

ある Fail-Safe 論理系について

京都大学大学院 高岡忠雄

序. およそ故障と呼び得る現象は非対称に生起し易く, また, 目的論的にみて, 非対称に起る故障は許容される場合が多い。遅れる時計はいつも遅れ易く, 遅刻して来る人はいつも遅刻しがちである。一方, 時計を使う立場から云えば, 進むことの方が, 遅れることより許容される場合が多い。デートの時間に, 早く来すぎることはあまり問題ないが, 遅れて来ることが破滅的状況へ導くこともあり得る。よくひきあいに出来る例だが, ある汽車が進むべきところを, 誤って止った場合, 単に時間の損失で済む場合が多いが, 止るべきところを, 誤って進んだ場合, 大変なことになるだろう。

以上のことを抽象化して, 論理回路の問題として扱うと, 入力, 及び論理素子が 0 が 1 のどちらかの値に非対称に誤るという条件の下で, 出力の値が誤ったとしても, 0 が 1 のどちらかの値, 即ち安全側の値に誤ることを保障できるような論理回路を構成できるか, ということになる。このような回路を Fail-Safe 論理回路という。そして, 文献(1), (2) で詳細に論ぜられている。文献(3) には, 同じ Fail-Safe 論理回路を 2 台並列に稼働させる構想が述べられている。今, 安全側の出力を 0 としよう。出力の組 $(0, 0)$, $(0, 1)$, $(1, 0)$,

(1, 1) のうち、後三者は正しく 1 と認定されるが、(0, 0) は正しい 0 が誤りの 0 が識別不可能である。しかし、これによって、出力の信頼度を可成り上昇させ得ることは事実である。

ここに、別の考え方を提起しよう。上記のシステムでは、出力の値が 0 にせよ 1 にせよ、システムは稼動していると考えられる。我々は深い山中で道に迷った時、右の道を行くか、左の道を行くかということをしせずに、ただその場に止って、救援を待てと教えられている。すなわち、論理回路の出力に新たに N の値をもうけ、0 と 1 は N にしか誤らないようにする。そして、 N の値が出力に出たときは、システム全体を止めて、故障の箇所を調べて、修理をする。このようなシステムを N -Fail-Safe であるという。ここで、入力及び論理素子は非対称に誤るものとし、相補二重系という一種の二重系を用いる。後でみるように、かかる論理関数も N -Fail-Safe 論理回路で実現できる。(古典的な Fail-Safe 系では、実現すべき論理関数に制限があった。) また、 N -Fail-Safe 系は、絶えず稼動している古典的な Fail-Safe 系に比して、能率という点では恐らく劣るが、完全に安全であるといえる。本研究の結果、筆者は、能率と安全性とは両立し難く、反対方向に増減するものだと結論している次第である。

1. N -Fail-Safe 関数

記号を次のように定める。

$$L^n = \{0, 1\}^n, \quad \tilde{L}^n = \{0, N, 1\}^n$$

$$x = (x_1, \dots, x_n)$$

定義 1. 1. 図 1 で表される誤りを抹消誤りという。

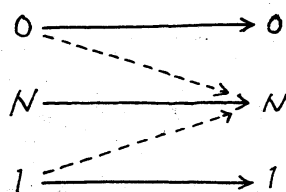


図 1. 抹消誤り

図 1 の左の値から右の値への変化において、実線は正しい場合であり、点線が抹消誤りを示す。

定義 2. 1. 三値論理関数 $f'(x)$ が二値論理関数 $f(x)$ の N -Fail-Safe 関数であるとは、 $\forall x \in L^n \quad f'(x) = f(x)$ であり、変数の変化 $x \in L^n \rightarrow y \in (\tilde{L}^n - L^n)$ が抹消誤り（これは x の各要素から y の要素への変化が正しいか、抹消誤りの場合をいう）のとき、出力の変化 $f(x) \rightarrow f'(y)$ がまた正しいか、抹消誤りのものをいう。

命題 1. 1. 関数 $f'_1(x_1), \dots, f'_m(x_m), g'(t_1, \dots, t_m)$ がそれぞれ $f_1(x_1), \dots, f_m(x_m), g(t_1, \dots, t_m)$ の N -Fail-Safe 関数とする。このとき、関数 $g'(f'_1(x_1), \dots, f'_m(x_m))$ は、関数 $g(f_1(x_1), \dots, f_m(x_m))$ の N -Fail-Safe 関数である。すなわ

ち, N -Fail-Safe の性質は関数の合成のもとで保存される。

定義 1.3. $f'(x)$ を二値関数 $f(x)$ の N -Fail-Safe 関数とする。このとき, $\tilde{L}^n$ の部分集合

$$S_{f'} = \{x \in (\tilde{L}^n - L^n) \mid f'(x) = N\}$$

を $f'(x)$ の情報損失集合という。

我々はこれから, $f(x)$ の N -Fail-Safe 関数の中で, この情報損失集合が最小になるものを探さねばならぬ。

こゝでいくつかの記号を導入しておこう。

$$x(i, r) = (x_{i_1}, \dots, x_{i_r}), \quad \bar{x}(i, r) = x - x(i, r)$$

とし, 変数集合 $x(i, r)$, $\bar{x}(i, r)$ のとり空間をそれぞれ, 二値の場合 L_i^r , L_i^{n-r} , 三値の場合 $\tilde{L}_i^r$, $\tilde{L}_i^{n-r}$ とする。

勿論

$$L^n = L_i^r \times L_i^{n-r}, \quad \tilde{L}^n = \tilde{L}_i^r \times \tilde{L}_i^{n-r}$$

であり, $r=0$ のときは $x(i, r)$ は空, $r=n$ のときは,

$\bar{x}(i, r)$ は空である。また $N(i, r)$ は $x_{i_1} = N, \dots, x_{i_r} = N$ を意味する。

定義 1.4. 二値関数 $f(x)$ に対し, $f(x)$ の完備 N -Fail-Safe 関数 $\hat{f}(x)$ を次のように定義する。

$$\forall x \in L^n \quad \hat{f}(x) = f(x)$$

また, $x \in \tilde{L}^n - L^n$ に対しては,

$$\forall x(i, r) \in L_i^r \quad f(x(i, r), \bar{x}(i, r)) = D$$

$$(D = 0 \text{ or } 1)$$

の成り立つとき,

$$f(N(i, r), \bar{x}(i, r)) = D$$

と定め, それ以外のとき, すなわち

$$\exists x(i, r), \exists x'(i, r) \in L_i^r \quad f(x(i, r), \bar{x}(i, r)) \neq f(x'(i, r), \bar{x}(i, r))$$

の成り立つとき,

$$f(N(i, r), \bar{x}(i, r)) = N$$

と定める。 i と r を適当に変えることによって, すべての $x \in \tilde{L}^n$ に対して, $\tilde{f}(x)$ の値は唯一通りに定められる。

命題 1. 2. $\tilde{f}(x)$ は $f(x)$ の N -Fail-Safe 関数である。

命題 1. 3. $f(x)$ の任意の N -Fail-Safe 関数 $f'(x)$ に対し,

$$\forall x \in \tilde{L}^n \quad (\tilde{f}(x) = N \Rightarrow f'(x) = N)$$

が成り立ち, それ故 $S_{\tilde{f}} \subseteq S_{f'}$ が成り立つ。

これで, $\tilde{f}(x)$ が我々の望む N -Fail-Safe 関数であることがわかった。

命題 1. 4. $x = (N(i, r), \bar{x}(i, r))$ に対して $\tilde{f}(x) = N$ のとき, $\bar{x}(i, r)$ の 0 か 1 のいくつかをさらに N に変えて得られる x' に対しても $\tilde{f}(x') = N$ である。

この命題は $\tilde{f}(x)$ を効果的に定義するアルゴリズムに有用である。次頁に, 1 変数関数, 2 変数関数の完備 N -Fail-Safe 関数の表を掲げる。

変数 x	定数	否定	恒等関数	定数
0	0	1	0	1
1	0	0	1	1
N	0	N	N	1

表 1. 1. 1 変数関数の完備 N -Fail-Safe 関数

変数		関数 $\tilde{f}_i(x_0, x_1)$															
x_0	x_1	$\tilde{f}_0$	$\tilde{f}_1$	$\tilde{f}_2$	$\tilde{f}_3$	$\tilde{f}_4$	$\tilde{f}_5$	$\tilde{f}_6$	$\tilde{f}_7$	$\tilde{f}_8$	$\tilde{f}_9$	$\tilde{f}_{10}$	$\tilde{f}_{11}$	$\tilde{f}_{12}$	$\tilde{f}_{13}$	$\tilde{f}_{14}$	$\tilde{f}_{15}$
0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	N	0	N	N	1	0	N	N	1	0	N	N	1	0	N	N	1
N	0	0	N	0	N	N	1	N	1	0	N	0	N	N	1	N	1
1	N	0	0	0	0	N	N	N	N	N	N	N	N	1	1	1	1
N	1	0	0	N	N	0	0	N	N	N	N	1	1	N	N	1	1
N	N	0	N	N	N	N	N	N	N	N	N	N	N	N	N	N	1

表 1. 2. 2 変数関数の完備 N -Fail-Safe 関数

これらの関数の中、特別なものは既に文献 (4), (5), (6) にみられる。

2. 関数の展開

二値関数 $f(x)$ を

$$f(x) = F(f_1(x), \dots, f_m(x)) \quad (2.1)$$

なる形に展開されたものとする。ここに、各 $f_i(x)$, および $F(t_1, \dots, t_m)$ はそれぞれ二値関数である。それぞれの完備 N -Fail-Safe 関数をつくり、簡単のため

$$\tilde{F}(\tilde{f}_1(x), \dots, \tilde{f}_m(x)) = \tilde{F}(x). \quad (2.2)$$

と表記する。

定理 2.1. D を 0 か 1 の値をとるものとする。このとき、

$$\forall x \in \tilde{L}^n \quad (\tilde{F}(x) = D \Rightarrow \tilde{f}(x) = D)$$

が成り立つ。

証明 $x \in L^n$ の場合は命題 1.1 より明らかである。そこで、 $x = (N(i, r), \bar{x}(i, r))$ と仮定し、この x に対し $\tilde{f}_{j_1}, \dots, \tilde{f}_{j_s}$ が値 N をとるものとする。ここで $\tilde{F}(x) = D$ と仮定し、

$$f = (f_1, \dots, f_m)$$

$$f(j, s) = (f_{j_1}, \dots, f_{j_s}), \quad \bar{f}(j, s) = f - f(j, s)$$

とすると定義より

$$\forall f(j, s) \in L_j^s \quad F(f(j, s), \bar{f}(j, s)) = D$$

となる。一方 $x(i, r)$ が L_i^r のすべての真をつくるとき、 $f_1, \dots, f_m$ の中で値を変えるのは $f_{j_1}, \dots, f_{j_s}$ だけであるから、
(2.1) より

$$\forall x(i, r) \in L_i^r \quad f(x(i, r), \bar{x}(i, r)) = D$$

が成り立つ。従って、

$$\tilde{f}(x) = \tilde{f}(N(i, r), \bar{x}(i, r)) = D$$

となる。

(証明終り)

定理 2.2.

$$\forall x \in \tilde{L}^n \quad (\tilde{f}(x) = N \Rightarrow \tilde{F}(x) = N), \quad S_{\tilde{f}} \subseteq S_{\tilde{F}}$$

が成り立つ。

証明. $\tilde{F}(x)$ が $f(x)$ の N -Fail-Safe 関数であること、命題 1.3 より明らかである。 (証明終り)

系. 関数 $f(x)$ の任意の冗数の展開形式に対して、定理 2.1 及び定理 2.2 が成り立つ。

これにより、関数 $f(x)$ を展開して、それから各関数について、完備 N -Fail-Safe 関数をつくること、 N -Fail-Safe 性は保存されるが、出力の情報損失度は高くなる。それ故、つぎに情報無損失な展開形式があるかどうかの問題になる。

定義 2.1. 展開

$$f(x) = F(f_1(x_1), \dots, f_m(x_m)) \quad (2.3)$$

$$x_i \cap x_j = \emptyset \quad i \neq j \quad (i, j = 1, \dots, m)$$

$$x = x_1 \cup \dots \cup x_m$$

を樹枝状展開という。

(2.3) 式の右辺の各関数について、その完備 N -Fail-Safe

関数をつくり，その三値出力を再び簡単のため

$$\tilde{F}(\tilde{f}_1(x_1), \dots, \tilde{f}_m(x_m)) = \tilde{F}(x) \quad (2.4)$$

と表記する。

定理 2.3. 樹枝状展開 (2.3) に対し，

$$\forall x \in L^n \quad \tilde{f}(x) = \tilde{F}(x) \quad , \quad S_{\tilde{f}} = S_{\tilde{F}}$$

が成り立つ。

証明. 定理 2.2 より

$$\tilde{f}(x) = N \Rightarrow \tilde{F}(x) = N$$

であるから，もし

$$\tilde{f}(x) = N \Leftarrow \tilde{F}(x) = N \quad (2.5)$$

が証明されれば

$$\tilde{f}(x) = N \Leftrightarrow \tilde{F}(x) = N$$

が得られ，等価的に

$$\tilde{f}(x) \neq N \Leftrightarrow \tilde{F}(x) \neq N$$

が得られる。このとき $\tilde{f}(x) = D$ ， $\tilde{F}(x) = D'$ ($D, D' = 0$ or 1)

とすると，定理 2.1 より $D = D'$ でなければならぬ。それ故，

(2.5) と等価な

$$\tilde{f}(x) \neq N \Rightarrow \tilde{F}(x) \neq N$$

を示そう。まず $\tilde{f}(x) = D$ ($D = 0$ or 1) と仮定する。 $x =$

$(N(i, r), \bar{x}(i, r))$ とすると，定義より，

$$\forall x(i, r) \in L_i^r \quad f(x(i, r), \bar{x}(i, r)) = D$$

が成り立つ。ここで $x(i, r) \in L_i^r$ が L_i^r のすべての実をく
すとき $f_{j_1}, \dots, f_{j_s}$ がその値を変えなものでしよう。(もし,
 $f_{j_1}, \dots, f_{j_m}$ のうちどれか値を変えなければ証明は明らか)。

このとき、もちろん D の値も D に留まるのであり、各 x_i は
互いに素であるから、 $f_{j_1}, \dots, f_{j_s}$ は互いに独立にその値を変
えることになり、 $f(j, s)$ は L_j^s のすべての実を尽くすこと
になる。このような状況の下で

$$\forall f(j, s) \in L_j^s \quad F(f(j, s), \bar{f}(j, s)) = D$$

である。ここで入力 $x = (N(i, r), \bar{x}(i, r))$ が入るとすると、

$$\tilde{f}_{j_1}(N(i, r), \bar{x}(i, r)) = N, \dots, \tilde{f}_{j_s}(N(i, r), \bar{x}(i, r)) = N \text{ となり、}$$

$$\tilde{F}(N(j, s), \bar{f}(j, s)) = D$$

となる。

(証明終り)

系. 任意の関数の樹枝状展開に対して定理 2.3 が成り立つ。

樹枝状展開ができれば、情報無損失であることがわかった。

しかし、任意の関数 $f(x)$ がいつも樹枝状に展開できるとは限
らない。任意の関数 $f(x)$ がもっと展開形式で、情報無損失なも
のがあるであろうか。次節でみるように、そのような展開形
式は存在する。

3. 素項展開と完全系

任意の二値関数 $f(x)$ は

$$\begin{aligned} f(x) = & f(0, 0, \dots, 0) \bar{x}_1 \bar{x}_2 \dots \bar{x}_n + f(0, 1, \dots, 0) \bar{x}_1 x_2 \dots \bar{x}_n \\ & + \dots + f(1, 1, \dots, 1) x_1 x_2 \dots x_n \end{aligned} \quad (3.1)$$

の形に展開される。これを最小項展開という。いくつかの変数やその否定との論理積を

$$\begin{aligned} C_{x_\alpha}(x) &= 1 \quad \text{for } x = x_\alpha \\ &= 0 \quad \text{for } x \neq x_\alpha \end{aligned}$$

で定義すると, (3.1) 式は

$$f(x) = \sum_{x_\alpha \in L^n} f(x_\alpha) C_{x_\alpha}(x) \quad (3.2)$$

の形に書ける。(3.2) 式において, 隣り合う項を一緒にしてまとめいくと, 最後に素項が残る。その素項展開を

$$f(x) = \sum C_{a_i}(x_i) \quad (3.3)$$

と表す。ここに x_i は x の部分集合で

$$\begin{aligned} C_{a_i}(x_i) &= 1 \quad \text{for } x_i = a_i \\ &= 0 \quad \text{for } x_i \neq a_i \end{aligned}$$

であり, $C_{a_i}(x_i), C_{a_j}(x_j) (i \neq j)$ は互いに素である。(3.3)

の形で用いられている関数は論理和, 論理積及び否定であり, これらの関数も完備 N -Fail-Safe 関数に変えて得られる三値関数の出力を簡単のため $\tilde{f}(x)$ と表記する。

定理 3.7. 素項展開 (3.3) に対し,

$$\forall x \in \tilde{L}^n \quad \tilde{f}(x) = \tilde{F}(x), \quad S_{\tilde{f}} = S_{\tilde{F}}$$

が成り立つ。

証明 定理 3.3 の証明と同様に

$$\tilde{f}(x) \neq N \Rightarrow \tilde{F}(x) \neq N$$

を示せばよい。まず, $\tilde{f}(x) = 0$ と仮定し, $x = (\nu(i, r), \bar{x}_\alpha(i, r))$ と仮定しよう。 $x(i, r)$ が L_i^r のすべての真をつくるとき, 素項 C 's の値はすべて 0 に留まることになる。それ故, すべての $\tilde{C}_{\alpha_i}(x_i)$ は値 0 をとり, 多変数論理和の完備 N -Fail-Safe 関数の性質より

$$\tilde{F}(x) = \sum \tilde{C}_{\alpha_i}(x_i) = 0$$

つぎに, $\tilde{f}(x) = 1$ と仮定しよう。

$$\forall x(i, r) \in L_i^r \quad f(x(i, r), \bar{x}_\alpha(i, r)) = 1$$

であるから, 素項の中に $C_{\bar{x}_\alpha(i, r)}(\bar{x}_\alpha(i, r))$ なる項が存在するか, さもなければ, この項より変数の減った項, すなわち上記の項を簡単のため C_α と書くと $C_\alpha \Rightarrow C_\beta$ となるような項 C_β が素項として存在する。入力 $x = (\nu(i, r), \bar{x}_\alpha(i, r))$ が入るとき $\tilde{C}_\alpha = 1$ であり, 従って $\tilde{C}_\beta = 1$ である。それ故, 関数 $\tilde{\Sigma}$ の性質より

$$\tilde{F}(x) = 1$$

となる。

(証明終り)

系. (3.3) 式の Σ , C はそれぞれ 2 変数関数の論理和, 論理積に到るまで樹枝状に展開できる. その結果の展開形式に用いられている関数をそれぞれ完備 N -Fail-Safe 関数にして得られた関数の三値出力を簡単のため $\hat{F}(x)$ と書くと

$$\forall x \in \tilde{L}^n \quad \tilde{f}(x) = \hat{F}(x), \quad S_{\tilde{f}} = S_{\hat{F}}$$

が成り立つ。

これによって, 素項展開が我々にとって最も望ましい展開形式の一つであることがわかった. なお, 素項展開はもっと簡単な形にして変数を減らすことができる. しかし, その場合, 一般には情報損失度が再び上昇する.

例 3.1. 次の関数を考えよう

(1) 最小項展開 $f(x_1, x_2, x_3) = x_1 x_2 x_3 + x_1 x_2 \bar{x}_3 + \bar{x}_1 x_2 x_3 + \bar{x}_1 \bar{x}_2 x_3$

(2) 素項展開 $f(x_1, x_2, x_3) = x_1 x_2 + x_2 x_3 + \bar{x}_1 x_3$

(3) 最簡形式 $f(x_1, x_2, x_3) = x_1 x_2 + \bar{x}_1 x_3$

この場合, (1), (3) の展開形式は (2) より情報損失度は高くなる. (入力 $(N, 1, 1)$ で確かめられる)

定理 3.2. 否定, 論理和, 論理積の完備 N -Fail-Safe 関数のセットで, 完備 N -Fail-Safe 関数の完全系をなす.

証明 上の定理の系から直接得られる. (証明終り)

補題 3.1. 否定, 論理和, 論理積の完備 N -Fail-Safe 関数をも簡単のため二値の場合と同じ記号を用いる. このとき,

$$\overline{x_1 + x_2} = \overline{x_1} \cdot \overline{x_2}$$

$$\overline{x_1 x_2} = \overline{x_1} + \overline{x_2}$$

が成り立つ。また、表 1.2 における $\tilde{f}_1$ を "1" で、 $\tilde{f}_7$ を記号 "↓" で表す。

$$x \uparrow x = \overline{x}, \quad x \downarrow x = \overline{x}$$

$$x_1 + x_2 = (x_1 \uparrow x_2) \uparrow (x_1 \uparrow x_2)$$

$$x_1 x_2 = (x_1 \downarrow x_2) \downarrow (x_1 \downarrow x_2)$$

が成り立つ。

それ故

定理 3.3. $(-, +)$, $(-, \cdot)$, "1", "↓" はそれぞれ、
完備 N -Fail-Safe 関数の完全系をなす

4. 相補二重系

三値変数 x に、次の表のようにして二値変数の組 (x', x'') を割り当てる。

x	(x', x'')
N	$(0, 0)$
0	$(1, 0)$
1	$(0, 1)$
R	$(1, 1)$

表 4.1.

変数 x' , x'' がともに 0 に誤るとき、変数 x は N に誤る。それ故抹消誤りとなる。 R は組み合わせ禁止となる。これを相補二

重系という。代表的な完備 N -Fail-Safe 関数の相補 = 重系を構成してみよう。

$$f(x) = \bar{x} : f' = x'', f'' = x'$$

論理和 $f_1(x_1, x_2) = x_1 + x_2$, 論理積 $f_2(x_1, x_2) = x_1 \cdot x_2$ に対しては, $R = (1, 1)$ が組み合わせ禁止であることを利用すれば, 次のような簡単な形で表示される。

$$f_1' = x_1' + x_2' , f_1'' = x_1'' \cdot x_2''$$

$$f_2' = x_1' \cdot x_2' , f_2'' = x_1'' + x_2''$$

以上を図示しておこう。

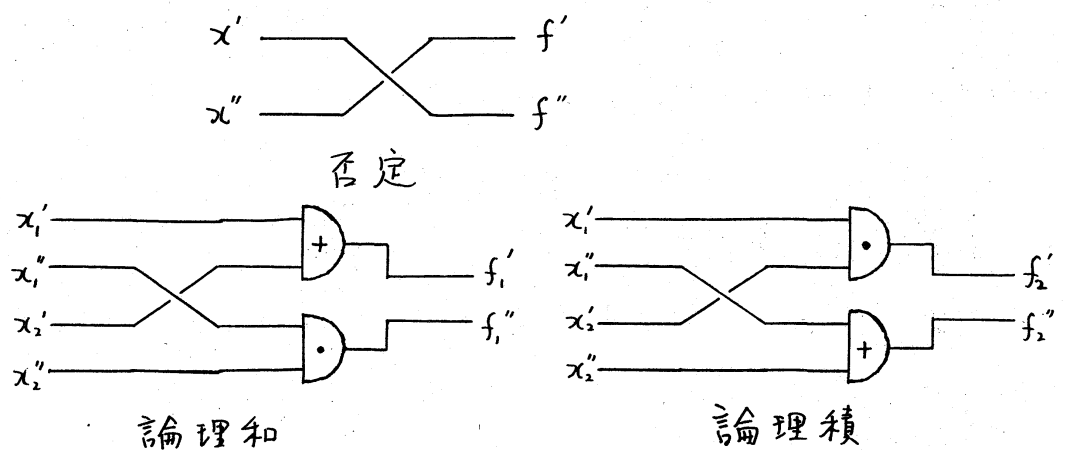


図 4.1. 代表的な完備 N -Fail-Safe 関数の相補 = 重系

上図において, 二値の OR 素子, AND 素子ともに 0 側に非対称に誤りをもつとする。これによって N -Fail-Safe 性が保たれる。上記の回路は偶然 von Neumann の Double line trick⁽⁷⁾ に一致している。なお, 誤り発見用に NOR 回路が利用できる。

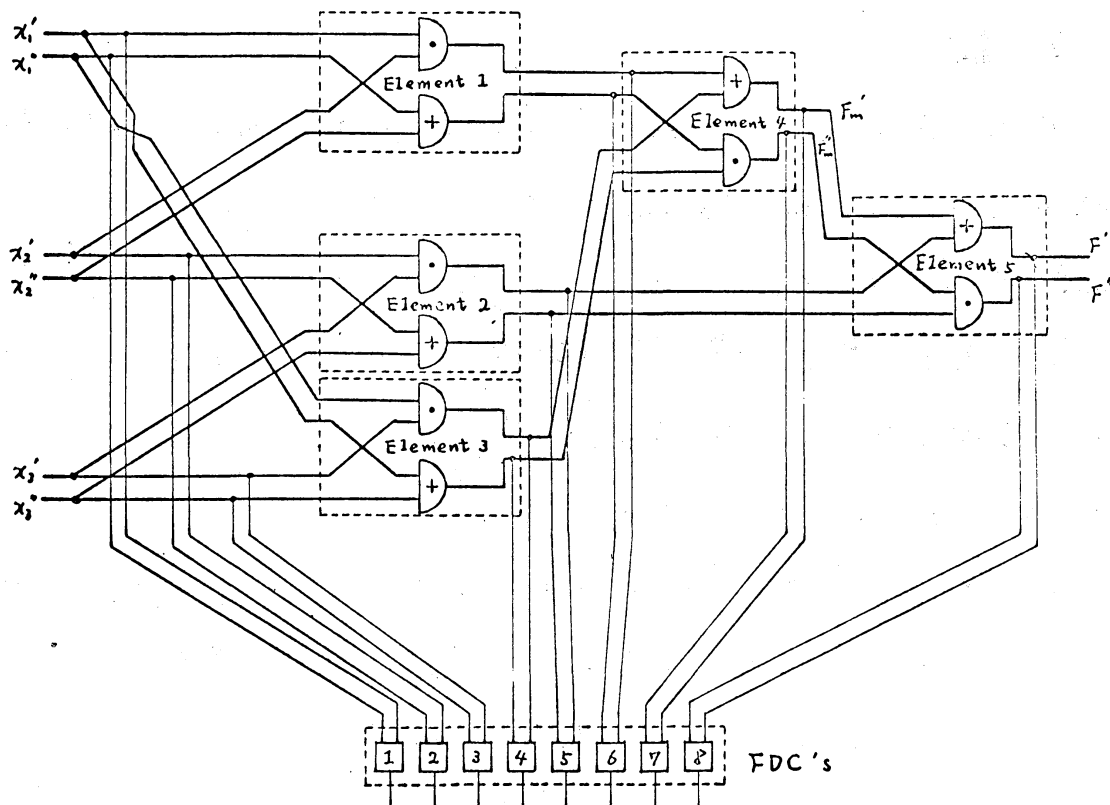


図 4.2. 例 3.1 の相補二重系

なお、本論文は著者らの草稿(8)を加筆訂正してまとめたものである。

最後に、いろいろ御討論頂いた長谷川助教授に感謝致します。

文 献

- (1) H. Mine and Y. Koga, "Basic Properties and a Construction Method for Fail-Safe Logical Systems," IEEE Trans. EC, vol EC-16, No. 3, pp. 282-289, June 1967.
- (2) 橋本, 都倉, 嵩, "非対称誤り素子によるフェイルセーフ論理回路と2重化論理," 電子通信学会誌, Vol. 50, No. 4, pp. 680-687, April 1967.
- (3) 渡辺, 高橋, "Fail-Safe 論理系と誤り訂正機能のある二重系の構成法," 信学会電算機研資, (昭41-01).
- (4) S. C. Kleene, Introduction to Meta-Mathematics, North-Holland Publishing Co., Amsterdam, Third reprint 1962.
- (5) M. Yoeli and S. Rinon, "Application of Ternary Algebra to the Study of Static Hazards," J. ACM 11, No. 1, pp. 84-97, January 1964.
- (6) 平山, 渡辺, 浦野, "Fail-Safe 論理系の構成理論," 電子通信学会誌, vol. 52-C, No. 1, pp. 33-40, January 1969.

- (7) J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," Automata Studies, Princeton University Press, 1956.
- (8) 三根 高岡, "非対称故障論理回路を用いた2重系の構成法," 信学会オートマトン研資, (昭42-09).

補 遺

本論文に述べた内容はもう少し数学的に定式化できるので、それについて述べる。

定義 1. 集合 $L = \{0, 1\}$, $\tilde{L} = \{0, 1, N\}$ とし, $\tilde{L}$ を順序, $0 < N$, $1 < N$ で半順序^{集合}として定義する。(図 7)

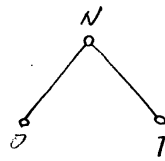


図 7.

定義 2. $\tilde{L}^n$ を半順序集合として定義する。ただし順序は $x, x' \in \tilde{L}^n$ に対して

$$x \leq x' \Leftrightarrow x_1 \leq x'_1, \dots, x_n \leq x'_n$$

とする。

定義 3.

$$f: L^n \rightarrow L \quad (\text{2 値関数})$$

$$f': \tilde{L}^n \rightarrow \tilde{L} \quad (\text{3 値関数})$$

f' が f の N -fail-safe 関数であるとは

$$\forall x \in L^n \quad f'(x) = f(x)$$

$$\forall x \in L^n, \forall x' \in \tilde{L}^n \quad x \leq x' \Rightarrow f'(x) \leq f'(x')$$

定義 4. $NFS(f)$ を 2 値関数 f の N -fail-safe 関数全体からなる集合。

定義 5. 2 値関数 f が与えられたとき, 3 値関数 $f^{\sim}$ および

$\hat{f}$ もつぎのように定める。

$$\forall x \in L^n \quad \tilde{f}(x) = f(x)$$

ある $x \in \tilde{L}^n$ に対して

$$\forall x' \in L^n \quad (x' \leq x) \wedge (f(x') = D)$$

ならば

$$\tilde{f}(x) = D \quad (D = 0 \text{ or } 1)$$

そうでないとき, すなわち

$$\exists x', \exists x'' \in L^n \quad (x' \leq x) \wedge (x'' \leq x) \wedge (f(x') \neq f(x''))$$

のとき

$$\tilde{f}(x) = N$$

と定める。 $\hat{f}$ については

$$\forall x \in L^n \quad \hat{f}(x) = f(x)$$

$$\forall x \in (\tilde{L}^n - L^n) \quad \hat{f}(x) = N$$

と定める。

$\tilde{f}, \hat{f} \in NFS(f)$ とする。

定義 6. $NFS(f)$ を半順序集合として定義する。順序は

$g_1, g_2 \in NFS(f)$ に対して

$$g_1 \leq g_2 \Leftrightarrow \forall x \in \tilde{L}^n \quad g_1(x) \leq g_2(x)$$

$NFS(f)$ の最小元, 最大元は唯一に決って $\tilde{f}, \hat{f}$ となる。

定義 7. $g_1, g_2 \in NFS(f)$ に対して $g_1 \vee g_2, g_1 \wedge g_2$ を以下のように定義する。

$$g_1 \vee g_2(x) = \text{Max}\{g_1(x), g_2(x)\}$$

$$g_1 \wedge g_2(x) = \text{Min}\{g_1(x), g_2(x)\}$$

補題 1. $g_1 \in NFS(f), g_2 \in NFS(f)$ に対して

$$g_1 \vee g_2 \in NFS(f), g_1 \wedge g_2 \in NFS(f)$$

補題 2. $g_1, g_2, g_3 \in NFS(f)$ に対して

$$g_1 \vee g_2 = g_2 \vee g_1 = \text{Min}\{g \mid g \geq g_1, g \geq g_2\}$$

$$g_1 \wedge g_2 = g_2 \wedge g_1 = \text{Max}\{g \mid g \leq g_1, g \leq g_2\}$$

$$g_1 \wedge (g_2 \vee g_3) = (g_1 \wedge g_2) \vee (g_1 \wedge g_3)$$

$$g_1 \vee (g_2 \wedge g_3) = (g_1 \vee g_2) \wedge (g_1 \vee g_3)$$

補題 3. 代数系 $\langle NFS(f), \leq, \vee, \wedge \rangle$ は分配束となる。

定義 8. $g \in NFS(f)$ に対して $\bar{g}$ を次のように定義する。

$$\forall x \in L^n \quad \bar{g}(x) = f(x)$$

$$\forall x \in \tilde{L}^n \quad (((\tilde{f}(x) = D) \wedge (g(x) = N)) \Rightarrow \bar{g}(x) = D)$$

$$\forall x \in \tilde{L}^n \quad (((\tilde{f}(x) = D) \wedge (g(x) = D)) \Rightarrow \bar{g}(x) = N)$$

$$\forall x \in \tilde{L}^n \quad ((\tilde{f}(x) = N) \Rightarrow (\bar{g}(x) = N))$$

補題 4. $g \in NFS(f)$ に対して

$$\bar{g} \in NFS(f)$$

$$g \wedge \bar{g} = \tilde{f}, g \vee \bar{g} = \hat{f}$$

従って

定理 7. 代数系 $\langle NFS(f), \leq, \vee, \wedge \rangle$ はブール束となる。

定義 9. 2 値演算 $\bar{x}$, $\bar{x}$ を

$$\bar{x} = 0 \quad \text{for } x = 1$$

$$= 1 \quad \text{for } x = 0$$

$$\bar{x} = (\bar{x}_1, \dots, \bar{x}_n)$$

で定義する。

定理 2. 4 つのブール関数 $NFS(f(x))$, $NFS(\bar{f}(x))$, $NFS(f(\bar{x}))$, $NFS(\bar{f}(\bar{x}))$ は互いに同型である。

定義 10. $a_i \in (\tilde{L}^n - L^n)$, $b_j \in (\tilde{L}^n - L^n)$ $i=1, \dots$, $j=1, \dots$ に対し

$$G_{a_i}(x) = \begin{cases} N & \text{for } x = a_i \\ 1 & \text{for } x \neq a_i \end{cases}$$

$$H_{b_j}(x) = \begin{cases} N & \text{for } x = b_j \\ 0 & \text{for } x \neq b_j \end{cases}$$

補題 5. 任意の N -fail-safe 関数 $f' \in NFS(f)$ が

$$\tilde{f}(a_i) = 1, \quad f'(a_i) = N \quad i=1, 2, \dots$$

$$\tilde{f}(b_j) = 0, \quad f'(b_j) = N \quad j=1, 2, \dots$$

のとき $f'(x)$ は

$$f'(x) = \tilde{f}(x) \prod_i G_{a_i}(x) + \sum_j H_{b_j}(x)$$

と表わされる。ここに、積は $\prod$ を、和は $\sum$ を表す。

証明

$$f'(x) = \tilde{f}(x) \quad \text{for } x \neq a_i \\ x \neq b_j$$

$$f'(a_i) = \tilde{f}(a_i) G_{a_i}(a_i) = N$$

$$f'(b_j) = \tilde{f}(b_j) + H_{b_j}(b_j) = N$$

補題 6.

$$G_{a_i}(x) = G_{\delta_1}(x_1) + \dots + G_{\delta_n}(x_n) \\ \text{for } a_i = (\delta_1, \dots, \delta_n)$$

$$H_{b_j}(x) = H_{\sigma_1}(x_1) \cdot \dots \cdot H_{\sigma_n}(x_n) \\ \text{for } b_j = (\sigma_1, \dots, \sigma_n)$$

$= 1 = G, H$ は

x	G_0	G_1	G_N	H_0	H_1	H_N
0	N	1	1	N	0	0
1	1	N	1	0	N	0
N	1	1	N	0	0	N

で与えられる。

補題 7. $\bar{x}$ と $\bar{0} = 1, \bar{1} = 0, \bar{N} = N$ で定義すると

$$H_{\delta}(x) = \bar{G}_{\delta}(\bar{x}), \quad G_N(x) = x + \bar{x}, \quad G_1(x) = G_0(\bar{x})$$

以上より

定理 3. 関数 $\widetilde{NAND}$ と G_0 , または $\widetilde{NOR}$ と G_0 だけで
 N -fail-safe 関数の完全系をなす。