LETTER

New 2-Factor Covering Designs for Software Testing

Noritaka KOBAYASHI[†*a)], ***Nonmember***, **Tatsuhiro TSUCHIYA**[†], ***and*** **Tohru KIKUNO**[†], ***Regular Members***

SUMMARY 2-Factor covering designs, a type of combinatorial designs, have recently received attention since they have industrial applications including software testing. For these applications, even a small reduction on the size of a design is significant, because it directly leads to the reduction of testing cost. In this letter, we report ten new designs that we constructed, which improve on the previously best known results.
key words: *factor covering design, software testing, finite field*

1. Introduction

Software testing often consumes up to half of the overall software costs. Even for simple software, exhaustive testing is infeasible because the number of possible test cases is typically prohibitively large. Much research has been aimed at simultaneously achieving high efficacy and reducing testing cost by selecting test cases appropriately. One such approach is to use a set of test cases that covers all pair-wise combinations of parameter values [1], [4], [7]. This approach is based on the observation that a significant number of faults are caused by parameter interactions in many applications. To cover all pair-wise combinations of parameter values, only a small number of test cases are sufficient if we can select them appropriately. Thus this approach can lead to reduce the cost of testing.

As an example, consider the problem of testing a telephone switch's ability to place telephone calls [1]. Table 1 shows four parameters that define a simple test model. Each of the four parameters has three values. The *Call Type* parameter tells the type of call. Its values are Local, Long Distance, and International. The other three parameters (*Billing*, *Access* and *Status*) also have three values. The table thus defines a total of 81 ($= 3^4$) different test cases. To cover all pair-wise parameter interactions, however, we need only nine test cases, as shown in Fig. 1.

A test set that covers all pair-wise parameter interactions can be viewed as a kind of a combinatorial design. Dalal and Mallows named this type of design 2-*factor covering design* [3]. Let p denote the number of test parameters, and let v denote the number of values that each parameter has. When we have t test cases that cover all pair-wise interactions, we refer to the 2-factor covering design as a $(t, [v^p])$ design. In the rest of the letter we call the t components just *tuples*, instead of test cases.

For example, the design shown in Fig. 1 is a $(9, [3^4])$ design. We refer to the number of tuples in a design D as the *size* of the design and denote it by $|D|$. Since the values in any column can be permuted, we assume without loss of generality that all parameters take 1 in the first tuple in any design.

The size of a 2-factor covering design depends on its construction; therefore much research has been aimed at reducing the size. In this letter, we focus on constructing 2-factor covering designs with $v = 3$. For the case $v = 2$, a construction that can generate optimal 2-factor covering designs has been proposed [5]. However, the problem of constructing optimal 2-factor covering designs for the case $v = 3$ is not solved. This problem is called the *ternary Spernery* problem [6] and has received recent attention. Dalal and Mallows [3] summarized the best known 2-factor covering designs with $v = 3$ as shown in Table 2.

In this letter, we present ten new designs with $v = 3$, which improve on the existing results. One of the ten designs is a $(17, [3^{16}])$ design, which improves on the

Table 1 Parameters for placing a telephone call.

	A:*Call Type*	B:*Billing*	C:*Access*	D:*Status*
#1	Local	Caller	Loop	Success
#2	Long Distance	Collect	ISDN	Busy
#3	International	800	PBX	Blocked

Test	A	B	C	D
#1	1	1	1	1
#2	1	2	2	2
#3	1	3	3	3
#4	2	1	2	3
#5	2	2	3	1
#6	2	3	1	2
#7	3	1	3	2
#8	3	2	1	3
#9	3	3	2	1

Fig. 1 A $(9, [3^4])$ design.

Manuscript received October 5, 2001.
Manuscript revised April 15, 2002.
Final manuscript received August 2, 2002.
[†]The authors are with the Graduate School of Information Science and Technology, Osaka University, Toyonaka-shi, 560-8531 Japan.
[*]Presently, with Nomura Research Institute, Ltd.
a) E-mail: n-kobays@ics.es.osaka-u.ac.jp

Table 2 Sizes of the best known $(t, [3^p])$ designs reported in [3].

p	4	5	6	13	16	18	56	126
t	9	11	12	15	18	21	24	27

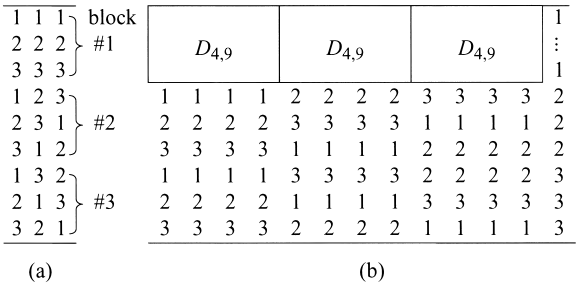

Fig. 2 (a) $B(3)$ and (b) $D_{13,15}(= D_{4,9} \times_R B(3))$.

#1	1	1	1	1	1	1	1
#2	2	2	2	2	1	2	2
#3	3	3	3	3	1	3	3
#4	1	3	2	3	2	1	2
#5	2	1	3	2	2	3	1
#6	3	2	1	3	3	2	1
#7	1	2	3	1	3	2	3
#8	3	1	2	1	3	3	2
#9	2	3	1	2	3	1	3
#10	2	1	2	3	2	2	3
#11	3	2	3	2	2	1	2
#12	2	3	2	1	2	2	1
#13	1	2	1	2	2	3	2

Fig. 3 A $(13, [3^7])$ design.

previously best known size for the case $p = 17$. Other nine designs have different numbers of parameters from those summarized in Table 2.

2. Basic Constructions

In this section we present three known constructions. Using these constructions we have obtained the ten new designs.

2.1 Heuristic Search-Based Algorithm [1]

AETG [1], an actual test generation system, employs a heuristic algorithm. The algorithm incrementally constructs a 2-factor covering design as follow. Assume that n tuples have already been constructed. The $(n+1)$th tuple is obtained by first generating some fixed number (e.g., 100 or 200) of different candidate tuples at random and then choosing one that covers the most new combinations of parameter values.

As stated later, the design shown in Fig. 3 was obtained by this algorithm. Using this design, we will explain the algorithm in brief. For example, suppose that the first two tuples, #1 and #2, have been obtained. Then the next tuple is generated as follows. First some number of tuples are generated randomly as candidates. Then for each candidate tuple, the number of new combinations it covers is computed. After that a tuple that covers the most new combinations is selected from the candidates.

In this case, since $p = 7$ and $v = 3$, there are a total of $v^2p(p-1)/2 = 189$ combinations to be covered, and each tuple can cover at most $p(p-1)/2 = 21$ combinations. The first two tuples cover 42 different combinations, so there remain 147 combinations that have not appeared yet. When we obtained this design, (3, 3, 3, 3, 1, 3, 3) happened to be generated as a candidate. This tuple covers 21 of the remaining combinations. Clearly there was no other candidate that covers more than 21 new combinations, and thus the tuple was selected as the third tuple. This process continues until all combinations are covered by at least one tuple.

2.2 Simple Product [8]

Let F_1 and F_2 be 2-factor covering designs for $[v^p]$ and $[v^q]$, respectively. The simple product, $F_1 \times F_2$, which is a 2-factor covering design for $[v^{pq}]$, can be constructed as follows. Let F_2' be a design that is obtained by removing the first tuple (that is, the tuple $(1, 1, \cdots, 1)$) from F_2.

For each tuple $(t_1, t_2, \cdots, t_p)$ in F_1, we include in the product the tuple

$$(t_1, t_2, \cdots, t_p, t_1, t_2, \cdots, t_p, \cdots\cdots, t_1, t_2, \cdots, t_p).$$

For each tuple $(t_1, t_2, \cdots, t_q)$ in F_2', we include in the product the tuple

$$(\underbrace{t_1, t_1, \cdots, t_1}_{p}, \underbrace{t_2, t_2, \cdots, t_2}_{p}, \cdots\cdots, \underbrace{t_q, t_q, \cdots, t_q}_{p}).$$

The size of the product is $|F_1| + |F_2| - 1$.

2.3 Reduced Product [2]

To describe this concept, we first define *block structured* 2-factor covering designs [2]. We assume that $v = z^j$ for some prime z and integer $j \geq 1$ (i.e., v is a prime power number). It is well known that this is the necessary and sufficient condition for a finite field with v elements to exist. A 2-factor covering design for $[v^p]$ is *block structured*, provided that the design can be partitioned into blocks such that each consists of the v translates, $(i \oplus x_1, \cdots, i \oplus x_p), i \in \{1, 2, \cdots, v\}$, of some tuple $(x_1, \cdots, x_p)$ in the block, where $\oplus$ denotes addition over the finite field. Suppose that blocks are numbered from one to the total number of the blocks, and that the block that includes the first tuple is the first block.

Now we describe the reduced product. Let F_1 and F_2 be 2-factor covering designs for $[v^p]$ and $[v^q]$, respectively. Assume that F_2 is block structured. The

reduced product, $F_1 \times_R F_2$, which is a 2-factor covering design for $[v^{pq+1}]$, can be constructed as follows [2]. Let F_2' be a design that is obtained by removing the first block from F_2. For each tuple $(t_1, t_2, \cdots, t_p)$ in F_1, we include in the product the tuple

$$(t_1, t_2, \cdots, t_p, t_1, t_2, \cdots, t_p, \cdots\cdots, t_1, t_2, \cdots, t_p, 1).$$

For each tuple $(t_1, t_2, \cdots, t_q)$ in F_2', we include in the product the tuple

$$(\underbrace{t_1, t_1, \cdots, t_1}_{p}, \underbrace{t_2, t_2, \cdots, t_2}_{p}, \cdots\cdots, \underbrace{t_q, t_q, \cdots, t_q}_{p}, i)$$

where i denotes the number assigned to the block that contains the tuple $(t_1, t_2, \cdots, t_q)$. The size of the reduced product is $|F_1| + |F_2| - v$.

In [2], it is shown that a finite field with v elements yields a $(v^2, [v^{v+1}])$ design and a block structured $(v^2, [v^v])$ design. Let $B(v)$ denote a block structured $(v^2, [v^v])$ design. In the remainder of this letter, we denote a $(t, [3^p])$ design by $D_{p,t}$. For example, a $(9, [3^4])$ design shown in Fig. 1 is $D_{4,9}$. Figure 2 shows $B(3)$ and $D_{4,9} \times_R B(3)$.

3. Results

In this section, we present the ten new designs obtained by using the three constructions described in the previous section. Table 3 shows the number of parameters, p, and the size, t, for each of the new designs. Due to space limits, we show the five smallest designs. The $(13, [3^7])$ design in Fig. 3 was constructed by the heuristic search-based algorithm. The $(17, [3^{16}])$ design and the $(20, [3^{24}])$ design in Figs. 4 and 5 were constructed as the simple products $D_{4,9} \times D_{4,9}$ and $D_{6,12} \times D_{4,9}$, respectively.

The $(18, [3^{19}])$ design and the $(19, [3^{22}])$ design in Figs. 6 and 7 were the reduced product $D_{6,12} \times_R B(3)$ and $D_{7,13} \times_R B(3)$, respectively. As shown in Table 3, the remaining five new designs, $D_{40,21}$, $D_{49,23}$, $D_{58,24}$,

Table 3 Constructions of the new $(t, [3^p])$ designs.

Design	p	t	Construction method
$D_{7,13}$	7	13	The heuristic algorithm
$D_{16,17}$	16	17	$D_{4,9} \times D_{4,9}$
$D_{19,18}$	19	18	$D_{6,12} \times_R B(3)$
$D_{22,19}$	22	19	$D_{7,13} \times_R B(3)$
$D_{24,20}$	24	20	$D_{6,12} \times D_{4,9}$
$D_{40,21}$	40	21	$D_{13,15} \times_R B(3)$
$D_{49,23}$	49	23	$D_{16,17} \times_R B(3)$
$D_{58,24}$	58	24	$D_{19,18} \times_R B(3)$
$D_{67,25}$	67	25	$D_{22,19} \times_R B(3)$
$D_{73,26}$	73	26	$D_{24,20} \times_R B(3)$

#1	1111	1111	1111	1111
#2	1222	1222	1222	1222
#3	1333	1333	1333	1333
#4	2123	2123	2123	2123
#5	2231	2231	2231	2231
#6	2312	2312	2312	2312
#7	3132	3132	3132	3132
#8	3213	3213	3213	3213
#9	3321	3321	3321	3321
#10	1111	2222	2222	2222
#11	1111	3333	3333	3333
#12	2222	1111	2222	3333
#13	2222	2222	3333	1111
#14	2222	3333	1111	2222
#15	3333	1111	3333	2222
#16	3333	2222	1111	3333
#17	3333	3333	2222	1111

Fig. 4 A $(17, [3^{16}])$ design.

#1	111111	111111	111111	111111
#2	122133	122133	122133	122133
#3	132321	132321	132321	132321
#4	131232	131232	131232	131232
#5	123312	123312	123312	123312
#6	221223	221223	221223	221223
#7	213333	213333	213333	213333
#8	233121	233121	233121	233121
#9	212212	212212	212212	212212
#10	312122	312122	312122	312122
#11	323231	323231	323231	323231
#12	331313	331313	331313	331313
#13	111111	222222	222222	222222
#14	111111	333333	333333	333333
#15	222222	111111	222222	333333
#16	222222	222222	333333	111111
#17	222222	333333	111111	222222
#18	333333	111111	333333	222222
#19	333333	222222	111111	333333
#20	333333	333333	222222	111111

Fig. 5 A $(20, [3^{24}])$ design.

#1	111111	111111	111111	1
#2	122133	122133	122133	1
#3	132321	132321	132321	1
#4	131232	131232	131232	1
#5	123312	123312	123312	1
#6	221223	221223	221223	1
#7	213333	213333	213333	1
#8	233121	233121	233121	1
#9	212212	212212	212212	1
#10	312122	312122	312122	1
#11	323231	323231	323231	1
#12	331313	331313	331313	1
#13	111111	222222	333333	2
#14	222222	333333	111111	2
#15	333333	111111	222222	2
#16	111111	333333	222222	3
#17	222222	111111	333333	3
#18	333333	222222	111111	3

Fig. 6 A $(18, [3^{19}])$ design.

Table 4 Sizes of the best known $(t, [3^p])$ designs.

p	4	5	6	7	13	16	19	22	24	40	49	58	67	73	126
t	9	11	12	13	15	17	18	19	20	21	23	24	25	26	27

#1	1111111	1111111	1111111	1
#2	2222122	2222122	2222122	1
#3	3333133	3333133	3333133	1
#4	1323212	1323212	1323212	1
#5	2132231	2132231	2132231	1
#6	3213321	3213321	3213321	1
#7	1231323	1231323	1231323	1
#8	3121332	3121332	3121332	1
#9	2312313	2312313	2312313	1
#10	2123223	2123223	2123223	1
#11	3232212	3232212	3232212	1
#12	2321221	2321221	2321221	1
#13	1212232	1212232	1212232	1
#14	1111111	2222222	3333333	2
#15	2222222	3333333	1111111	2
#16	3333333	1111111	2222222	2
#17	1111111	3333333	2222222	3
#18	2222222	1111111	3333333	3
#19	3333333	2222222	1111111	3

Fig. 7 A $(19, [3^{22}])$ design.

$D_{67,25}$, and $D_{73,26}$, can be constructed by making use of $D_{13,15}$ (shown in Fig. 2) and the above four designs.

Note that these designs improve the best known results for a large range of values of p. For example, since we have $(17, 3^{16})$ and $(18, 3^{19})$ designs, the best known size is improved from 18 to 17 for $14 \leq p \leq 16$ and from 21 to 18 for $17 \leq p \leq 19$. Table 4 summarizes the best known results updated by our results.

References

[1] 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. Software Eng., vol.23, no.7, pp.437–443, July 1997.

[2] 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.

[3] S.R. Dalal and C.L. Mallows, "Factor-covering designs for testing software," Technometrics, vol.40, no.3, pp.234–243, Aug. 1998.

[4] I.S. Dunietz, W.K. Ehrlich, B.D. Szablak, C.L. Mallows, and A. Iannino, "Applying design of experiments to software testing," Proc. IEEE Int'l Conf. Software Eng., pp.205–215, 1997.

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

[6] N.J.A. Sloane, "Covering arrays and intersecting codes," J. Combinatorial Designs, vol.1, no.1, pp.51–63, 1993.

[7] C.H. West, "Protocol validation—Principles and applications," Computer Networks and ISDN Systems, vol.24, no.3, pp.219–242, May 1992.

[8] A.W. Williams and R.L. Probert, "A practical strategy for testing pair-wise coverage of network interfaces," Proc. IEEE 7th Int'l Symp. Software Reliability Eng., pp.246–254, 1997.