2　計算と論理関数

[I] 2進法による表現

- 数を 2 進法で表し 0 と 1 の有限列と考え、それに演算をほどこすための実際の機構を考える。
- 数 a を 2 進法で表すには ...
  - $\boxed{\text{方法 1}}$
    - (1) a が偶数ならば 0 を、奇数ならば 1 を置く。
    - (2) $[a/2]$ を 2 進法で表し、(1) の左に置く。
  - $\boxed{\text{方法 2}}$
    - (1) $2^n \leq a < 2^{n+1}$ となる n をさがし、1 の右に n 桁の場所を用意する。
    - (2) $a - 2^n$ を 2 進法で表し、(1) に重ねる。
- 例: 21 を 2 進法で表す;
  - $\boxed{\text{方法 1}}$で⋯
    - (1) 21 は奇数だから、1 を置く。
    - (2) $[21/2] = 10$ を 2 進法で表せば 1010 なので、 10101 が 21 の 2 進表記である。
  - $\boxed{\text{方法 2}}$で⋯
    - (1) $16 \leq 21 < 32$ だから $n = 4$、したがって $1 \cdots\cdot$ となる。
    - (2) $21 - 16 = 5$ を 2 進法で表せば 101 なので、 10101 を得る。

2^n の表

n	0	1	2	3	4	5	6	7	8	9	10
2^n	1	2	4	8	16	32	64	128	256	512	1024

n	11	12	13	14	15	16	17
2^n	2048	4096	8192	16384	32768	65536	131072

[II] ２進法における筆算

- ２進法で表された数の筆算は十進法に比べてやさしい。

$$
\begin{array}{ccccccc}
 & & 1 & 0 & 1 & 1 & 0 \\
+) & & & 1 & 1 & 0 & 1 \\
\hline
 & 1 & 0 & 0 & 0 & 1 & 1
\end{array}
\qquad
\begin{array}{ccccccccc}
 & & & & & 1 & 0 & 1 & 1 \\
 & & & \times) & & 1 & 1 & 0 & 1 \\
\hline
 & & & & & 1 & 0 & 1 & 1 \\
 & & & 1 & 0 & 1 & 1 & & \\
 & & 1 & 0 & 1 & 1 & & & \\
\hline
 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1
\end{array}
$$

[III] 真偽表・論理関数

- ２進数表示された数の和を計算するために、まず次のような表を考える:

c	x	y	d	z
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

- ２つの数の２進数表示が

$$a = x_n x_{n-1} \cdots x_1 x_0, \qquad b = y_n y_{n-1} \cdots y_1 y_0$$

  で与えられるとき、a, b の和は次のようにして計算できる。

  (1) $k = 0,\ c = 0$ とする。

  (2) $x = x_k,\ y = y_k$ とし、c, x, y の値にしたがって、表から d, z を求める。

  (3) $c = d,\ z_k = z$ とおく。

  (4) $k+1$ をあらためて k とし (2) に戻り、k が n になるまで繰り返す。

  (5) 以上によって求まった z_k を並べて、和 $dz_n \cdots z_1 z_0$ を得る。

- d, z はそれぞれ c, x, y に対して決まる。このような対応は**論理関数**と呼ばれ、

$$d = f(c, x, y), \qquad z = g(c, x, y)$$

  のように表される。また、上の表は論理関数 f, g の**真偽表**と呼ばれる。

[IV] 基本論理関数

- 写像 $\{0,1\}^n \to \{0,1\}$ を n 変数の**論理関数**という。
- 以下のように定義される
  $\mathrm{NOT}(x) = \overline{x}, \quad \mathrm{AND}(x,y) = x \wedge y, \quad \mathrm{OR}(x,y) = x \vee y$
  を**基本論理関数**という;

x	$\overline{x}$
0	1
1	0

x	y	$x \wedge y$	$x \vee y$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

- 公式 — 論理代数の公理
  NOT, AND, OR の間には以下の公式が成り立つ;

(1) 交換法則
$$x \wedge y = y \wedge x, \qquad x \vee y = y \vee x$$
(2) 結合法則
$$x \wedge (y \wedge z) = (x \wedge y) \wedge z,$$
$$x \vee (y \vee z) = (x \vee y) \vee z$$
(3) 吸収法則
$$x \vee (y \wedge x) = (x \vee y) \wedge x = x$$
(4) 分配法則
$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z),$$
$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$$
(5) 二重否定と相補法則
$$\overline{\overline{x}} = x, \qquad x \wedge \overline{x} = 0, \qquad x \vee \overline{x} = 1$$
(6) de Morgan の法則
$$\overline{x \wedge y} = \overline{x} \vee \overline{y}, \qquad \overline{x \vee y} = \overline{x} \wedge \overline{y}$$
(7) 0, 1 の性質
$$x \wedge 0 = 0, \qquad x \wedge 1 = x,$$
$$x \vee 0 = x, \qquad x \vee 1 = 1$$

- 和を計算するために考えた $d = f(c,x,y)$, $z = g(c,x,y)$ は、たとえば
$$d = (\overline{c} \wedge (x \wedge y)) \vee (c \wedge (x \vee y)),$$
$$z = (\overline{c} \wedge \overline{x} \wedge y) \vee (\overline{c} \wedge x \wedge \overline{y}) \vee (c \wedge \overline{x} \wedge \overline{y}) \vee (c \wedge x \wedge y)$$
  のように表すことができる。

[V] 論理関数の構成

- 一般に、n 変数の論理関数は $2^{(2^n)}$ 個ある。
- 1 変数の論理関数は、恒等関数、NOT および定数関数（つねに 0、つねに 1）の 4 つのみで、恒等関数は $\mathrm{AND}(x,x) = x \wedge x = x$ で定義でき、また、定数関数は以下のように定義できる。

$$\mathrm{AND}(x, \mathrm{NOT}(x)) = x \wedge \overline{x} = 0, \qquad \mathrm{OR}(x, \mathrm{NOT}(x)) = x \vee \overline{x} = 1$$

- **補題**　$D(x,y,z) = (x \wedge \overline{z}) \vee (y \wedge z)$ と定めれば、

$$D(x,y,z) = \begin{cases} x & (z=0 \text{ のとき}), \\ y & (z=1 \text{ のとき}) \end{cases}$$

が成り立つ。

- **定理 2-1**　すべての論理関数は、NOT、AND、OR の合成で得られる。

[**証明**]　変数の個数に関する数学的帰納法によって証明する。1 変数の場合は上で見たようにすべて NOT、AND、OR の合成で得られている。$n > 1$ として、n 変数論理関数 $f(x_1, \ldots, x_n)$ に対して

$$g(x_1, \ldots, x_{n-1}) = f(x_1, \ldots, x_{n-1}, 0),$$
$$h(x_1, \ldots, x_{n-1}) = f(x_1, \ldots, x_{n-1}, 1)$$

と定めると、これらは $n-1$ 変数だから帰納法の仮定により、NOT、AND、OR の合成で表され，さらに前補題の関数 D を用いて

$$f(x_1, \ldots, x_n) = D(g(x_1, \ldots, x_{n-1}), h(x_1, \ldots, x_{n-1}), x_n)$$

と書けるから，$f(x_1, \ldots, x_n)$ も NOT、AND、OR の合成で表すことができる。　（証明終わり）

- NAND 関数、NOR 関数を

$$\mathrm{NAND}(x,y) = \mathrm{NOT}(\mathrm{AND}(x,y)) = \overline{x \wedge y}$$
$$\mathrm{NOR}(x,y) = \mathrm{NOT}(\mathrm{OR}(x,y)) = \overline{x \vee y}$$

によって定義すると，

$$\mathrm{NOT}(x) = \mathrm{NAND}(x,x) = \mathrm{NOR}(x,x)$$
$$\mathrm{AND}(x,y) = \mathrm{NOT}(\mathrm{NAND}(x,y)) = \mathrm{NOR}(\mathrm{NOT}(x), \mathrm{NOT}(y))$$
$$\mathrm{OR}(x,y) = \mathrm{NOT}(\mathrm{NOR}(x,y)) = \mathrm{NAND}(\mathrm{NOT}(x), \mathrm{NOT}(y))$$

- **定理 2-2**　(1) すべての論理関数は、NAND の合成で得られる。
(2) すべての論理関数は、NOR の合成で得られる。