サーベイ論文

ペアワイズテスト——ソフトウェアテストの効率化を求めて——

土屋　達弘[†a)]　　菊野　亨[†]

A Survey of Pairwise Testing

Tatsuhiro TSUCHIYA[†a)] and Tohru KIKUNO[†]

あらまし　本論文は，ペアワイズテストと呼ばれるソフトウェアテスト手法に関するサーベイである．ペアワイズテストとは，ソフトウェアの入力パラメータの相互作用に注目し，フォールトを効率良く顕在化する手法である．具体的には，入力パラメータのペアのすべてに対して，それらがとり得る値の組合せすべてを網羅するようにテストを行う．ここでは，ペアワイズテストの定義と，この手法の背後にある根拠について述べた後，ペアワイズテストの条件を満たすテスト集合の構成問題について論じる．更に，ペアワイズテストの有効性について，知られている研究成果を紹介する．

キーワード　ペアワイズテスト，全ペアテスト，カバリングアレー，組合せデザイン，組合せテスト

1. まえがき

ソフトウェアの不具合，すなわち，フォールトを検出するためのテストは，ソフトウェアシステムの開発において不可欠であると同時に，多くの時間・コストを必要する．したがって，フォールト検出能力に優れたテストケースの作成は，限られたコストで高い信頼を有するシステムを実現する上で非常に重要な課題である．

このような問題を解決する一つの有効な手段として，ペアワイズテスト（pairwise testing，全ペアテスト（all-pairs testing））が注目を集めている．具体的には，あらゆる 2 個の入力パラメータについて，とり得る値の組合せすべてを網羅するようにテストを行うことで，効果的にフォールトを発見するという手法である．この手法の背景にある基本的な根拠は，フォールトの多くは，少数の特定のパラメータ値の組合せによって顕在化するという知見である．近年出版されたソフトウェアテストに関する書物のほとんどにおいて，ペアワイズテストの説明があり，多くは独立した章が割かれている [1]〜[5]．

† 大阪大学大学院情報科学研究科，吹田市
Graduate School of Information Science and Technology, Osaka University, 1–5 Yamadaoka, Suita-shi, 565–0871 Japan
a) E-mail: t-tutiya@ist.osaka-u.ac.jp

ペアワイズテストの自然な拡張として，2 パラメータではなく k $(= 2, 3, \cdots)$ 個のパラメータについて，パラメータ値の組合せを網羅するという方法論も存在する．このような場合は，k-way テストと呼ばれる．以降，k 個のパラメータについて値の組合せすべてが網羅されるという基準を，k-way カバレッジ（k-way coverage），特に $k = 2$ の場合をペアワイズカバレッジと呼ぶものとする．また，こうしたパラメータ値の組合せを網羅することを基準としたテスト手法全般を，組合せテスト（combinatorial testing）と呼ぶものとする．

ただし，ペアワイズカバレッジを達成することで，フォールトを効率良く検出できるとしても，少数のテストケースでその基準を満たせなければ，この手法の有効性は消失してしまう．そのため，いかに少数のテストケースによって，ペアワイズカバレッジを達成するかについて，活発に研究が行われている．

本論文では，主に $k = 2$ の場合，すなわち，ペアワイズテストを中心に，関連する研究結果について説明する．まず，**2.** では，ペアワイズテストの定義と，適用例について述べる．**3.** では，ペアワイズカバレッジを満たすテスト集合を作成する方法について説明する．以降，この条件を満たすテスト集合をペアワイズテスト集合と呼ぶ．**4.** では，ペアワイズテストの有効性に関する研究結果について紹介する．最後に，**5.** で結論と今後の展望について述べる．

2. ペアワイズテスト

2.1 基本概念

ペアワイズテストを議論する際に通常想定されるモデルでは，システムはいくつかの入力パラメータからなり，各入力パラメータにはとり得る値の有限集合が定められている．テストケースは，すべての入力パラメータについて，とり得る値を一つ定めることによって得られるベクトルである．このとき，ペアワイズカバレッジを満たすテスト集合は，どの二つの入力パラメータのとり得るどの値の組合せも，少なくとも一つのテストケースに現れるようなテストケースの集合と定義できる．

ペアワイズテストについて，表 1 で表される単純化されたテストのモデルを例に説明する．ここでは，ウェブブラウザを通じて何らかの操作を行うシステムを想定している．このモデルでは 4 種類のパラメータが存在しており，それらは OS，ブラウザ，ネットワーク，最適化のレベルとする．また，それぞれのパラメータは，三つの異なる値をとることが可能であるとする．例えば，ブラウザであれば，IE，Firefox，Opera という値を設定可能である．テストケースは，各パラメータの値の組合せとする．したがって，テストケースの総数は $3 \times 3 \times 3 \times 3 = 81$ 通りであり，すべての可能なパラメータ値の組合せをテストするためには，これだけのテストケースが必要となる．

一方，ペアワイズカバレッジを満たすテスト集合（以降，ペアワイズテスト集合と呼ぶ）は，表 2 のようにわずか 9 個のテストケースによって構成可能である．実際，どの二つのパラメータを選んでも，それらのとり得る 9 種類の値の組合せが，9 個のテストケースのいずれかに含まれていることが確認できる．なお，このテスト集合は **3.1.1** で述べる方法で構成したものである．

この例では，どのようにテストケースを選んでも，ペアワイズカバレッジを満たすには最低でも 9 個の異なるテストケースが必要なのは明らかである．したがって，表 2 のテスト集合はテスト数が最小なペアワイズテスト集合であることが分かる．

先に述べたように，ペアワイズテストの背景にある基本的な根拠は，フォールトの多くは少数の特定のパラメータ値の組合せによって顕在化するという知見である．近年になり，この経験的な知見を根拠づける研究が Kuhn らによって公表された [6]．Kuhn らは，

表 1 パラメータと値
Table 1 Parameters and values.

OS	Browser	Network	Optimization
NT	IE	Modem	L1
2000	Firefox	Wire	L2
XP	Opera	Wireless	L3

表 2 ペアワイズテスト集合
Table 2 A test set for pairwise coverage.

	OS	Browser	Network	Optimization
1	NT	IE	Modem	L1
2	2000	Firefox	Wire	L1
3	XP	Opera	Wireless	L1
4	NT	Firefox	Wireless	L2
5	2000	Opera	Modem	L2
6	XP	IE	Wire	L2
7	NT	Opera	Wire	L3
8	2000	IE	Wireless	L3
9	XP	Firefox	Modem	L3

種々のソフトウェアについて，フォールトの顕在化が，何個のパラメータ値に依存しているかについて大規模な調査を行っている．その結果，大部分のフォールトは，一つまたは二つの特定のパラメータの値の組合せによって，顕在化することが示されている．これは，ペアワイズテストの根拠を実証的に裏づける結果であるといえる．

2.2 適用事例

ペアワイズテストの嚆矢となるのは，Mandl による Ada 用のコンパイラのテストに関する研究である [7]．以降，以下のように，種々のソフトウェアシステムに対し，適用が試みられている．

- 電子メールシステム [8], [9]
- ルールベースシステム [10]
- パーサ，コンパイラ [7], [10]
- （グラフィカル）ユーザインタフェース [10]〜[12]
- ネットワーク保守支援システム [13]
- 通信プロトコル [14]
- 人工衛星用人工知能システム [15]
- 電子マネー用 IC チップファームウェア [16]

ペアワイズテストは，その適用対象となる開発段階や，ソフトウェアの構成要素とは独立した概念であるが，上記の適用例の多くでは，システムの動作が外部仕様どおりであるかを確認するための機能テストにおいて，ペアワイズテストを利用している．

テスト集合を生成するツールについて報告した例としては，文献 [11], [17]〜[19] 等がある．例として，図 1

```
D:\pict>cat ModelFile.txt
OS:             NT, 2000, XP
Browser:        IE, Firefox, Opera
Network:        Modem, Wire, Wireless
Optimization:   L1, L2, L3

D:\pict>pict ModelFile.txt
OS      Browser Network Optimization
2000    Opera   Modem   L3
XP      IE      Wire    L1
2000    IE      Wireless        L2
XP      Firefox Modem   L2
NT      Firefox Wireless        L1
NT      Opera   Modem   L1
NT      Firefox Wire    L3
XP      Opera   Wireless        L2
2000    Opera   Wire    L1
NT      IE      Modem   L3
NT      Opera   Wire    L2
2000    Firefox Wireless        L3
XP      Firefox Modem   L3
```

図 1　PICT ツールの実行例
Fig. 1　A PICT model and the output.

に，文献 [19] で報告されている PICT と呼ばれるツールの動作の様子を示す．ファイル `ModelFile.txt` は，表 1 のテストモデルを PICT ツールの入力形式で表現しており，この入力に対し，合計で 13 個のテストケースからなるペアワイズテスト集合が出力されている．なお，このツールでは，**3.2.1** で後述する探索に基づく逐次的な手法によりペアワイズテスト集合を求めている．

3. テスト集合の設計

本章では，ペアワイズカバレッジを満たすテスト集合を作成する問題について議論する．まず，最もテストケース数の少ないペアワイズテスト集合を求めるという問題について考える．計算量については，この問題がクラス NP に含まれることは明らかであるが，それ以上のことは，まだほとんど明らかにされていない(注1)．

しかし，現状では，最適解，及び，最適解の大きさ（テストケース数）を求めることは非常に困難な場合が多い．これを端的に示すのは，パラメータ数 13 で，各パラメータが 12 種類の値を有す場合である．この場合，テストケース数最小のペアワイズテスト集合のテストケース数が 144 であれば，それは，次数 12 の有限射影平面の存在を意味する．この有限射影平面の有無は，組合せデザイン（combinatorial design）と呼ばれる数学分野における著名な未解決問題の一つである(注2)．組合せデザインの分野では，ペアワイズテスト集合は，カバリングアレー（covering array）という名で呼ばれている [25].

ただし，ある条件のもとでは，最適な構成法が知られている．代表的な二つの場合を以下に挙げる．

- すべてのパラメータが 2 値をとる場合 [26], [27].
- すべてのパラメータのとり得る値の総数が同じであり，その数（V）が素数か素数のべき乗，かつ，パラメータ数が $V+1$ 以下である場合．

特に後者については，**3.1.1** にて，その構成法を説明する．なお，最小のテスト数の上下限に関する研究として，文献 [28]〜[30] 等が知られている．

一般の場合については，最適な構成方法は知られていないため，できるだけテスト数の少ないペアワイズテスト集合を構成するため，数々の手法が提案されてきている．文献 [31] では，これらの手法を，非決定的な構成法と決定的な構成法に分け，その上で更に細かく種類分けする分類法を提案している．本論文では，この分類には従わず，構成法を代数の性質を利用したものと，計算機による探索を用いた手法とに分け，それぞれについて概説する．

3.1 代数的手法

組合せデザインの分野では，主に代数的な手法によって，小さいカバリングアレー，つまり，テストケースの少ないペアワイズテスト集合を構成する様々な方法が検討されている．以下では，このような手法の中で，文献 [32]〜[34] 等の議論をもとに，特に理解が容易なものについて紹介する．

3.1.1 パラメータ数の少ない場合

ここでは，パラメータのとり得る値の総数は，どのパラメータでも同一とし，その数を V (≥ 2) で表す．また，パラメータの総数を n (≥ 2) で表す．

以下に示す方法によって，V と n に関して，次の 2

(注1)：NP 完全と述べている文献が存在するが（例, [20]），上記のとおり証明はされていない．文献 [21] では証明を試みたが失敗したという事例を報告している．
(注2)：組合せデザインに関する問題を網羅的に扱った文献として [22] がある．また，邦書では，理論面については文献 [23] に，応用を含む話題については文献 [24] に詳しい．

表 3　代数的手法（$n = 4$, $V = 3$）
Table 3　A test set constructed by an algebraic method ($n = 4$, $V = 3$).

i	j	p=1	p=2	p=3	p=4
0	0	0 (= (0 ∗ 0 + 0)%3)	0 (= (0 ∗ 1 + 0)%3)	0 (= (0 ∗ 2 + 0)%3)	0
0	1	1 (= (0 ∗ 0 + 1)%3)	1 (= (0 ∗ 1 + 1)%3)	1 (= (0 ∗ 2 + 1)%3)	0
0	2	2 (= (0 ∗ 0 + 2)%3)	2 (= (0 ∗ 1 + 2)%3)	2 (= (0 ∗ 2 + 2)%3)	0
1	0	0 (= (1 ∗ 0 + 0)%3)	1 (= (1 ∗ 1 + 0)%3)	2 (= (1 ∗ 2 + 0)%3)	1
1	1	1 (= (1 ∗ 0 + 1)%3)	2 (= (1 ∗ 1 + 1)%3)	0 (= (1 ∗ 2 + 1)%3)	1
1	2	2 (= (1 ∗ 0 + 2)%3)	0 (= (1 ∗ 1 + 2)%3)	1 (= (1 ∗ 2 + 2)%3)	1
2	0	0 (= (2 ∗ 0 + 0)%3)	2 (= (2 ∗ 1 + 0)%3)	1 (= (2 ∗ 2 + 0)%3)	2
2	1	1 (= (2 ∗ 0 + 1)%3)	0 (= (2 ∗ 1 + 1)%3)	2 (= (2 ∗ 2 + 1)%3)	2
2	2	2 (= (2 ∗ 0 + 2)%3)	1 (= (2 ∗ 1 + 2)%3)	0 (= (2 ∗ 2 + 2)%3)	2

条件が成り立つとき，V^2 個のテストケースからなるペアワイズテスト集合が構成できる [32], [35].

（1） V が素数か素数のべき乗

（2） $n \leq V + 1$

二つのパラメータ間の値の組合せの総数が V^2 なので，この手法で構成されるペアワイズテスト集合は，テストケースの数に関して，明らかに最適（最小）である．したがって，$n = V + 1$ についてこのテスト集合が得られれば，$n \leq V$ の場合についても，単に余分なパラメータを除けば最適なものが得られるので，以降 $n = V + 1$ として考える．

V^2 個のテストそれぞれを $t_{i,j}$ $(0 \leq i, j \leq V - 1)$ で表す．このとき，テスト $t_{i,j}$ の p $(1 \leq p \leq n)$ 番目のパラメータの値 $t^p_{i,j}$ を以下のように定めることで，ペアワイズテスト集合が得られる．

$$t^p_{i,j} = \begin{cases} (i * (p-1) + j)\%V & 1 \leq j \leq V \\ i & j = V + 1 \end{cases}$$

ただし % は剰余を計算する演算子とする．

表 3 は，$V = 3$, $n = 4$ の場合，この方法で構成したテスト集合を示している．表 2 と見比べると，両者が全く同じものであることが分かる．

なお代数的には，この手法は，素数を法とする剰余環が有限体となる性質を利用して，有限射影平面を求めているものと解釈することができる [32], [35]．有限体は要素数が素数のべき乗の場合も存在するので，上記の手法を一般化して，このような場合，例えば，$V = 4, 8, 9, 16, \cdots$ の場合に，パラメータ数 $n = V+1$，テストケース数 V^2 のペアワイズテスト集合を構成することができる（上式の加法 $(+)$，乗法 $(*)$ を有限体上のそれとすればよい）．

V が素数のべき乗でない場合は，V を超える最小の素数のべき乗に対し，上記の方法でテスト集合を構成し，有効でない値のみを変更することで，ペアワイズテスト集合が構成できる．

3. 1. 2　パラメータ数の多い場合

3. 1. 1 では，パラメータ数 n がパラメータ値の総数 V に比べ，たかだか 1 しか多くない場合（$n \geq V + 1$）について，ペアワイズテスト集合の構成法を説明した．この条件が成り立たない場合，すなわち，

$$n > V + 1$$

の場合については，再帰的構成法を用いてペアワイズテスト集合を構成できる．簡単に述べると，この方法は，パラメータ数の少ない場合に対するペアワイズテスト集合を組み合わせることで，パラメータ数を増やす方法である．

様々な再帰的構成法がこれまで提案されているが（例えば, [32]〜[34]），ここでは最も単純なものについて説明する．その他の再帰的構成法については，文献 [36] に詳しい．

先に表 3 の $n = 4$, $V = 3$ に対するペアワイズテスト集合を得ているが，このテスト集合を組み合わせることで，$n = 16$ に対して再帰的構成法により得られたペアワイズテスト集合を表 4 に示す．

ここで，上から 9 個のテストケースは，表 3 のテスト集合を横に（パラメータ方向に）四つ並べることによって構成している．これらの 9 個のテストケースにより，1 番目から 4 番目までのパラメータに関しては，どの 2 パラメータ間の値のペアもすべて網羅されている．これは，5 番目から 8 番目まで，9 番目から 12 番目まで，13 番目から 16 番目までという範囲についても同様である．

残る問題は，例えば，1 番目と 5 番目のパラメータ間のように，これら 9 個のテストケースで網羅されていない可能性のある部分である．これらのパラメータ間の値の組合せをカバーするために，やはり表 3 のテスト集合が利用できる．具体的には，表 3 の i 番目の

表 4　表 3 を組み合わせて得られたペアワイズテスト集合
Table 4　A test set recursively constructed from Table 3.

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0
2	2	2	0	2	2	2	0	2	2	2	0	2	2	2	0
0	1	2	1	0	1	2	1	0	1	2	1	0	1	2	1
1	2	0	1	1	2	0	1	1	2	0	1	1	2	0	1
2	0	1	1	2	0	1	1	2	0	1	1	2	0	1	1
0	2	1	2	0	2	1	2	0	2	1	2	0	2	1	2
1	0	2	2	1	0	2	2	1	0	2	2	1	0	2	2
2	1	0	2	2	1	0	2	2	1	0	2	2	1	0	2
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
2	2	2	2	2	2	2	2	2	2	2	2	0	0	0	0
0	0	0	0	1	1	1	1	2	2	2	2	1	1	1	1
1	1	1	1	2	2	2	2	0	0	0	0	1	1	1	1
2	2	2	2	0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	0	2	2	2	2	1	1	1	1	2	2	2	2
1	1	1	1	0	0	0	0	2	2	2	2	2	2	2	2
2	2	2	2	1	1	1	1	0	0	0	0	2	2	2	2

表 5　$V = 3$ として得られたペアワイズテスト集合
Table 5　A pairwise test set for $V = 3$.

0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	1	1	0	1	1	1	0	1
2	2	2	0	d	d	d	0	d	d	d	0	d
0	1	2	1	0	1	d	1	0	1	d	1	0
1	2	0	1	1	d	0	1	1	d	0	1	1
2	0	1	1	d	0	1	1	d	0	1	1	d
0	2	1	2	0	d	1	d	0	d	1	d	0
1	0	2	2	1	0	d	d	1	0	d	d	1
2	1	0	2	d	1	0	d	d	1	0	d	d
1	1	1	1	1	1	1	1	1	1	1	1	0
2	2	2	2	d	d	d	d	d	d	d	d	0
0	0	0	0	1	1	1	1	d	d	d	d	1
1	1	1	1	d	d	d	d	0	0	0	0	1
2	2	2	2	0	0	0	0	1	1	1	1	1
0	0	0	0	d	d	d	d	1	1	1	1	d
1	1	1	1	0	0	0	0	d	d	d	d	d
2	2	2	2	1	1	1	1	0	0	0	0	d

（d は don't care 値を表す.）

パラメータの値を，$(i-1)*4+1$ 番目，$(i-1)*4+2$ 番目，$(i-1)*4+3$ 番目，$i*4$ 番目の値として用いることで，パラメータ数が 4 倍の 16 パラメータに対する 9 個のテストケースを構成する．先のテストケースに現れていない値の組合せは，これらの新たに得られた 9 個のテストケースのいずれかによって，必ずカバーされる．

このようにして得られた合計 18 個のテストケース中には，0 のみからなるテストケースが重複して現れるので，一方を取り除くことで，16 個のパラメータに対し，17 個のテストケースでペアワイズカバレッジを満たすテスト集合が求まる．

一般には，パラメータ数，大きさが，それぞれ n，s と n'，s' のペアワイズテストを組み合わせることで，パラメータ数 $n*n'$，大きさ $s+s'-1$ のペアワイズテストを構成できる．

このようにして得られたテスト集合もペアワイズテスト集合であるので，同じ手法を繰り返し適用して，任意のパラメータ数にまでテスト集合を拡大できる．例えば，表の 16 パラメータのテスト集合に，もとの 4 パラメータのテスト集合を組み合わせることで，64 パラメータに対する，25 個のテストケースからなるペアワイズテスト集合を構成できる．

3.1.1 の方法で生成したパラメータ数 $V+1$，テスト数 V^2 のテスト集合をもとに，上記の手法を i 回繰り返し適用して得られるテスト集合のテストケースの総数を s_i，パラメータ数を n_i とおくと，以下の漸化式が得られる．

$$\begin{cases} s_0 = V^2, n_0 = V+1 \\ s_i = s_{i-1} + s_0 - 1, n_i = n_{i-1} * n_0 \quad (i \geq 1) \end{cases}$$

これらの式から，パラメータ数は乗法的に増加するのに対し，テストケース数は加法的にしか増えないことが見てとれる．つまり，パラメータ数に対し，ペアワイズカバレッジを満たすために必要なテストケースの数は，対数的にしか増加しない．

3.1.3　パラメータ値の総数が異なる場合

これまで，各パラメータは V 種類の値をとり得ることを前提としてきた．ここでは，パラメータによって，とり得る値の総数が異なる場合にペアワイズテスト集合を構成する手法について議論する．基本的なアプローチとして，以下の 2 種類の考え方を取り上げる．

（1）　最も値の種類の多いパラメータを基準にしてテスト集合を生成する．

（2）　値の総数が同じパラメータについてペアワイズテスト集合を構成する．得られたテスト集合を結合し，カバーされていない値の組に対するテストケースを追加する．

これらの手法の説明のため，例として，1 番目から 4 番目のパラメータは 3 値を，5 番目から 13 番目のパラメータは 2 値をとる場合を考える．表 5 は，（1）の手法，すなわち，すべてのパラメータが 3 値をとると考えて構成したテスト集合から，無効な値（この場合であれば 2）を変更したものである．無効な値は，有効な任意のパラメータ値に変更してもペアワイズカバレッジには影響ないので，このような値は don't care として d の文字を表記している．

表 6　複数のテスト集合を組み合わせて得られたペアワイズテスト集合
Table 6　A pairwise test set constructed by combining different test sets.

0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	1	0	1	1	0	1	1	0
2	2	2	0	0	1	1	0	1	1	0	1	1
0	1	2	1	1	0	1	1	0	1	1	0	1
1	2	0	1	1	1	1	1	1	1	0	0	0
2	0	1	1	0	0	0	1	1	1	1	1	1
0	2	1	2	1	1	1	0	0	0	1	1	1
1	0	2	2	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*
2	1	0	2	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*
0	0	0	0	1	1	1	1	1	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1
2	2	2	2	0	0	0	0	0	0	0	0	0
2	2	2	2	1	1	1	1	1	1	1	1	1

（d は don't care 値を表す.）

一方，表 6 は，（2）の手法を用いて構成したペアワイズテスト集合である．具体的には，まず，**3.1.1** の手法で $V = 3$，$n = 4$ のペアワイズテストを，**3.1.1** と **3.1.2** の手法で $V = 2$，$n = 9$ のペアワイズテストを得た後，これらをパラメータ方向に組み合わせている．この状態では，パラメータの組で，一方が 1 番目から 4 番目までのいずれか，もう一方が 5 番目から 13 番目までのいずれかであるようなものについては，値がカバーされていない可能性がある．そこで，5 個のテストケースを追加し，これらの組合せを網羅している．

より詳細な場合分けをもとに，更に小さいテスト集合の構成を目指している研究として，文献 [37], [38] がある．

3.1.4　直交表を用いた手法

直交表（orthogonal array）は，典型的な組合せデザインであり，カバレングアレーの特殊な場合に相当する．直交表は，実験計画の分野において，効率的な実験を実現するために用いられている重要なツールである．直交表からテストケースを得るという試みとしては，文献 [8], [39] がある．

最近になって，我が国で直交表を基本としたテスト手法である HAYST 法が注目を集めている [40]．HAYST 法では，直接対応する直交表が存在しないようなシステムに対しても，様々な技術を用意することで，直交表をもとに多くのパラメータ値の組合せをカバーした，効果的なテストケースを作成することができる．ペアワイズテストと異なり，HAYST 法では，100%のペアワイズカバレッジは保障しないが，背景にある基本的な考えはペアワイズテストと同様である．

```
TS ⟵ ∅
repeat
  新たなテストケース t を生成
  TS ⟵ TS ∪ {t}
until ペアワイズカバレッジが満たされる
TS を出力
```

図 2　逐次的な構成手法
Fig. 2　Incremental test set construction.

複写機用ソフトウェアのテストに適用した結果，従来のアドホックなテストに比べ，非常に高い割合でパラメータ値の組合せをカバーすることができ，テスト効率も高まった旨が報告されている [41].

3.2　探索に基づく手法

3.1 で説明した手法は，ペアワイズテスト集合のもつ数学的な性質を利用したものであった．本節では，ペアワイズテスト集合の構成を組合せ最適化問題としてとらえ，計算機を利用した解探索を基本としたテスト集合の構成方法について説明する．

3.2.1　逐次的な構成手法

一つずつテストケースを生成することで，最終的にペアワイズカバレッジを満たす手法である．概略を図 2 に示す．この図において TS は求めたテストケースの集合であり，空集合から始めて，ペアワイズカバレッジが満たされるまで，生成されたテストケースを随時追加していく．

この手法で重要な点は，ループにおいて TS に加えられるテストケースの生成方法である．直感的には，その時点で，いまだカバーされていない値の組を多く含むテストケースを追加すれば，全体として少ない数のテストケースでペアワイズカバレッジが満たされるように思われる．実際，文献 [11] では，そのつど，最も多くの新しい値の組を含むテストケースを追加することによって最終的に得られるテスト集合の大きさが，パラメータ数に対して対数的にしか増加しないことを示している．

しかしながら，新しい値の組の数に関し最適なテストケースを選ぶ問題は，最適なペアワイズテスト集合を求める問題と異なり，NP 完全問題であることが分かっている [36]．したがって，現実的には，ヒューリスティックな手法を用いて，必ずしも最適ではないが良好なテストケースを選択する必要がある．

商用のシステムである AETG [11] では，ランダムにパラメータを並べ，順に最も有望な値を設定してテ

ストケースを作成するという手順を 100 回程度繰り返し，得られた中で最良のテストケースを一つ選択するという手法を用いている．

AETG と似通ったアルゴリズムとして，TCG [42]，及び，DDA [43] が提案されている．文献 [44] では，これらの 3 アルゴリズム（AETG，TCG，DDA）を統一的に表現することのできる枠組みを与え，アルゴリズムの設計のどの部分が，テストケース数を少なくするのに貢献しているかを，統計的に分析している．結果として，一つのテストケースを求める際に，どのパラメータから値を決めていくかという，パラメータの選択基準が重要であることが示唆されている．

文献 [45] では，求解能力の高い遺伝アルゴリズムやアントコロニーアルゴリズムを，テストケースの生成問題に適用することで，最終的により小さいテスト集合が得られることを報告している．

なお，逐次的な構成手法で，図 2 の枠組みに当てはまらないものとして，IPO アルゴリズムと呼ばれる手法が知られている [46]．IPO アルゴリズムでは，まず少数の一部のパラメータのみに対するペアワイズテスト集合を構成する．次に，まだ考慮されていないパラメータ一つに対し，その値を含むように，それまでに得られているペアワイズテスト集合を拡張する．パラメータを追加することによって，ペアワイズカバレッジが満たされなくなった場合は，新たにテストを追加する．考慮されていないパラメータがまだあれば，再度それらの中から一つを取り上げ，同様の操作でペアワイズテスト集合を拡張する．このように，テスト集合を表として見た場合，縦軸方向（テスト数）だけでなく，横軸方向（パラメータ数）にもテスト集合の拡張を繰り返すという点，及び，拡張に際し，決定的なアルゴリズムを用いている点が，IPO アルゴリズムが，その他の逐次的構成法と異なる点である．

3.2.2 一括的な構成手法

一つひとつのテストを逐次的に求めるのではなく，テスト集合全体を探索によって求める手法もよく研究されている．文献 [47] に基づき，図 3 にこの手法の大まかな手順を示す．

この手法では，まず指定された数のテストケースをランダムに生成する．そして，これらのテストケースを少しずつランダムに変更していくことで，最終的にペアワイズカバレッジを満たすテスト集合を求める．図では，ある時点において得られているテスト集合を TS とし，TS から，あるテストケースに対し一つのパラメータ値を変更して得られたテスト集合を，TS' としている．ここで，カバーされる値の組の数が，TS よりも TS' の方が多いか，同じ場合，TS を TS' に更新する．逆に少ない場合は，更新を行うか否かは確率的に決定する．

```
TS ⟵ ∅;
repeat 指定されたテストケース数と同回数
  ランダムにテストケース t を生成
  TS ⟵ TS ∪ {t}
while TS がペアワイズカバレッジを満たしていない do
  ランダムに TS に含まれる 1 テストケースの 1 パラメータの
  値を変更して得られるテスト集合を TS′ とする
  if 少なくとも TS と同数の値の組を TS′ がカバーする
    then TS ⟵ TS′
    else ある確率で TS ⟵ TS′ を実行
TS を出力
```

図 3　一括的な構成手法
Fig. 3　Direct test set construction.

表 7　ヒルクライミングの実行結果
Table 7　The results obtained by hill climbing.

(a) 初期状態（47 種類の値の組合せがカバーされていない）

1	1	0	1	1	1	0	0	1	1	0	1	0
1	2	1	0	0	0	0	1	0	1	1	0	0
2	1	1	1	1	0	0	0	1	1	1	0	1
1	0	0	1	1	0	1	0	0	1	0	1	0
1	2	2	0	0	0	0	1	1	1	0	1	0
2	1	0	1	1	0	1	0	1	0	1	0	0
0	2	1	0	0	0	0	0	1	1	0	1	0
0	2	0	1	0	0	0	0	1	1	0	0	0
2	0	2	2	1	0	0	0	1	0	0	1	1
2	1	2	1	1	1	1	1	1	1	1	1	1
0	1	2	0	0	1	0	1	1	0	0	1	0
2	0	2	0	0	0	1	0	1	0	1	0	1

(b) 最終結果

1	1	0	2	1	0	1	1	1	0	1	1	1
1	2	1	0	1	1	0	1	1	1	1	0	1
2	1	1	1	1	1	1	0	1	1	0	0	0
1	0	2	1	1	0	0	1	1	1	1	1	1
0	2	0	0	1	0	1	1	1	1	0	1	0
0	0	0	1	1	1	0	1	0	0	0	0	1
0	0	1	2	1	1	0	1	0	0	0	1	0
2	2	1	1	0	0	0	0	1	0	0	1	0
1	1	2	2	0	0	1	0	0	0	0	0	0
2	2	2	2	0	1	0	1	0	1	1	0	0
0	1	2	0	0	1	0	0	0	0	1	1	1
2	0	0	0	0	1	1	0	0	0	0	0	1

この確率を 0 とした場合は，更新によってカバーされる値の組の総数が減少することはなくなる．これはヒルクライミングと呼ばれる探索手法に相当する．

表 7 は，**3.1.3** で用いた 4 個のパラメータが 3 値を，9 個のパラメータが 2 値をとる例について，実際にこの方法を用いてテスト集合を求めた過程を示している．テストケース数は 12 としている．表 7 (a) は

最初にランダムに生成されたテスト集合である．これらのテストケースには，まだ 47 種類のパラメータ値の組合せが現れていない．これを逐次ランダムに改善していくと，1000 回程度の **while** ループの実行によってペアワイズカバレッジを満たすことができた．表 7 (b) は最終的に得られたペアワイズテスト集合である．

ただし，ヒルクライミングは，一時的にすら解が悪くなることを許していないので，局所最適に陥ることが多くなる．シミュレーテッドアニーリングは，カバーできる値の組が減っても，TS の更新をある程度の確率で許すことで局所最適にとどまることを防ぎ，徐々にこの確率を減少させることで，最終的に良好な解を得る手法である．実際に，ヒルクライミングよりも優れていることが報告されている [47]．

また，tabu サーチも，シミュレーテッドアニーリング同様，よく知られた探索手法であり，図 3 とほぼ同様の枠組みで，ペアワイズテスト集合を求める問題に利用できる [48]．

なお，一括的な構成手法ではあるが，図 3 の枠組みとは異なるアプローチとして，ペアワイズテスト集合を得る問題を，他の最適化問題に帰着させて解く方法が提案されている．具体的には，整数計画問題 [49]，制約プログラミング問題 [50]，ブール式の充足可能性判定問題 [51] への変換が知られている．ただし，これらの手法では，小さい問題に対してしか，現実的な時間で問題を解くことができないことが多い．

3.3 議　　論

3.1，**3.2** では，代数的手法と探索による手法について説明した．代数的手法は，ペアワイズテスト集合のもつ数学的な性質を利用した構成法であり，計算機を用いずに，大規模なテスト集合を作成することが可能である．一方，探索に基づく手法は，計算機の能力に依存しており，問題の規模が大きくなると，良好なテストケースの作成に時間が掛かる．反面，柔軟性に優れており，以下で説明するパラメータ値の組合せに関する制約や，k-way テストへの拡張が容易である．

3.3.1 求 解 性 能

本章で紹介したように，ペアワイズテスト集合を得るため，様々な手法が検討されている．この背景には，数学的な興味とともに，テストケースの削減がソフトウェア開発の効率化につながるという実用上の動機がある．得られるペアワイズテスト集合の大きさに関して，代数的手法と探索的手法を比較した場合，問題の規模が大きくなく，解の計算に十分な時間を費やせるのであれば，探索に基づく手法が優れていることが多い．

例えば，tabu サーチを用いた前述の研究 [48] では，いくつかの問題に対し，これまでに知られている最小のペアワイズテスト集合を発見することができた旨を報告している．ただし，同じ探索に基づく手法であっても，逐次的な構成手法よりも，一括的な手法の方が，より小さいペアワイズテスト集合を得るという点に関しては優れている．文献 [47] では，ペアワイズテスト集合の大きさに関して，逐次的な構成手法と一括的な構成手法とを比較しており，逐次的な構成手法の方が，1〜2 割程度テストケース数が多かったという結果を報告している．

ただし，一括的な構成手法では，最初に指定するテストケース数（図 3 を参照のこと）が小さすぎる場合，解となるペアワイズテスト集合が探索空間中にまばらにしか存在せず発見が困難であったり，そもそもその大きさのペアワイズテスト集合が存在しなかったりする場合がある．そのような場合，探索に極めて長い時間を要したり，探索を中断してテストケース数を増やした上で，再度探索を行うといった処理が必要となる．逐次的な構成手法にはこのような問題はないため，1〜2 割程度のテストケース数の増加が，実用上問題とならない場合であれば，逐次的な構成手法を用いてテストを得ることは理に適っているといえる．

代数的手法については，パラメータ数とパラメータ値の種類に関して，各手法ごとにその手法が有効に動作する条件が存在し，その条件のもとでは極めて小さいペアワイズテスト集合が得られることが多い．得られるペアワイズテスト集合が最適であることが知られている代数的手法として，**3.** の冒頭で述べた例がある [26], [27], [32]．

3.3.2 パラメータ値の組合せに関する制約

実際のソフトウェアでは，同時に選択できないパラメータ値の組合せが多数存在することが普通である．

例として，表 1 の場合を再度取り上げ，L3 は OS が XP の場合は設定できないという新たな制約を仮定する．この場合，表 2 の 9 番目のテストケース（XP, Firefox, Modem, L3）は設定できない．このテストケースを削除し，新たに制約を満たす二つのテストケースを追加して得られたペアワイズテスト集合を，表 8 に示している．

このような制約が存在した場合，代数的手法は直接

表 8　制約を考慮したペアワイズテスト集合
Table 8　A pairwise test set satisfying additional constraints.

	OS	Browser	Network	Optimization
1	NT	IE	Modem	L1
2	2000	Firefox	Wire	L1
3	XP	Opera	Wireless	L1
4	NT	Firefox	Wireless	L2
5	2000	Opera	Modem	L2
6	XP	IE	Wire	L2
7	NT	Opera	Wire	L3
8	2000	IE	Wireless	L3
9	XP	Firefox	Modem	L2
10	2000	Firefox	Modem	L3

適用することができない．一方，探索に基づく手法では，制約を満たさないテストケースを解の候補としないことで，無効なテストケースの生成を回避できる．

制約の存在は，可能なテストケース数，及び，可能なペアの数が少なくなることを意味するので，一見，必要なテストケース数も減少するように思える．しかしながら，経験的には，多くの場合，逆に必要なテストケース数が増えるようである．これは，テストケース選択の自由度が減るためと考えられる．

3.3.3　k 個のパラメータ間の値のテスト

これまで，二つのパラメータ間の値を網羅するテスト，すなわち，ペアワイズテストに議論を限定してきた．ここでは，任意の k $(1 \leq k \leq n)$ 個のパラメータを網羅する k-way テストについて考える．

探索に基づく構成手法では，2 以外の k への拡張は容易である．逐次的な構成手法であれば，k 個のパラメータ間での値の組合せをできるだけカバーするようそのつどテストケースを選択すればよい．一括的な構成手法であれば，更新を行う繰返しの終了条件を，ペアワイズカバレッジではなく k-way カバレッジに変更すればよい [11], [52].

一方，代数的な構成手法については，本論文で説明した手法はもとより，知られている多くのものは $k = 2$ の場合に特化しており，$k \geq 3$ の場合については，更なる研究が望まれている．$k = 3$ の場合の研究に関しては，文献 [53] に詳しい．また，代数的な手法と探索に基づく手法を組み合わせ，3-way カバレッジを満たすテスト集合を求めている研究として，文献 [54] がある．

また，更に問題を一般化して，パラメータごとに k の値が異なるようなテストについて考えることも可能である．つまり，重要なパラメータについては，大きい k の値を用いるという手法である．このような場合も，探索による構成手法であれば，対応は比較的容易である．関連研究としては，シミュレーテッドアニーリングを用いた文献 [55] を挙げることができる．

4.　有効性評価に関する研究成果

2.2 で適用例について述べたが，ペアワイズテストの有効性を，他のテスト手法と比較する形で定量的に評価した例は多くない．

いくつかの研究では，コードカバレッジを評価することで，ペアワイズテストの有効性を評価している．コードカバレッジは，テストの有効性を示す重要な指標と考えられている(注3)．

文献 [11] では，C 言語で記述された 1000 行程度のプログラムに，ペアワイズテスト，ランダムテスト，全数テストを実施した場合のコードカバレッジを報告している．その結果によると，テストケース 300 のランダムテストで，達成されたコードカバレッジ（ブロックカバレッジ）が 67%だったのに対し，ペアワイズテストでは，200 個のテストケースで，400 個のテストからなる全数テストと同様の，92%のカバレッジが達成できたとのことである．また，文献 [57] では，UNIX の sort プログラムを対象とした実験の結果，ランダムテストより良好なカバレッジが得られたことが報告されている．

文献 [13] では，ペアワイズテストだけでなく，k-way テストを行った場合についても，コードカバレッジの評価を行っている．ブロックカバレッジに関しては，$k = 2$，すなわち，ペアワイズテストの場合，全数テストとほぼ同等のカバレッジを達成できたことが報告されている．一方，分岐カバレッジについては，k の値がある程度大きくないと，全数テストと遜色ないカバレッジは達成できなかった．

一方，コードカバレッジではなく，フォールト検出率を評価した例を以下に示す．

文献 [58] では，ブール式でモデル化された論理テストを対象に，ランダムテスト等の手法と，ペアワイズテストとのフォールト検出能力の比較を行い，ペアワイズテストの優位性を結論している．

文献 [59] では，ネットワークシステムにおけるセキュリティに関連した実際の欠陥のデータを用いて，

(注3)：最近の研究において，対象によっては，コードカバレッジがテストの有効性を示す指標とはならないことが指摘されている [56].

ランダムテストとペアワイズテストの有効性を比較評価している．その結果，特にパラメータ値の総数が多い場合に，ペアワイズテストの優位性が顕著になることが分かった旨を報告している．

逆に，ペアワイズテストの有効性に否定的な結論を導いている研究として Schroeder らによるものがある [60]．この研究では，二つの実際のプログラムに対し，人為的にフォールトを注入することで，多数のミュータントプログラムを作成し，それらのフォールトが検出できるか否かを調べることで，テスト方法の性能を評価している．その結果，ペアワイズテストとランダムテストで，有意な差は見られなかったとしている．

Schroeder らの研究を踏まえて，文献 [61] では，ペアワイズテストを用いる際の，いくつかの問題点を指摘している．例えば，パラメータ値が任意の整数値をとる場合など，パラメータ値すべてをテストすることはできず，代表値を適切に選択することが必要となる．このとき，代表値を選ぶために行われる境界値分析が失敗すれば，フォールトを顕在化させる値が選ばれないことになり，ペアワイズテストを実行しても，フォールトを検出することはできない．文献 [61] では，ペアワイズテストは，従来のテスト手法に取って代わるものではなく，それらとともに適切に用いることで効果が得られるという点を強調している．

5. むすび

本論文では，ソフトウェアテストの一手法である組合せテストについて概説した．まず，その概念，及び，適用事例について紹介した後，ペアワイズテストを中心に，理解が容易なテスト集合の作成手法について，例を用いて説明した．また，テスト集合の生成手法，及び，有効性評価に関する研究を紹介した．

組合せデザインの分野においては，小さいテスト集合を求める構成法の研究が活発に行われている．本論文で触れることのできなかったこれらの数学的な研究については，文献 [36], [62] で概観することができる．

既にペアワイズテストの概念は，ソフトウェア開発者に広く認知されるに至っているが，日本国内での適用事例は, [16] を例外として，ほとんどないのが現状である．ペアワイズテストの普及には，実システムへの適用と，それにより得られた知見の集積が不可欠と考えられる．

謝辞 資料整理に協力頂いた紀本眞氏に感謝する．本研究の一部は，日本学術振興会科学研究費若手（B）課題番号 17700033，及び，文部科学省 21 世紀 COE プログラム（研究拠点形成費補助金）の研究助成によるものである．ここに記して謝意を表す．

文　献

[1] R.D. Craig and S.P. Jaskiel, Systematic Software Testing, Artech House Publishers, Boston, 2002.（成田光彰，宗　雅彦（共訳），体系的ソフトウェアテスト入門，日経 BP，2005.）

[2] L. Copeland, A Practitioner's Guide to Software Test Design, Artech House Publishers, Boston, 2003.（宗 雅彦（訳），はじめて学ぶソフトウェアのテスト技法，日経 BP，2004.）

[3] C. Kaner, J. Bach, and B. Pettichord, Lessons Learned in Software Testing: A Context Driven Approach, John Wiley & Sons, New York, 2002.

[4] S. Splaine and S.P. Jaskiel, The Web Testing Handbook, STQE Publishing, Orange Park, 2001.

[5] J.D. McGregor and D.A. Sykes, A Practical Guide to Testing Object-Oriented Software, Addison-Wesley, Boston, 2001.

[6] D.R. Kuhn, D.R. Wallace, and A.M. Gallo, Jr., "Software fault interactions and implications for software testing," IEEE Trans. Softw. Eng., vol.30, no.6, pp.418–421, 2004.

[7] R. Mandl, "Orthogonal latin squares: An application of experiment design to compiler testing," Commun. ACM, vol.28, no.10, pp.1054–1058, 1985.

[8] R. Browmlie, J. Prowse, and M.S. Phadke, "Robust testing of AT&T PMX/StarMAIL using OATS," AT&T Technical J., vol.71, no.3, pp.41–47, 1992.

[9] K. Burr and W. Young, "Combinatorial test techniques: Table-based automation, test generation and code coverage," Proc. International Conference on Software Testing, Analysis, and Review (STAR '98), pp.503–513, 1998.

[10] S.R. Dalal, A. Jain, N. Karunanithi, J.M. Leaton, C.M. Lott, G.C. Patton, and B.M. Horowitz, "Model-based testing in practice," Proc. 21st International Conference on Software Engineering (ICSE '99), pp.285–294, 1999.

[11] D.M. Cohen, S.R. Dalal, M.L. Fredman, and G.C. Patton, "The AETG system: An approach to testing based on combinatorial design," IEEE Trans. Softw. Eng., vol.23, no.7, pp.437–444, 1997.

[12] L.J. White, "Regression testing of GUI event interactions," Proc. 1996 International Conference on Software Maintenance (ICSM '96), pp.350–358, 1996.

[13] I.S. Dunietz, W.K. Ehrlich, B.D. Szablak, C.L. Mallows, and A. Iannino, "Applying design of experiments to software testing: Experience report," Proc. 19th International Conference on Software Engineering (ICSE '97), pp.205–215, 1997.

[14] K.B.A. Jain and R.L. Erickson, "Improved quality

of protocol testing through techniques of experimental design," Proc. IEEE International Conference on Communications (ICC '94), pp.745–752, 1994.

[15] B. Smith, M. Feather, and N. Muscettola, "Challenges and methods in testing the remote agent planner," Proc. Fifth Int'l Conf. Artificial Intelligence Planning Systems, 2000.

[16] 太田豊一，栗田太郎，松尾谷徹，"all-pair 法を応用した携帯電話組み込み用モバイル FeliCa IC チップファームウェアの評価に関する報告，" ソフトウェアテストシンポジウム 2006 (JaSST '06)，東京，2006.

[17] G. Sherwood, "Effective testing of factor combinations," Proc. Third International Conference on Software Testing, Analysis and Review (STAR '94), 1994.

[18] G.T. Daich, "New spreadsheet tool helps determine minimal set of test parameter combinations," CrossTalk: Journal of Defense Software Engineering, vol.16, no.8, pp.26–30, Aug. 2003.

[19] J. Czerwonka, "Pairwise testing in real world. Practical extensions to test case generators," Proc. 24th Annual Pacific Northwest Software Quality Conference, pp.419–430, 2006.

[20] S. Maity and A. Nayak, "Improved test generation algorithms for pair-wise testing," Proc. 16th IEEE International Symposium on Software Reliability Engineering (ISSRE '05), pp.235–244, 2005.

[21] A.W. Williams, Software Component Interaction Testing: Coverage Measurement and Generation of Configurations, Ph.D. Dissertation, University of Ottawa, 2002.

[22] C. Colbourn and J. Dinitz, eds., Handbook of Combinatorial Designs, 2nd ed., CRC Press, 2006.

[23] 永尾　汎，群とデザイン，岩波書店，1972.

[24] 高橋磐郎，組合せ理論とその応用，岩波書店，1979.

[25] N.J.A. Sloane, "Covering arrays and intersecting codes," J. Combinatorial Designs, vol.1, pp.51–63, 1993.

[26] G.O.H. Katona, "Two applications (for search theory and truth functions) of Sperner type theorems," Periodica Mathematica Hungarica, vol.1-2, pp.19–22, 1973.

[27] D. Kleitman and J. Spencer, "Families of k-independent sets," Discrete Math., vol.6, pp.255–262, 1973.

[28] A. Godbole, D. Skipper, and R. Sunley, "t-Covering arrays: Upper bounds and Poisson approximations," Combinatorics, Probability and Computing, pp.105–118, 1996.

[29] B. Stevens, L. Moura, and E. Mendelsohn, "Lower bounds for transversal covers," J. Combinatorial Designs, vol.15, no.3, pp.273–299, Dec. 1998.

[30] N. Ido and T. Kikuno, "Lower bounds estimation of factor-covering design sizes," J. Combinatorial Designs, vol.11, no.2, pp.89–99, 2003.

[31] M. Grindal, J. Offutt, and S.F. Andler, "Combination testing strategies: A survey," Software Testing, Verification and Reliability, vol.15, no.3, pp.167–199, 2005.

[32] D.M. Cohen and M.L. Fredman, "New techniques for designing qualitatively independent systems," J. Combinatorial Designs, vol.6, no.6, pp.411–416, 1998.

[33] N. Kobayashi, T. Tsuchiya, and T. Kikuno, "A new method for constructing pair-wise covering designs for software testing," Inf. Process. Lett., vol.81, no.2, pp.85–81, 2002.

[34] A.W. Williams and R.L. Probert, "A practical strategy for testing pair-wise coverage of network interfaces," Proc. 7th International Symposium on Software Reliability Engineering (ISSRE '96), pp.246–254, 1996.

[35] S. Poljak and Z. Tuza, "On the maximum number of qualitatively independent partitions," J. Combinatorial Theory Series A, vol.51, no.1, pp.111–116, 1989.

[36] C.J. Colbourn, Combinatorial Aspects of Covering Arrays, Le Matematiche, 2005.

[37] L. Moura, J. Stardom, B. Stevens, and A. Williams, "Covering arrays with mixed alphabet sizes," J. Combinatorial Designs, vol.11, no.6, pp.413–432, 2003.

[38] C.J. Colbourn, S.S. Martirosyan, G.L. Mullen, D. Shasha, G.B. Sherwood, and J.L. Yucas, "Products of mixed covering arrays of strength two," J. Combinatorial Designs, vol.14, no.2, pp.124–138, 2006.

[39] E. Heller, "Using design of experiment structures to generate software test cases," Proc. 12th Int'l Conf. on Testing Computer Software, pp.33–41, 1995.

[40] 秋山浩一，直交表による組み合わせテスト入門，vol.2, ソフトウェア・テスト PRESS，Dec. 2005.

[41] 山本訓稔，秋山浩一，"直交表を利用したソフトウェアテスト—HAYST 法，" ソフトウェアテストシンポジウム 2004 (JaSST '04)，pp.13–17, 東京，2004.

[42] Y.-W. Tung and W.S. Aldiwan, "Automating test case generation for the new generation mission software system," Proc. 2000 IEEE Aerospace Conference, pp.431–437, Big Sky, USA, 2000.

[43] C. Colbourn, M. Cohen, and R. Turban, "A deterministic density algorithm for pairwise interaction coverage," Proc. IASTED Intl. Conference on Software Engineering, pp.242–252, 2004.

[44] R.C. Bryce, C.J. Colbourn, and M.B. Cohen, "A framework of greedy methods for constructing interaction test suites," Proc. 27th International Conference on Software Engineering (ICSE '05), pp.146–155, 2005.

[45] T. Shiba, T. Tsuchiya, and T. Kikuno, "Using artificial life techniques to generate test cases for combinatorial testing," Proc. 28th Annual International Computer Software and Applications Conference (COMPSAC '04), pp.71–77, 2004.

[46] K.C. Tai and Y. Lie, "A test generation strategy for

pairwise testing," IEEE Trans. Softw. Eng., vol.28, no.1, pp.109–111, 2002.
[47] M.B. Cohen, P.B. Gibbons, W.B. Mugridge, and C.J. Colbourn, "Constructing test suites for interaction testing," Proc. 25th International Conference on Software Engineering (ICSE '03), pp.38–48, 2003.
[48] K.J. Nurmela, "Upper bounds for covering arrays by tabu search," Discrete Appl. Math., vol.138, no.1-2, pp.143–152, 2004.
[49] A.W. Williams and R.L. Probert, "Formulation of the interaction test coverage problem as an integer program," Proc. 14th International Conference on the Testing of Communicating Systems (TestCom 2002), pp.283–298, Berlin, Germany, 2002.
[50] B. Hnich, S.D. Prestwich, E. Selensky, and B.M. Smith, "Constraint models for the covering test problem," Constraints, vol.11, no.2-3, pp.199–219, 2006.
[51] J. Yan and J. Zhang, "Backtracking algorithms and search heuristics to generate test suites for combinatorial testing," Proc. 30th Annual International Computer Software and Applications Conference (COMPSAC '06), pp.385–394, 2006.
[52] A. Dumitrescu, "Efficient algorithms for generation of combinatorial covering suites," Proc. International Symposium on Algorithms and Computation (ISAAC2003), LNCS, vol.2906, pp.300–308, 2003.
[53] M. Chateauneuf and D.L. Kreher, "On the state of strength-three covering arrays," J. Combinatorial Designs, vol.10, no.4, pp.217–238, 2002.
[54] M.B. Cohen, C.J. Colbourn, and A.C.H. Ling, "Constructing strength three covering arrays with augmented annealing," Discrete Mathematics (to appear).
[55] M.B. Cohen, P.B. Gibbons, W.B. Mugridge, C.J. Colbourn, and J.S. Collofello, "Variable strength interaction testing of components," Proc. 27th Annual International Computer Software and Applications Conference (COMPSAC '03), pp.413–418, 2003.
[56] X. Cai and M.R. Lyu, "The effect of code coverage on fault detection under different testing profiles," SIGSOFT Softw. Eng. Notes, vol.30, no.4, pp.1–7, 2005.
[57] D.M. Cohen, S.R. Dalal, J. Parelius, and G.C. Patton, "The combinatorial design approach to automatic test generation," IEEE Softw., vol.13, no.5, pp.83–88, 1996.
[58] N. Kobayashi, T. Tsuchiya, and T. Kikuno, "Non-specification-based approaches to logic testing for software," J. Inf. Softw. Technol., vol.44, no.2, pp.113–121, 2002.
[59] K.Z. Bell and M.A. Vouk, "On effectiveness of pairwise methodology for testing network-centric software," Proc. ITI 3rd International Conference on Information and Communications Technology, 2005: Enabling Technologies for the New Knowledge Society, pp.221–235, Dec. 2005.
[60] P.J. Schroeder, P. Bolaki, and V. Gopu, "Comparing the fault detection effectiveness of n-way and random test suites," Proc. 2004 International Symposium on Empirical Software Engineering (ISESE2004), pp.49–59, 2004.
[61] J. Bach and P.J. Schroeder, "Pairwise testing: A best practice that isn't," Proc. 22nd Annual Pacific Northwest Software Quality Conference, pp.175–191, 2004.
[62] S.R. Dalal and C.L. Mallows, "Factor-covering designs for testing software," Technometrics, vol.40, no.3, pp.234–243, 1998.

（平成 18 年 12 月 25 日受付，19 年 3 月 1 日再受付）

土屋　達弘　（正員）

平 7 大阪大学大学院基礎工学研究科前期課程了．博士（工学）．現在，大阪大学大学院情報科学研究科准教授．

菊野　亨　（正員：フェロー）

昭 50 大阪大学大学院基礎工学研究科博士後期課程了．工博．広島大学工学部助教授，大阪大学基礎工学部教授を経て，現在，大阪大学大学院情報科学研究科教授．