Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Matrix Expression of Logic and Fuzzy Control

Daizhan Cheng, Hongsheng Qi
Institute of Systems Science, Chinese Academy of Sciences
Beijing 100080, P.R.China
dcheng@iss03.iss.ac.cn

***Abstract*— This paper gives a matrix expression of logic. Under matrix expression a general description of the logical operations is proposed, which is very convenient in logical inference. Then based on matrix expression the logic operators have been extended to multi-valued logic, which provides a foundation for fuzzy systems. Finally, the logic-based fuzzy control is considered.**

I. PRELIMINARIES

Mathematical logic [6] is the foundation for design of logic-based intelligent systems [11] and fuzzy control [9], [7], [8]. But unlike standard logic, the multi-valued logic still doesn't have an axiomatic foundation. Moreover, there exists a gap between fuzzy logic and fuzzy control.

Recently, a new matrix product, called semi-tensor product (STP) and denoted by $\ltimes$ has been proposed [2], [4] and applied to several control and related problems [3]. We refer to the Appendix for basic concepts and properties of STP.

It is shown in this paper that the semi-tensor product is a suitable tool to describe the logical operators. Under this matrix expression the properties of operators can be revealed easily. Moreover, this expression can easily be extended to multi-valued logic. Finally, the multi-valued logic is used to provide a framework for the fuzzy logic and logic-based fuzzy control.

In one word, the purpose of this paper is to provide a new mathematical foundation for logic-based fuzzy control by using semi-tensor product.

Next, we give some definitions and notations for the statement ease.

Definition 1.1 *A pure logical domain, denoted by D_l, is defined as*

$$D_l = \{T = 1, F = 0\}; \tag{1}$$

A k-valued logical domain ($k > 2$), denoted by D_k, is defined as

$$D_k = \{i/(k-1) | i = 0, 1, \cdots, k-1\}; \tag{2}$$

A fuzzy logical domain, denoted by D_f, is defined as

$$D_f = \{r \mid 0 \le r \le 1\}. \tag{3}$$

Remark. *1. In the sequel we use D for D_l, D_k, or D_f. In fact, fuzzy control doesn't use D_f. It uses only D_k and (in most cases) $k \le 9$. So in the sequel we consider D_l and D_k only, unless elsewhere stated.*

2. We may consider D_l as a particular case of D_k for $k = 2$.

Definition 1.2 *A basic (s-ary) logical operator (or relation) is a mapping*

$$\sigma : \underbrace{D \times D \times \cdots \times D}_{s} \to D.$$

To use matrix expression we identify the elements in D_k with a vector as

$$e_i = \frac{k-i}{k-1} \Leftrightarrow \delta_i^k, \quad i = 1, 2, \cdots, k$$

where δ_i^k is the i-th column of identity matrix I_k.

Definition 1.3 *A $k \times k^s$ matrix M_σ is called the structure matrix of an s-ary logical operator σ if*

$$\sigma(P_1, \cdots, P_s) = M_\sigma \ltimes P_1 \ltimes \cdots \ltimes P_s. \tag{4}$$

Note that all the matrix products throughout this paper are semi-tensor product and we omit the symbol $\ltimes$ hereafter.

Next, let's see how to construct the structure matrix for an s-ary logic operator. Let σ be an s-ary operator and

$$m_{i_1, i_2, \cdots, i_s} = \sigma(\delta_{i_1}, \delta_{i_2}, \cdots, \delta_{i_s}), \quad 1 \le i_1, i_2, \cdots, i_s \le k.$$

Using this notation, we have

Proposition 1.4 *The structure matrix of σ is*

$$M_\sigma = \begin{bmatrix} m_{11\cdots1} & m_{11\cdots2} & \cdots & m_{kk\cdots k} \end{bmatrix}. \tag{5}$$

In fact, it provides a one-one correspondence between structure matrices and the logical operators:

Definition 1.5 *A $k \times k^s$ matrix is called a logical matrix, if its columns are elements of D_k.*

It is obvious that a logical matrix is a structure matrix of an s-ary logic operator. From the sequel one sees that it might be degenerated (i.e., of lower orders).

Proposition 1.6 *The product of two (k-valued) logic matrices is still a logic matrix.*

II. MATRIX EXPRESSION OF FUNDAMENTAL LOGICAL OPERATORS

In this section considers the standard logic. The truth table of conjunction, disjunction, implication, and equivalence are as follows [10]:

*Work was supported by NNSF 60274012, 60221301, 60334040 of China

Table 1. Truth Table

P	Q	$P \vee Q$	$P \wedge Q$	$P \rightarrow Q$	$P \leftrightarrow Q$
T	T	T	T	T	T
T	F	T	F	F	F
F	T	T	F	T	F
F	F	F	F	T	T

Using (5) and truth table it is easy to construct the structure matrices of the corresponding operators.

Proposition 2.1 *1. For unary operator "negation" $\neg$, its matrix form (M_n) is*

$$M_n = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{6}$$

2. The matrices for binary operators: "conjunction", $\wedge$, (M_c); "disjunction", $\vee$, (M_d); "implication", $\rightarrow$, (M_i); "equivalence", $\leftrightarrow$, (M_e) are respectively

$$\begin{array}{l} M_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}; \quad M_d = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \\ M_i = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}; \quad M_e = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}. \end{array} \tag{7}$$

It is well known that there are 4 different unary operators, 16 binary operators. In general, we have 2^{2^s} different mappings over s-ary operators. (k^{k^s} for k-valued logic.)

In case of $s = 2$. The truth table for 16 different mappings are:

Table 2. Truth Tables For Binary Operators

P	Q	σ_0	σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7
		F	$\downarrow$	$-_w$	$\neg P$	$-$	$\neg Q$	$\veebar$	$\uparrow$
1	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	1	1	1
0	1	0	0	1	1	0	0	1	1
0	0	0	1	0	1	0	1	0	1

P	Q	σ_8	σ_9	σ_{10}	σ_{11}	σ_{12}	σ_{13}	σ_{14}	σ_{15}
		$\wedge$	$\leftrightarrow$	Q	$\rightarrow$	P	$\rightarrow_w$	$\vee$	T
1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1
0	1	0	0	1	1	0	0	1	1
0	0	0	1	0	1	0	1	0	1

Where $\sigma_0 = F$, $\sigma_{15} = T$ are constant operators; $\sigma_3 = \neg P$, $\sigma_5 = \neg Q$, $\sigma_{10} = Q$, and $\sigma_{12} = P$ are unary operators. We have the following new binary operators:

- Exclusive OR (σ_6) ($P \veebar Q$):

$$P \veebar Q \Leftrightarrow \neg(P \leftrightarrow Q); \tag{8}$$

- NAND (σ_7, not and) ($P \uparrow Q$):

$$P \uparrow Q \Leftrightarrow \neg(P \wedge Q); \tag{9}$$

- NOR (σ_1, not or) ($P \downarrow Q$):

$$P \downarrow Q \Leftrightarrow \neg(P \vee Q); \tag{10}$$

- NIMP (σ_4, not implementation) ($P - Q$):

$$P - Q \Leftrightarrow \neg(P \rightarrow Q); \tag{11}$$

- NIIMP (σ_2, not inv. impl.) ($P -_w Q$):

$$P -_w Q \Leftrightarrow \neg(Q \rightarrow P); \tag{12}$$

- ICOD (σ_{13}, inv. cond.) $P \rightarrow_w Q$:

$$P \rightarrow_w Q \Leftrightarrow Q \rightarrow P. \tag{13}$$

Using the structure matrices and the properties of semi-tensor product, logical relations can easily be proved without any special knowledge of logic.

Example 2.2 *1. Prove*

$$P \rightarrow Q \Leftrightarrow \neg Q \rightarrow \neg P. \tag{14}$$

Proof.

$$\begin{array}{rcl} RHS & = & M_i M_n Q M_n P = M_i M_n (I_2 \otimes M_n) QP \\ & = & M_i M_n (I_2 \otimes M_n) W_{[2]} PQ. \end{array}$$

Since

$$\begin{array}{l} M_i M_n (I_2 \otimes M_n) W_{[2]} \\ = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} = M_i, \end{array}$$

(14) follows.

2. Prove

$$(P \vee Q) \wedge (P \rightarrow R) \wedge (Q \rightarrow R) \Rightarrow R. \tag{15}$$

Proof. Assume the right hand side is false, i.e., $R = (0,1)^T$, we check the left hand side:

$$\begin{array}{l} (P \vee Q) \wedge (P \rightarrow R) \wedge (Q \rightarrow R) \\ = M_c M_d P Q M_c M_i P R M_i Q R \\ = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ 1-p \end{bmatrix} \begin{bmatrix} q \\ 1-q \end{bmatrix} \\ \quad \ltimes \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ 1-p \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ \quad \ltimes \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} q \\ 1-q \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} X(p,q) \\ Y(p,q) \end{bmatrix}, \end{array}$$

where

$$\begin{array}{rcl} X(p,q) & = & (p+q-pq)(1-p)(1-q) \\ Y(p,q) & = & (p+q-pq) + (1-p)^2(1-q)^2. \end{array}$$

We have four cases to check (1) $p = 0, q = 0$, (2) $p = 0, q = 1$, (3) $p = 1, q = 0$, and (4) $p = 1, q = 1$. In each case the last matrix is $(0,1)^T$. □

Next, we define a matrix, called the order-reducing matrix, as

$$M_r = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}. \tag{16}$$

Its name is from the following property.

Lemma 2.3 *Let A be a logic variable. Then for any $p \times 4q$ matrix Ψ*

$$\Psi A^2 = \Psi M_r A. \tag{17}$$

In a logic expression a logic variable is constant if its value is assigned in advance, it is called a free variable if its value can be arbitrary. Using this concept and above Lemma, we have

Theorem 2.4 *Any logic expression $L(P_1, \cdots, P_s)$ with free logic variables $P_1, \cdots, P_s$ can be expressed in a standard form as*

$$L(P_1, \cdots, P_s) = M_L P_1 P_2 \cdots P_s, \tag{18}$$

where M_L is a 2×2^s logic matrix.

Using (6)-(7) and M_r, the matrices of binary operators (8)-(13) can be easily deduced. Moreover M_i and M_e (then all the binary operators) can be expressed by M_c, M_d, and M_n, that is,

$$M_i = M_d M_n, \tag{19}$$

$$M_e = M_c M_d M_n (I_4 \otimes M_d M_n)(I_2 \otimes M_r)(I_2 \otimes W_{[2]}) M_r. \tag{20}$$

We use an example to show how convenient the matrix expression in logical inference.

Example 2.5 *There are three persons A, B and C. A said B is a liar, B said C is a liar, and C said A and B are liars. Who is a liar?*

The logic expression of the statement is

$$(A \leftrightarrow \neg B) \wedge (B \leftrightarrow \neg C) \wedge (C \leftrightarrow \neg A \wedge \neg B).$$

Its matrix form, $L(A, B, C)$, is

$$M_c^2 (M_e A M_n B)(M_e B M_n C)(M_e C M_c M_n A M_n B) \tag{21}$$

Its standard form can be computed as:

$$L(A, B, C) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \end{bmatrix} ABC.$$

L is true only if:

$$A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Conclusion: Only B is honest. □

III. Multi-valued Logic

One of the advantages of the matrix form of logic is it can easily be extended to multi-valued logic. First generalization is, using scaler form, we define:

Definition 3.1 *Let P and Q be two logic variables. Then their disjunction is defined as*

$$P \vee Q = max(P, Q); \tag{22}$$

their conjunction is defined as

$$P \wedge Q = min(P, Q). \tag{23}$$

The negation is defined as

$$\neg P = 1 - P. \tag{24}$$

Example 3.2 *Let $k = 3$. Then we can easily obtained*

$$M_d^3 = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{25}$$

$$M_c^3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}, \tag{26}$$

$$M_n^3 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}. \tag{27}$$

□

Definition 3.1 is usually universal. But for other logic operators, them could be defined in various ways. For instance, there are three well known 3-Valued Logics, Kleene-type(K), Łukasiewicz-type(Ł), Bochvar-type(B) are listed as follows ($T = 1$, $U = 1/2$, $F = 0$) [8].

Table 3. 3*-Valued Logics*

		K		Ł		B	
P	Q	$\rightarrow$	$\leftrightarrow$	$\rightarrow$	$\leftrightarrow$	$\rightarrow$	$\leftrightarrow$
T	T	T	T	T	T	T	T
T	U	U	U	U	U	U	U
T	F	F	F	F	F	F	F
U	T	T	U	T	U	U	U
U	U	U	U	T	T	U	U
U	F	U	U	U	U	U	U
F	T	T	F	T	F	T	F
F	U	T	U	T	U	U	U
F	F	T	T	T	T	T	T

For k-valued logic, it is reasonable to define logic operators via their structure matrices. Now let's use (19) to define implication for $k = 3$. It is easy to calculate that

$$M_i^3 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{28}$$

We can also use (20) to define the structure matrix of equivalence. We need one more notation. Let δ_i^k be the i-th column of I_k, $i = 1, \cdots, k$. It is easy to prove that the order-reducing matrix for k-valued logic is

$$M_r^k = \begin{bmatrix} \delta_1^k & 0 & \cdots & 0 \\ 0 & \delta_2^k & \cdots & 0 \\ & & & \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \cdots & \delta_k^k \end{bmatrix}. \tag{29}$$

Using it to (20), we have

$$M_e^3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}. \tag{30}$$

It is obvious that the implicitly defined 3-valued logic coincides with Kleene-type logic.

We use the following example to show how to use the matrix expression to perform fuzzy logical inference.

Example 3.3 ([8]) *A detective has the following clues for a murder:*

1) 80% *sure that either A or B is criminal;*
2) *If A is the killer, it is very likely that the committing time is not before midnight;*
3) *If B's statement is true, the room's light at midnight was on;*
4) *If B's statement is false, it is very likely that the committing time is before midnight;*
5) *There is evidence to assure that the room's light was off at midnight;*

We assume (as a common understanding) "very likely" is stronger than "80%", and quantize: "T", "very likely", "80%, "$1-80\%$", "very unlikely", "F" as 6 logic levels and consider the problem over 6-valued logic. Denote the statement propositions as:

1) *A: A is the killer;*
2) *B: B is the killer;*
3) *M: The committing time is before midnight;*
4) *S: B's statement is true;*
5) *L: The room's light was on at midnight;*

Then we have the following fuzzy logical equations

$$\begin{array}{rcl} A\vee B &=& (0,0,1,0,0,0)^T,\\ A\to\neg M &=& (0,1,0,0,0,0)^T,\\ S\to L &=& (1,0,0,0,0,0)^T,\\ \neg S\to M &=& (0,1,0,0,0,0)^T,\\ \neg L &=& (1,0,0,0,0,0)^T. \end{array} \tag{31}$$

We use the matrix expression to perform the fuzzy logical inference. First, from $\neg L=(1,0,0,0,0,0)^T$ we have

$$L=(0,0,0,0,0,1)^T.$$

Then from $S\to L=(1,0,0,0,0,0)^T$ we have the matrix form as

$$M_i^6 SL = M_i^6 W_{[6]} LS := \Psi_1 S = (0,1,0,0,0,0)^T;$$

It is easy to calculate that

$$\Psi_1=\begin{bmatrix}0&0&0&0&0&1\\0&0&0&0&1&0\\0&0&0&1&0&0\\0&0&1&0&0&0\\0&1&0&0&0&0\\1&0&0&0&0&0\end{bmatrix}$$

Then S can be solved as $S=(0,0,0,0,0,1)^T$. Similarly, from $\neg S\to M=(0,1,0,0,0,0)^T$ we can solve M out as

$$M=(0,1,0,0,0,0)^T.$$

From $A\to\neg M = M_i^6 A M_n^6 M = (0,1,0,0,0,0)^T$ we can solve

$$A=(0,0,0,0,1,0)^T.$$

Finally, from $A\vee B = M_d^6 AB = (0,0,1,0,0,0)^T$ we can solve

$$B=(0,0,1,0,0,0)^T.$$

We conclude that "very unlikely" that A is the killer, and "80%" possibly B is the killer. □

Comparing our inference with it in [8] one sees that [8] created several un-axiomatic artificial rules for fuzzy logical inference and then used them to the previous example. While we don't need any of them and obtained the same conclusion.

IV. Fuzzy Control Systems

Instead of fuzzy logic, we use multi-valued logic in fuzzy control systems. The reason is, as the authors' knowledge, in fuzzy system analysis and fuzzy control design only logic with finite values is useful. (As indicated in literatures, mostly, $k\leq 9$.) The following framework can be considered as the "quantized version" of [1].

A fuzzy (control) system is described by an input-output mapping: $U\to Y$. The goal of a fuzzy control is to get control $u(y)$ via fuzzy logic. We give a formal description as:

- Universes of Discourse:
  - Output Space: $Y\subset\mathbb{R}^p$;
  - Input (Control) Space: $U\subset\mathbb{R}^m$;
- Linguistic Propositions:

$$\mathcal{A}_Y=\{A_1,A_2,\cdots,A_s\};$$
$$\mathcal{B}_U=\{B_1,\cdots,B_t\};$$

- Corresponding membership functions:

$$\mu_{A_i}(y)\in D_k,\quad y\in Y;$$
$$\mu_{B_j}(u)\in D_k,\quad u\in U;$$

- Linguistic Rules:
  If $A_1^1,\cdots,A_{s_1}^1$ satisfy R_1, then B_1;
  $\vdots$
  If $A_1^t,\cdots,A_{s_t}^t$ satisfy R_t, then B_t, where $A_j^i\in\mathcal{A}$;
- Corresponding to R_i, the logic expression is L_i, $i=1,\cdots,t$.

Linguistic Rules are expressed in logical expression as

$$\begin{cases} L_1(A_1^1,\cdots,A_{s_1}^1)\to B_1\\ \quad\vdots\\ L_t(A_1^t,\cdots,A_{s_t}^t)\to B_t. \end{cases} \tag{32}$$

Using Mandani implication ($A\to B=A\wedge B$, which is commonly used in fuzzy control), the membership degree for i-th Rule is

$$\mu_{R_i}=\mu_{A_1^i}(y)\wedge\cdots\wedge\mu_{A_{s_i}^i}(y)\wedge\mu_{B_i}(u),\quad i=1,\cdots,t. \tag{33}$$

Note that in most recent fuzzy control only a special case is considered [9]. That is, there are only two (or three) linguistic propositions: say, A_1, A_2; and one rule, say, R. The linguistic rule is simply:

If A_1 and A_2 satisfy R, then B.

Then the linguistic rule becomes

$$\mu_R=\mu_{A_1}(y)\wedge\mu_{A_2}(y)\wedge\mu_B(u).$$

In this framework, the linguistic rules could be more general. We can use their logical expressions directly. Then the lookup rule table is not necessary any more.

We use a simple example to demonstrate it.

Example 4.1 *To keep a container on constant temperature, we use an electric heater. The temperature, T, and its varying rate, E, are quantized as in 7 levels as $T_i = \{1, \cdots, 7\}$, $E_i = \{1, \cdots, 7\}$. Let the output, $Y = E_i + 2 * T_j$, and quantize it, by $Q(Y)$, into 7 levels as $3-4$, $5-8$, $9-11$, $12-13$, $14-16$, $17-19$, $20-21$, and consider $Q(Y) \in D_7$.*

Now assume the control U is the voltage of the electrical heater and U is also in 7 levels. It is reasonable to assume that

$$u = \neg Q(Y). \tag{34}$$

Then the rule table follows as

Table 4. Rule Table

E\U\T	*-3*	*-2*	*-1*	*0*	*1*	*2*	*3*
-3	*3*	*2*	*2*	*1*	*1*	*0*	*-1*
-2	*3*	*2*	*2*	*1*	*0*	*-1*	*-1*
-1	*2*	*2*	*1*	*1*	*0*	*-1*	*-2*
0	*2*	*2*	*1*	*0*	*-1*	*-1*	*-2*
1	*2*	*1*	*1*	*0*	*-1*	*-2*	*-2*
2	*2*	*1*	*0*	*-1*	*-1*	*-2*	*-3*
3	*1*	*1*	*0*	*-1*	*-2*	*-2*	*-3*

Comparing with the classical method [7], it provides same control rule. But we don't need pre-assigned rule table because it is generated automatically by control rule. One more advantage is we can use logic-based control for general multi-input (control) multi-output (measurement) case without any (complicated higher dimensional) table. □

Remark. *In fact, in most cases of fuzzy control problem the numbers of levels for different variables are different. So we have to consider mix-valued logic variables. That is, the logic variables can take values from $D_{k_1} \oplus \cdots \oplus D_{k_s}$. It is a sub-space of fuzzy logic, i.e.,*

$$D_{k_1} \oplus \cdots \oplus D_{k_s} \subset D_f \oplus \cdots \oplus D_f.$$

So all the results of fuzzy logic remain true for mix-valued logic. But a significant advantage of mix-valued logic is the matric approach is available. Fuzzy control based on logical relations is called a logic-based fuzzy control. We refer to [5] for details.

V. Conclusion

Using semi-tensor product of matrices, the matrix expression of logical operators has been proposed. All the logical relations can be easily proved. They have also been extended to multi-valued logic. A canonical form of general logical expression is presented. It was also shown that the matrix form of multi-valued logic is very powerful in fuzzy logical inference. Finally, matric expression of multi-logic is also applied to fuzzy control systems, and a logic-based fuzzy control is proposed.

[5] is a complete version of this paper.

VI. Appendix

This appendix gives a brief review on semi-tensor product of matrices. We refer to [2], [4] for detail.

Definition A.1 *1. Let X be a row vector of dimension np, and Y be a column vector with dimension p. Then we split X into p equal-size blocks as $X^1, \cdots, X^p$, which are $1 \times n$ rows. Define the left semi-tensor product, denoted by $\ltimes$, as*

$$\begin{cases} X \ltimes Y = \sum\limits_{i=1}^{p} X^i y_i \in \mathbb{R}^n, \\ Y^T \ltimes X^T = \sum\limits_{i=1}^{p} y_i (X^i)^T \in \mathbb{R}^n. \end{cases} \tag{35}$$

2. Let $A \in M_{m\times n}$ and $B \in M_{p\times q}$. If either n is a factor of p, say $nt = p$ and denote it as $A \prec_t B$, or p is a factor of n, say $n = pt$ and denote it as $A \succ_t B$, then we define the left semi-tensor product of A and B , denoted by $C = A \ltimes B$, as the following: C consists of $m \times q$ blocks as $C = (C^{ij})$ and each block is

$$C^{ij} = A^i \ltimes B_j, \quad i = 1, \cdots, m, \quad j = 1, \cdots, q,$$

where A^i is i-th row of A and B_j is the j-th column of B.

Remark. *Note that when $n = p$ the left semi-tensor product coincides the conventional matrix product. Therefore, the left semi-tensor product is only a generalization of the conventional product. For convenience, we may omit the product symbol $\ltimes$.*

Some fundamental properties of the left semi-tensor product are collected in the following:

Proposition A.2 [2] *The left semi-tensor product satisfies (as long as the related products are well defined)*

1. (Distributive rule)

$$\begin{aligned} &A \ltimes (\alpha B + \beta C) = \alpha A \ltimes B + \beta A \ltimes C; \\ &(\alpha B + \beta C) \ltimes A = \alpha B \ltimes A + \beta C \ltimes A, \quad \alpha, \beta \in \mathbb{R}. \end{aligned} \tag{36}$$

2. (Associative rule)

$$(A \ltimes B) \ltimes C = A \ltimes (B \ltimes C). \tag{37}$$

Proposition A.3 *Let $A \in M_{p\times q}$ and $B \in M_{m\times n}$. If $q = km$, then*

$$A \ltimes B = A(B \otimes I_k); \tag{38}$$

If $kq = m$, then

$$A \ltimes B = (A \otimes I_k)B. \tag{39}$$

Proposition A.4 *1. Assume A and B are of the proper dimensions such that $A \ltimes B$ is well defined. Then*

$$(A \ltimes B)^T = B^T \ltimes A^T. \tag{40}$$

2. In addition assume both A and B are invertible, then

$$(A \ltimes B)^{-1} = B^{-1} \ltimes A^{-1}. \tag{41}$$

Proposition A.5 *Assume $A \in M_{m\times n}$ is given.*

1. Let $Z \in \mathbb{R}^t$ be a row vector. Then

$$A \ltimes Z = Z \ltimes (I_t \otimes A); \tag{42}$$

2. Let $Z \in \mathbb{R}^t$ be a column vector. Then

$$Z \ltimes A = (I_t \otimes A) \ltimes Z. \tag{43}$$

Note that when $\xi \in \mathbb{R}^n$ is a column or a row, then $\xi^k := \underbrace{\xi \ltimes \cdots \ltimes \xi}_{k}$ is well defined.

A swap matrix, $W_{[m,n]}$ is an $mn \times mn$ matrix constructed in the following way: label its columns by $(11, 12, \cdots, 1n, \cdots, m1, m2, \cdots, mn)$ and its rows by $(11, 21, \cdots, m1, \cdots, 1n, 2n, \cdots, mn)$. Then its element in the position $((I, J), (i, j))$ is assigned as

$$w_{(IJ),(ij)} = \delta_{i,j}^{I,J} = \begin{cases} 1, & I = i \text{ and } J = j, \\ 0, & \text{otherwise.} \end{cases} \tag{44}$$

When $m = n$ we simply denote by $W_{[n]}$ for $W_{[n,n]}$.

Let $A \in M_{m\times n}$, i.e., A is an $m \times n$ matrix. Denote by $V_r(A)$ the row stacking form of A, that is,

$$V_r(A) = (a_{11} \cdots a_{1n} \ \cdots \ a_{m1} \cdots a_{mn})^T,$$

and by $V_c(A)$ the column stacking form of A, that is,

$$V_c(A) = (a_{11} \cdots a_{m1} \ \cdots \ a_{1n} \cdots a_{mn})^T.$$

The following "swap" property shows the meaning of the name.

Proposition A.6 *1. Let $X \in \mathbb{R}^m$ and $Y \in \mathbb{R}^n$ be two columns. Then*

$$W_{[m,n]} \ltimes X \ltimes Y = Y \ltimes X, \quad W_{[n,m]} \ltimes Y \ltimes X = X \ltimes Y. \tag{45}$$

2. Let $A \in M_{m\times n}$. Then

$$W_{[m,n]} V_r(A) = V_c(A), \quad W_{[n,m]} V_c(A) = V_r(A). \tag{46}$$

Proposition A.7

$$W_{[m,n]}^T = W_{[m,n]}^{-1} = W_{[n,m]}. \tag{47}$$

Proposition A.8

$$W_{[pq,r]} = \left(W_{[p,r]} \otimes I_q\right)\left(I_p \otimes W_{[q,r]}\right). \tag{48}$$

Taking transpose on both sides of (48) yields

$$W_{[r,pq]} = \left(I_p \otimes W_{[r,q]}\right)\left(W_{[r,p]} \otimes I_q\right). \tag{49}$$

References

[1] R. Babuska, An overview of fuzzy modeling and model-based fuzzy control, in *Fuzzy logic Control, Advances in Applications*, H.B. Verbruggen (ed.), World Scientific Publishing, Singapore, 1999.

[2] D. Cheng, *Matrix and Polynomial Approach to Dynamic Control Systems*, Science Press, Beijing, 2002.

[3] D. Cheng, Y. Dong, Semi-tensor product of matrices and its applications to physics, *Methods and Applications of Analysis*, Vol. 10, No. 4, 565-588, 2003.

[4] D. Cheng, X. Hu, Y. Wang, Non-regular Feedback Linearization of Nonlinear Systems via a Normal Form Algorithm, *Automatica*, Vol. 40, No. 3, 439-447, 2004.

[5] D. Cheng, H. Qi, Matrix expression of logic and its applications to fuzzy logical inference and fuzzy control, submitted for pub.

[6] A.G. Hamilton, *Logic for Mathematicians*, revised ed., Cambridge University Press, Cambridge, 1988.

[7] J. Han, Y. Li, *Fuzzy Control Technology*, Chongqing University Press, Chongqing, 2003 (in Chinese).

[8] Z. Liu, Y. Liu, *Fuzzy Logic and Neural Network*, BUAA Press, Beijing, 1996 (in Chinese).

[9] K.M. Passino, S. Yurkovich, *Fuzzy Control*, Addison Wesley Longman, 1998.

[10] L. Rade, B. Westergren, *Mathematics Handbook for Science and Engineering*, 4th ed, Studentlitteratur, Sweden, 1998.

[11] K. Truemper, *Design of Logic-based Intelligent Systems*, Wiley & Sons, New Jersey, 2004.