c, R^i, R^o, B, \Omega, \Theta) \triangleq \bigoplus_{i=1}^n S_i(C_i, R_i^c, R_i^i, R_i^o, B_i, \Omega_i, \Theta_i) \\
 & = S(C, R^c, R^i, R^o, B, \Omega, \Theta \mid C = \bigcup_{i=1}^n C_i, R^c = \boxplus_{i=1}^n R_i^c, R^i = \boxplus_{i=1}^n R_i^i, \\
 & \quad R^o = \boxplus_{i=1}^n R_i^o, B = \boxplus_{i=1}^n B_i, \Omega = \boxplus_{i=1}^n \Omega_i, \Theta = \bigcup_{i=1}^n \Theta_i) \\
 & \quad \parallel \prod_{i=1}^n S_i(C_i, R_i^c, R_i^{i'}, R_i^{o'}, B_i, \Omega_i, \Theta_i \mid R_i^{i'} = R_i^i \cup \{C \times C_i\}, \\
 & \quad \quad R_i^{o'} = R_i^o \cup \{C_i \times C\}, \Theta_i' = \Theta_i \cup C)
 \end{aligned} \tag{53}$$

Figure 5. Basic types of system structural relations in system compositions

No.	Type of composition	Syntax	Example
1	Parallel	$S_1 \parallel S_2$	$S \triangleq S_1 \parallel S_2 \parallel \dots \parallel S_n$
2	Serial	$S_1 \rightarrow S_2$	$S \triangleq S_1 \rightarrow S_2 \rightarrow \dots \rightarrow S_n$
3	Nested	$S_1 \rightsquigarrow S_2$	$S \triangleq S_1 \rightsquigarrow S_2 \rightsquigarrow \dots \rightsquigarrow S_n$

system compositions can be referred to related definitions in RTPA (Wang, 2002, 2003, 2006c, 2007c, 2008a, 2008d).

According to Definition 33, a system can be integrated from the bottom up by a series of compositions, level-by-level, in a system hierarchy.

Example 3. A composed system $S(C, R, B, \Omega, \Theta)$ as given in Figure 6 can be formally described as follows:

$$S(C, R, B, \Omega, \Theta) \triangleq S_1 \parallel S_2 \parallel \dots \parallel S_x$$

in which the subsystems of S can be refined as in Box 13.

System Difference

Definition 34. The difference between a closed systems \widehat{S} and its subsystem \widehat{S}_1 , denoted by Ξ , results in a closed subsystem \widehat{S}_2 that is formed

by the difference of sets of components (C_i), and the differences of the internal relations, behaviors, and constraints (R_i, B_i, Ω_i) with their incremental counterparts ($\Delta R_{12}, \Delta B_{12}, \Delta \Omega_{12}$), see Box 14.

According to Definition 34, a difference of a subsystem from a system \widehat{S} , will result in the removal of not only the given subsystem but also all interrelations and incremental behaviors between the subsystem and other subsystem in it.

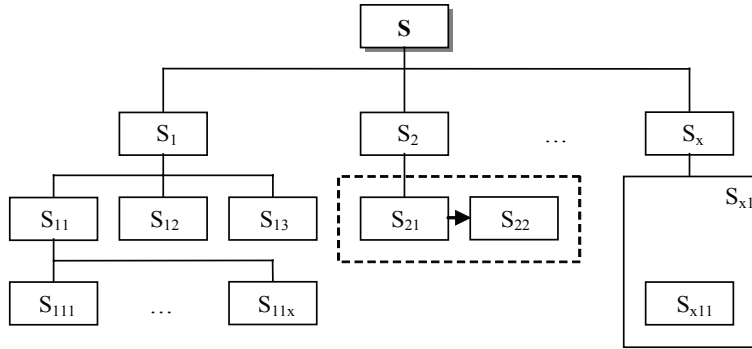
Similarly, the difference of open systems can be defined as follows.

Definition 35. The difference between an open systems S and its subsystem S_1 , denoted by Ξ , results in an open subsystem S_2 that is formed by the difference of sets of components and I/O relations (C_i, R_i^i, R_i^o), and the differences of the internal relations, behaviors, and constraints (R_i^c, B_i, Ω_i) with their incremental counterparts (ΔR_{12}^c ,

Box 13.

$$\begin{aligned}
 S_1(C_1, R_1, B_1, \Omega_1, \Theta_1) &\hat{=} S_{11} \parallel S_{12} \parallel S_{13} = S_{111} \parallel \dots \parallel S_{11x} \parallel S_{12} \parallel S_{13} \\
 S_2(C_2, R_2, B_2, \Omega_2, \Theta_2) &\hat{=} S_{21} \rightarrow S_{22} \\
 S_x(C_x, R_x, B_x, \Omega_x, \Theta_x) &\hat{=} S_{x1} \succ S_{x11}
 \end{aligned}$$

Figure 6. The hierarchical structure of system compositions



Box 14.

$$\begin{aligned}
 \widehat{S}(C, R, B, \Omega) &\boxminus \widehat{S}_1(C_1, R_1, B_1, \Omega_1) \\
 &\hat{=} \widehat{S}_2(C_2, R_2, B_2, \Omega_2 \mid C_2 = C \setminus C_1, R_2 = R \setminus (R_1 \cup \Delta R_{12}), \\
 &\quad B_2 = B \setminus (B_1 \cup \Delta B_{12}), \Omega_2 = \Omega \setminus (\Omega_1 \cup \Delta \Omega_{12}))
 \end{aligned} \tag{54}$$

$\Delta B_{12}, \Delta \Omega_{12}$, see Box 15.

The operation of open system difference can also be illustrated by Figure 4, where S_1 and all related I/O relations should be removed in the operation $S_2 = S \boxminus S_1$.

System Decomposition

A system decomposition is an inverse operation of system composition that breaks up a system into two or more subsystems. System decomposition can be described based on the concept of system difference, which is an inversed operation of the

incremental union of sets.

Definition 36. The decomposition of an open systems S , denoted by \boxminus , is to break up S into two or more subsystems at a given level of the system hierarchy by one of the compositional relations $R_c = \{\parallel, \rightarrow, \succ\}$, see Box 16.

As specified in Definition 36, the decomposition operation results in the removal of all internal relations $\Delta R_{ij}^c = C_i \times C_j, 1 \leq i, j \leq n$ that are no longer belong to any of its subsystems.

Similarly, the decomposition of a closed

Box 15.

$$\begin{aligned}
 & S(C, R^c, R^i, R^o, B, \Omega, \Theta) \boxminus S_1(C_1, R_1^c, R_1^i, R_1^o, B_1, \Omega_1, \Theta_1) \\
 & \triangleq S_2(C_2, R_2^c, R_2^i, R_2^o, B_2, \Omega_2, \Theta_2 \mid C_2 = C \setminus C_1, R_2^c = R^c \setminus (R_1^c \cup \Delta R_{12}^c), \\
 & \quad R_2^i = R^i \setminus R_1^i, R_2^o = R^o \setminus R_1^o, B_2 = B \setminus (B_1 \cup \Delta B_{12}), \\
 & \quad \Omega_2 = \Omega \setminus (\Omega_1 \cup \Delta \Omega_{12}), \Theta_2 = \Theta_1)
 \end{aligned} \tag{55}$$

Box 16.

$$\begin{aligned}
 & S(C, R^c, R^i, R^o, B, \Omega, \Theta) \prod_{i=1}^n \mathfrak{S}_i C_i, R_i^c, R_i^i, R_i^o, B_i, \Omega_i, \Theta_i) \\
 & \triangleq \mathop{\bigcap}_{i=1}^n \{ S_i(C_i, R_i^c, R_i^i, R_i^o, B_i, \Omega_i, \Theta_i \mid R_i^i = R^i \cup \{C \times C_i\}, \\
 & \quad R_i^o = R^o \cup \{C_i \times C\}, \Theta_i = \Theta_i \cup C) \\
 & \quad \mid S(C, R^c, R^i, R^o, B, \Omega, \Theta) \boxminus S_i(C_i, R_i^c, R_i^i, R_i^o, B_i, \Omega_i, \Theta_i) \}
 \end{aligned} \tag{56}$$

system into multiple subsystems can be defined as follows.

Definition 37. The decomposition of a closed system \widehat{S} , denoted by \bowtie , is to break up \widehat{S} into two or more subsystems at a given level of the system hierarchy by one of the compositional relations $R_c = \{||, \rightarrow, \rightsquigarrow\}$, see Box 17.

Definitions 36 and 37 indicate that either an open or a closed system can be resolved from the top down by a series of decompositions, level-by-level, in the system hierarchy. It is noteworthy that both open and closed system decompositions result in a set of open subsystems.

System Aggregation

Definition 38. The aggregation of a closed system \widehat{S} from a set of n peer systems $\widehat{S}_i, 1 \leq i \leq n$, denoted by \Leftarrow , is an aggregation of \widehat{S} with the elicitation of interested subsets of components

C'_i behaviors B'_i , and constraints Ω'_i , see Box 18. Similarly, the aggregation of open systems can be defined as follows.

Definition 39. The aggregation of an open system S from a set of n peer systems $S_i, 1 \leq i \leq n$, denoted by \Leftarrow , is an aggregation of S with the elicitation of interested subsets of components C'_i behaviors B'_i and constraints Ω'_i ; and at the same time, it establishes new associations among all aggregated systems, see Box 19. System aggregation is also known as system elicitation.

According to Definitions 32 and 39, the difference between system composition and aggregation is that the former constructs a new system by integrating a set of entire systems as subsystems while the latter constructs a new system by eliciting interested subsets of components, behaviors, and/or constraints from a set of individual and independent systems.

Box 17.

$$\begin{aligned}
 & \widehat{S}(C, R, B, \Omega) \prod_{i=1}^n \widehat{S}_i(C_i, R_i, B_i, \Omega_i) \\
 & \triangleq \prod_{i=1}^n \{ S_i(C_i, R_i^c, R_i^i, R_i^o, B_i, \Omega_i, \Theta_i \mid R_i^i = (C \times C_i), R_i^o = (C_i \times C), \Theta_i = C) \\
 & \quad \parallel \widehat{S}(C, R, B, \Omega) \boxplus \widehat{S}_i(C_i, R_i, B_i, \Omega_i) \\
 & \quad \}
 \end{aligned} \tag{57}$$

Box 18.

$$\begin{aligned}
 S(C, R, B, \Omega) & \Leftarrow \prod_{i=1}^n \delta_i(C_i, R_i, B_i, \Omega_i) \\
 & \triangleq S(C, R, B, \Omega \mid C = \bigcup_{i=1}^n C'_i \subseteq C_i, R = C \times C, \\
 & \quad B = \bigcup_{i=1}^n B_i \subseteq B_i, \Omega = \bigcup_{i=1}^n C'_i \subseteq C_i)
 \end{aligned} \tag{58}$$

Box 19.

$$\begin{aligned}
 S(C, R^c, R^i, R^o, B, \Omega, \Theta) & \Leftarrow \prod_{i=1}^n S_i(C_i, R_i^c, R_i^i, R_i^o, B_i, \Omega_i, \Theta_i) \\
 & \triangleq S(C, R^c, R^i, R^o, B, \Omega, \Theta \mid C = \bigcup_{i=1}^n C'_i \subseteq C_i, R^c = \{C \times C\}, \\
 & \quad R^i = \bigcup_{i=1}^n (R_i^i \cup \{C_i \times C\}), R^o = \bigcup_{i=1}^n (R_i^o \cup \{C \times C_i\}), \\
 & \quad B = \bigcup_{i=1}^n B'_i \subseteq B_i, \Omega = \bigcup_{i=1}^n C'_i \subseteq C_i, \Theta \subseteq C_\Theta \cup \bigcup_{i=1}^n \Theta_i) \\
 & \parallel \prod_{i=1}^n S_i(C_i, R_i^c, R_i^{i'}, R_i^{o'}, B_i, \Omega_i, \Theta'_i \mid R_i^{i'} = R_i^i \cup \{C \times C_i\}, \\
 & \quad R_i^{o'} = R_i^o \cup \{C_i \times C\}, \Theta'_i = \Theta_i \cup C)
 \end{aligned} \tag{59}$$

System Specification

A system specification is an inverse operation of system aggregation. System specification is usually operated in a series of refinements.

Definition 40. *The specification of a closed sys-*

tem \widehat{S}_0 by a set of n refined systems $\widehat{S}_i, 1 \leq i \leq n$, denoted by \vdash , is a specification of \widehat{S}_0 with a total order of a series of refinements by increasingly specific and detailed components C_i , behaviors B_i , and constraints Ω_i , see Box 20.

Similarly, the specification of open systems

Box 20.

$$\begin{aligned}
 & (\widehat{S}_n \vdash \dots \vdash \widehat{S}_2 \vdash \widehat{S}_1) \vdash \widehat{S}_0(C_0, R_0, B_0, \Omega_0) \\
 & \triangleq \widehat{S}_0(C_0, R_0, B_0, \Omega_0 \mid C_0 = \prod_{i=1}^n (C_{i-1} \subset C_i), R_0 = (C_0 \times C_0), \\
 & \quad B_0 = (\prod_{i=1}^n B_{i-1} \subset B_i), \Omega_0 = (\prod_{i=1}^n B_{i-1} \subset B_i))
 \end{aligned} \tag{60}$$

can be defined as follows.

Definition 41. *The specification of an open system S_0 by a set of n refined systems S_i , $1 \leq i \leq n$, denoted by \vdash , is a specification of S_0 with a total order of a series of refinements by increasingly specific and detailed components C_i , behaviors B_i and constraints Ω_{ij} ; and at the same time, it establishes new associations among all refining systems, see Box 21.*

System specification is a refinement process where more specific and detailed components, behaviors, and constraints are developed in a consistent and coherent top-down hierarchy. The major tasks of system specifications are

system architecture (component) and behavior specifications, which can be further refined by the Component Logical Models (CLMs) and processes as provided in RTPA (Wang, 2002, 2003, 2006a, 2006c, 2007c, 2008a, 2008d).

CONCLUSION

A new mathematical structure of abstract systems has been presented as the most complicated mathematical entities beyond sets, functions, concepts, and processes. A formal and rigorous treatment of abstract systems as well as their taxonomy and properties has been described. System algebra has been introduced as a set of relational and compositional operations for manipulating abstract systems and their com-

Box 21.

$$\begin{aligned}
 & (S_n \vdash \dots \vdash S_2 \vdash S_1) \vdash S_0(C_0, R_0^c, R_0^i, R_0^o, B_0, \Omega_0, \Theta_0) \\
 & \triangleq S_0(C_0, R_0^c, R_0^i, R_0^o, B_0, \Omega_0, \Theta_0 \mid C_0 = \prod_{i=1}^n (C_{i-1} \subset C_i), R_0^c = \{C_0 \times C_0\}, \\
 & \quad R_0^i = \bigcup_{i=1}^n (R_i^i \cup \{C_i \times C_0\}), R_0^o = \bigcup_{i=1}^n (R_i^o \cup \{C_0 \times C_i\}), \\
 & \quad B_0 = \prod_{i=1}^n (B_{i-1} \subset B_i), \Omega_0 = \prod_{i=1}^n (B_{i-1} \subset B_i), \Theta_0 = \Theta_1 = \Theta_2 = \dots = \Theta_n) \\
 & \parallel \prod_{i=1}^n S_i(C_i, R_i^c, R_i^i, R_i^o, B_i, \Omega_i, \Theta_i \mid R_i^i = R_i^i \cup \{C_0 \times C_i\}, \\
 & \quad R_i^{o'} = R_i^o \cup \{C_i \times C_0\}, \Theta_i = \Theta_i \cup C_0)
 \end{aligned} \tag{61}$$

posing rules. The former have been elicited as the algebraic operations of *independent, related, overlapped, equivalent, subsystem, and super-system*. The latter have been identified as the algebraic operations of *inheritance, tailoring, extension, substitute, difference, composition, decomposition, aggregation, and specification*. A wide range of applications of system algebra has been recognized in cognitive informatics, system science, system engineering, computing, and software engineering. System algebra has formed a fundamental theory for denoting the rigorous semantics of conventional object-oriented design notations and methodologies such as UML in software and intelligent system engineering.

ACKNOWLEDGMENT

The author would like to acknowledge the Natural Science and Engineering Council of Canada (NSERC) for its partial support to this work. The author would like to thank anonymous reviewers for their valuable comments and suggestions.

REFERENCES

- Ashby, W. R. (1958). Requisite variety and implications for control of complex systems. *Cybernetica*, 1, 83-99.
- Ashby, W. R. (1962) Principles of the self-organizing system. In von H. Foerster & G. Zopf (Eds.), *Principles of self-organization* (pp. 255-278). Oxford, England: Pergamon.
- Eigen, M., & Schuster, P. (1979). *The hypercycle: A principle of natural self-organization*. Berlin, Germany: Springer.
- Ellis, D. O., & Fred, J. L. (1962). *Systems philosophy*. Prentice Hall.
- Ford, J. (1986). Chaos: Solving the Unsolvable, predicting the unpredictable. In *Chaotic dynamics and fractals*. Academic Press.
- Haken, H. (1977). *Synergetics*. New York: Springer-Verlag.
- Heylighen, F. (1989). Self-organization, emergence and the architecture of complexity. In *Proceedings of the First European Conference on System Science (AF CET)* (pp. 23-32). Paris.
- Klir, G. J. (1992). *Facets of systems science*. New York: Plenum Press.
- Klir, R. G. (1988). Systems profile: the emergence of systems science. *Systems Research*, 5(2), 145-156.
- Prigogine, I., & Nicolis, G. (1972). Thermodynamics of evolution. *Physics Today*, 25, 23-28.
- Rapoport, A. (1962). Mathematical aspects of general systems theory. *General Systems Yearbook*, 11, 3-11.
- Skarda, C. A., & Freeman, W. J. (1987). How brains make chaos into order. *Behavioral and Brain Sciences*, 10.
- von Bertalanffy, L. (1952). *Problems of life: An evolution of modern biological and scientific thought*. London: C. A. Watts.
- Wang, Y. (2002). The real-time process algebra (RTPA). *The International Journal of Annals of Software Engineering*, 14, 235-274.
- Wang, Y. (2003). Using process algebra to describe human and software system behaviors. *Brain and Mind*, 4(2), 199-213.
- Wang, Y. (2005). System science models of software engineering. In *Proceedings of the Eighteenth Canadian Conference on Electrical and Computer Engineering (CCECE'05)* (pp. 1802-1805). Saskatoon, Saskatchewan, Canada: IEEE CS Press.
- Wang, Y. (2006a). Cognitive informatics and contemporary mathematics for knowledge representation and manipulation (Invited plenary talk). In *Proceedings of the First International Conference on Rough Set and Knowledge Technology (RSKT'06)* (LNAI 4062, pp. 69-78). Chongqing, China: Springer.
- Wang, Y. (2006b). On abstract systems and system algebra. In *Proceedings of the Fifth IEEE International Conference on Cognitive Informatics (ICCI'06)* (pp. 332-343), Beijing, China: IEEE CS Press.
- Wang, Y. (2006c, March). On the informatics laws and deductive semantics of software. *IEEE Transactions on Systems, Man, and Cybernetics (C)*, 36(2), 161-171.
- Wang, Y. (2007a). Keynote speech, on theoretical

foundations of software engineering and denotational mathematics. In *Proceedings of the Fifth Asian Workshop on Foundations of Software* (pp. 99-102). Xiamen, China: BHU Press.

Wang, Y. (2007b). The OAR model of neural informatics for internal knowledge representation in the brain. *The International Journal of Cognitive Informatics and Natural Intelligence*, 1(3), 64-75.

Wang, Y. (2007c). Software engineering foundations: A software science perspective. In *CRC Series in Software Engineering: Vol. 2*. CRC Press.

Wang, Y. (2007d). The theoretical framework of cognitive informatics. *The International Journal of Cognitive Informatics and Natural Intelligence*, 1(1), 1-27.

Wang, Y. (2008a, April). Deductive semantics of RTPA. *The International Journal of Cognitive Informatics and Natural Intelligence*, 2(2), 95-121.

Wang, Y. (2008b, April). On concept algebra: A denotational mathematical structure for knowledge and software modeling. *The International Journal*

of Cognitive Informatics and Natural Intelligence, 2(2), 1-19.

Wang, Y. (2008c). On the big-R notation for describing iterative and recursive behaviors. *The International Journal of Cognitive Informatics and Natural Intelligence*, 2(1), 17-28.

Wang, Y. (2008d, April). RTPA: A denotational mathematics for manipulating intelligent and computing behaviors. *The International Journal of Cognitive Informatics and Natural Intelligence (IJCINI)*, 2(2), 44-62.

Wang, Y., & Wang, Y. (2006). On cognitive informatics models of the brain. *IEEE Transactions on Systems, Man, and Cybernetics (C)*, 36(2), 203-207.

Zadeh, L. A. (1965). Fuzzy sets and systems. In J. Fox (Ed.), *Systems theory* (pp. 29-37). Brooklyn, NY: Polytechnic Press.

Zadeh, L. A. (1973). Outline of a new approach to analysis of complex systems. *IEEE Trans. on Sys., Man and Cyb.*, 1(1), 28-44.

Yingxu Wang is professor of cognitive informatics and software engineering, director of International Center for Cognitive Informatics (ICfCI), and director of Theoretical and Empirical Software Engineering Research Center (TESERC) at the University of Calgary. He received a PhD in software engineering from The Nottingham Trent University, UK, in 1997, and a BSc in electrical engineering from Shanghai Tiedao University in 1983. He was a visiting professor in the Computing Laboratory at Oxford University and Department of Computer Science at Stanford University during 1995 and 2008, respectively, and has been a full professor since 1994. He is the editor-in-chief of the International Journal of Cognitive Informatics and Natural Intelligence (IJCINI), and editor-in-chief of the CRC Book Series in Software Engineering. He has published over 300 journal and conference papers and 11 books in software engineering and cognitive informatics, and won dozens of research achievement, best paper, and teaching awards in the last 28 years, particularly the IBC 21st Century Award for Achievement "in recognition of outstanding contribution in the field of Cognitive Informatics and Software Science," and the groundbreaking book on Software Engineering Foundations: A Software Science Perspective.