Communication
Art
Technology
Systems
CATS
CESL

直交表 超入門（1）

— ソフトウェアテストの効率的網羅のための実験計画 —

キャッツ組込みソフトウェア研究所　川崎 秀二*

組込みソフトウェアにおいて、膨大なソフトウェアテストをどのように効率的に行うかというテストプロセスの設計は開発プロセス全体の中で、重要な課題となっている．ソフトウェアテストは、単体ごとに開発された機能を複数同時に組合わせて使用する際のテストである、機能組合わせテストをできるだけ網羅する事になる．機能の全組合わせ方についてテストケースを網羅する事は、テストケース数の多さから、不可能に近いか多大な工数を要する．本稿では、効率的でバランスの良いテストケース網羅法として知られている、直交表について基礎的事項を紹介する．

In embedded softwares, designing a test process that considers how to perform tests in an efficient way, has been an important problem in a whole development process. By the software test, we are to conduct those test cases that verify if actions of multiple functions simultaneously work correctly and to make the ratio of coverage of test cases as much as possible. Covering all the possible test cases, however, is usually impossible or close to it, or requires great amount of costs. As a solution to this, a design method of tests, called the orthogonal array, is known to provide an efficient and well-balanced coverage. In this paper, we give an introductory exposition of the orthogonal arrays.

1　はじめに — ソフトウェアテストの背景と現状

組込みソフトウェアの規模は、増大の一途を辿っている．コード行数で言えば、数百万〜数千万行に及ぶものも珍しくなくなった．それと同時に、商品として市場に出てからバグが報告されるケースが頻発するようになった．バグを抑えるには、開発プロセスに伴うテストプロセスが重要である事が認識されている [1].

テストは一般に、V 字モデルで言う単体機能のテストから統合機能のテストという流れに沿って行われるから、バグには、いくつかの機能や条件が同時に組み合わされた時に生じるものが多い．そのため多数の機能や条件の組合わせテストが必要になる．

ただ、一つの単体テストあるいは統合テストをするにおいても、しばしばテストすべき機能や条件が数十〜数百個程度ある．それらの全組合わせを総当たり的に網羅した実験計画は、組合わせの数が指数的に増加する事から、テスト自体に多大な時間・労力をかける事になり効率が悪いか、あるいは実質的に不可能でさえある．

これに対し、直交表による実験計画では、全組合わせの網羅を諦め、代償として少ないテスト数（テストケース数の増大度は高々線形的）で効率やバランスの良い実験計画を行うものである．

上述の機能組合わせテストでは、網羅率を高めようとすれば、単機能テストから始まり、2機能間組合わせテスト、3機能間組合わせテスト、$\cdots$ と多数のテストが必要になる．これらの m 機能間組合わせのテストが、開発案件で考えられる最大の組合わせ数 M まで全て必要か？と言えば、必ずしもそうではない．バグが m 機能間組合わせに含まれている可能性は、m が小さいほど高い事が知られている [10]．実際、次のような指針が与えられている．

- 2機能間までの組合わせテストにより、70〜90%のバグを発見できる
- 3機能間までの組合わせテストにより、90〜99%のバグを発見できる

*キャッツ組込みソフトウェア研究所, CATS Embedded Software Laboratory

因子	水準 1	水準 2
OS	Windows Vista	Windows XP
ソフトウェア	PowerPoint2003	Word2003
互換性	上位への	下位への

(a)

Test No.	OS	ソフトウェア	互換性
1	Windows Vista	PPT2003	下位互換性
2	Windows Vista	Word2003	上位互換性
3	Windows XP	PPT2003	上位互換性
4	Windows XP	Word2003	下位互換性

(b)

表 1: ソフトウェアの各 OS での上位／下位互換性のテスト：(a) 組合わせテストしたい因子とその水準 (b) 直交表 L_4 へのテストケースの割付け

- ６機能間までの組合わせテストにより、ほぼ100%のバグを発見できる
- 多くの場合、４機能組合わせテストまでを考慮すれば良い

従って、小さい m から高い網羅率でバランス良く、順次網羅されているような実験計画を組むのが妥当な考え方であろう．直交表は、以下に紹介するように、２機能間組合わせを 100%網羅し、３機能間以上の組合わせを出来るだけ高い網羅率でバランス良く、かつ少ないテスト数で実現するような、効率の良いテストを実現する．

2 概要 — 直交表とは

生産や設計の現場において、実験により様々な条件・要因・パラメータ等の項目の組合わせによる効果の違いをテストしたいとする．各項目はそれぞれ幾つかの値を取り得るであろう．この項目の事を**因子**、取り得る値の事を**水準**という．ソフトウェアテストの場合は、この「効果の違い」として機能組合せによるバグが生じないかをテストしたいという事になる．

Test No.	OS	ソフトウェア	互換性
1	Windows Vista	PPT2003	上位互換性
2	Windows Vista	Word2003	下位互換性
3	Windows XP	PPT2003	下位互換性
4	Windows XP	Word2003	上位互換性

表 2: 表 1(b) に現れていない組合わせ

2.1 直交表とはどんなものか

いま、例題として 2 つのソフトウェア PowerPoint2003 と Word2003 で作成されたファイルの、上位バージョン（PowerPoint2007, Word2007）および下位バージョン（PowerPoint2000, Word2000）への互換性について、テストしたいとする．因子および水準として、表 1(a) のようなテスト項目を仮定する．この３因子につき、もし水準値の全組合わせをテストするなら、

$$2 \times 2 \times 2 = 8$$

通りのテストケースを実施する事になる．

これに対し、直交表 （L_4 と呼ばれるもの）を用いた組合わせを構成したのが表 1(b) である．例えば、テストケース１では、（OS、ソフトウェア、互換性）＝（Windows Vista, PPT2003, 下位への互換性）という組合わせでのテストを行う．この４行のテストケースにより、

- OS とソフトウェアの全組合わせ４通り
- ソフトウェアと互換性の全組合わせ４通り
- 互換性と OS の全組合わせ４通り

がちょうどうまく組合されている事に注意する．この「２因子組合わせの網羅率は 100%」である事が、直交表の特長である．

では、表 1(b) に現れていない組合わせはどうなるのか？ それを知るために、いま、テストケース１を標準条件（全てデフォルト）と見做して、現れていない組合わせを見てみる（表 2）．表 2 の４行目は、３因子ともデフォルト以外の水準値にしたものである．１行目〜３行目では１因子のみがデフォルト以外であり、つまり、その因子のデフォルトでない水準値のテストケースになっている．これは、他の２つの因子をデフォル

トに固定した上での、その１因子の単機能テストである．ソフトウェアテストでは、単純な組合わせのテストから順番に複雑な組合わせへと実施される．従って、２因子間の組合わせテストの前段階で、単機能テストが出来ている事が一般に想定されており、表２の１行目〜３行目のテストケースはテスト済みである．故に、全組合わせによるテストケース８通りのうち、漏れのテストケースは表２の４行目のみ、という事になる．この場合、３因子間組合わせの網羅率は $7/8 = 87.5\%$ である．しかも、全組合わせの場合のテストケース８に対し、テストケース数は４で済む．

要約すると、

- ２因子間組合わせは 100%網羅されている
- ３因子間組合わせでの網羅率も高い
- テストケース数は全組合わせより少ない

これが直交表の特長である．

2.2 直交表の定義と性質

$S = \{0, 1, \cdots, s-1\}$, $s \in \mathbb{N}$ とおく．S は水準の集合、s は水準数に対応する．直交表のフォーマルな定義は次のように与えられる．

直交表の定義 [9]

S の元を要素とする $N \times R$ の２次元配列が直交表であるとは、ある自然数 t, λ が存在して、任意の $N \times t$ 部分配列 $(0 \leq t \leq R)$ が、行方向の t 個の要素の組に対し、同じ組を λ 回ずつ含んでいる事をいう．

t, λ はそれぞれ A の強度 (intensity)、指数 (index) と呼ばれる．定義から λ は自動的に

$$\lambda = N/s^t \tag{1}$$

と求まる．s^t は、S の要素の t 個の並びの組の種類の数である．簡単な考察から

- 行あるいは列の順序を入れ替えても同じパラメータの直交表となる
- 任意の列の水準ベクトルの要素の順序を入れ替えても同じパラメータの直交表となる

等、いくつかの性質が分かる．

例．２水準系の直交表 $L_4(2^3)$, $L_8(2^7)$.

図１ (a),(b) はそれぞれ２水準系の直交表

- L_4 $(N = 4, R = 3)$,
- L_8 $(N = 8, R = 7)$

である．L_4 では AB, AC, BC のどの２列をとっても $(0,0), (0,1), (1,0), (1,1)$ が１回ずつ現われている．また、L_8 では、AB, BC, $\cdots$, FG のどの２列をとってもこれらの４パターンが２回ずつ現われている．従って、L_4 は強度２、指数１の直交表、L_4 は強度２、指数２の直交表である．

テストケース数に関しては、機能数の増加とともに必要なテストケース数は線形的に増加するが、全組合わせでは傾きが２機能組合わせで４、３機能組合わせで８、$\cdots$ であるのに対し、直交表ではほぼ１である [1].

「直交表」という名前は、定義からはイメージしにくいが、各列ベクトル同士の相互相関係数が全て０になる事を意味している（付録を参照）．もちろん、自分自身の列との自己相関係数は１である．これは、直交表においては各列の因子が互いに独立である事を意味する．線形空間では正規直交基底が最も無駄の無い基底であったが、この相関の直交性が、効率的で無駄の無いテストケース構成に繋がっている．

2.3 直交表のサイズ

テストすべき因子・水準がリストアップされたら、適用する直交表の必要なサイズ（行数）が次のように決まる．いま、各因子 $F_j, j = 1, \cdots, J$ がそれぞれ S_j 個の水準を持つとする．このとき、$S_j - 1$ を因子 F_j の**自由度**という．基本的には、少なくとも因子の自由度の和+1 だけの行数が要ることになる．最後の＋１は標準条件（全因子の水準がデフォルト値）でのテストケースに対応する．

これに対し、N 水準系の直交表を適用するには、この行数を、最も近い N の冪乗へ切上げる．すなわち、

$$K = 1 + \sum_{j=1}^{J} (S_j - 1) \tag{2}$$

テストケース＼因子	A	B	C
1	0	0	0
2	0	1	1
3	1	0	1
4	1	1	0

(a)

テストケース＼因子	A	B	C	D	E	F	G
1	0	0	0	0	0	0	0
2	0	0	0	1	1	1	1
3	0	1	1	0	0	1	1
4	0	1	1	1	1	0	0
5	1	0	1	0	1	0	1
6	1	0	1	1	0	1	0
7	1	1	0	0	1	1	0
8	1	1	0	1	0	0	1

(b)

図 1: 2 水準系直交表 (a)L_4 (b) L_8

とおくと、N 水準系直交表を適用する際の必要なサイズは

$$N^{\lceil \log_N K \rceil} \tag{3}$$

で与えられる．ただし、$\lceil x \rceil$ は実数 x の整数への切上げである．特に (2) で、リストアップされた因子の水準数が全て同じ（$S_j \equiv S$）であれば

$$K = 1 + (S-1)J \tag{4}$$

となる．さらに $S = 2$ なら $K = 1 + J$ である．

(2) から分かるように、因子数 J が $J+1$ に増加したとき、K の増加分は F_{J+1} の自由度 $S_{J+1} - 1$ だけであり、水準数が同じなら $S-1$ だけである．$S = 2$ あるいは 3 なら増加分は 1 あるいは 2 だけである．

一方、全組合わせでは、

$${}_{J+1}C_m - {}_JC_m = {}_JC_{m-1} \tag{5}$$

であり、例えば $J = 32$ とすると、2 因子間組合わせ ($m = 2$) で ${}_{32}C_1 = 32$ 通り増加し，3 因子間組合わせ ($m = 3$) で ${}_{32}C_2 = 496$ 通り増加する事になる．このように、直交表によるテストケース数は、全組合わせよりも少なく、J や m が大きくなると、その差はかなり大きくなる．

なお、必要な行数の見積もりとして、水準数が最大となる上位 2 つの因子を取ってそれらの水準数の積（2 因子の水準組合わせ数）を下限とする事もしばしば用いられる．

3 原理 — 直交表の成立ち

前述のように、本稿の対象とする直交表は N 水準系と混合系である．N 水準系は、各因子の水準数が同じ（$= S$）であり、$L_N(S^r)$ のように記す．

本節では、N 水準系および多水準系の直交表の作成の仕方を述べる．N 水準系直交表の作成の仕方は、多水準系の場合の基礎にもなっている．

3.1 N 水準系

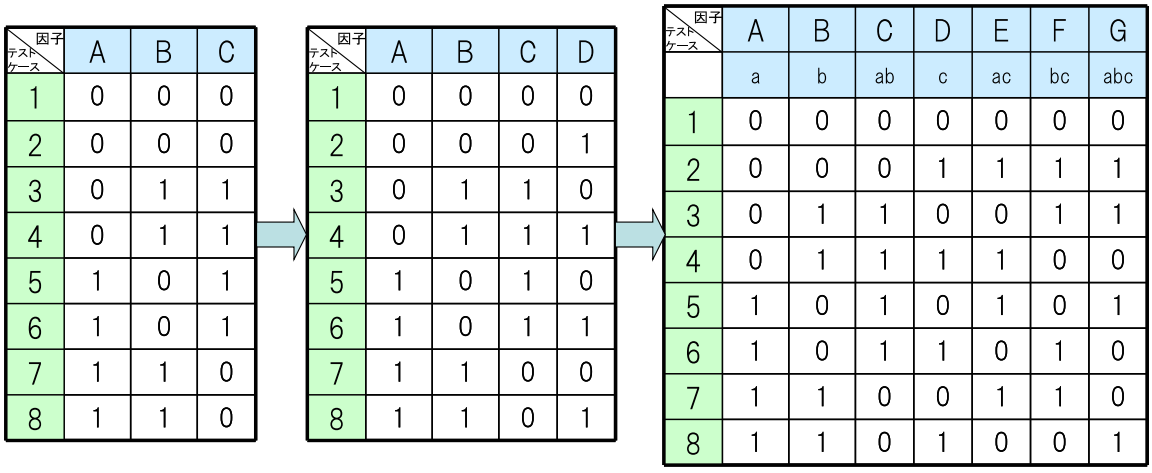
簡単のため、$N = 2$ について、最小の直交表である L_4 を拡張して L_8 を、次に L_8 を拡張して L_{16} を、$\cdots$ と順次作るやり方を述べる．$N \geq 3$ についても同様である．

$\{0, 1\}$ の 2 値を取る論理変数 A, B の排他的論理和 (XOR) $A \oplus B$ は以下のような演算であった事を思い出しておこう：

$$\begin{aligned} A \ \oplus \ B &= \bar{A} \cdot B + A \cdot \bar{B} \\ &= \begin{cases} 1 & \text{if } A = B \\ 0 & \text{if } A \neq B, \end{cases} \end{aligned} \tag{6}$$

ただし、$\bar{A} = (A \text{ の否定})$.

L_4 から L_8 を作るやり方は次の通り：



テスト＼因子	A	B	C
1	0	0	0
2	0	0	0
3	0	1	1
4	0	1	1
5	1	0	1
6	1	0	1
7	1	1	0
8	1	1	0

テスト＼因子	A	B	C	D
1	0	0	0	0
2	0	0	0	1
3	0	1	1	0
4	0	1	1	1
5	1	0	1	0
6	1	0	1	1
7	1	1	0	0
8	1	1	0	1

テスト＼因子	A	B	C	D	E	F	G
	a	b	ab	c	ac	bc	abc
1	0	0	0	0	0	0	0
2	0	0	0	1	1	1	1
3	0	1	1	0	0	1	1
4	0	1	1	1	1	0	0
5	1	0	1	0	1	0	1
6	1	0	1	1	0	1	0
7	1	1	0	0	1	1	0
8	1	1	0	1	0	0	1

図 2: 直交表 L_4 から L_8 を作成; 左から順に (i), (ii), (iii) の手順を行ったもの

L_4 から L_8 を作る

(i). L_4 の各行（行 1.〜4.）を複製して上から順に新 1 行目=旧 1 行目、新 2 行目=旧 1 行目、新 3 行目=旧 2 行目、新 4 行目=旧 2 行目、$\cdots$、新 8 行目=旧 4 行目、のようにし、8 行 3 列の表とする

(ii). ${}^t[0\ \ 1\ \ 0\ \ 1\cdots]$ からなる一列を第 4 列目（D 列とする）に追加し、8 行 4 列の行とする

(iii). 第 5 列〜第 7 列（それぞれ E, F, G 列とする）を

(A 列) $\oplus$ (D 列) = (E 列)
(B 列) $\oplus$ (D 列) = (F 列)
(C 列) $\oplus$ (D 列) = (G 列)

により追加し、8 行 4 列の行とする

二列間の XOR は、二列の各行成分に対して取る．図 2 に、手順 (i)-(iii) による L_4 から L_8 の作成を示す．同様の手続きで、L_8 から L_{16}、L_{16} から L_{32}、$\cdots$ を作る事ができる．

直交表における、上記のような二列間の XOR は、その二因子の**交互作用**成分を表す．図 2 で、L_8 の各列の成分を a, b, ab 等で示してある．a, b は原因子を、ab は交互作用を表す．

この手順により、先ず $A = a$, $B = b$, $C = ab$ を互いに直交する 3 列とし、そこへ $D = c = {}^t[0\,1\,0\,1\,0\,1\,0\,1]$ という縦ベクトルを選んで $A = a$, $B = b$, $C = ab$, $D = c$ の 4 列が互いに直交するようにする．次いで、$E = ac$ により $A = a$, $B = b$, $C = ab$, $D = c$, $E = ac$ の 5 列が互いに直交するようにする．このようにして、L_8 は直交表となっている．

別の見方をすれば、$A = a, D = c, E = ac$ も互いに直交する 3 つの列であるが、ここで $c = a \cdot ac$ と見做せるから、$D = c, A = a, E = ac$ の 3 列は互いに直交するという事になる．$D = c$ を $B = b, F = bc$ から $c = b \cdot bc$ により作ったと見做せば、同様に $B = b, F = bc$ から $c = b \cdot bc$ $D = c, B = b, F = bc$ の 3 列は互いに直交するという事にもなる．

3.2 多水準系

次に、多水準系直交表の作り方として、最も基本的な 2 水準系に 4 水準の因子を作るには、4 水準因子の自由度は 3 であるから、任意の 2 列とそれらの交互作用からなる 3 列をまとめて 1 つの列とすれば良い．それは、このような 3 列のまとまりと残りの列が $t = 2$ として互いに直交しているからである．実際、図 2 の、作成された L_8 でいうと、$A = a$, $B = b$, $C = ab$ の 3 列をまとめて ABC 列とすると、例えば (ABC, D) は $(000, 0)$, $(000, 1)$, $\cdots$, $(110, 1)$ が全て 1 回ずつ現れている．これは (ABC, E) 等の他の列との組合わせについても同じで、つまり、$\lambda = 1$ の新たな直交表になっている．

因子 / テストケース	ABC	D	E	F	G
1	000	0	0	0	0
2	000	1	1	1	1
3	011	0	0	1	1
4	011	1	1	0	0
5	101	0	1	0	1
6	101	1	0	1	0
7	110	0	1	1	0
8	110	1	0	0	1

図 3: $L_8(2^7)$ の因子をまとめて 4 水準の因子を持つ直交表にする

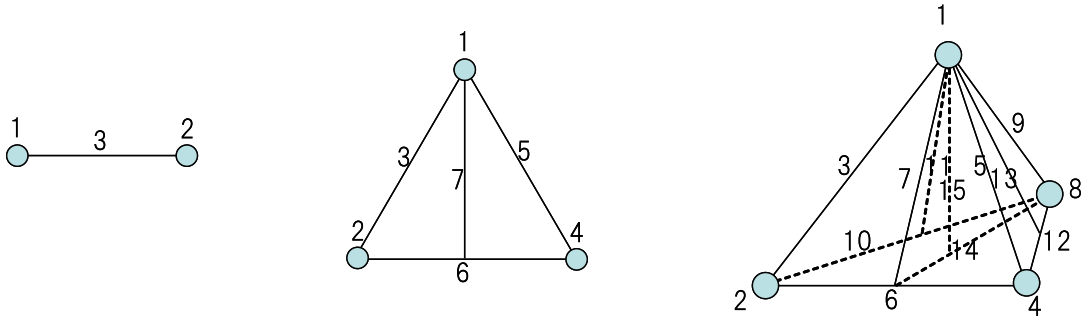


図 4: 2 水準系の線点図： 左から 4 水準、8 水準、16 水準

ここで、でたらめに3列を組合わせてまとめる事はできない事に注意する．例えば、$B = b$, $C = ab$, $D = c$ をまとめて BCD 列とすると、$BCD = 001$（2行目）に対し、(BCD, E), (BCD, F), (BCD, G) の組合わせでは「$(001, 1)$ が1回」のみ生じているが、(BCD, A) の組合わせでは「$(001, 0)$ が1回」のみ生じており共通の指数 λ が存在しない．まとめられるのは、原因子とその交互作用因子からなる列である．

3.3 線点図

3.1, 3.2 節のようにして定まる直交表の型は、可能なパターンが決まっているものであり、ユーザがその都度新たに考案する必要は無い．ユーザの行うべき事は、何を主効果あるいは交互作用として列に配置するかを定め、それに対し適切な直交表パターンを選択する事である．その選択の際に参考となるのが、**線点図**である．

線点図は、いくつかの頂点とそれらを結ぶ辺からなる図形である．頂点が主効果となる原因子を表し、辺が交互作用因子を表す．図4に、2水準系の線点図 [7] を示す．

2水準因子をまとめて多水準因子を作るのに、(a) は4水準因子を作る場合に第1列と第2列および交互作用の第3列をまとめたものを取る事を表している．得られる4水準因子の自由度は $(2-1)\times 3 = 3$ となる．(b) は8水準因子を作る場合に第1列と第2列および交互作用の第3列、…、第1列と第6列および交互作用の第7列をまとめたものを取る事を表している．得られる8水準因子の自由度は $(2-1)\times 7 = 7$ となる．同様に、(c) は16水準因子を作る場合のまとめるべき列を表しており、得られる16水準因子の自由度は15となる．

図6上は2水準系の $L_{16}(2^{15})$（各列の成分付き）であり、その下は3つずつ列をまとめて4水準系の $L_{16}(4^5)$ を作ったものである．因子の $ABC(1/2/3)$ 等は、元の第1列、第2列、第3列である A 列、B 列、C 列をまとめたものである．

渡辺による「組込みソフトウェア開発課題への挑戦〜網羅度〜」 [8] の 3.2 節では、カーオーディオの3つの操作ボタンの機能組合わせテストの事例における4水準機能の割当てに関し、(1,2,3) 列、(4,8,12) 列などを1つのまとまりと捉えて列の割当てを行っているのは、16水準直交表が、図6上に示されているような、2つの列とその交互作用という構成になっているからである．(1,2,3) 列、(4,8,12) 列、(5,10,15) 列、(6,11,13) 列、(7,9,14) 列は互いに直交する4水準因子になっている．

このように、直交表の列は原因子と交互作用因子からなっており、これにより直交性がもたらされている．行数と列数の必然性もここにある．行数 N に対し、列数は

$$\begin{aligned}&\big(\text{原因子の列数}\big) + \big(1\text{ 次の交互作用の列数}\big)\\ &\quad + \big(2\text{ 次の交互作用の列数}\big) + \cdots < N\end{aligned} \tag{7}$$

となる左辺の和のうち、最大のものである．例えば、$N = 4$ の $L_4(2^3)$ では

$$\big(\{a, b\}\text{ の }2\text{ 列}\big) + \big(\{ab\}\text{ の }1\text{ 列}\big) = 3\text{ 列} < 4 \tag{8}$$

であり、同様に $L_8(2^7)$ では

$$\begin{aligned}&\big(\{a, b, c\}\text{ の }3\text{ 列}\big)\\ &\quad + \big(\{ab, bc, ac\}\text{ の }3\text{ 列} = {}_3C_2\big)\\ &\quad + \big(\{abc\}\text{ の }1\text{ 列} = {}_3C_3\big)\\ &= 7\text{ 列} < 8\end{aligned} \tag{9}$$

また $L_{16}(2^{15})$ では

$$\begin{aligned}&\big(\{a, b, c, d\}\text{ の }4\text{ 列}\big)\\ &\quad + \big(\{ab, ac, ad, bc, bd, cd\}\text{ の }6\text{ 列} = {}_4C_2\big)\\ &\quad + \big(\{abc, acd, bcd, abd\}\text{ の }4\text{ 列} = {}_4C_3\big)\\ &\quad + \big(\{abcd\}\text{ の }1\text{ 列} = {}_4C_4\big)\\ &= 15\text{ 列} < 16\end{aligned} \tag{10}$$

である．

4 網羅率

複数の因子の中には、両立しない組合わせが一般には存在する．そのような組合わせを禁則という．例えば、あるワープロソフトにおける書式指定「箇条書きと段落番号」の「箇条書き」と「段落番号」は同時に設定する事ができないという事例があり、この2つの項目を因子として採用する場合には、両方を同時に”ON”

因子 / テストケース	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	a	b	ab	c	ac	bc	abc	d	ad	bd	abd	cd	acd	bcd	abcd
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
3	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
4	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
5	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
6	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0
7	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
8	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1
9	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
10	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0
11	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0
12	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1
13	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
14	1	1	0	0	1	1	0	1	0	0	1	1	0	0	1
15	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
16	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0

因子 / テストケース	ABC (1/2/3)			DHL (4/8/12)			EJO (5/10/15)			FKM (6/11/13)			GIN (7/9/14)		
	a	b	ab	c	d	cd	ac	bd	abcd	bc	abd	acd	abc	ad	bcd
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	1	1	0	1	1	0	1	1	0	1	1
3	0	0	0	1	0	1	1	0	1	1	0	1	1	0	1
4	0	0	0	1	1	0	1	1	0	1	1	0	1	1	0
5	0	1	1	0	0	0	0	1	1	1	1	0	1	0	1
6	0	1	1	0	1	1	0	0	0	1	0	1	1	1	0
7	0	1	1	1	0	1	1	1	0	0	1	1	0	0	0
8	0	1	1	1	1	0	1	0	1	0	0	0	0	1	1
9	1	0	1	0	0	0	1	0	1	0	1	1	1	1	0
10	1	0	1	0	1	1	1	1	0	0	0	0	1	0	1
11	1	0	1	1	0	1	0	0	0	1	1	0	0	1	1
12	1	0	1	1	1	0	0	1	1	1	0	1	0	0	0
13	1	1	0	0	0	0	1	1	0	1	0	1	0	1	1
14	1	1	0	0	1	1	1	0	1	1	1	0	0	0	0
15	1	1	0	1	0	1	0	1	1	0	0	0	1	1	0
16	1	1	0	1	1	0	0	0	0	0	1	1	1	0	1

図 5: $L_{16}(2^{15})$ から $L_{16}(4^5)$ を作成

因子 / テストケース	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	a	b	ab	c	ac	bc	abc	d	ad	bd	abd	cd	acd	bcd	abcd
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
3	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
4	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
5	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
6	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0
7	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
8	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1
9	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
10	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0
11	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0
12	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1
13	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
14	1	1	0	0	1	1	0	1	0	0	1	1	0	0	1
15	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
16	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0

因子 / テストケース	ABC (1/2/3)			DHL (4/8/12)			EJO (5/10/15)			FKM (6/11/13)			GIN (7/9/14)		
	a	b	ab	c	d	cd	ac	bd	abcd	bc	abd	acd	abc	ad	bcd
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	1	1	0	1	1	0	1	1	0	1	1
3	0	0	0	1	0	1	1	0	1	1	0	1	1	0	1
4	0	0	0	1	1	0	1	1	0	1	1	0	1	1	0
5	0	1	1	0	0	0	0	1	1	1	1	0	1	0	1
6	0	1	1	0	1	1	0	0	0	1	0	1	1	1	0
7	0	1	1	1	0	1	1	1	0	0	1	1	0	0	0
8	0	1	1	1	1	0	1	0	1	0	0	0	0	1	1
9	1	0	1	0	0	0	1	0	1	0	1	1	1	1	0
10	1	0	1	0	1	1	1	1	0	0	0	0	1	0	1
11	1	0	1	1	0	1	0	0	0	1	1	0	0	1	1
12	1	0	1	1	1	0	0	1	1	1	0	1	0	0	0
13	1	1	0	0	0	0	1	1	0	1	0	1	0	1	1
14	1	1	0	0	1	1	1	0	1	1	1	0	0	0	0
15	1	1	0	1	0	1	0	1	1	0	0	0	1	1	0
16	1	1	0	1	1	0	0	0	0	0	1	1	1	0	1

図 6: $L_{16}(2^{15})$ から $L_{16}(4^5)$ を作成

に設定するようなテストケースを排除しなければならない.

そのような禁則を排除した上で、構成した直交表の r 因子間組合わせ網羅率 C_r （$1 \leq r \leq R$）は

$$C_r = \frac{p_r}{a_r - b_r} \tag{11}$$

で定義される [1]. ただし、

- p_r: 直交表に出現する r 因子間の組合わせ数
- a_r: 全ての r 因子間の組合わせ数
- b_r: r 因子間の禁則の組合わせ数.

さきに述べたように、ソフトウェアテストでは、単因子テスト ($r = 1$) から始まり、直交表によるテストフェーズでは 2 因子間 ($r = 2$) の組合わせ網羅率を 100%にし、出来るだけ 3 因子間 ($r = 3$) の組合わせ網羅率をも高める事を考える. どの r までの組合わせ網羅を考慮するかは、因子の重要性や機能のクリティカリティ等、およびテストの規模に対する効果と効率性に鑑み、案件ごとに判断する.

5 むすび

本稿では、ソフトウェアテストのための有効な実験計画技術である直交表の紹介の第 1 回目として、ごく基礎的な事項を説明した. 次回は、応用として状態遷移パスに関わる因子を直交表に割り付ける事によって、各パスのテストがなされる事を説明する.

A 付録

- 直交表の定義から相関係数の意味での直交性「異なる 2 列の相互相関係数が 0」が導かれる事

第 j 列の要素を $s_{j,n}$, $n = 1, \cdots, N$ で表す. 任意の第 j 列、第 i 列に対し直交性を示すだけなら相互相関係数の分子である共分散

$$\sum_{n=1}^{N} (s_{j,n} - \bar{s}_j) \cdot (s_{i,n} - \bar{s}_i)$$

が 0 になる事を示せばよい. ただし、$\bar{s}_j = (1/N) \sum_{n=1}^{N} s_{j,n}$ であるが、各列で成分が同じ回数だけ出現しているので、$\bar{s}_j = \bar{s}_i \equiv \bar{s}$ である. いま、簡単のため、$t = 2$ の場合につき示す. $t \geq 3$ の場合も基本的には同様である.

直交表の異なる 2 列からなる部分配列 $N \times 2$ には、

$$(s_{j,n}, s_{i,n}) = \{(0,0), (0,1), \cdots, (s-1, s-1)\}$$

の s^2 個の組が、各組 λ 個ずつある. 従って、共分散は

$$\sum_{n=1}^{N} (s_{j,n} - \bar{s})(s_{i,n} - \bar{s}) = \lambda \Big[(0-\bar{s})(0-\bar{s}) + \cdots + (s-1-\bar{s})(s-1-\bar{s}) \Big]$$

と書け、さらに $s_{j,n} = 0$ から $s-1$ までの値で仕分けすると

$$= \lambda \Big[\Big\{ (0-\bar{s})(0-\bar{s}) + \cdots + (0-\bar{s})(s-1-\bar{s}) \Big\} + \cdots + \Big\{ (s-1-\bar{s})(0-\bar{s}) + \cdots + (s-1-\bar{s})(s-1-\bar{s}) \Big\} \Big].$$

このとき、$\Big\{ (0-\bar{s})(0-\bar{s}) + \cdots + (0-\bar{s})(s-1-\bar{s}) \Big\}$ は 0 となる. { } の中は、互いに符号の反転した項のペアからなり、キャンセルするからである. $s_{j,n} = 1, \cdots, s-1$ についても同様に { } の中は 0 となる.

参考文献

[1] 吉澤、秋山、仙石. ソフトウェアテスト — Hayst 法入門、日科技連出版、2007.

[2] リー・コープランド、宗 訳. はじめて学ぶソフトウェアのテスト技法、日経 BP 社、2005.

[3] ボーリス・バイザー、小野間・山浦 訳. ソフトウェアテスト技法 — 自動化・品質保証、そしてバグの未然防止のために、日経 BP 出版センター、1994.

[4] 秋山．直交表を活用したソフトウェアテストの効率化—HAYST 法の活用—、http://www.swtest.jp/jasst05w/S4-1.pdf, JaSST'05, 2005.

[5] 山本、秋山．直交表を利用したソフトウェアテスト—HAYST 法—、JaSST'04, 2004.

[6] 高橋他．組合わせテストを用いたソフトウェアテストとその限界、http://www.juse.or.jp/software/pdf/22_spc/5/5_2_1_report.pdf

[7] 田口、小西．直交表による実験のわりつけ方 —例題と演習、日科技連出版、1959.

[8] 渡辺．組込みソフトウェア開発課題への挑戦〜網羅度〜, キャッツ株式会社 CESL 資料, 2009.

[9] A. S. Hedayat, N. J. A. Sloane and J. Stuflen, *Orthogonal Arrays: Theory and Applications*, Springer, 1999.

[10] D. R. Kuhn and D. R. Wallace, A. M. Gallo. "Software Fault Interactions and Implications for Software Testing", IEEE Trans. Software Eng., VOL. 30, pp.418-421, 2004.

川崎　秀二

1991 年 電気通信大学・電気通信・通信工 卒．1993 年慶大・理工・電気 修士．1996 年慶大・理工・数理 単位取得退学．博士（工学）．日本数学会会員．

現在、キャッツ株式会社 CESL（組込みソフトウェア研究所）研究員．