数理解析研究所講究録
第81巻 1970年 290-312

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。よくひきあいに出される例だが，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か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か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台並列に稼動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か誤りの0か識別不可能である。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，いかなる論理関数も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い得る。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。

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

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

定義1.1. 図1で表される誤りを抹消誤りという。

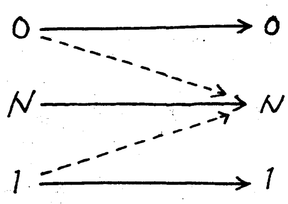


図1. 抹消誤り

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，$^\forall x\in L^n$ $f'(x)=f(x)$ であり，変数の変化 $x\in L^n\to y\in(\tilde{L}^n-L^n)$ 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 $f'(x)\to f'(y)$ がまた正しいか，抹消誤りのものをいう。

命題1.1. 関数 $f_1'(x_1),\cdots,f_m'(x_m)$，$g'(t_1,\cdots,t_m)$ がそれぞれ $f_1(x_1),\cdots,f_m(x_m)$，$g(t_1,\cdots,t_m)$ の N-Fail-Safe 関数とする。このとき，関数 $g'(f_1'(x_1),\cdots,f_m'(x_m))$ は，関数 $g(f_1(x_1),\cdots,f_m(x_m))$ 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。

ここでいくつかの記号を導入しておこう。

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

とし，変数集合 $x(i, r)$，$\bar{x}(i, r)$ のとる空間をそれぞれ，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 $x_{i_1} = N, \cdots, x_{i_r} = N$ 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 $\tilde{f}(x)$ を次のように定義する。

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

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，

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

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

$$f(x)=F(f_1(x),\cdots,f_m(x)) \qquad (2.1)$$

なる形に展開されたものとする。ここに，各 $f_i(x)$，および $F(t_1,\cdots,t_m)$ 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

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

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，

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

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 $\tilde{f}_{j_1},\cdots,\tilde{f}_{j_s}$ が値Nをとるものとする。ここで $\tilde{F}(x)=D$ と仮定し，

$$\mathfrak{f}=(f_1,\cdots,f_m)$$

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

とすると定義より

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

となる。一方 $x(i,r)$ が L_i^r のすべての点をつくすとき，$f_1,\cdots,f_m$ の中で値を変えるのは $f_{j_1},\cdots,f_{j_s}$ 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.

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

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 $f(x)$ の任意の次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が問題になる。

定義 2.1. 展開

$$f(x) = F(f_1(x_1), \cdots, f_m(x_m)) \tag{2, 3}$$

$$x_i \cap x_j = \phi \quad i \neq j \quad (i, j = 1, \cdots, m)$$

$$x = x_1 \cup \cdots \cup x_m$$

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

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

と表記する。

定理 2.3. 樹枝状展開(2.3)に対し，

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

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

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

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 $D = D'$ でなければならない。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。$x = (x(i, r), \bar{x}(i, r))$ とすると，定義より，

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

が成り立つ。ここで $x(i,r)\in L_i^r$ が L_i^r のすべての点をつくすとき $f_{j_1},\cdots,f_{j_s}$ がその値を変えるものとしよう。(もし，$f_1,\cdots,f_m$ のうちどれも値を変えなければ証明は明らか)。このとき，もちろん F 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，$f_{j_1},\cdots,f_{j_s}$ は互いに独立にその値を変えることになり，$\mathfrak{f}(j,s)$ は L_j^s のすべての点を尽くすことになる。このような状況の下で

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

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

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

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

となる。　　　　　　　　　　　　　　　　(証明終り)

系．任意の次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 $f(x)$ がもつ展開形式で，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

$$f(x) = f(0,0,\cdots,0)\bar{x}_1\bar{x}_2\cdots\bar{x}_n + f(0,1,\cdots,0)\bar{x}_1 x_2\cdots\bar{x}_n + \cdots + f(1,1,\cdots,1)x_1x_2\cdots x_n \tag{3.1}$$

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

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

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

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

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，これらの関数を完備 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，$x = (N(i,r), \bar{x}_\alpha(i,r))$ と仮定しよう。$x(i,r)$ が L_i^r のすべての点をつくすとき，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 $\tilde{C}_{a_i}(x_i)$ 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

$$\tilde{F}(x) = \tilde{\Sigma}\, \tilde{C}_{a_i}(x_i) = 0$$

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 $C_{\bar{x}_\alpha(i,r)}(\bar{x}(i,r))$ 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 $x = (N(i,r), \bar{x}_\alpha(i,r))$ 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

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

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1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 $f(x_1, x_2, x_3) = x_1x_2x_3 + x_1x_2\bar{x}_3 + \bar{x}_1x_2x_3 + \bar{x}_1\bar{x}_2x_3$

(2) 素項展開 $f(x_1, x_2, x_3) = x_1x_2 + x_2x_3 + \bar{x}_1x_3$

(3) 最簡形式 $f(x_1, x_2, x_3) = x_1x_2 + \bar{x}_1x_3$

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，

$$\overline{x_1+x_2} = \bar{x}_1 \cdot \bar{x}_2$$

$$\overline{x_1 x_2} = \bar{x}_1 + \bar{x}_2$$

が成り立つ．また，表7.2における$\tilde{f}_1$を"|"で，$\tilde{f}_7$を記号"↓"で表すと

$$x \mid x = \bar{x}, \quad x \downarrow x = \bar{x}$$

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

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

が成り立つ。

それ故

定理3.3. (−, +), (−, ·), "|", "↓" 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関数の相補二重系を構成してみよう。

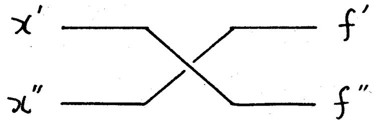
$$f(x)=\bar{x} \quad : \quad f'=x'', \ f''=x'$$

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。

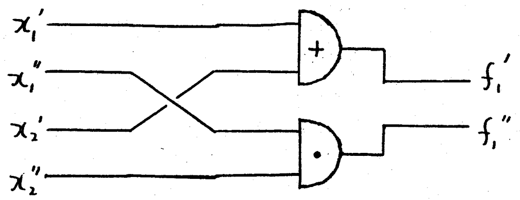
$$f_1'=x_1'+x_2' \ , \quad f_1''=x_1''\cdot x_2''$$

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

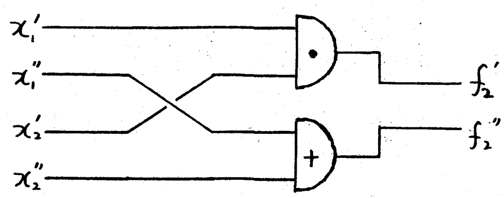
以上を図示しておこう。



否定



論理和



論理積

図 4.1. 代表的な完備N-Fail-Safe関数の相補二重系

上図において，二値のOR素子，AND素子ともに0側に非対称に誤るものとする。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。

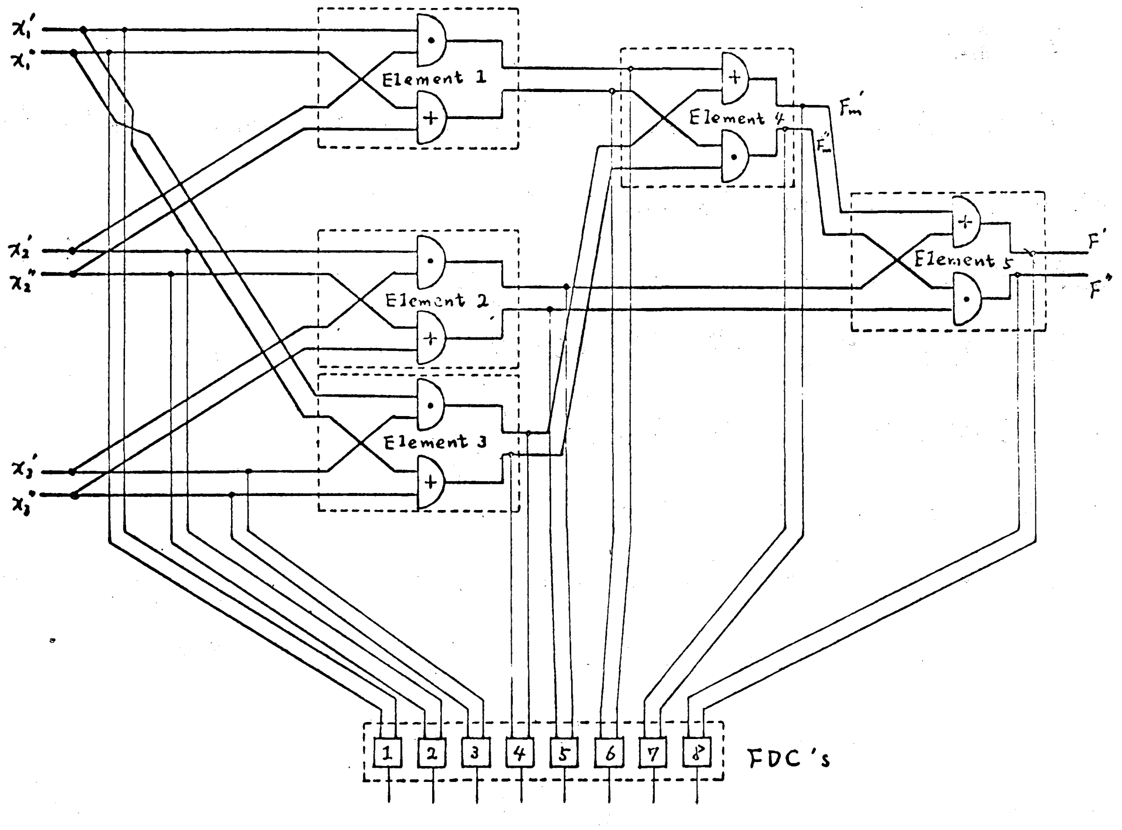


図 4.2. 例 3.1 の相補二重系

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イフ論理回路と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一構成法，" 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重系の一構成法," 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1）

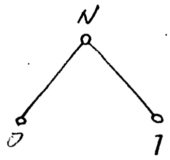


図 1.

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

$$x \leq x' \Leftrightarrow x_1 \leq x_1', \cdots, x_n \leq x_n'$$

とする。

定義3．

$$f : L^n \to L \quad (\text{2値関数})$$

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

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 $\tilde{f}$ および

$\hat{f}$を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

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

なら

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

そうでないとき，すなわち

$$\exists x', \exists x'' \in L^n \ (x' \le x) \wedge (x'' \le x) \wedge (f(x') \ne f(x''))$$

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。

$\tilde{f}, \hat{f} \in NFS(f)$となる。

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

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

$NFS(f)$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．

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

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

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

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

補題 2. $g_1, g_2, g_3 \in NFS(f)$ に対して

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

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

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

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

補題 3. 代数系 $\langle NFS(f), \leq, \vee, \wedge \rangle$ は分配束をなす。

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

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

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

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

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

補題 4. $g \in NFS(f)$ に対して

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

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

従って

定理 7. 代数系 $\langle NFS(f), \leq, \vee, \wedge \rangle$ はブール束をなす。

定義9. 2値演算 $\bar{x}$, $\bar{\mathbf{x}}$ を

$$\begin{aligned} \bar{x} &= 0 \quad \text{for } x = 1 \\ &= 1 \quad \text{for } x = 0 \\ \bar{\mathbf{x}} &= (\bar{x}_1, \cdots, \bar{x}_n) \end{aligned}$$

で定義する。

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

定義10. $\mathbf{a}_i \in (\tilde{L}^n - L^n)$, $\mathbf{b}_j \in (\tilde{L}^n - L^n)$ $i = 1, \cdots$, $j = 1, \cdots$ に対し

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

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

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

$$\begin{aligned} \tilde{f}(\mathbf{a}_i) &= 1, \quad f'(\mathbf{a}_i) = N \qquad i = 1, 2, \cdots \\ \tilde{f}(\mathbf{b}_j) &= 0, \quad f'(\mathbf{b}_j) = N \qquad j = 1, 2, \cdots \end{aligned}$$

のとき $f'(\mathbf{x})$ は

$$f'(\mathbf{x}) = \tilde{f}(\mathbf{x}) \prod_i G_{\mathbf{a}_i}(\mathbf{x}) + \sum_j H_{\mathbf{b}_j}(\mathbf{x})$$

と表わされる。ここに、積は $\tilde{\cdot}$ を、和は $\hat{+}$ 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.

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

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

ここに 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. $\bar{x}$ を $\bar{0} = 1$, $\bar{1} = 0$, $\bar{N} = N$ で定義すると

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

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。