J Control Theory Appl 2008 6 (1) 123–133
DOI 10.1007/s11768-008-7190-z

Logic and logic-based control

Hongsheng QI, Daizhan CHENG,
(Institute of Systems Science, Chinese Academy of Sciences, Beijing 100080, China)

Abstract: This paper gives a matrix expression of logic. Under the matrix expression a general description of the logical operators is proposed. Using the semi-tensor product of matrices, the proofs of logical equivalences, implications, etc., can be simplified a lot. Certain general properties are revealed. Then, based on matrix expression, the logical operators are extended to multi-valued logic, which provides a foundation for fuzzy logical inference. Finally, we propose a new type of logic, called mix-valued logic, and a new design technique, called logic-based fuzzy control. They provide a numerically computable framework for the application of fuzzy logic for the control of fuzzy systems.

Keywords: Semi-tensor product; Matrix expression of logic; Mix-valued logic; Logic-based fuzzy control

1 Introduction

Mathematical logic [1] is the foundation for designing logic-based intelligent systems [2] and fuzzy control [3∼5]. However, unlike standard logic, the multi-valued logic does not have an axiomatic foundation yet. 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 and applied to several control and related problems [6∼8]. We refer to [6] for the basic concepts and properties of STP.

Under the matrix expression with semi-tensor product the properties of logical operators can be revealed or proved easily by matrix computation without any special knowledge on logic.

Moreover, this expression can easily be extended to multi-valued logic. Using some examples, we show that under matrix expression, the fuzzy logical inference can be converted to some matrix calculations.

Then, a new type of logic, called mix-valued logic, is introduced. It can be considered as a sub-Morgan algebra of the fuzzy algebra. Therefore, all the properties of fuzzy logic are true for mix-valued logic. With this, a new technique, called logic-based fuzzy control, is proposed. Compared with the existing method, it is more convenient and more widely applicable.

In brief, the purpose of this paper is to provide a new mathematical foundation for logic, multi-valued logic, and mix-valued logic via a matrix form of logic and semi-tensor product of matrices. They are useful for logic-based fuzzy inference and fuzzy control.

The rest of the paper is organized as follows: Section 2 gives a matrix expression for logical operators. The structure matrix and its properties of logic are investigated in Section 3. In Section 4, the basic properties of logic are re-visited and proved via their matrix expressions. Section 5 presents a canonical matrix form for all logical expressions. In Section 6, most results in previous sections are extended to multi-valued logic. Section 7 proposes a new type of logic, called mix-valued logic, which is a sub-Morgan algebra of fuzzy logic. Finally, in Section 8, logic-based fuzzy control is proposed and investigated via mix-valued logic. It is not only easy in computation but also more widely applicable. Section 9 is the conclusion.

2 Matrix expression of logical operators

Definition 1 1) A classical/logical domain, denoted by D_ℓ, is defined as

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

2) A classical/logical variable, P, is a variable taking values in D_ℓ, i.e., $P \in D_\ell$;

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

$$D_f = \{\mu \mid 0 \leqslant \mu \leqslant 1\}; \tag{2}$$

4) A fuzzy logical variable, P, is a variable taking values in D_f, i.e., $P \in D_f$.

5) A k-valued logical domain, denoted by D_k, is defined as

$$D_k = \left\{ \frac{i}{k-1} \middle| i = 0, 1, \cdots, k-1 \right\}; \tag{3}$$

6) A k-valued logical variable, P, is a variable taking values in D_k, i.e., $P \in D_k$.

It is obvious that a classical/logical variable is a k-valued logical variable with $k = 2$; and a k-valued logical variable is also a fuzzy logical variable.

Received 14 September 2007.
This work was partly supported by the National Natural Science Foundation of China (No.60274010, 60343001, 60221301, 60334040).

We reserve $T_0 \equiv T \equiv 1$ and $F_0 \equiv F \equiv 0$ as constant variables for "True" and "False" respectively.

Definition 2 1) An r-ary logical operator is a mapping $\sigma : \underbrace{D_f \times D_f \times \cdots \times D_f}_{r} \to D_f$.

2) An r-ary k-valued logical operator is a mapping $\sigma : \underbrace{D_k \times D_k \times \cdots \times D_k}_{r} \to D_k$. Here, $k = 2$ is allowed.

In the rest of this section we consider the classical logic only. To use matrix expression we identify logical values as

$$T := 1 \equiv \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad F := 0 \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{4}$$

Definition 3 A 2×2^r matrix M_σ is called the structure matrix of an r-ary logical operator σ if

$$\sigma(P_1, \cdots, P_r) = M_\sigma \ltimes P_1 \ltimes \cdots \ltimes P_r := M_\sigma P_1 \cdots P_r. \tag{5}$$

Note that all the matrix products throughout this paper are STP and we omit the symbol $\ltimes$ hereafter (as in the last term of (5)).

First, we consider how to construct the structure matrix for an operator. Consider the four fundamental binary operators [9]: Disjunction, $P \vee Q$; Conjunction, $P \wedge Q$; Implication, $P \to Q$; Equivalence, $P \leftrightarrow Q$.

Their truth tables are as in Table 2 [9].

Table 1 Truth tables.

P	Q	$P \vee Q$	$P \wedge Q$	$P \to Q$	$P \leftrightarrow Q$
1	1	1	1	1	1
1	0	1	0	0	0
0	1	1	0	1	0
0	0	0	0	1	1

According to the truth tables, we can produce the structure matrix easily. Taking "$\vee$" (disjunction) as an example, Its values in the truth table are $(1, 1, 1, 0)^{\mathrm{T}}$. Then we define a matrix as

$$M_d = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

In general, if the truth table of an r-ary operator σ is $(s_1, s_2, \cdots, s_{2^r})^{\mathrm{T}}$, then its structure matrix is:

$$M_\sigma = \begin{bmatrix} s_1 & s_2 & \cdots & s_{2^r} \\ 1 - s_1 & 1 - s_2 & \cdots & 1 - s_{2^r} \end{bmatrix}. \tag{6}$$

The following result is fundamental, which is a consequence of the structure of truth tables and the associativity of the semi-tensor product of matrices.

Theorem 1 The M_σ defined in (6) is the structure matrix of σ. That is, for this M_σ the equation (5) holds.

Proof Split M_σ into two $2 \times 2^{r-1}$ blocks as $M_\sigma = (M_\sigma^1, M_\sigma^2)$. Then the right hand side of (5) becomes

$$\begin{bmatrix} M_\sigma^1 & M_\sigma^2 \end{bmatrix} P_1 P_2 \cdots P_r = \left(\begin{bmatrix} M_\sigma^1 & M_\sigma^2 \end{bmatrix} P_1 \right) P_2 \cdots P_r. \tag{7}$$

Now if $P_1 = (1, 0)^{\mathrm{T}}$, (7) becomes $M_\sigma^1 P_2 \cdots P_r$, which corresponds to the block where the first variable $P_1 = 1$. When $P_1 = (0, 1)^{\mathrm{T}}$, (7) becomes $M_\sigma^2 P_2 \cdots P_r$, which corresponds to the block where the first variable $P_1 = 0$. Continuing this procedure of deduction, finally, it will get the precise value in the truth table for corresponding values of variables.

Using Theorem 1, the structure matrices of four fundamental binary operators are as follows:

$$\begin{cases} M_\vee := M_d = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\ M_\wedge := M_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}, \\ M_\to := M_i = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \\ M_\leftrightarrow := M_e = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}. \end{cases} \tag{8}$$

In the following, we would like to answer the question: How many logical operators in general do we have?

Theorem 2 1) A logical operator σ has a unique structure matrix. 2) A 2×2^r matrix is a structure matrix of a logical operator, iff all its columns are elements in D_ℓ.

Proof 1) By the construction, it is obvious because the truth table for any σ is unique.

2) Let ξ_s be the s-th column of the 2×2^r matrix M. Express $s - 1$ into binary form as $s_1 s_2 \cdots s_r$. (Note that since $s - 1 \leqslant 2^{2^r} - 1$, the length, l, of $s - 1$ in binary form is less than or equal to 2^r. If $l < 2^r$, add $2^r - l$ zeros ahead of it to form a binary number of length 2^r, e.g., say $r = 2$ and $s = 6$, then $s - 1 = 101$, and set $s_1 s_2 s_3 s_4 = 0101$). If $s_i = 0$ choose $P_i = (1, 0)^{\mathrm{T}}$; if $s_i = 1$, choose $P_i = (0, 1)^{\mathrm{T}}$. (For our example of $r = 2$ and $s = 6$, $P_1 = (1, 0)^{\mathrm{T}}$, $P_2 = (0, 1)^{\mathrm{T}}$, $P_3 = (1, 0)^{\mathrm{T}}$, and $P_4 = (0, 1)^{\mathrm{T}}$). Then it is easy to check that

$$M_\sigma P_1 P_2 \cdots P_k = \xi_s.$$

Hence, $\xi_s \in D_\ell$.

Conversely, assume all the columns of M are in D_ℓ, and denote the first row of M by M_1. Set the truth table of σ be M_1^{T}, then according to the proof of Theorem 1, it is obvious that $M = M_\sigma$.

3 General structure of logical operators

Theorem 2 provides a general picture for the set of logical operators. In fact, we know now there are 2^r different variable cases, and corresponding to each variable case we have 2 possible values to be assigned to an operator. Hence, there are totally 2^{2^r} different r-ary logical operators. (There are k^{k^r} for r-ary k-valued logic operators).

In this section, we consider them case by case for $r = 1$ and $r = 2$.

First, consider $r = 1$. In general, we can have four operators:

Table 2 Unary logical operators.

P	σ_0^1	σ_1^1	σ_2^1	σ_3^1
	F_0	$\neg$	$\equiv$	T_0
1	0	0	1	1
0	0	1	0	1

The structure matrices of these unary logical operators are as follows,

$$M_{F_0} = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}, \quad M_n = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix};$$

$$M_{id} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad M_{T_0} = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}.$$

Only "$\neg$" is commonly used. However, for completeness and convenience in later use, the other three are also presented here.

Next, we consider binary operators: We summarize them in Table 3.

Table 3 Truth tables for binary operators.

P	Q	σ_0^2	σ_1^2	σ_2^2	σ_3^2	σ_4^2	σ_5^2	σ_6^2	σ_7^2
		F_0	$\downarrow$	$-^*$	$\neg_1$	$-$	$\neg_2$	$\veebar$	$\uparrow$
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_8^2	σ_9^2	σ_{10}^2	σ_{11}^2	σ_{12}^2	σ_{13}^2	σ_{14}^2	σ_{15}^2
		$\wedge$	$\leftrightarrow$	Q	$\rightarrow$	P	$\rightarrow^*$	$\vee$	T_0
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

In addition to using some meaningful symbols for the operators, we also propose a notation as σ_j^i. It is very convenient in use. Here the superscript i indicates the degree of the operator and the subscript j, when converted to the binary form, is the truth table of the operator. Using this, the structure matrix of σ_j^i can be obtained immediately.

Example 1 Consider $\neg_2$, its alterative notation is σ_5^2. Now $5 = 101 = 0101$, so

$$M_{\neg_2} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}.$$

4 Some properties of basic logical operators

According to the structure of the truth tables of the basic logical operators, we have the following negation property.

Proposition 1 Given an r-ary logical operator, σ_a^r, its negation operator is $\sigma_{2^{2^r}-a-1}^r$. That is,

$$\neg\sigma_a^r(P,Q) = \sigma_{2^{2^r}-a-1}^r(P,Q). \tag{9}$$

Proof Since $2^{2^r} - 1 = \underbrace{1\,1\,\cdots\,1}_{2^r}$. Express a into binary form as (by possibly adding zeros ahead of the number) 2^r digital binary number as:

$$a = a_1 a_2 \cdots a_{2^r}.$$

Then, $(a_1, a_2, \cdots, a_{2^r})^{\mathrm{T}}$ is the truth table of σ_a^r. Now in binary form

$$2^{2^r} - 1 - a = (1 - a_1)(1 - a_2)\cdots(1 - a_r),$$

which is the binary form of $2^{2^r} - a - 1$. In other words, the truth table of $\sigma_{2^{2^r}-a-1}^r$ is $(1-a_1, 1-a_2, \cdots, 1-a_r)^{\mathrm{T}}$. (9) follows immediately.

Definition 4 Two logical expressions are said to be (absolute) logically equivalent, if they have the same logical value for all possible values of the variables in D_ℓ (D_f).

Proposition 2 Assume two logical expressions contain same number of logical variables and each variable appears to each expression only once. Then they are absolute logically equivalent, iff they are logical equivalent.

Proof Under the matrix expression one sees that a logical expression is a linear mapping about each individual variable as long as this variable appears in the expression only once. Now the conditions assure that both expressions are multi-linear about all its variables. If their matrix expressions are equal for $P_i = (1,0)^{\mathrm{T}}$ and $P_i = (0,1)^{\mathrm{T}}$, a linear combination shows that they are equal for $P_i = (\mu, 1-\mu)^{\mathrm{T}}$, $0 \leqslant \mu \leqslant 1$.

Proposition 3 The following are absolute logical equivalence:

1) $\neg\neg P \Leftrightarrow P$;

2) $(P \wedge Q) \wedge R \Leftrightarrow P \wedge (Q \wedge R)$;

3) $(P \vee Q) \vee R \Leftrightarrow P \vee (Q \vee R)$;

4) $\neg(P \wedge Q) = \neg P \vee \neg Q$;

5) $\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$;

6) $P \rightarrow Q \Leftrightarrow \neg P \vee Q$;

7) $\neg(P \rightarrow Q) \Leftrightarrow P \wedge \neg Q$;

8) $P \rightarrow Q \Leftrightarrow \neg Q \rightarrow \neg P$;

9) $P \rightarrow (Q \rightarrow R) \Leftrightarrow (P \wedge Q) \rightarrow R$;

10) $\neg(P \leftrightarrow Q) \Leftrightarrow P \leftrightarrow \neg Q$;

Proof We prove (8) only, and by Proposition 2, we have only to prove it is logically equivalent.

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

Since

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

(8) follows.

Definition 5 An r-ary operator is said to be symmetric if

$$M_\sigma P_1P_2\cdots P_k=M_\sigma P_{\lambda(1)}P_{\lambda(2)}\cdots P_{\lambda(k)},\ \forall\lambda\in S_k,\quad(10)$$

where S_k is the k-th order permutation group.

Proposition 4 A binary basic operator is symmetric, iff in its truth table $(s_1,s_2,s_3,s_4)^{\mathrm T}$

$$s_2=s_3.$$

Proof Note that the structure matrix of this operator, σ, is

$$M_\sigma=\begin{bmatrix}s_1&s_2&s_3&s_4\\1-s_1&1-s_2&1-s_3&1-s_4\end{bmatrix},$$
$$\sigma(P,Q)=M_\sigma PQ=M_\sigma W_{[2]}QP$$
$$=\begin{bmatrix}s_1&s_3&s_2&s_4\\1-s_1&1-s_3&1-s_2&1-s_4\end{bmatrix}QP$$
$$=M_\sigma QP=\sigma(Q,P).$$

Example 2 Consider the binary operators in Table 3. A straightforward computation shows that among them F_0, $\downarrow$, $\veebar$, $\uparrow$, $\wedge$, $\leftrightarrow$, $\vee$, and T_0 are symmetric.

Proposition 5 The following are logical equivalence:

1) $P\vee P\Leftrightarrow P$;
2) $P\wedge P\Leftrightarrow P$;
3) $R\vee(P\wedge\neg P)\Leftrightarrow R$;
4) $R\wedge(P\vee\neg P)\Leftrightarrow R$;
5) $P\wedge(Q\vee R)\Leftrightarrow(P\wedge Q)\vee(P\wedge R)$;
6) $P\vee(Q\wedge R)\Leftrightarrow(P\vee Q)\wedge(P\vee R)$;
7) $P\leftrightarrow Q\Leftrightarrow(P\to Q)\wedge(Q\to P)$;
8) $P\leftrightarrow Q\Leftrightarrow(P\wedge Q)\vee(\neg P\wedge\neg Q)$.

Proof We prove (3) only. Let $R=(\delta,1-\delta)^{\mathrm T}$ and $P=(\mu,1-\mu)^{\mathrm T}$. Then

$$LHS=M_\vee RM_\wedge PM_\neg P$$
$$=\begin{bmatrix}1&1&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}\delta\\1-\delta\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&1&1\end{bmatrix}\begin{bmatrix}\mu\\1-\mu\end{bmatrix}$$
$$\cdot\begin{bmatrix}0&1\\1&0\end{bmatrix}\begin{bmatrix}\mu\\1-\mu\end{bmatrix}$$
$$=\begin{bmatrix}\delta+(1-\delta)\mu(1-\mu)\\(1-\delta)[\mu^2+\mu(1-\mu)+(1-\mu)^2)]\end{bmatrix}.$$

As long as $\mu\in\{0,1\}$ we have $LHS=(\delta,1-\delta)^{\mathrm T}=R$.

Proposition 6 The following logical implications are true.

1) $P\wedge Q\Rightarrow P$;
2) $P\wedge Q\Rightarrow Q$;
3) $P\Rightarrow P\vee Q$;
4) $Q\Rightarrow P\vee Q$;
5) $\neg P\Rightarrow P\to Q$;
6) $Q\Rightarrow P\to Q$;
7) $\neg(P\to Q)\Rightarrow P$;
8) $\neg(P\to Q)\Rightarrow\neg Q$;
9) $\neg P\wedge(P\vee Q)\Rightarrow Q$;
10) $P\wedge(P\to Q)\Rightarrow Q$;
11) $\neg Q\wedge(P\to Q)\Rightarrow\neg P$;
12) $(P\to Q)\wedge(Q\to R)\Rightarrow P\to R$;
13) $(P\vee Q)\wedge(P\to R)\wedge(Q\to R)\Rightarrow R$.

Proof We prove (13) only. Assume the right hand side is false, i.e., $R=(0,1)^{\mathrm T}$, we check the left hand side:

$$(P\vee Q)\wedge(P\to R)\wedge(Q\to R)$$
$$=M_\wedge M_\vee PQM_\wedge M_\to PRM_\to QR$$
$$=\begin{bmatrix}1&0&0&0\\0&1&1&1\end{bmatrix}\begin{bmatrix}1&1&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}p\\1-p\end{bmatrix}\begin{bmatrix}q\\1-q\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&1&1\end{bmatrix}$$
$$\ltimes\begin{bmatrix}1&0&1&1\\0&1&0&0\end{bmatrix}\begin{bmatrix}p\\1-p\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}1&0&1&1\\0&1&0&0\end{bmatrix}\begin{bmatrix}q\\1-q\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix}$$
$$=\begin{bmatrix}(p+q-pq)(1-p)(1-q)\\p^2q^2-2p^2q-2pq^2+p^2+q^2+3pq-p-q+1\end{bmatrix}.$$

We have four cases to check

1) $p=0,\ q=0$,
2) $p=0,\ q=1$,
3) $p=1,\ q=0$,
4) $p=1,\ q=1$.

In each case, the last matrix is $(0,1)^{\mathrm T}$.

Finally, we consider some equivalences to EOR, NAND, and NOR. Using the matrix expression, the proofs of the following are similar.

Proposition 7 The following are logically equivalent.

1) $P\veebar Q\Leftrightarrow Q\veebar P$;
2) $(P\veebar Q)\veebar R\Leftrightarrow P\veebar(Q\veebar R)$;
3) $P\wedge(Q\veebar R)\Leftrightarrow(P\wedge Q)\veebar(P\wedge R)$;
4) $P\veebar Q\Leftrightarrow(P\wedge\neg Q)\vee(\neg P\wedge Q)$;
5) $P\veebar Q\Leftrightarrow\neg(P\leftrightarrow Q)$;
6) $P\uparrow Q\Leftrightarrow Q\uparrow P$;
7) $P\downarrow Q\Leftrightarrow Q\downarrow P$;
8) $P\uparrow(Q\uparrow R)\Leftrightarrow\neg P\vee(Q\wedge R)$;
9) $(P\uparrow Q)\uparrow R\Leftrightarrow(P\wedge Q)\vee\neg R$;
10) $P\downarrow(Q\downarrow R)\Leftrightarrow\neg P\wedge(Q\vee R)$;

11) $(P \downarrow Q) \downarrow R \Leftrightarrow (P \vee Q) \wedge \neg R$.

5 Canonical logical expression

This section considers the statement deduction.

Lemma 1 A logical statement with r variables $\sigma(A_1, \cdots, A_r)$ can always be expressed as

$$\sigma(A_1, \cdots, A_k) = M_\sigma A_1^{p_1} A_2^{p_2} \cdots A_r^{p_r}, \tag{11}$$

where M_σ is a $2 \times 2^{p_1+\cdots+p_r}$ structure matrix.

Proof Using the matrix expression of each operator, a logical expression can always be expressed as

$$\sigma(A_1, \cdots, A_k) = M_{i_1} A_{i_1} \cdots M_{i_s} A_{i_s}, \tag{12}$$

where

$$A_{i_j} \in \{A_1, A_2, \cdots, A_k\}.$$

Since the STP of matrices is associative, (12) can be transformed as

$$\sigma(A_1, \cdots, A_k) = M A_{i_1} \cdots A_{i_s}. \tag{13}$$

We can prove the following fact: Let $A_i \in \mathbb{R}^n$, $i = 1, \cdots, s$. Define

$$P = \underbrace{I_n \otimes \cdots \otimes I_n}_{i-1} \otimes W_{[n]} \otimes \underbrace{I_n \otimes \cdots \otimes I_n}_{s-i-1},$$

then

$$P A_1 \cdots A_i A_{i+1} \cdots A_s = A_1 \cdots A_{i+1} A_i \cdots A_s. \tag{14}$$

Note that $P^{-1} = P$. Using it and (14), (13) can be easily converted into (11).

Lemma 2 Given a matrix expression of a logic statement as MA^2, where M is a $p \times 4q$ matrix. Spliting M as $M = \begin{bmatrix} M_1 & M_2 & M_3 & M_4 \end{bmatrix}$ with M_i as a $p \times q$ matrices, then

$$MA^2 = \begin{bmatrix} M_1 & M_4 \end{bmatrix} A. \tag{15}$$

Proof Using the property of semi-tensor product of matrices, it is easy to verify that (15) holds for both $A = (1, 0)^{\mathrm{T}}$ and $A = (0, 1)^{\mathrm{T}}$.

Motivated by Lemma 2, we define a matrix, called the power-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, which is an immediate consequence of Lemma 2.

Lemma 3 Let A be a logical variable. Then for any $p \times 4q$ matrix $\varPsi$

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

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

Theorem 3 Any logical expression $L(P_1, \cdots, P_r)$ with free logical variables $P_1, \cdots, P_r$ can be uniquely expressed in a canonical form as

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

where M_L is a 2×2^r structure matrix.

Proof Use Lemma 1 to convert the matrix expression as (11). Then, use Lemma 3 (i.e., equation (16)) to reduce the power of each variable to 1. Note that the dimension requirement for the STP is automatically satisfied. Finally, since the expression (18) is a bilinear form with respect to distinct variables, it should be unique.

In the following application, we try to use M_c, M_d, and M_n to express M_i and M_e.

Proposition 8 M_i and M_e can be expressed by M_c, M_d, and M_n respectively as

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

and

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

Proof Note that $P \to Q \Leftrightarrow \neg P \wedge Q$. Expressing it in matrix form, we have $M_i PQ = M_d(M_n P)Q = M_d M_n PQ$. (19) follows immediately. As for (20), we have $P \leftrightarrow Q \Leftrightarrow (P \to Q) \wedge (Q \to P)$. So we have

$$M_e PQ = M_c(M_i PQ)(M_i QP).$$

Now the right hand side is

$$\begin{aligned} & M_c M_i PQ M_i QP \\ &= M_c M_d M_n (I_4 \otimes M_d M_n) PQQP \\ &= M_c M_d M_n (I_4 \otimes M_d M_n) P (M_r) QP \\ &= M_c M_d M_n (I_4 \otimes M_d M_n)(I_2 \otimes M_r) P W_{[2]} PQ \\ &= M_c M_d M_n (I_4 \otimes M_d M_n)(I_2 \otimes M_r)(I_2 \otimes W_{[2]}) PPQ \\ &= M_c M_d M_n (I_4 \otimes M_d M_n)(I_2 \otimes M_r)(I_2 \otimes W_{[2]}) M_r PQ. \end{aligned}$$

(20) is proved.

Remark 1 From (19), (20), one can easily verify that the structure matrices of all binary operators can be expressed by M_c, M_d, and M_n. This is not surprising because $\{\vee, \wedge, \neg\}$ is an adequate set. But later on, they can be used to define corresponding operators for the multi-valued and mix-valued cases.

Next, we use the following example to show the application of Theorem 3.

Example 3 Person A said that person B is a liar, person B said person C is a liar, and person C said that both persons A and B are liars. Who is a liar ?

Denote A: person A is honest; B: person B is honest; and C: person C is honest. Then the logical expression is

$$(A \leftrightarrow \neg B) \wedge (B \leftrightarrow \neg C) \wedge (C \leftrightarrow \neg A \wedge \neg B). \tag{21}$$

The problem becomes finding when (21) is true. The matrix expression of (21) is

$$\begin{aligned}
&M_c^2 M_e A M_n B M_e B M_n C M_e C M_c M_n A M_n B\\
&= \begin{bmatrix}1&0&0&0\\0&1&1&1\end{bmatrix}^2 \begin{bmatrix}1&0&0&1\\0&1&1&0\end{bmatrix} A \begin{bmatrix}0&1\\1&0\end{bmatrix} B\\
&\cdot \begin{bmatrix}1&0&0&1\\0&1&1&0\end{bmatrix} B \begin{bmatrix}0&1\\1&0\end{bmatrix} C\\
&\ltimes \begin{bmatrix}1&0&0&1\\0&1&1&0\end{bmatrix} C \begin{bmatrix}1&0&0&0\\0&1&1&1\end{bmatrix} \begin{bmatrix}0&1\\1&0\end{bmatrix} A \begin{bmatrix}0&1\\1&0\end{bmatrix} B. \qquad (22)
\end{aligned}$$

Now we have to reduce it to the normal form as (18). A straightforward computation shows that (22) equals to

$$\begin{bmatrix}0&0&0&0&0&1&0&0\\1&1&1&1&1&0&1&1\end{bmatrix} ABC. \qquad (23)$$

(23) is the canonical form of (22). To make (23) true (i.e., (23) $= (1,0)^{\mathrm{T}}$) the only solution is

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

So person A and person C are liars and person B is honest.

The following two formulas are used to reduce the complexity of computation:

Proposition 9

$$\begin{cases} W_{[n,n^k]} = \prod\limits_{i=0}^{k-1} I_{n^{k-1-i}} \otimes W_{[n]} \otimes I_{2^i},\\ W_{[n^k,n]} = \prod\limits_{i=k-1}^{0} I_{n^{k-1-i}} \otimes W_{[n]} \otimes I_{2^i}. \end{cases} \qquad (24)$$

6 Multi-valued logic

One of the advantages of the matrix form of logic is that it can easily be extended to multi-valued logic. Assume P and Q are two k-valued logical variables, i.e., they take values from D_k ($k \geqslant 2$). The first generalization is, using scale form, we define some basic operations as:

Definition 6 Let P and Q be two k-valued logical variables. Then their disjunction is defined as

$$P \vee Q = \max(P, Q); \qquad (25)$$

their conjunction is defined as

$$P \wedge Q = \min(P, Q). \qquad (26)$$

The negation is defined as

$$\neg P = 1 - P. \qquad (27)$$

To use the matrix expression, we have to give the logical values a vector form, which is the counterpart of (4). We define

$$\frac{i}{k-1} \equiv d^k_{k-i},\ i = 0, 1, \cdots, k-1, \qquad (28)$$

where d^k_j is the j-th column of the identity matrix I_k. Precisely, we have

$$1 \equiv \begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix};\ \frac{k-2}{k-1} \equiv \begin{bmatrix}0\\1\\\vdots\\0\end{bmatrix};\ \cdots;\ 0 \equiv \begin{bmatrix}0\\0\\\vdots\\1\end{bmatrix}.$$

Using vector form of logic values, the matrices of the operators defined in Definition 6 can be easily calculated.

For notational ease, in most places of this section, we assume $k = 3$. All the arguments are applicable to any $k \geqslant 2$.

Proposition 10 Assume $k = 3$. Then the structure matrices for disjunction, conjunction and negation are

$$M_d^3 = \begin{bmatrix}1&1&1&1&0&0&1&0&0\\0&0&0&0&1&1&0&1&0\\0&0&0&0&0&0&0&0&1\end{bmatrix}; \qquad (29)$$

$$M_c^3 = \begin{bmatrix}1&0&0&0&0&0&0&0&0\\0&1&0&1&1&0&0&0&0\\0&0&1&0&0&1&1&1&1\end{bmatrix}; \qquad (30)$$

$$M_n^3 = \begin{bmatrix}0&0&1\\0&1&0\\1&0&0\end{bmatrix}. \qquad (31)$$

Definition 6 is a natural generalization of the classical logic and widely used. However, the implication can be defined in various ways. Hence, there are many different 3-valued logics. For instance, three known 3-valued logics, Kleene-Dienes-type (KD), Luekasiewic-type (L), Bochvar-type (B) are listed as follows ($T = 1$, $U = 0.5$, $F = 0$) [4].

Table 4 3-valued logics.

		KD		L		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 logical operators via their structure matrices.

Now let us use (19) to define the implication for $k = 3$. It is easy to calculate that

$$M_i^3 = \begin{bmatrix}1&0&0&1&0&0&1&1&1\\0&1&0&0&1&1&0&0&0\\0&0&1&0&0&0&0&0&0\end{bmatrix}. \qquad (32)$$

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

$$M_r^k = \begin{bmatrix} d_1^k & 0 & \cdots & 0 \\ 0 & d_2^k & \cdots & 0 \\ 0 & 0 & & 0 \\ 0 & 0 & \cdots & d_s^k \end{bmatrix}. \tag{33}$$

Note that using the same argument one sees easily that k-valued logic (20) becomes

$$M_e = M_c M_i (I_{k^2} \otimes M_i)(I_k \otimes M_r)(I_k \otimes W_{[k]}) M_r. \tag{34}$$

Using this formula, we can calculate that

$$M_e^3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}. \tag{35}$$

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

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

Example 4 [4] A detective has the following clues for a murder:

1) 80% sure that either A or B is the 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

$$\begin{cases} A \vee B = (0,0,1,0,0,0)^{\mathrm{T}}, \\ A \rightarrow \neg M = (0,1,0,0,0,0)^{\mathrm{T}}, \\ S \rightarrow L = (1,0,0,0,0,0)^{\mathrm{T}}, \\ \neg S \rightarrow M = (0,1,0,0,0,0)^{\mathrm{T}}, \\ \neg L = (1,0,0,0,0,0)^{\mathrm{T}}. \end{cases} \tag{36}$$

We use the matrix expression to perform the fuzzy logical inference. First, from $\neg L = (1,0,0,0,0,0)^{\mathrm{T}}$ we have $L = (0,0,0,0,0,1)^{\mathrm{T}}$. Then from $S \rightarrow L = (1,0,0,0,0,0)^{\mathrm{T}}$ we have the matrix form as $M_i^6 SL = M_i^6 W_{[6]} LS := \varPsi_1 S = (1,0,0,0,0,0)^{\mathrm{T}}$. It is easy to calculate that

$$\varPsi_1 = M_i^6 W_{[6]} L = \begin{bmatrix} 0&0&0&0&0&1 \\ 0&0&0&0&1&0 \\ 0&0&0&1&0&0 \\ 0&0&1&0&0&0 \\ 0&1&0&0&0&0 \\ 1&0&0&0&0&0 \end{bmatrix}.$$

Then S can be solved as $S = (0,0,0,0,0,1)^{\mathrm{T}}$. Similarly, from $\neg S \rightarrow M = (0,1,0,0,0,0)^{\mathrm{T}}$ we have $M_i^6 M_n^6 SM = (0\ 1\ 0\ 0\ 0\ 0)^{\mathrm{T}}$. Then M can be solved as

$$M = (0,1,0,0,0,0)^{\mathrm{T}}.$$

Consider $A \rightarrow \neg M = M_i^6 A M_n^6 M = (0,1,0,0,0,0)^{\mathrm{T}}$. Using some properties of semi-tensor product, we have

$$\begin{aligned} M_i^6 A M_n^6 M &= M_i^6 (I_6 \otimes M_n^6) AM \\ &= M_i^6 (I_6 \otimes M_n^6) W_{[6]} MA \\ &:= \psi_2 A. \end{aligned}$$

Since ψ_2 is easily computable, then A can be solved as $A = (0,0,0,0,1,0)^{\mathrm{T}}$. Finally, from $A \vee B = M_d AB = (0,0,1,0,0,0)^{\mathrm{T}}$ we can solve $B = (0,0,1,0,0,0)^{\mathrm{T}}$. We conclude that "very unlikely" that A is the killer, and 80% possibly B is the killer.

Comparing our inference with it in [4], one sees that [4] created several un-axiomatic artificial rules for fuzzy logical inference and then used them in the previous example. However, we do not need any of them and obtained the same conclusion.

Next, we consider the canonical form of a logic expression. Extracting the same argument shows that Theorem 3 remains true. Precisely,

Theorem 4 Any k-valued logical expression $L(P_1, \cdots, P_r)$ with free logical variables $P_1, \cdots, P_r$ can be uniquely expressed in a canonical form as

$$L(P_1, \cdots, P_r) = M_L P_1 P_2 \cdots P_r, \tag{37}$$

where M_L is a $k \times k^r$ structure matrix.

Finally, we mentioned before that multi-valued logics (25)$\sim$(27) and (34) are commonly used. However, the implication has various forms. They are useful in some cases, emphasizing certain logical properties. So far, for simplicity we use only (19). We list some others [10] with their structure matrices for $k = 3$. They are useful for fuzzy control, etc.

· Zadeh: $A \to B = \neg A \vee (A \wedge B)$,

$$M_i^{\mathrm{Z}} = \begin{bmatrix} 1&0&0&0&0&0&1&1&1 \\ 0&1&0&1&1&1&0&0&0 \\ 0&0&1&0&0&0&0&0&0 \end{bmatrix};$$

· Lukasiewicz: $A \to B = (\neg A + B) \wedge T_0$,

$$M_i^{\mathrm{L}} = \begin{bmatrix} 1&0&0&1&1&0&1&1&1 \\ 0&1&1&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \end{bmatrix};$$

· Mamdani: $A \to B = A \wedge B$,

$$M_i^{\mathrm{M}} = M_c^3, \text{ in } (30);$$

· Gaines-Rescher: $A \to B = \begin{cases} T_0, & A \leqslant B, \\ 0, & \text{otherwise}, \end{cases}$

$$M_i^{\mathrm{GR}} = \begin{bmatrix} 1&0&0&1&1&0&1&1&1 \\ 0&0&0&0&0&0&0&0&0 \\ 0&1&1&0&0&1&0&0&0 \end{bmatrix};$$

· Gödel: $A \to B = \begin{cases} T_0, & A \leqslant B, \\ B, & \text{otherwise}, \end{cases}$

$$M_i^{\mathrm{Gl}} = \begin{bmatrix} 1&0&0&1&1&0&1&1&1 \\ 0&1&0&0&0&1&0&0&0 \\ 0&0&1&0&0&0&0&0&0 \end{bmatrix};$$

· Kleene-Dienes: $A \to B = \neg A \vee B$,

$$M_i^{\mathrm{KD}} = M_i^3, \text{ in } (32).$$

This is what we use throughout the paper.

· Wang: $A \to B = \begin{cases} T_0, & A \leqslant B, \\ \neg A \vee B, & \text{otherwise}, \end{cases}$

$$M_i^{\mathrm{W}} = \begin{bmatrix} 1&0&0&1&1&0&1&1&1 \\ 0&1&0&0&0&0&0&0&0 \\ 0&0&1&0&0&1&0&0&0 \end{bmatrix}.$$

Among these implications only M_i^{Z}, M_i^{M}, and M_i^{KD} involve standard logical operations. M_i^{KD} is what we used mostly in this paper; $M_i^{\mathrm{M}} = M_c$. As for M_i^{Z}, it is easy to prove that

$$M_i^{\mathrm{Z}} = M_d M_n (I_k \otimes M_c) M_r. \tag{38}$$

For $k > 3$, the corresponding structure matrices of implication can also be easily figured out. Then all other binary operators can be defined via their structure matrices.

Next, we extend all the binary operators to k-valued logic. They can be done via corresponding structure matrices. Of course, we need the operator "negation", which is defined in (27). Now, $\sigma_8 = \wedge$ and $\sigma_{14} = \vee$ have been defined in (26) and (25), respectively. Then for any k, their structure matrices can be calculated easily. The structure matrices for

$$\sigma_0 = F_0, \ \sigma_{15} = T_0, \ \sigma_3 = \neg_1,$$
$$\sigma_5 = \neg_2, \ \sigma_{10} = Q, \ \sigma_{12} = P$$

can be constructed easily.

Now, the implication σ_{11} is defined variously as in the above. Then for equivalence, M_e can be calculated by using (34). Finally, for EOR (σ_6), NAND (σ_7), NOR (σ_1), NIMP (σ_4), NIIMP (σ_2), ICOD (σ_{13}), their structure matrices can be calculated by their definitions respectively.

Now, all the binary operators are well defined so it is natural to ask such a question: which formulas in section 4 remain available? Of course, it depends on the definition of implication. Under some implications there are no tautologies (or $A \to A \nLeftrightarrow T_0$). In the following, we consider the basic properties of some multi-valued logics with tautologies.

Example 5 Consider Propositions 3, 5, and 6 for Gödel (Gl), Lukasiewicz (L), Wang (W), and Gaines-Reescher (GR). Using structure matrices, it is easy to check whether the expressions are correct (denoted by "T") or not (denoted by "F"):

Table 5 Check Proposition 3 for some multi-valued logics.

	1)	2)	3)	4)	5)	6)	7)	8)	9)	10)
Gl	T	T	T	T	T	F	F	F	T	F
L	T	T	T	T	T	F	F	T	F	F
W	T	T	T	T	T	F	F	T	F	F
GR	T	T	T	T	T	F	F	T	F	F

Table 6 Check Proposition 5 for some multi-valued logics.

	1)	2)	3)	4)	5)	6)	7)	8)
Gl	T	T	F	F	T	T	T	F
L	T	T	F	F	T	T	T	F
W	T	T	F	F	T	T	T	F
GR	T	T	F	F	T	T	T	F

Table 7 Check Proposition 6 for some multi-valued logics.

	1)	2)	3)	4)	5)	6)	7)	8)	9)	10)	11)	12)	13)
Gl	T	T	T	T	F	T	F	T	F	T	F	T	T
L	T	T	T	T	T	T	T	T	F	F	F	F	F
W	T	T	T	T	T	T	T	T	F	F	F	F	F
GR	T	T	T	T	F	F	F	F	F	T	T	T	T

7 Mix-valued logic

In fuzzy control, it happens frequently that the variables can take different numbers of values. Formally, in a problem, the logical variables, $P_i, i = 1, \cdots, s$, may take values from D_{k_i}, where k_i may differ from each other. We give a precise definition:

Definition 7 A mix-valued logic is a quartet: $\mathcal{L}_m := \{D_{k_1} \bigcup \cdots \bigcup D_{k_s},\ \wedge,\ \vee,\ \neg\}$. A logical variable P in this logic can take values from $D_{k_1} \bigcup \cdots \bigcup D_{k_s}$. The operations $\wedge$, $\vee$, $\neg$ are defined as in (26), (25), and (27) respectively.

It is easy to see that a logical variable in this logic may take a value from D_ℓ, where ℓ may differ from all k_i, $i = 1, \cdots, s$. Since $\{0, 1\}$ are assumed to be common elements in $D_k, \forall\, k$, we have

$$\min\{k_i \,|\, 1 \leqslant i \leqslant s\} \leqslant \ell \leqslant \sum_{i=1}^{s} k_i - 2s + 2. \tag{39}$$

Note that since $D_{k_1} \bigcup \cdots \bigcup D_{k_s} \subset D_f$, the following is obvious.

Proposition 11 $\mathcal{L}_m$ is a sub-algebra of the Morgan algebra $\{D_f,\ \wedge,\ \vee,\ \neg\}$.

From Proposition 11, all the results in fuzzy logic can be used for mix-valued logic $\mathcal{L}_m$ without any further requirement.

However, a significant advantage of mix-valued logic is that the matrix approach is available. Therefore, it is convenient in both theoretical analysis and numerical realization. The rest of this section is devoted to the matrix expression of the operators in $\mathcal{L}_m$.

First, we consider the structure matrix of binary operators.

Definition 8 Let $P \in D_k$, $Q \in D_\mu$. A binary operator σ is a mapping $D_k \times D_\mu \to D_k \bigcup D_\mu$, defined uniquely by its structure matrix, M_σ.

We give an example to describe it.

Example 6 Assume $P \in D_3$ and $Q \in D_4$. Then

$$\begin{aligned} D_3 \bigcup D_4 &= \{0, 1/2, 1\} \bigcup \{0, 1/3, 2/3, 1\} \\ &= \{0, 1/3, 1/2, 2/3, 1\} \simeq D_5. \end{aligned}$$

Note that in fuzzy logic, the real values in D_k are meaningless; only the order of the entries is important. Therefore, we can convert entries into their vector form by the order only. Then we can construct the structure matrix of disjunction, $\vee$, denoted by $M_d^{(3,4)}$, as

$$M_d^{(3,4)} = \begin{bmatrix} 1&1&1&1&1&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1 \end{bmatrix}. \tag{40}$$

Similarly, the structure matrix of conjunction, $\wedge$, denoted by $M_c^{(3,4)}$, is

$$M_c^{(3,4)} = \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&1&1&1&1&1 \end{bmatrix}. \tag{41}$$

Using the same technique, the structure matrices of all binary operators (and hence the operator itself) can be defined. Say, the structure matrix of implication, $\rightarrow$, (in KD sense), denoted by $M_i^{(3,4)}$, is

$$M_i^{(3,4)} = M_d^{(3,4)} M_n^3 = \begin{bmatrix} 1&0&0&0&1&0&0&0&1&1&1&1 \\ 0&1&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&1&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&0&0 \end{bmatrix}. \tag{42}$$

A binary operator $\sigma : D_k \times D_\mu \to D_\ell$ is symmetric if

$$P\sigma Q = Q\sigma P, \ \text{ or } \ \sigma(P, Q) = \sigma(Q, P).$$

To check the symmetry of an operator, we have

Proposition 12 A binary operator $\sigma : D_k \times D_\mu \to D_\ell$ is symmetric, iff its structure matrix satisfies

$$M_\sigma^{(k,\mu)} W_{[\mu,k]} = M_\sigma^{(\mu,k)}. \tag{43}$$

Proof Let $P \in D_k$ and $Q \in D_\mu$. Symmetry means

$$M_\sigma^{(k,\mu)} PQ = M_\sigma^{(\mu,k)} QP.$$

Since

$$LHS = M_\sigma^{(k,\mu)} W_{[\mu,k]} W_{[k,\mu]} PQ = M_\sigma^{(k,\mu)} W_{[\mu,k]} QP,$$

the conclusion follows.

The canonical expression form remains true for mix-valued logic:

Theorem 5 Let $P_i \in D_{k_i}$, $i = 1, \cdots, r$ be a set of multi-valued logic variables. $\bigcup\limits_{i=1}^{r} D_{k_i} \simeq D_\ell$. Then any logic expression $L(P_1, \cdots, P_r)$ can be uniquely expressed as

$$L(P_1, \cdots, P_r) = M_L P_1 P_2 \cdots P_r, \tag{44}$$

where M_L is a unique $\ell \times \prod\limits_{i=1}^{r} k_i$ matrix, called the structure matrix of L.

8 Logic-based fuzzy control

Review a fuzzy control system [5, 11], and one sees easily that what was really used is mix-valued logic, including multi-valued logic as its particular case.

A fuzzy (control) system is described in Fig.1, where $r(t)$ is a reference input; $u(t) \in U$ is the control; $y(t) \in Y$ is the output; FC is the fuzzy controller; F, LBR, and DF are fuzzification, logic-based rule, and defuzzification, respectively.

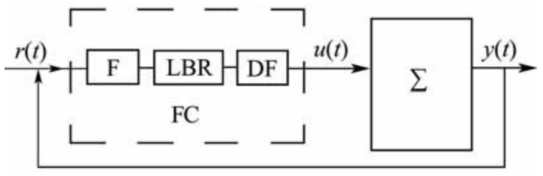

Fig. 1 A fuzzy control system.

The goal of a fuzzy control is to get control $u(y)$ via fuzzy logic-based rule. We give a formal description as:

· Output Space: Y (e.g., $Y \subset \mathbb{R}^p$);

· Input (Control) Space: U (e.g., $U \subset \mathbb{R}^m$);

· Linguistic Propositions (w.r.t Y and U respectively):

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

· Corresponding membership functions:

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

· Linguistic Rules:

If $A^1_1, \cdots, A^1_{s_1}$ satisfy R_1, then B_1;

$\vdots$

If $A^t_1, \cdots, A^t_{s_t}$ satisfy R_t, then B_t, where $A^i_j \in \mathcal{A}$;

· Corresponding to R_i, the logical expression is L_i, $i = 1, \cdots, t$.

Linguistic Rules are expressed in logical expression as

$$\begin{cases} L_1(A^1_1, \cdots, A^1_{s_1}) \to B_1, \\ \vdots \\ L_t(A^t_1, \cdots, A^t_{s_t}) \to B_t. \end{cases} \tag{45}$$

Using Mandani implication (which is commonly used in fuzzy control), the membership degree for i-th Rule is

$$\mu_{R_i} = \mu_{A^i_1}(y) \wedge \cdots \wedge \mu_{A^i_{s_i}}(y) \wedge \mu_{B_i}(u),\ i = 1, \cdots, t. \tag{46}$$

Then a defuzzification method is used to convert decisions into actions. Say, center of gravity (COG) is used, then

$$u^{\text{crisp}} = \sum_{i=1}^{t} B_i \int \mu(i) / \sum_{i=1}^{t} \int \mu(i). \tag{47}$$

Or using center-average (CA), we have

$$u^{\text{crisp}} = \sum_{i=1}^{t} B_i \mu_{R_i} / \sum_{i=1}^{t} \mu_{R_i}. \tag{48}$$

In most fuzzy control problems, (45) is replaced by a lookup rule table. In fact, such rules do not reveal the input-output logical connection. Therefore, it is essentially an experience-based fuzzy control.

The logic-based fuzzy control, to be proposed in the sequel, is to realize (45) by a mix-valued logical expression. The expression is defined on $D_s \times D_t$, and will produce feedback control u from y directly.

We use a simple example to show how convenient and powerful the logic-based fuzzy control is over the experience-based fuzzy control.

Example 7 [5] 23∼47 Consider an inverted pendulum on a cart. Let $e(t)$, $\dot{e}(t)$, and $u(t)$ be error, change-in-error, and force, respectively. The input membership functions are as in Fig.2 (note that e and $\dot{e}$ have the same membership function with different unions), and the output membership function is as in Fig. 3.

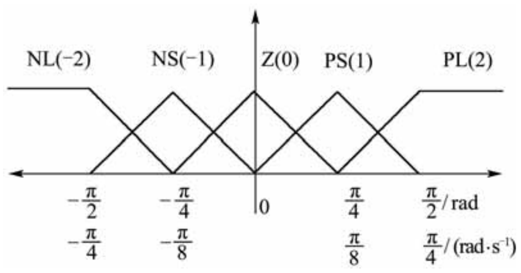

Fig. 2 Input membership functions.

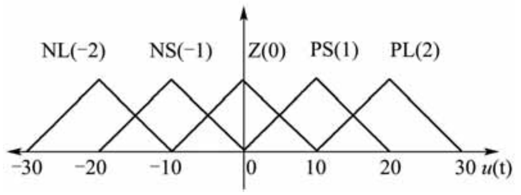

Fig. 3 Output membership function.

The rule table used in [5] is as in Table 7.

Table 8 Rule table for inverted pendulum. u

e	$\dot{e}$: −2	−1	0	1	2
−2	2	2	2	1	0
−1	2	2	1	0	−1
0	2	1	0	−1	−2
1	1	0	−1	−2	−2
2	0	−1	−2	−2	−2

Note that from Table 7 one sees easily that, roughly speaking, $u(t)$ is logically related to $e(t) + \dot{e}(t)$. Formally, we normalize (re-scaling) e and $\dot{e}$ as $E = 4e/\pi,\ \dot{E} = 8\dot{e}/\pi$. Then we use A_i and B_i, $i = 1, 2, 3, 4, 5$ for $E + \dot{E}$ and $u(t)$ to be NL, NS, Z, PS, PL respectively. Then $\mathcal{A}, \mathcal{B} \in D_5$. The logical rule of (45) can be easily obtained as

$$\mathcal{B} = \neg \mathcal{A}. \tag{49}$$

In the following, let us see how to design a logic-based fuzzy control.

Using classical membership function for $E + \dot{E}$, we have

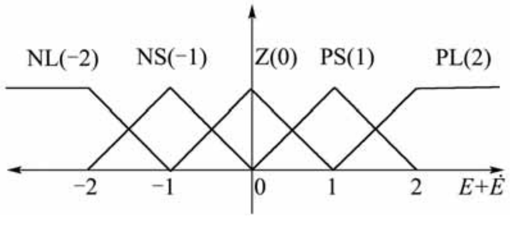

Fig. 4 Membership function of $E + \dot{E}$.

In Table we choose some particular values of e and $\dot{e}$ to compare the controls $u(t)$ with experience-based (E-B) and logic-based (L-B) approaches, respectively.

Table 9 Comparing fuzzy controls via experience-based approach vs logic-based approach. u

e		$-3\pi/8$			0			$5\pi/16$		
$\dot{e}$		$-\pi/32$	$3\pi/32$	$9\pi/32$	$-\pi/32$	$3\pi/32$	$9\pi/32$	$-\pi/32$	$3\pi/32$	$9\pi/32$
COG	E-B	16.84	3.18	−10	8.68	−6.82	−18.06	−5	−20	−20
	L-B	16.67	2.5	−10	8.33	−7.5	−18.33	−5	−20	−20
CA	E-B	16.82	6.82	−6.82	3.18	−6.82	−20	−10	−20	−20
	L-B	17.5	7.5	−7.5	2.5	−7.5	−20	−10	−20	−20

Remark 2 1) From Example 7, one can see easily that designing logic-based fuzzy control requires less computation. 2) The method has wide applicability, e.g., most of the examples in [5] can be treated in this way. 3) Obviously, the logic-based approach can be used for more general logical relations, where the experience-based one is not applicable.

9 Conclusions

A new matrix product, called the semi-tensor product, was introduced for the matrix expression of logical operations. It was proved that it is very convenient in performing logical operators, simplifying logical expressions, proving formulas, etc. Then the method was extended to multi-valued logic. Through some examples, we show that the matrix expression is very convenient in (fuzzy) logical inference, because it converts the problem to solving linear algebraic equations. Finally, to meet the requirement in fuzzy control, we introduced a new type of logic, called the mix-valued logic. Certain properties have been investigated via its matrix expression. A framework for logic-based fuzzy control was introduced and compared with traditional experience-based fuzzy control.

References

[1] A. G. Hamilton. *Logic for Mathematicians*[M]. Revised ed. Cambridge: Cambridge University Press, 1988.

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

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

[4] Z. Liu, Y. Liu. *Fuzzy Logic and Neural Network*[M]. Beijing: Beihang Press, 1996 (in Chinese).

[5] K. M. Passino, S. Yurkovich. *Fuzzy Control*[M]. California: Addison Wesley Longman, Inc., 1998.

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

[7] D. Cheng, Y. Dong. Semi-tensor product of matrices and its applications to physics[J]. *Methods and Applications of Analysis*, 2003, 10(4): 565 –588.

[8] D. Cheng, X. Hu, Y. Wang. Non-regular feedback linearization of nonlinear systems via a normal form algorithm[J]. *Automatica*, 2004, 40(3): 439 – 447.

[9] L. Rade, B. Westergren. *Mathematics Handbook for Science and Engineering*[M]. 4th ed. New York: Springer, 1998.

[10] G. Wang. *Non-Classical Mathematical Logic and Fuzzy Reasoning*[M]. Beijing: Science Press, 2000 (in Chinese).

[11] R. Babuska. An overview of fuzzy modeling and model-based fuzzy control[M]//*Fuzzy Logic Control, Advances in Applications*. Singapore: World Scientific Publishing, 1999.

Hongsheng QI was born in 1982 in Anhui, China. He received the B.S. degree from Anhui University in 2003. He is currently a Ph.D. student in Institute of Systems Science, Chinese Academy of Sciences. His research interests include nonlinear control, logic-based control.

Daizhan CHENG received his Ph.D. degree from Washington University, St. Louis, in 1985. He is currently a professor with Institute of Systems Science, Chinese Academy of Sciences, Chairman of Technical Committee on Control Theory (2003-), Chinese Association of Automation, Chairman of IEEE CSS Beijing Chapter, and IEEE Fellow. His research interests include nonlinear systems, numerical method, logic-based control etc.