Chapter 5
Matrix Expression of Logic

Mathematical logic is a discipline of both philosophy and natural science. From natural science point of view, it is produced by the efforts of mathematicians to reveal the essence of mathematical thinking and mathematical deductions. The basic concepts/results can be found in any standard textbooks, e.g., [4].

The purpose of this chapter is to express logical variables, logical operators, and logical equations into matrix forms by using semi-tensor product. We first introduce the matrix expression of logic. Using this form and the semi-tensor product, many fundamental properties of logic can be discovered. These results are then extended to the multi-valued logic and its operations. Finally, we consider some of its applications, including logical inference.

5.1 Logic and Its Expression

A logical variable means a proposition. Usually, a proposition can either be "true" or "false". When the proposition is true, we say that the logical variable takes value "T" or "1", and when it is false, the logical variable takes value "F" or "0". We consider some simple examples. We refer to any standard textbook on Mathematical Logic, e.g., [4], for proposition and basic concepts and properties of classical logic.

Example 5.1. Consider the following propositions.

A: A dog has 4 legs;
B: The snow is black;
C: There is another human in universe.

It is obvious that $A = 1$, $B = 0$. As for C, it could be either 1 or 0. But C should be one of them, though we still do not know the answer so far.

In classical logic a logical variable can only take values from $\{0,1\}$. But in real world a proposition may not be described precisely by only "true" or "false". For instance, “Mr. Smith is an old man”. If Mr. Smith is 20 years old, the statement is obviously “False”. If Mr. Smith is 80 years old, the statement is surely “True”. But if this person is 40 or 50 years old, then what can we say? It seems that we need some values between 0 (“False”) and 1 (“True”) to describe this statement, and hence the classical logic is not enough for analyzing such problems. Fuzzy logic allows a logical variable to take any value from internal $[0,1]$.

Usually, we use a membership function to describe the value of a fuzzy logical variable. For instance, we may use the following membership functions to describe the statement x : “Somebody is old”.

$$f(x)=\begin{cases}0, & x\le 20\\ 0.01(x-20), & 20<x\le 40\\ 0.2+0.04(x-40), & 40<x\le 60\\ 0.8+0.01(x-60), & 60<x\le 80\\ 1, & x>80.\end{cases} \tag{5.1}$$

This function is depicted in Fig. 5.1. We refer to [5] for more details about fuzzy logic. Its applications in fuzzy control systems can be found in [10], [8], [9].

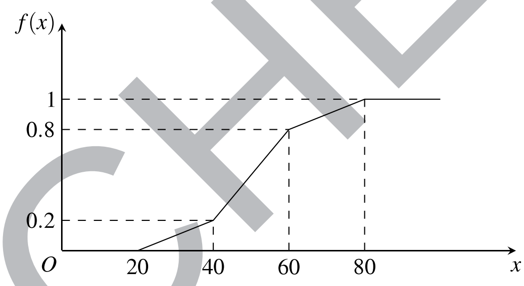

Fig. 5.1 Membership function of x

In classical logic a logical variable can take only two different values 0 or 1, while in fuzzy logic a logical variables can take continuous values between 0 and 1. It is obvious that using continuous values can describe a logical statement more precise than the classical two value case. But in many cases it may be too diverse and complicated to consider the continuous logic values. For instance, consider the statement x : “Mr. Smith is an old man”. It is hard to tell what is the difference between $x=0.41$ and $x=0.42$. Hence, a precise value may not have much sense when it is used to describe a proposition. Then we may consider to quantize the continuous membership function. For instance, in the age problem, we may classify different ages into three categories: “young”, “middle aged”, and “old” and use “0”, “0.5, and “1” for them respectively. A quantized membership function of $f(x)$ in (5.1) becomes

$$q(x) = \begin{cases} 0, & x \leq 40 \\ 0.5, & 40 < x \leq 60 \\ 1, & x > 60, \end{cases} \tag{5.2}$$

which is depicted in Fig. 5.2.

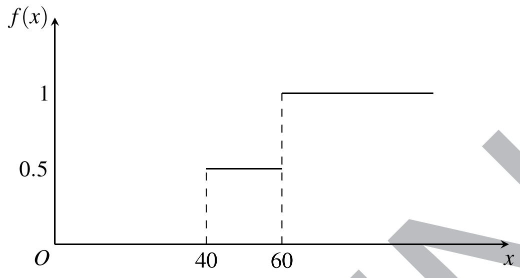

Fig. 5.2 Quantized membership function of x

In general, a logic, where a logical variable can take k different values between 0 and 1 is called the k-valued logic. When $k = 2$ it is classical logic, and when $k > 2$ it is called a multi-valued logic. Readers who are interested in multi-valued logic may refer to [3].

Definition 5.1. 1. The domain of (classical) logic is denoted by

$$\mathscr{D} := \{T = 1, F = 0\}. \tag{5.3}$$

A logical variable x takes value from $\mathscr{D}$, that is $x \in \mathscr{D}$.

2. The domain of k-valued logic is denoted by

$$\mathscr{D}_k := \left\{ T = 1, \frac{k-2}{k-1}, \frac{k-3}{k-1}, \cdots, \frac{1}{k-1}, F = 0 \right\}. \tag{5.4}$$

A k-valued logical variable x takes value from $\mathscr{D}_k$, that is $x \in \mathscr{D}_k$.

3. The domain of fuzzy logic is denoted by

$$\mathscr{D}_f := [0, 1]. \tag{5.5}$$

A k-valued logical variable x takes value from $\mathscr{D}_f$, that is $x \in [0, 1]$.

Next, we define the logical operators.

Definition 5.2. [1] An r-ary (multi-valued, fuzzy) logical operator is a mapping σ : $\underbrace{D \times D \times \cdots \times D}_{r} \to D$ (correspondingly, $\underbrace{D_k \times D_k \times \cdots \times D_k}_{r} \to D_k$, $\underbrace{D_f \times D_f \times \cdots \times D_f}_{r} \to D_f$).

An r-ary logical operator can also be called a logical function with r arguments. A classical logical function is also called a Boolean function. Mathematically, they are the same. But in applications they have a mild difference. Conventionally, "operator" is mostly used for $r = 1, 2$, and the operator has obvious logical meaning, such as "conjunction", "disjunction" etc. While, the function is used for more general case. Most likely, a logical function is composed of its arguments connected by operators. You may consider that a logical function is a "compounded function".

In the rest of this section we consider the classical logic only. We first introduce some fundamental operators.

(i) Negation: A unary logical operator, denoted by $\neg$. Negation is defined as

$$\neg x = \begin{cases} 0, & x = 1 \\ 1, & x = 0. \end{cases} \tag{5.6}$$

(ii) Conjunction: A binary logical operator, denoted by $\wedge$. Conjunction is defined as

$$x \wedge y = \begin{cases} 1, & x = 1 \text{ and } y = 1 \\ 0, & \text{otherwise.} \end{cases} \tag{5.7}$$

(iii) Disjunction: A binary logical operator, denoted by $\vee$. Disjunction is defined as

$$x \vee y = \begin{cases} 0, & x = 0 \text{ and } y = 0 \\ 1, & \text{otherwise.} \end{cases} \tag{5.8}$$

(iv) Conditional: A binary logical operator, denoted by $\rightarrow$. Conditional is defined as

$$x \rightarrow y = \begin{cases} 0, & x = 1 \text{ and } y = 0 \\ 1, & \text{otherwise.} \end{cases} \tag{5.9}$$

(v) Biconditional: A binary logical operator, denoted by $\leftrightarrow$. Biconditional is defined as

$$x \leftrightarrow y = \begin{cases} 1, & x = y \\ 0, & \text{otherwise.} \end{cases} \tag{5.10}$$

A conventional way to depict the values of an operator is using a table, called the truth table. For instance, for "negation", we have Table 5.1.

Table 5.1 Truth Table for "negation"

x	$\neg x$
1	0
0	1

Similarly, we can have the truth table for “conjunction”, “disjunction”, “conditional”, and “biconditional” etc. respectively as in Table 5.2.

Table 5.2 Truth Table for $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\bar{\nabla}$, $\uparrow$, $\downarrow$

x	y	$x\wedge y$	$x\vee y$	$x\rightarrow y$	$x\leftrightarrow y$	$x\bar{\nabla}y$	$x\uparrow y$	$x\downarrow y$
1	1	1	1	1	1	0	0	0
1	0	0	1	0	0	1	1	0
0	1	0	1	1	0	1	1	0
0	0	0	0	1	1	0	1	1

The truth value of a logical function can easily be obtained from the truth tables of basic connectives. We use an example to depict this.

Example 5.2. 1. Let $p = x\vee(\neg y)$. Then the truth table of x is shown in Table 5.3.
2. Let $q = (x\wedge y)\leftrightarrow(\neg z)$. Then the truth table of y is shown in Table 5.4.

Table 5.3 Truth Table for p

x	y	$\neg y$	$p = x\vee(\neg y)$
1	1	0	1
1	0	1	1
0	1	0	0
0	0	1	1

Table 5.4 Truth Table for q

x	y	z	$x\wedge y$	$\neg z$	$q = x\wedge y\leftrightarrow(\neg z)$
1	1	1	1	0	0
1	1	0	1	1	1
1	0	1	0	0	1
1	0	0	0	1	0
0	1	1	0	0	1
0	1	0	0	1	0
0	0	1	0	0	1
0	0	0	0	1	0

For statement ease, we define the vector truth table of a Boolean function.

Definition 5.3. Let $f(x_1,\cdots,x_k)$ be a Boolean function. Denote the column of f in its truth table by V_f, and call it the vector truth table of f.

Example 5.3. 1. Consider $f(x,y) = x\vee(\neg y)$. According to Table 5.3,

$$V_f = [1\ 1\ 0\ 1]^T.$$

2. Consider $g(x,y,z) = x \wedge y \leftrightarrow (\neg z)$. According to Table 5.4,

$$V_g = [0\ 1\ 1\ 0\ 1\ 0\ 1\ 0]^T.$$

We introduce some notations:

(i)

$$\delta_n^i := \mathrm{Col}_i(I_n), \quad i = 1, \cdots, n.$$

(ii)

$$\Delta_n := \mathrm{Col}(I_n), \text{ when } n = 2,\ \Delta := \Delta_2.$$

(iii) $L \in \mathscr{M}_{m\times n}$ is called a logical matrix if $\mathrm{Col}(L) \subset \Delta_m$. The set of $m \times n$ logical matrices is denoted by $\mathscr{L}_{m\times n}$.

(vi) Let $L \in \mathscr{L}_{m\times n}$. Then L can be expressed as

$$L = \left[\delta_m^{i_1}\ \delta_m^{i_2}\ \cdots\ \delta_m^{i_n}\right].$$

For notational compactness, we denote L briefly as

$$L = \delta_m[i_1\ i_2\ \cdots\ i_n].$$

To use matrix expression of logic, we identify

$$T = 1 \sim \delta_2^1, \quad F = 0 \sim \delta_2^2,$$

and call it the vector form of logic. Then in vector form an r-ary Boolean function f becomes a mapping $f : \Delta^r \to \Delta$.

Definition 5.4. Let $f(x_1, \cdots, x_r)$ be an r-ary Boolean function. $L_f \in \mathscr{L}_{2\times 2^r}$ is called the structure matrix of f, if in vector form we have

$$f(x_1, \cdots, x_r) = L_f \ltimes_{i=1}^r x_i. \tag{5.11}$$

Proposition 5.1. *Let $f(x_1, \cdots, x_r)$ be an r-ary Boolean function. Then there exists a unique structure matrix $L_f \in \mathscr{L}_{2\times 2^r}$ such that (5.11) holds.*

Proof. Assume the vector truth table of f is V_f. Construct L_f as follows:

$$\mathrm{Row}_1(L_f) := V_f^T; \quad \mathrm{Row}_2 = \neg(\mathrm{Row}_1).$$

Here we use $\neg\mathrm{Row}_1$ for taking negation on every elements of Row_1. It follows from the construction of truth table and the definition of semi-tensor product that the constructed L_f satisfies (5.11) and such structure matrix must be unique. □

Using Proposition 5.1, the structure matrices of some fundamental operators are obtained as

$$\begin{array}{l} M_\neg := M_n = \delta_2[2\ 1]; \\ M_\vee := M_d = \delta_2[1\ 1\ 1\ 2];\ M_\wedge := M_c = \delta_2[1\ 2\ 2\ 2]; \\ M_\rightarrow := M_i = \delta_2[1\ 2\ 1\ 1];\ M_\leftrightarrow := M_e = \delta_2[1\ 2\ 2\ 1]. \end{array} \tag{5.12}$$

Remark 5.1. 1. Note that the structure matrix of a logical function is unique. In fact, it is clear from the construction that there is a one-to-one correspondence between logical functions and the structure matrices. Hence determining a logical function is equivalent to determining a structure matrix.
2. From Proposition 5.1 one also sees that there is a one-to-one correspondence between vector truth tables and structure matrices.

In most applications it is not convenient to construct the structure matrix of a logical function by using its truth table. We give a method to construct it. To begin with, we need a tool, called the power-reducing matrix, which is defined as

$$M_r := \delta_4[1\ 4]. \tag{5.13}$$

The following lemma shows the power-reducing matrix can reduce the power of a logical variable. It can be proved by a straightforward computation.

Lemma 5.1. *Given a logical variable $x \in \Delta$. Then*

$$x^2 = M_r x. \tag{5.14}$$

Lemma 5.2. *A logical function $f(x_1, \cdots, x_k)$ can be expressed in vector form as*

$$f(x_1, \cdots, x_k) = L x_1^{n_1} \cdots x_k^{n_k}. \tag{5.15}$$

Proof. First, using the structure matrices of $\neg$, $\wedge$, $\vee$, etc., we can express the function as a product as

$$f(x_i, \cdots, x_k) = \ltimes_{j=1}^{r} \xi_j, \tag{5.16}$$

where ξ_j is either a structure of a unary or binary logical operator, or an argument x_i. Assume there are two factors as $x_i M_\sigma$, where x_i is an argument and M_σ is the structure matrix of operator σ. Using Theorem 2.10, we can swap two factors as

$$x_i M_\sigma = [I_2 \otimes M_\sigma] x_i.$$

Using this technique, we can move all the arguments to the rear of the product. Then use the swap matrix

$$x_i^p x_j^q = W_{[2^q, 2^p]} x_j^q x_i^p$$

we can re-arrange the order of arguments into the required order. □

We give a simple example to depict this:

Example 5.4. let $f(x,y) = (x \vee y) \to (x \wedge y)$. Then in vector form we have

$$\begin{aligned} f(x,y) &= M_i (x \vee y)(x \wedge y) \\ &= M_i M_d xy M_c xy \\ &= M_i M_d (I_4 \otimes M_c) xyxy \\ &= M_i M_d (I_4 \otimes M_c) x W_{[2]} xy^2 \\ &= M_i M_d (I_4 \otimes M_c) \left[I_2 \otimes W_{[2]} \right] x^2 y^2. \end{aligned}$$

Using Lemmas 5.1 and 5.2, we can give an alternative proof for Proposition 5.1. In fact, starting from (5.15), we can use (5.14) to reduce the powers of each x_i to 1. After some additional swaps, (5.11) can be obtained.

Example 5.5. (Continuing Example 5.4)

$$\begin{aligned} f(x,y) &= M_i M_d (I_4 \otimes M_c) \left[I_2 \otimes W_{[2]}\right] x^2 y^2 \\ &= M_i M_d (I_4 \otimes M_c) \left[I_2 \otimes W_{[2]}\right] M_r x M_r y \\ &= M_i M_d (I_4 \otimes M_c) \left[I_2 \otimes W_{[2]}\right] M_r \left[I_2 \otimes M_r\right] xy \end{aligned}$$

Finally, we conclude that

$$f(x,y) = (x \vee y) \to (x \wedge y) = Lxy,$$

where

$$\begin{aligned} L &= M_i M_d (I_4 \otimes M_c) \left[I_2 \otimes W_{[2]}\right] M_r \left[I_2 \otimes M_r\right] \\ &= \delta_2[1\ 2\ 2\ 1]. \end{aligned}$$

5.2 General Structure of Logical Operators

According to Proposition 5.1 and Remark 5.1, the number of r-ary logical functions is the same as the number of vector truth tables. It is obvious that if there are r variables and each variable can only take two possible values, then there are 2^r different value combinations of variables. Moreover, each variable value combination may correspond to two function values. Hence there are 2^{2^r} different vector truth tables. That is, there are 2^{2^r} different r-ary logical functions.

Remark 5.2. 1. Let $s < r$. Then an s-ary logical function can be considered as a special r-ary logical function, which is independent of $r-s$ logical variables. In constructing logical functions this observation should be taken into consideration.
2. Consider k-valued logic. The number of r-ary functions is k^{k^r}.

The following theorem is very useful in recovering the logical form of a logical function from its structure matrix.

Theorem 5.1. *Let $f(x_1,\cdots,x_k)$ be an r-ary logical function, with structure matrix $L_f \in \mathscr{L}_{2\times 2^r}$. Sprit L_f into two equal-size blocks as*

$$L_f = [L_f^1\ L_f^2].$$

Then $f(x_1,\cdots,x_k)$ can be expressed as

$$f(x_1,\cdots,x_r) = [x_1 \wedge f_1(x_2,\cdots,x_r)] \vee [\neg(x_1) \wedge f_2(x_2,\cdots,x_r)], \tag{5.17}$$

where f_i has L_f^i as its structure matrix, $i = 1,2$.

We leave the proof of this theorem to the reader.

Remark 5.3. Using (5.17) repetitively, we finally can express any logical function as compounded by $\neg$, $\wedge$, and $\vee$ with logical variables. This fact shows that the logical operators of ary 1 or 2 are of particular importance.

Next, we consider all the logical operators with ary $r = 1$ or $r = 2$.

Assume $r = 1$. In general, we have 4 logical operators, which are listed in Table 5.5.

Table 5.5 1-ary Operators

P	σ_0^1	σ_1^1	σ_2^1	σ_3^1
	F	$\neg$	$\equiv$	T
1	0	0	1	1
0	0	1	0	1

Here "F" is the constant "False" operator, and "T" is the constant "True" operator, "$\neg$" is the negation, and "$\equiv$" is the identity operator.

Their structure matrices are as follows.

$$M_F = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}; \quad M_n = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \quad M_{id} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \quad M_T = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}. \tag{5.18}$$

Next, we assume $r = 2$. In addition to the four operators ($\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$) the following 3 are also commonly used. (Refer to [6].)

(i) EOR (exclusive or), $x \bar{\vee} y$, it is true whenever either x or y, but not both are true;
(ii) NAND (not and), $x \uparrow y$, defined by $x \uparrow y = \neg(x \wedge y)$.
(iii) NOR (not or), $x \downarrow y$, defined by $x \downarrow y = \neg(x \vee y)$.

In addition to these 7 commonly used operators, we still have $2^{2^2} - 7 = 9$ other 2-ary operators. We listed these 16 operators in Table 5.6.

Here $\sigma_0^2 = F$, $\sigma_{15}^2 = T$ are two constant operators; $\sigma_3^2 = \neg x$, $\sigma_5^2 = \neg y$, $\sigma_{10}^2 = x$ and $\sigma_{12}^2 = y$ are essentially 1-ary operators. The following are some now 2-ary operators.

- EOR (σ_6^2, exclusive or) ($x \bar{\vee} y$):

$$x \bar{\vee} y = \neg(x \leftrightarrow y); \tag{5.19}$$

- NAND (σ_7^2, not and) ($x \uparrow y$):

$$x \uparrow y = \neg(x \wedge y); \tag{5.20}$$

- NOR (σ_1^2, not or) ($x \downarrow y$):

$$x \downarrow y = \neg(x \vee y); \tag{5.21}$$

Table 5.6 2-ary Operators

x	y	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	$\downarrow$	$-^*$	$\neg_1$	$-$	$\neg_2$	$\bar{\vee}$	$\uparrow$
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
x	y	σ_8^2	σ_9^2	σ_{10}^2	σ_{11}^2	σ_{12}^2	σ_{13}^2	σ_{14}^2	σ_{15}^2
		$\wedge$	$\leftrightarrow$	y	$\rightarrow$	x	$\rightarrow^*$	$\vee$	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

- NC (σ_4^2, not conditional) $(x - y)$:

$$x - y = \neg(x \rightarrow y); \tag{5.22}$$

- NINVC (σ_2^2, not inverse conditional) $(x -^* y)$:

$$x -^* y = \neg(y \rightarrow x); \tag{5.23}$$

- INVC (σ_{13}^2, inverse conditional) $x \rightarrow^* y$:

$$x \rightarrow^* y = y \rightarrow x. \tag{5.24}$$

The equivalence of (5.24) comes from its definition. The equivalences (5.19) – (5.23) will be proved later.

For convenience, we use σ_j^i to represent all the operators, where the superscript i means the ary of the operator, and the subscript j means the order of the operator. It is interesting that when the order, counting from zero, is converted to binary number, it is exactly the vector truth table of the operator. Hence the structure matrix of s_j^i can easily be constructed.

Example 5.6. Consider $\neg_2$, which is the negation of the second variable. Its alternative notation is σ_5^2. Since $5 = 101 = 0101$, we have

$$M_{\neg_2} = \delta_2[2\ 1\ 2\ 1].$$

5.3 Fundamental Properties of Logical Operators

According to the structure of truth tables, we have the following conjugate properties.

Proposition 5.2. *Given an r-ary operator σ_a^r, its negation operator is $\sigma_{2^{2^r}-a-1}^r$. That is,*

$$\neg\sigma_a^r(P,Q) = \sigma_{2^{2^r}-a-1}^r(P,Q). \tag{5.25}$$

Proof. Since $2^{2^r}-1 = \underbrace{1\ 1\ \cdots\ 1}_{2^r}$. Expressing a positive integer a into 2^r digital binary number (adding some zeros in the front if necessary), say, it is

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

Then $[a_1, a_2, \cdots, a_{2^r}]^{\mathrm{T}}$ is the vector truth table of σ_a^r. In binary form we have $2^{2^r} - 1 - a = [(1-a_1),(1-a_2),\cdots,(1-a_r)]$. Hence we know that the vector truth table of $\sigma_{2^{2^r}-a-1}^r$ is $(1-a_1, 1-a_2, \cdots, 1-a_r)^{\mathrm{T}}$. That implies (5.25). □

Remark 5.4. Using Proposition 5.2, we can prove (5.19) – (5.23) immediately.

Definition 5.5. 1. Two logical expressions are said to be logically equivalent, if for any particulary chosen values of logical variables from $\mathscr{D}$ the two structure matrix expressions have the same value. If $f(x_1,\cdots,x_k)$ and $g(x_1,\cdots,x_k)$ are logically equivalent, it is denoted as

$$f(x_1,\cdots,x_k) \Leftrightarrow g(x_1,\cdots,x_k).$$

2. Two logical expressions are said to be absolutely logically equivalent, if for any particulary chosen values of logical variables from $\mathscr{D}$ the two structure matrix expressions have the same value. If $f(x_1,\cdots,x_k)$ and $g(x_1,\cdots,x_k)$ are absolutely logically equivalent, it is denoted as

$$f(x_1,\cdots,x_k) \overset{\Leftrightarrow}{=} g(x_1,\cdots,x_k).$$

Proposition 5.3. *Assume two logical expressions f and g have same arguments, moreover, every arguments appear to f (g) precisely once. Then the logical equivalence of f and g is equivalent to absolutely logically equivalence.*

Proof. Assume f and g are logically equivalent. Recall the proof of Lemma 5.2, one sees easily that for $x_i \in \mathscr{D}_f$ (5.15) remains true. Denote the L in (5.15) for f and g by L_f and L_g respectively. Now by the assumption, $n_i = 1$, $i = 1,\cdots,k$. It follows that L_f and L_g are the structure matrices of f and g as $x_i \in \mathscr{D}$. Since f and g are logically equivalent, we have $L_f = L_g$. The conclusion follows. □

Using Proposition 5.3, we can have the following conclusions.

Proposition 5.4. *The following are absolutely logically equivalent.*

$$\neg\neg x \Leftrightarrow x; \tag{5.26}$$
$$(x \wedge y) \wedge z \Leftrightarrow x \wedge (y \wedge z); \tag{5.27}$$
$$(x \vee y) \vee z \Leftrightarrow z \vee (y \vee z); \tag{5.28}$$
$$\neg(x \wedge y) \Leftrightarrow \neg x \vee \neg y; \tag{5.29}$$
$$\neg(x \vee y) \Leftrightarrow \neg x \wedge \neg y; \tag{5.30}$$
$$x \to y \Leftrightarrow \neg x \vee y; \tag{5.31}$$
$$\neg(x \to y) \Leftrightarrow x \wedge \neg y; \tag{5.32}$$
$$x \to y \Leftrightarrow \neg y \to \neg x; \tag{5.33}$$
$$x \to (y \to z) \Leftrightarrow (x \wedge y) \to z; \tag{5.34}$$
$$\neg(x \leftrightarrow y) \Leftrightarrow x \leftrightarrow \neg y. \tag{5.35}$$

Proof. We prove (5.33) only. The proves of others are similar and we leave them to the reader. According to Proposition 5.3, we have only to prove they are logically equivalent.

$$\begin{aligned} RHS &= M_i M_n y M_n x = M_i M_n (I_2 \otimes M_n) yx \\ &= M_i M_n (I_2 \otimes M_n) W_{[2]} xy. \end{aligned}$$

Since

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

(5.33) follows. □

Definition 5.6. An r-ary operator is said to be symmetric if

$$M_\sigma P_1 P_2 \cdots P_k = M_\sigma P_{\lambda(1)} P_{\lambda(2)} \cdots P_{\lambda(k)}, \quad \forall \lambda \in \mathbf{S}_k. \tag{5.36}$$

Recall that $\mathbf{S}_k$ is the k-th order symmetric group.

Proposition 5.5. *A binary operator, σ is symmetric, iff in its vector truth table $V_\sigma = [s_1\ s_2\ s_3\ s_4]^{\mathrm{T}}$*

$$s_2 = s_3.$$

Proof. Note that the structure matrix of σ is

$$M_\sigma = \begin{bmatrix} s_1 & s_2 & s_3 & s_4 \\ 1-s_1 & 1-s_2 & 1-s_3 & 1-s_4 \end{bmatrix}.$$

Then

$$
\begin{aligned}
\sigma(x,y) =& M_\sigma xy = M_\sigma W_{[2]} yx \\
=& \begin{bmatrix} s_1 & s_3 & s_2 & s_4 \\ 1-s_1 & 1-s_3 & 1-s_2 & 1-s_4 \end{bmatrix} QP \\
=& M_\sigma yx = \sigma(y,x).
\end{aligned}
$$

□

Example 5.7. Consider the binary operators in Table 5.6. According to Proposition 5.5, we have that F, $\downarrow$, $\bar{\vee}$, $\uparrow$, $\wedge$, $\leftrightarrow$, $\vee$, and T are symmetric, and the others are not.

Proposition 5.6. *The followings are logically equivalent:*

$$x \vee x \Leftrightarrow x; \tag{5.37}$$

$$x \wedge x \Leftrightarrow x; \tag{5.38}$$

$$y \vee (x \wedge \neg x) \Leftrightarrow y; \tag{5.39}$$

$$y \wedge (x \vee \neg x) \Leftrightarrow y; \tag{5.40}$$

$$x \wedge (y \vee z) \Leftrightarrow (x \wedge y) \vee (x \wedge z); \tag{5.41}$$

$$x \vee (y \wedge z) \Leftrightarrow (x \vee y) \wedge (x \vee z); \tag{5.42}$$

$$x \leftrightarrow y \Leftrightarrow (x \to y) \wedge (y \to x); \tag{5.43}$$

$$x \leftrightarrow y \Leftrightarrow (x \wedge y) \vee (\neg x \wedge \neg y). \tag{5.44}$$

Proof. We prove (5.39) only. Assume $y = (\delta, 1-\delta)^{\mathrm{T}}$, $x = (\mu, 1-\mu)^{\mathrm{T}}$. Then

$$
\begin{aligned}
LHS =& M_\vee y M_\wedge x M_\neg x \\
=& \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta \\ 1-\delta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mu \\ 1-\mu \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \mu \\ 1-\mu \end{bmatrix} \\
=& \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta \\ 1-\delta \end{bmatrix} \begin{bmatrix} \mu(1-\mu) \\ \mu^2 + \mu(1-\mu) + (1-\mu)^2 \end{bmatrix} \\
=& \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta[\mu(1-\mu)] \\ \delta[\mu^2 + \mu(1-\mu) + (1-\mu)^2] \\ (1-\delta)[\mu(1-\mu)] \\ (1-\delta)[\mu^2 + \mu(1-\mu) + (1-\mu)^2] \end{bmatrix} \\
=& \begin{bmatrix} \delta + (1-\delta)\mu(1-\mu) \\ (1-\delta)[\mu^2 + \mu(1-\mu) + (1-\mu)^2] \end{bmatrix}
\end{aligned}
$$

Since $\mu \in \mathscr{D}$, we have $LHS = (\delta, 1-\delta)^{\mathrm{T}} = y$. □

It is easy to verify that (5.37)–(5.44) are not absolutely logically equivalent. For instance, consider (5.39), in the above proof taking $\mu = 0.5$, then $LHS = \begin{bmatrix} 0.25 + 0.75\delta \\ 0.75 - 0.75\delta \end{bmatrix} \neq y$.

The following proposition is very useful. We leave the proof to the reader.

Proposition 5.7 (De Morgan's Law).

1. $$\neg(x \wedge y) = (\neg x) \vee (\neg y). \tag{5.45}$$

2. $$\neg(x \vee y) = (\neg x) \wedge (\neg y). \tag{5.46}$$

Definition 5.7. 1. A logical expression is called a tautology, if it is always "true" no matter what values the arguments take.
2. A logical expression is called a contradiction, if it is always "false" no matter what values the arguments take.
3. Let L_1 and L_2 be two logical expressions. If $L_1 \to L_2$ is a tautology, then we say that L_1 tautologically implicates L_2, denoted by $L_1 \Rightarrow L_2$.

Proposition 5.8. *$L_1 \Rightarrow L_2$, if and only if, when $L_2 = \delta_2^2$, $L_1 = \delta_2^2$.*

Proof. (Sufficiency) Assume $L_2 = \delta_2^1$, and $L_1 = \begin{bmatrix} \alpha \\ 1-\alpha \end{bmatrix}$. Then

$$L_1 \to L_2 = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ 1-\alpha \end{bmatrix} \delta_2^1 = \delta_2^1.$$

Assume $L_2 = \delta_2^2$. Then according to the assumption, $L_1 = \delta_2^2$. Hence

$$L_1 \to L_2 = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \delta_2^2 \delta_2^2 = \delta_2^1.$$

(Necessary) Assume $L_2 = \delta_2^2$ but $L_1 = \delta_2^1$. Then

$$L_1 \to L_2 = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \delta_2^1 \delta_2^2 = \delta_2^2,$$

which means $L_1 \to L_2$ is not a tautology.

Note that the physical meaning of Proposition 5.8 is clear: As long as L_1 is "true", L_2 must be "true".

The following tautological implications can be proved by using Proposition 5.8.

Proposition 5.9. *The followings are tautological implications.*

$$x \wedge y \Rightarrow x; \tag{5.47}$$
$$x \wedge y \Rightarrow y; \tag{5.48}$$
$$x \Rightarrow x \vee y; \tag{5.49}$$
$$y \Rightarrow x \vee y; \tag{5.50}$$
$$\neg x \Rightarrow x \rightarrow y; \tag{5.51}$$
$$y \Rightarrow x \rightarrow y; \tag{5.52}$$
$$\neg(x \rightarrow y) \Rightarrow x; \tag{5.53}$$
$$\neg(x \rightarrow y) \Rightarrow \neg y; \tag{5.54}$$
$$\neg x \wedge (x \vee y) \Rightarrow y; \tag{5.55}$$
$$x \wedge (x \rightarrow y) \Rightarrow y; \tag{5.56}$$
$$\neg y \wedge (x \rightarrow y) \Rightarrow \neg x; \tag{5.57}$$
$$(x \rightarrow y) \wedge (y \rightarrow z) \Rightarrow x \rightarrow z; \tag{5.58}$$
$$(x \vee y) \wedge (x \rightarrow z) \wedge (y \rightarrow z) \Rightarrow z. \tag{5.59}$$

Proof. We prove (5.59) only. Using Proposition 5.8, assume the right hand side is "false", i.e., $z = \delta_2^2$. We check the left hand side.

$$\begin{aligned}
&(x \vee y) \wedge (x \rightarrow z) \wedge (y \rightarrow z) \\
&= M_\wedge M_\vee xy M_\wedge M_\rightarrow xz M_\rightarrow yz \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ 1-p \end{bmatrix} \begin{bmatrix} q \\ 1-q \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \\
&\quad \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ 1-p \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} q \\ 1-q \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} (p+q-pq)(1-p)(1-q) \\ (p+q-pq)^2 + (1-p)^2(1-q)^2 + (p+q-pq)(1-p)(1-q) \end{bmatrix}.
\end{aligned}$$

We need to check four cases: (i) $p = 0, q = 0$, (ii) $p = 0, q = 1$, (iii) $p = 1, q = 0$, (iv) $p = 1, q = 1$. No matter which case was chosen. The result is the same as δ_2^2. □

Finally, we use the structure matrix method to prove some logical equivalences on "EOR""NAND", and "NOR". The proves are similar, and we leave them to the reader.

Proposition 5.10. *The followings are logical equivalences.*

$$x\bar{\nabla}y \Leftrightarrow y\bar{\nabla}x; \tag{5.60}$$
$$(x\bar{\nabla}y)\bar{\nabla}z \Leftrightarrow x\bar{\nabla}(y\bar{\nabla}z); \tag{5.61}$$
$$x \wedge (y\bar{\nabla}z) \Leftrightarrow (x \wedge y)\bar{\nabla}(x \wedge z); \tag{5.62}$$
$$x\bar{\nabla}y \Leftrightarrow (x \wedge \neg y) \vee (\neg x \wedge y); \tag{5.63}$$
$$x\bar{\nabla}y \Leftrightarrow \neg(x \leftrightarrow y); \tag{5.64}$$
$$x \uparrow y \Leftrightarrow y \uparrow x; \tag{5.65}$$
$$x \downarrow y \Leftrightarrow y \downarrow x; \tag{5.66}$$
$$x \uparrow (y \uparrow z) \Leftrightarrow \neg x \vee (y \wedge z); \tag{5.67}$$
$$(x \uparrow y) \uparrow z \Leftrightarrow (x \wedge y) \vee \neg z; \tag{5.68}$$
$$x \downarrow (y \downarrow z) \Leftrightarrow \neg x \wedge (y \vee z); \tag{5.69}$$
$$(x \downarrow y) \downarrow z \Leftrightarrow (x \vee y) \wedge \neg z. \tag{5.70}$$

5.4 Logical System and Logical Deduction

Assume two logical expressions $f(x_1,\cdots,x_k)$ and $g(x_1,\cdots,x_k)$ are logically equivalent. That is,

$$f(x_1,\cdots,x_k) \Leftrightarrow g(x_1,\cdots,x_k).$$

We can simply use the conventional expression as

$$f(x_1,\cdots,x_k) = g(x_1,\cdots,x_k).$$

So for logical expressions $\Leftrightarrow$ is the same as $=$.

Definition 5.8. A static logical system (briefly, logical system) is expressed as

$$\begin{cases} f_1(x_1,x_2,\cdots,x_k) = c_1, \\ f_2(x_1,x_2,\cdots,x_k) = c_2, \\ \vdots \\ f_m(x_1,x_2,\cdots,x_k) = c_m, \end{cases} \tag{5.71}$$

where f_i, $i = 1,\cdots,m$ are logical functions, x_i, $i = 1,\cdots,n$ are logical arguments (unknowns), and c_i, $i = 1,\cdots,m$ are logical constants. A set of logical constants d_i, $i = 1,\cdots,n$, which make

$$x_i = d_i, \quad i = 1,\cdots,n \tag{5.72}$$

satisfy (5.71), is said to be a solution of the logical system (5.71).

In this section we first consider how to solve the logical system (5.71). Assume the structure matrix of f_i is M_i, $i = 1,\cdots,m$. Then in vector form (5.71) can be expressed as

$$\begin{cases} M_1 \ltimes_{i=1}^n x_i = \delta_2^{r_1}, \\ M_2 \ltimes_{i=1}^n x_i = \delta_2^{r_2}, \\ \vdots \\ M_m \ltimes_{i=1}^n x_i = \delta_2^{i_m}, \end{cases} \tag{5.73}$$

where $r_i \in \Delta$, $i = 1, \cdots, m$.

To further simplify (5.73), we need the some preparations.

Proposition 5.11. *Assume*

$$\begin{cases} Y = M_y \ltimes_{i=1}^n x_i \\ Z = M_z \ltimes_{i=1}^n x_i, \end{cases} \tag{5.74}$$

where $M_y \in \mathscr{L}_{p \times 2^n}$ and $M_z \in \mathscr{L}_{q \times 2^n}$. Then

$$YZ = (M_y * M_z) \ltimes_{i=1}^n x_i. \tag{5.75}$$

(Here $*$ is Khatri-Rao product. We refer to Chapter 1 for the definition.)

Proof. First of all, using the properties of semi-tensor product and the power-reducing matrix, we can prove that

$$YZ = M_y \ltimes_{i=1}^n x_i M_z \ltimes_{i=1}^n x_i = M_{yz} \ltimes_{i=1}^n x_i, \tag{5.76}$$

where $M_{yz} \in \mathscr{L}_{pq \times 2^n}$. In fact, using Lemmas 5.1 and 5.2, same computation process as for structure matrix of a logical function yields M_{yz}.

Assume $\ltimes_{i=1}^n x_i = \delta_{2^n}^r$. Then $Y = \mathrm{Col}_r(M_y)$ and $Z = \mathrm{Col}_r(M_z)$. That is,

$$\mathrm{Col}_r(M_{yz}) = \mathrm{Col}_r(M_y) \ltimes \mathrm{Col}_r(M_z) = \mathrm{Col}_r(M_y) \otimes \mathrm{Col}_r(M_z).$$

Now $1 \le r \le 2^n$ is arbitrary, the conclusion follows. □

Using Proposition 5.11, system (5.73) can be converted into the following form:

$$M_1 * M_2 * \cdots * M_m \ltimes_{i=1}^n x_i = \delta_{2^n}^r, \tag{5.77}$$

where

$$r = \sum_{i=1}^{n-1} (r_i - 1) 2^{n-i} + r_n.$$

In other words,

$$[r_1 - 1 \; r_2 - 1 \; \cdots \; r_n - 1]$$

is the binary form of $r - 1$. From this observation one sees easily that system (5.73) is equivalent to

$$Mx = \delta_{2^n}^r, \tag{5.78}$$

where

$$M = M_1 * M_2 * \cdots * M_m, \quad x = \ltimes_{i=1}^n x_i.$$

Note that $x = \ltimes_{i=1}^n x_i \in \mathscr{D}_{2^n}$. System (5.78) can be solved immediately.

Proposition 5.12. *$x = \delta_{2^n}^k$ is the solution of (5.78), if and only if,*

$$\mathrm{Col}_k(M) = \delta_{2^n}^r. \tag{5.79}$$

Finally, we need one more tool in solving a logical system. In general, an equation of f_i in system (5.71) may not involve some arguments. Then how to get (5.73)? For instance, we consider the following system

$$\begin{cases} x_1 \wedge x_2 = 0 \\ x_2 \vee x_3 = 1 \\ x_3 \leftrightarrow x_1 = 1. \end{cases} \tag{5.80}$$

To get the component algebraic form (5.73), we have to add some fabricated arguments to each equation. We introduce the following dummy matrix.

$$M_u = \delta_2[1\ 1\ 2\ 2]. \tag{5.81}$$

A straight forward computation shows the following proposition.

Proposition 5.13. *In vector form we have*

$$M_u xy = x. \tag{5.82}$$

It is obvious that the dummy matrix can be used to add fabricated arguments to each equations if necessary. We give an example to depict this.

Example 5.8. Consider system (5.80). To convert it into the form of (5.73), we have for the first equation that

$$M_c x_1 x_2 = \delta_2^2.$$

The missing x_3 can be plug as

$$M_c x_1 M_u x_2 x_3 = \delta_2^2.$$

Equivalently, we have

$$M_c [I_2 \otimes M_u] x_1 x_2 x_3 = \delta_2^2.$$

Setting $x = x_1 x_2 x_3$, we have

$$\delta_2[1\ 2\ 1\ 2\ 2\ 2\ 2\ 2]x = \delta_2^2.$$

Similarly, for the second and third equations we have

$$\delta_2[1\ 1\ 1\ 1\ 1\ 2\ 1\ 2]x = \delta_2^1;$$
$$\delta_2[1\ 2\ 2\ 1\ 1\ 2\ 2\ 1]x = \delta_2^1.$$

Next, the system can further be converted into the form (5.78) as

$$\delta_8[1\ 2\ 5\ 8\ 6\ 5\ 6\ 7]x = \delta_8^5.$$

Finally, according to Proposition 5.12, the solutions of system (5.80) are $\delta_8^3 \sim (1,0,1)$ and $\delta_8^6 \sim (0,1,0)$.

Next, we consider the problem of logical inference by solving logical equations. We refer to [7] for logical deduction in intelligent systems.

Our basic approach technique is to convert the problem into a logical system. Solving the system provides the solution of the logical problem. We use some simple examples to demonstrate this.

Example 5.9. A says: "B is a liar", B says: "C is a liar", C says: "both A and B are liars". Who is the liar?

To solve this problem we define 3 logical variables as

- x: A is honest;
- y: B is honest;
- z: C is honest.

Then the three statements can be expressed in logical version as

$$\begin{aligned} &x \Leftrightarrow \neg y \\ &y \Leftrightarrow \neg z \\ &z \Leftrightarrow \neg x \wedge \neg y. \end{aligned} \tag{5.83}$$

Let $c = \delta_2^1$. Then equation (5.83) can be converted into an algebraic form as

$$\begin{cases} M_e x M_n y = c \\ M_e y M_n z = c \\ M_e z M_c M_n x M_n y = c. \end{cases} \tag{5.84}$$

It is easy to convert (5.84) into an algebraic equation as

$$L\xi = b, \quad \text{where} \quad \xi = xyz,\ b = c^3 = \delta_8^1,$$

and

$$L = \delta_8[8,5,2,3,4,1,5,8]$$

Since only $\mathrm{Col}_6(L) = b$, we have unique solution

$$\xi = \delta_8^6,$$

which implies that

$$x = 0, \quad y = 1, \quad z = 0.$$

We conclude that only B is honest.

Next, we consider the logical minimization. Consider system (5.71) again. Assume we have a performance criteria as

$$J = J(x_1, \cdots, x_n).$$

The purpose is to find a best feasible solution $x^* = (x_1^*, \cdots, x_n^*)$, such that

$$J(x^*) = \min_{x \text{ satisfing } (5.71)} J(x).$$

We give an example to depict this.

Example 5.10 ([7]). Consider the following problem: The weather is either sunshine or rain. We take either the bus or a taxi to go to work. Suppose it is raining. If we use the bus, we must walk to the bus stop and hence use an umbrella. If we use a taxi, we do not have to use an umbrella.

We use the variables S (sunshine), R (rain), B (bus), T (taxi), and U (umbrella). Then we have the following equations:

$$\begin{cases} S \leftrightarrow \neg R = 1 \\ B \leftrightarrow \neg T = 1 \\ (R \wedge B) \leftrightarrow U = 1. \end{cases}$$

We may consider two cases: Case 1: $R = 1$. That is it is rain. Case 2: $R = 0$.

Suppose that the fare for the bus is \$3 and for the taxi is \$4, and that we view the inconvenience of handling an umbrella to be equivalent to a cost of \$2. Using vector form, the cost function becomes

$$J = 3 \times (\delta_2^1)^T B + 4 \times (\delta_2^1)^T T + 2 \times U.$$

Case 1: The component-wise algebraic form of the system becomes

$$\begin{cases} M_e S M_n R = \delta_2^1 \\ M_e B M_n T = \delta_2^1 \\ M_e M_c R B U = \delta_2^1 \\ R = \delta_2^1. \end{cases}$$

The algebraic form is

$$L\xi = b_1, \quad \text{where } \xi = SRBTU,\ b_1 = \delta_{16}^1,$$

and

$$\begin{aligned} L = [\ & 13\ 15\ 9\ 11\ 11\ 9\ 15\ 13\ 8\ \ 6\ \ 4\ \ 2\ \ 4\ \ 2\ \ 8\ \ 6 \\ & 5\ \ 7\ \ 1\ 3\ \ 3\ \ 1\ 7\ \ 5\ \ 16\ 14\ 12\ 10\ 12\ 10\ 16\ 14]. \end{aligned} \tag{5.85}$$

The solutions are

$$\delta_{32}^{19} \sim (0,1,1,0,1), \quad \delta_{32}^{22} \sim (0,1,0,1,0).$$

The optimal solution is

$$\delta_{32}^{22} \sim (0,1,0,1,0).$$

Case 2: The component-wise algebraic form of the system becomes

$$\begin{cases} M_e S M_n R = \delta_2^1 \\ M_e B M_n T = \delta_2^1 \\ M_e M_c R B U = \delta_2^1 \\ R = \delta_2^2 \end{cases}$$

The algebraic form is

$$L\xi = b_2, \quad \text{where } \xi = SRBTU,\ b_2 = \delta_{16}^2,$$

and L is same as (5.85).
The solutions are

$$\delta_{32}^{12} \sim (1,0,1,0,0), \quad \delta_{32}^{14} \sim (1,0,0,1,0).$$

The optimal solution is

$$\delta_{32}^{12} \sim (1,0,1,0,0).$$

5.5 Multi-valued Logic

One of the advantages of the matrix expression is that it can easily be extended to multi-valued logic. Let x be a k-valued logical variable. That is, $x \in \mathscr{D}_k$. To use the matrix approach, we identify

$$\frac{i}{k-1} \sim \delta_k^{k-i}, \quad i = 1,2,\cdots,n.$$

That is,

$$1 \sim \delta_k^1, \quad \frac{k-2}{k-1} \sim \delta_k^2, \quad \cdots, \quad \frac{1}{k-1} \sim \delta_k^{k-1}, 0 \sim \delta_k^k.$$

Then in vector form we have $x \in \Delta_k$.

Definition 5.9. Let x and y be two k-valued logical variables. Define

(i) (Negation)

$$\neg x = 1 - x; \tag{5.86}$$

(ii) (Disjunction)

$$x \vee y = \max(x, y); \tag{5.87}$$

(iii) (Conjunction)

$$x \wedge y = \min(x, y). \tag{5.88}$$

Using vector form, it is easy to calculate the structure matrices of the k-valued logical operators in Definition 5.9.

For notational ease, we introduce a set of k-dimensional vectors as:

$$\begin{aligned} U_s &= (1\ 2\ \cdots\ s-1\ \underbrace{s\ \cdots\ s}_{k-s+1}) \\ V_s &= (\underbrace{s\ \cdots\ s}_{s}\ s+1\ s+2\ \cdots\ k), \quad s = 1, 2, \cdots, k. \end{aligned}$$

Proposition 5.14. *1. For k-valued negation, its structure matrix is*

$$M_n^k = \delta_k[k\ k-1\ \cdots\ 1]. \tag{5.89}$$

When $k = 3$ we have

$$M_n^3 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}. \tag{5.90}$$

2. For k-valued disjunction, its structure matrix is

$$M_d^k = \delta_k[U_1\ U_2\ \cdots\ U_k]. \tag{5.91}$$

When $k = 3$ we have

$$M_d^3 = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.92}$$

3. For k-valued conjunction, its structure matrix is

$$M_c^k = \delta_k[V_1\ V_2\ \cdots\ V_k]. \tag{5.93}$$

When $k = 3$ we have

$$M_c^3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{5.94}$$

Definition 5.9 is a natural extension of the corresponding objects in classical logic. When $k = 2$ they coincide with objects in classical logic. Next, we consider "conditional" and "biconditional". It is not so obvious. There are many different definitions. In the following we consider the case of $k = 3$, and give 3 different definitions: (i) the type of Kleene-Dienes (KD), (ii) the type of Luekasiewic (L), (iii) the type of Bochvar (B). They are listed in Table 5.7 (where $T=1$, $U=0.5$, $F=0$) [5].

Table 5.7 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

Since we have already defined the "negation", "disjunction", and "conjunction" for k-valued logic. We may use them to define the "conditional" and "biconditional" as follows.

Definition 5.10. For k-valued logic, we define

(vi) (Conditional)

$$x \rightarrow y \Leftrightarrow \neg x \vee y. \tag{5.95}$$

(v) (Biconditional)

$$x \leftrightarrow y \Leftrightarrow (x \rightarrow y) \wedge (y \rightarrow x). \tag{5.96}$$

Note that (5.95) is from property (5.31) of conventional logic, and (5.96) is from (5.43). So definition 5.10 is a natural extension of the conventional logic.

Using (5.95), we have

$$M_i^k xy = M_d^k M_n^k xy.$$

Hence the structure matrix of $\rightarrow$ can be calculated as

$$M_i^k = M_d^k M_n^k. \tag{5.97}$$

It is easy to calculate that when $k=3$ we have

$$M_i^3 = \begin{bmatrix} 1&0&0&1&0&0&1&1&1 \\ 0&1&0&0&1&1&0&0&0 \\ 0&0&1&0&0&0&0&0&0 \end{bmatrix}. \tag{5.98}$$

To calculate the structure matrix of biconditional, we need the k-valued power-reducing matrix. Define the k-valued order reduce matrix as

$$M_r^k = \begin{bmatrix} \delta_k^1 & 0 & \cdots & 0 \\ 0 & \delta_k^2 & \cdots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \cdots & \delta_k^k \end{bmatrix}. \tag{5.99}$$

Then it is easy to prove the following result.

Proposition 5.15. *Let $x \in \Delta_k$. Then we have*

$$x^2 = M_r^k x. \tag{5.100}$$

Now we are ready to calculate the structure matrix of biconditional. Using (5.96), we have

$$\begin{aligned} M_e^k xy &= M_c^k M_i^k xy M_i^k yx \\ &= M_c^k M_d^k M_n^k xy M_d^k M_n^k yx \\ &= M_c^k M_d^k M_n^k [I_{k^2} \otimes M_d^k M_n^k] xy^2 x \\ &= M_c^k M_d^k M_n^k [I_{k^2} \otimes M_d^k M_n^k] x W_{[k,k^2]} xy^2 \\ &= M_c^k M_d^k M_n^k [I_{k^2} \otimes M_d^k M_n^k][I_k \otimes W_{[k,k^2]}] x^2 y^2 \\ &= M_c^k M_d^k M_n^k [I_{k^2} \otimes M_d^k M_n^k][I_k \otimes W_{[k,k^2]}] M_r^k x M_r^k y \\ &= M_c^k M_d^k M_n^k [I_{k^2} \otimes M_d^k M_n^k][I_k \otimes W_{[k,k^2]}] M_r^k [I_k \otimes M_r^k] xy. \end{aligned}$$

Hence we have

$$M_e^k = M_c^k M_d^k M_n^k [I_{k^2} \otimes M_d^k M_n^k][I_k \otimes W_{[k,k^2]}] M_r^k [I_k \otimes M_r^k]. \tag{5.101}$$

Using this formula, we can calculate that as $k = 3$ we have

$$M_e^3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}. \tag{5.102}$$

It is easy to check that when $k = 3$ Definition 5.10 coincides with type Kleene-Dienes logic.

Throughout this book we assume the default k-valued logic is defined by Definitions 5.9 and 5.10, unless elsewhere stated.

Next, we consider the k-valued logical system. Recall system (5.71) and assume the unknowns $x_1, \cdots, x_n \in \mathscr{D}_k$ and the constants $c_1, \cdots, c_m \in \mathscr{D}_k$. Then (5.71) becomes a k-valued logical system. A step by step verification shows that the following deductions remain true and we can obtain the algebraic form of a k-valued logical system, which is similar to (5.78):

$$Mx = \delta_{k^m}^r, \tag{5.103}$$

where $M \in \mathscr{L}_{k^m \times k^n}$. Similar to Proposition 5.12, we have

Proposition 5.16. *$x = \delta_{k^n}^p$ is the solution of (5.103), if and only if,*

$$\mathrm{Col}_p(M) = \delta_{k^m}^r. \tag{5.104}$$

In the following we give an example to show how to use multi-valued logic to carry out the logical inference.

Example 5.11. A detective is investigating a murder case. He has the following clues:

- 80% for sure that either A or B is the murderer;
- if A is the murderer, it is very likely that the murder happened after midnight;
- if B's confession is true, the light at midnight was on;
- if B's confession is false, it is very likely that the murder happened before midnight;
- there is an evidence that the light in the room of the murder was off at midnight.

What conclusion he can get?

First, we have to figure out the levels of logical values. Say, "very likely" is more possible that "80%", then we may quantize the logical values into six levels as: "T", "very likely", "80%", "1-80%", "very unlikely", and "F". Hence we may consider the problem as a problem of 6-valued logical inference.

Define the logical variables(unknowns) as

- A: A is the murderer;
- B: B is the murderer;
- M: the murder happened before the midnight;
- S: B's confession is true;
- L: the light in the room was on at midnight.

Then we can convert the statements into logical equations as

$$\begin{aligned} A \vee B &= \delta_6^3 \\ A \to \neg M &= \delta_6^2 \\ S \to L &= \delta_6^1 \\ \neg S \to M &= \delta_6^2 \\ \neg L &= \delta_6^1. \end{aligned} \tag{5.105}$$

We may use general method, described in Proposition 5.16, to solve that 6-valued system. But since this system has certain special form. We may use "substitution" to solve it. The so called substitution is exactly the same as the one in solving linear system in high school algebra: We use some equations to solve some unknowns first, and then substitute the solved unknowns of the other equations by solved values to solve other unknowns.

First, since $\neg L = \delta_6^1$, we have

$$L = \delta_6^6.$$

Next, the equation $S \to L = \delta_6^1$ provides the following matrix equation:

$$M_i^6 SL = M_i^6 W_{[6]} LS := \Psi_1 S = \delta_6^1.$$

It is easy to calculate that

$$\Psi_1 = M_i^6 W_{[6]} L = \delta_6[6\ 5\ 4\ 3\ 2\ 1].$$

It follows that $S = \delta_6^6$. Similarly, from $\neg S \to M = \delta_6^2$ we have

$$M_i^6 M_n^6 SM = \delta_6^2.$$

M can be solved as

$$M = \delta_6^2.$$

Next, consider $A \to \neg M = M_i^6 A M_n^6 M = \delta_6^2$. Using the properties of semi-tensor product, we have

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

Since

$$\psi_2 = M_i^6 (I_6 \otimes M_n^6) W_{[6]} M = \delta_6[5\ 5\ 4\ 3\ 2\ 1],$$

we have that

$$A = \delta_6^5.$$

Finally, from $A \vee B = M_d AB = \delta_6^3$ we have

$$B = \delta_6^3.$$

We conclude that, A is "very unlikely" the murder, and 80% that B is the murder.

Exercise 5

1. Prove Lemma 5.1. Show that when $x \in \mathscr{D}_f$ formula (5.2) is incorrect.
2. Prove Theorem 5.1. (Refer to [2])
3. Prove the other equivalences (except (5.33)) in Proposition 5.4.
4. Prove the other equivalences (except (5.39)) in Proposition 5.6.
5. Prove the tautological implications (5.47)-(5.58) in Proposition 5.9.
6. Prove the logical equivalences (5.60)-(5.70) in Proposition 5.10.
7. a. Prove De Morgan's Law (Proposition 5.7).
   b. Prove the following general De Morgan's Law:

$$\neg(x_1 \wedge \cdots \wedge x_n) = (\neg x_1) \vee \cdots \vee (\neg x_n). \tag{5.106}$$

$$\neg(x_1 \vee \cdots \vee x_n) = (\neg x_1) \wedge \cdots \wedge (\neg x_n). \tag{5.107}$$

8. Given a logical equation as

$$\begin{cases} f_1(x_1,\cdots,x_n) = g_1(x_1,\cdots,x_n) \\ f_2(x_1,\cdots,x_n) = g_2(x_1,\cdots,x_n) \\ \vdots \\ f_m(x_1,\cdots,x_n) = g_m(x_1,\cdots,x_n). \end{cases} \tag{5.108}$$

Give a general procedure to solve this equation.

9. Consider the following logical system

$$\begin{cases} x_1 = x_2 \sigma x_3 \\ x_2 = x_3 \sigma x_4 \\ \vdots \\ x_{n-1} = x_n \sigma x_1 \\ x_n = x_1 \sigma x_2. \end{cases} \tag{5.109}$$

a. Assume $\sigma = \wedge$. Solve system (5.109).
b. Assume $\sigma = \vee$. Solve system (5.109).

References

1. Barnes, D., Mac, J.: An Algebraic Introduction to Mathematical Logic. Springer-Verlag, New York (1975)
2. Cheng, D., Qi, H., Li, Z.: Analysis and Control of Boolean Networks: A Semi-tensor Product Approach. Springer (2010)
3. Gerla, G.: Fuzzy Logic, Mathematical Tools for Approximate Reasoning. Kluwer (2001)
4. Hamilton, A.: Logic for Mathematicians, revised edn. Cambridge Univ Press, Cambridge (1988)
5. Liu, Z., Liu, Y.: Fuzzy Logic and Newral Network. BUAA Press, Beijing (1996). (in Chinese)
6. Rade, L., Westergren, B.: Mathematics Handbook for Science and Engineering, 4th edn. Studentlitteratur, Sweden (1998)
7. Truemper, K.: Design of Logic-based Intelligent Systems. Wiley & Sons, New Jersey (2004)
8. Verbruggen, H., Babuška, R.: Fuzzy Logic Control: Advances in Applications. World Scientific Pub. Co. Inc., Singapore (1999)
9. Wang, L.: A Course in Fuzzy Systems and Control. Prentice-Hall, Inc. Upper Saddle River, NJ, USA (1996)
10. Zadeh, L., Kacprzyk, J.: Fuzzy Logic for the Management of Uncertainty. John Wiley & Sons, Inc. New York, NY, USA (1992)