Products of Mixed Covering Arrays of Strength Two

Charles J. Colbourn and Sosina S. Martirosyan
Dept. of Computer Science and Engineering
Arizona State University
P.O. Box 878809
Tempe, Arizona 85287
{charles.colbourn,sosina.martirosyan}@asu.edu

Gary L. Mullen
Department of Mathematics
The Pennsylvania State University
University Park, PA 16802, USA
mullen@math.psu.edu

Dennis Shasha
Department of Computer Science
Courant Institute of Mathematical Sciences
New York University
251 Mercer Street, New York, NY 10012
shasha@cs.nyu.edu

George B. Sherwood
Testcover.com LLC
41 Clover Hill Road
Colts Neck, NJ 07722
gsherwood@att.net

Joseph L. Yucas
Department of Mathematics
Southern Illinois University
Carbondale, IL 62901
jyucas@math.siu.edu

Abstract

A *covering array* $CA(N; t, k, v)$ is an $N \times k$ array such that every $N \times t$ sub-array contains all t-tuples from v symbols *at least* once, where t is the *strength* of the array. Covering arrays are used to generate software test suites to cover all t-sets of component interactions. The particular case when $t = 2$ (pairwise coverage) has been extensively studied, both to develop combinatorial constructions and to provide effective algorithmic search techniques. In this paper, a simple "cut-and-paste" construction is extended to covering arrays in which different columns (factors) admit different numbers of symbols (values); in the process an improved recursive construction for covering arrays with $t = 2$ is derived.

1 Introduction

Component based software development poses many challenges for the software tester. Interactions among components are complex and numerous. Components are prone to unexpected interaction faults. Ideally we would test all possible interactions, but this is usually infeasible. Consequently, we are interested in generating test suites that provide coverage of the most prevalent interactions.

Suppose that we have 20 components. If two of these have four possible configurations, while the rest have three, we have $4^2 \times 3^{18}$ or 6,198,727,824 possible interactions. We can cover all of the two-way interactions among these components with as few as 19 tests. Likewise, we can cover the three way interactions with only 90 tests. Recently, these methods have been applied to the generation of software test suites allowing one to guarantee certain interaction coverage in software systems [3, 4, 5, 6, 10, 11, 25, 26, 27, 28].

At the current time there are two primary areas of active research on combinatorial designs for software testing. The mathematics community is focusing on building smaller designs of higher interaction strength [1, 2, 21, 23, 24]. The software testing community is focusing on greedy search algorithms to build arrays in a more flexible environment, one that more closely matches real testing needs [3, 4, 10, 11, 25, 26, 27]; in addition, more powerful search techniques such as simulated annealing have been employed recently [5, 6]. Ideally we ought to combine these ideas to build interaction test suites that are minimal and efficient to generate. An initial investigation along these lines was conducted in [7]. Since the methods of building covering arrays for testing are varied, a trade-off must occur between computational power and the cost of running the final test suites. In this paper we examine some methods of combining computational search and recursive combinatorial construction to build test suites efficiently. Part of our main objective is to develop a flexible combinatorial construction; then heuristic search can be used to find ingredients for use in the construction.

2 Covering Arrays

The problems faced in software interaction testing are not unique. Similar problems exist for testing in other disciplines such as agriculture, pharmaceuticals, manufacturing and medicine [8, 12, 19]. The primary combinatorial objects used to satisfy the coverage criteria for these types of problems are orthogonal arrays and covering arrays. We begin with a few definitions.

An *orthogonal array* $OA_\lambda(t,k,v)$ is an $\lambda v^t \times k$ array on v symbols such that every $\lambda v^t \times t$ sub-array contains each ordered subset of size t from v symbols *exactly* λ times. When $\lambda = 1$ we omit the subscript. We do not need such a stringent object for software testing. Instead we can use a covering array that allows some duplication of coverage.

A *covering array* $CA_\lambda(N;t,k,v)$ is an $N \times k$ array such that every $N \times t$ sub-array contains all tuples from v symbols of size t *at least* λ times each. When N is unknown or unspecified, the notation $CA_\lambda(t,k,v)$ is also used. When $\lambda = 1$ we omit the subscript. The *covering array number* $CAN(t,k,v)$ is the minimum number N of rows required to produce a $CA(N;t,k,v)$. In a covering array $CA(t,k,v)$, t is the *strength*, k the *degree*, and v the *order*. We focus on strength $t = 2$ in this paper.

It can happen that some of the entries of the array are not needed in order to cover all t-tuples. In this case, we can replace the entry of the array by $\star$, to indicate a "don't care" position. When such a replacement is made, t-tuples containing a $\star$ are deemed not to match a t-tuple of the v symbols. The *profile* $(d_1,\ldots,d_k)$ of an $N \times k$ array is a k-tuple in which the entry d_i is the number of $\star$ entries in the ith column.

Often factors have different numbers of levels. A *mixed level covering array* $MCA(N;t,k,(v_1,\ldots,v_k))$ is an $N \times k$ array in which the entries of the ith column arise from an alphabet of size v_i; in addition, choosing any t distinct columns $i_1,\ldots,i_t$, every t-tuple containing, for $1 \le j \le t$, one of the v_{i_j} entries of column i_j, appears in columns $i_1,\ldots,i_t$ in at least one of the N rows. We sometimes use exponential notation, writing $s_1^{u_1} \cdots s_\ell^{u_\ell}$ to indicate that there are $k = \sum_{i=1}^{\ell} u_i$ factors, of which u_i have s_i levels for $1 \le i \le \ell$. As for covering arrays, we can omit N when it is not determined; we can also introduce $\star$ entries as before, and define a profile to indicate their distribution in the MCA. An example of type $5^2 3^6 2^{13}$ is given in Figure 1; while it is optimal in terms of number of tests (since $5 \times 5 = 25$), other distributions of $\star$ entries can arise.

1	1	⋆	⋆	⋆	1	⋆	⋆	⋆	1	⋆	⋆	⋆	⋆	⋆	1	⋆	⋆	⋆	⋆	⋆
1	2	2	⋆	⋆	2	2	2	⋆	⋆	2	⋆	2	⋆	⋆	⋆	2	⋆	⋆	2	2
1	3	3	3	3	3	3	3	1	1	2	1	1	1	1	1	2	1	1	1	⋆
1	4	1	1	1	1	2	3	2	2	1	2	2	1	2	2	1	2	2	2	1
1	5	1	2	2	2	1	1	1	1	1	1	1	2	2	1	1	1	2	1	2
2	1	2	3	1	2	3	1	2	⋆	2	2	1	2	⋆	2	⋆	2	⋆	1	2
2	2	1	⋆	3	1	1	2	1	⋆	1	1	2	1	1	2	1	1	1	⋆	2
2	3	3	2	2	1	2	2	2	2	⋆	2	⋆	2	1	⋆	⋆	⋆	2	2	1
2	4	⋆	2	⋆	3	1	3	2	1	2	2	2	2	2	2	2	⋆	1	⋆	2
2	5	2	1	3	3	2	1	⋆	1	1	1	⋆	1	1	1	2	2	1	2	1
3	1	3	1	2	2	1	3	⋆	2	⋆	⋆	⋆	1	2	⋆	1	1	1	⋆	⋆
3	2	1	3	⋆	3	3	2	⋆	1	2	⋆	⋆	⋆	⋆	1	1	2	2	2	1
3	3	⋆	2	1	⋆	⋆	1	⋆	⋆	1	⋆	2	⋆	⋆	⋆	2	⋆	⋆	⋆	2
3	4	2	⋆	3	2	2	2	1	2	⋆	2	1	2	1	1	⋆	1	⋆	1	1
3	5	3	3	1	1	3	2	2	2	1	1	2	⋆	⋆	2	⋆	⋆	⋆	⋆	⋆
4	1	1	2	3	⋆	2	2	2	1	1	⋆	2	1	2	1	2	1	2	2	1
4	2	3	1	1	1	3	1	1	⋆	2	⋆	1	2	1	⋆	⋆	2	1	1	2
4	3	2	3	2	2	1	⋆	⋆	⋆	⋆	⋆	⋆	⋆	2	2	1	2	⋆	⋆	⋆
4	4	⋆	⋆	2	3	3	1	⋆	2	⋆	1	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆
4	5	⋆	⋆	⋆	⋆	⋆	3	⋆	⋆	2	2	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆
5	1	⋆	⋆	1	3	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1
5	2	2	2	2	1	3	3	2	2	2	2	2	2	2	2	2	2	2	1	2
5	3	1	1	3	2	⋆	2	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆
5	4	3	3	⋆	⋆	2	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆
5	5	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆

Figure 1: An $MCA(25; 2, 21, 5^2 3^6 2^{13})$

3 Products of Strength Two Arrays

When a $CA(N; 2, k, v)$ and a $CA(M; 2, \ell, v)$ both exist, it is an easy matter to produce a $CA(N + M; 2, k\ell, v)$. To be specific, let $A = (a_{ij})$ be a $CA(N; 2, k, v)$ and let $B = (b_{ij})$ be a $CA(M; 2, \ell, v)$. Form an $(N + M) \times k\ell$ array $C = (c_{i,j}) = A \otimes B$ by setting $c_{i,(f-1)k+g} = a_{i,g}$ for $1 \leq i \leq N$, $1 \leq f \leq \ell$, and $1 \leq g \leq k$. Then set $c_{N+i,(f-1)k+g} = b_{i,f}$ for $1 \leq i \leq M$, $1 \leq f \leq \ell$, and $1 \leq g \leq k$. In essence, k copies of $B = (b_{ij})$ are being appended to ℓ copies of $A = (a_{ij})$ as shown in Figure 2. Since two different columns of C arise either from different columns of A or from two different columns of B, the result is a $CA(N + M; 2, k\ell, v)$. This appears, in different vernacular, in [18]; it is also the essence of the *block recursive construction* from [23].

We first consider two extensions of this simple concatenation, to allow mixed level covering arrays and to exploit "don't care" positions. The extension to mixed levels is treated in [15], without exploiting the $\star$ positions. To simplify the presentation, we assume a factor with v values always takes on values from $\{1, \ldots, v\}$, and hence the corresponding column of the array contains

$$\begin{array}{r|cccc|cccc|c|cccc|}
\cline{2-15}
 & a_{11} & a_{12} & \cdots & a_{1k} & a_{11} & a_{12} & \cdots & a_{1k} & \cdots & a_{11} & a_{12} & \cdots & a_{1k} \\
 & a_{21} & a_{22} & \cdots & a_{2k} & a_{21} & a_{22} & \cdots & a_{2k} & \cdots & a_{21} & a_{22} & \cdots & a_{2k} \\
N \text{ rows} & \vdots & & & & \vdots & & & & \cdots & \vdots & & & \\
 & a_{N1} & a_{N2} & \cdots & a_{Nk} & a_{N1} & a_{N2} & \cdots & a_{Nk} & \cdots & a_{N1} & a_{N2} & \cdots & a_{Nk} \\
\cline{2-15}
 & b_{11} & b_{11} & \cdots & b_{11} & b_{12} & b_{12} & \cdots & b_{12} & \cdots & b_{1\ell} & b_{1\ell} & \cdots & b_{1\ell} \\
 & b_{21} & b_{21} & \cdots & b_{21} & b_{22} & b_{22} & \cdots & b_{22} & \cdots & b_{2\ell} & b_{2\ell} & \cdots & b_{2\ell} \\
M \text{ rows} & \vdots & & & & \vdots & & & & \cdots & \vdots & & & \\
 & b_{M1} & b_{M1} & \cdots & b_{M1} & b_{M2} & b_{M2} & \cdots & b_{M2} & \cdots & b_{M\ell} & b_{M\ell} & \cdots & b_{M\ell} \\
\cline{2-15}
\end{array}$$

Figure 2: The structure of $A \otimes B$

only these symbols, and possibly $\star$.

Theorem 3.1 *Suppose that there exist*

1. *an $MCA(N; 2, k, (v_1, \ldots, v_k))$, A, with profile $(d_1, \ldots, d_k)$;*
2. *for each $1 \leq i \leq k$, an $MCA(M_i; 2, \ell_i, (w_{i1}, \ldots, w_{i,\ell_i}))$, B_i, with profile $(f_{i1}, \ldots, f_{i,\ell_i})$, and for which $w_{ij} \leq v_i$ for $1 \leq j \leq \ell_i$.*

Then for $T = N + max_{i=1}^{k}(M_i - d_i)$, there exists an

$$MCA(T; 2, \sum_{i=1}^{k} \ell_i, (w_{11}, \ldots, w_{1,\ell_1}, \cdots, w_{k1}, \ldots, w_{k,\ell_k})).$$

Proof. Form an array C with T rows and $\sum_{i=1}^{k} \ell_i$ columns, indexing columns as (i, j) for $1 \leq i \leq k$ and $1 \leq j \leq \ell_i$. On the first N rows, column (i, j) is column i of A. (At this stage, it is possible that $v_i > w_{ij}$; if so, each occurrence of $v_i - w_{ij}$ of the symbols can be changed to $\star$.) Now for $i = 1, \ldots, k$, select the d_i rows of A in which the ith column contains a $\star$, and from the last $T - N$ rows choose any $M_i - d_i$ rows, so that M_i rows are selected in total. Then *on these rows in the chosen order*, place the entries of the jth column of B_i in the column (i, j), for $1 \leq j \leq \ell_i$.

Two columns (i_1, j_1) and (i_2, j_2) of the result cover all pairs when $i_1 \neq i_2$; indeed these are covered on the first N rows. When $i_1 = i_2$, restricting to rows arising from B_{i_1}, we find all pairs covered. ■

While C is sufficient to cover all pairs, there is much redundant coverage. A simple argument takes advantage of this. Call a row of an MCA *constant* if the only pairs that it covers are of the form (x, x). Suppose that we require every ingredient, without loss of generality, to contain a constant row consisting of all "1" entries. If this is done, then by always selecting the same row of C in which to place the constant row of each B_i, the resulting matrix C has two constant rows of "1" entries. One is redundant and can be removed, leaving no pair uncovered but reducing the size of C. Although this can be done in general, it saves one test only. This simple argument imposes too stringent a condition, since we do not need duplicated rows to have redundant rows. Phrased more generally, then, since the rows of C arising from A, in two columns indexed by (i, j_1) and (i, j_2) cover *all* pairs of the form (x, x), we can delete all constant rows of each B_i prior to applying

Theorem 3.1. This can reduce the size of C by many rows in some instances. Moura, Stardom, Stevens, and Williams [15] exploit constant rows effectively to reduce the size of the resulting array.

Unfortunately, in some cases we must sacrifice columns to obtain many constant rows. When q is a prime power, the standard $CA(q^2; 2, q, q)$ from the finite field on q elements has q disjoint constant rows, but its extension to a $CA(q^2; 2, q+1, q)$ can have at most one, no matter how the symbols are relabeled. For example, taking two $CA(25; 2, 6, 5)$s allows us to produce a $CA(49; 2, 36, 5)$ while using instead one $CA(25; 2, 5, 5)$ with five disjoint constant rows allows us to obtain a $CA(45; 2, 30, 5)$. We examine a generalization that enables us to obtain a $CA(45; 2, 35, 5)$, sacrificing one column in the product rather than one column in an ingredient.

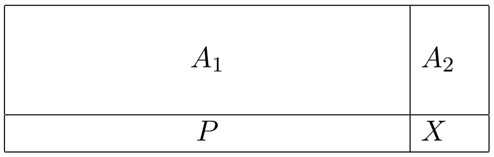

Figure 3: A partitioned covering array (PCA)

For the sake of clarity, we develop a construction for covering arrays first, and then generalize to mixed level covering arrays. We consider covering arrays exhibiting a specific structure. Consider a $CA(N; 2, k_1 + k_2, v)$, shown in Figure 3. Here A_1, A_2, and X are $(N - v) \times k_1$, $(N - v) \times k_2$, and $v \times k_2$ arrays, respectively. However P is a $v \times k_1$ array with a specific structure, namely that every column is a permutation of $\{1, \dots, v\}$. When a $CA(N; 2, k_1 + k_2, v)$ admits such a partition, it is a partitioned covering array $PCA(N; 2, (k_1, k_2), v)$. The structure is not altered by applying (possibly different) permutations to the v symbols in each column, and hence without loss of generality P can be assumed to be the matrix D in which each column is the identity permutation. If we assume in addition that X is a constant matrix (all entries equal), and without loss of generality that X is the all ones matrix O, we obtain a further restriction: An $SCA(N; 2, (k_1, k_2), v)$ is defined to be a $CA(N; 2, k_1 + k_2, v)$ in which, for $1 \leq i \leq v$, row $N - v + i$ is a $(k_1 + k_2)$-tuple in which the first k_1 entries are equal to i and the last k_2 symbols are equal to 1. When q is a prime power, an $OA(2, q + 1, q)$ yields an $SCA(q^2; 2, (q, 1), q)$.

Now we turn to the main product construction for covering arrays:

Theorem 3.2 *If a $PCA(N; 2, (k_1, k_2), v)$ and an $SCA(M; 2, (\ell_1, \ell_2); v)$ both exist, then a $PCA(N + M - v; 2, (k_1\ell_1, k_1\ell_2 + k_2\ell_1), v)$ also exists.*

Proof. Take a $PCA(N; 2, (k_1, k_2), v)$ with a partition as in Figure 3 into A_1, A_2, D and X; and an $SCA(M; 2, (\ell_1, \ell_2), v)$ with partition B_1, B_2, D, and O. Form an array as in Figure 4. In the products of the form $A_i \otimes B_j$, the first $N - v$ rows arise from A_i while the next $M - v$ arise from B_j (see Figure 2. Here $k_1 X$ is obtained by repeating the array X k_1 times. D and O are the matrix of identity permutations and of all ones, of appropriate dimension.

We claim that the result R is an $PCA(N + M - v; 2, ((k_1\ell_1, k_1\ell_2 + k_2\ell_1), v)$. Among the first $k_1\ell_1$ columns, two columns arising from different columns of A_1 have all pairs covered in the first $N - v$ and last v rows of R, since these form columns of the PCA. When they arise from the same column of A_1, they arise from different columns of B_1 and hence all pairs are covered in the last M rows of R, since these form columns of the SCA.

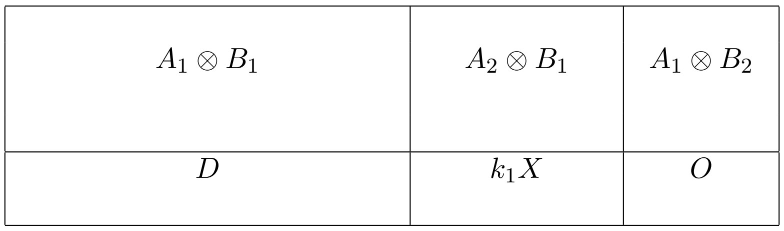

Figure 4: The product of a PCA and an SCA

If both columns arise from the next $k_2\ell_1$ columns of R, the same argument assures that all pairs are covered when they arise from different columns of A_2. If they arise from the same column of A_2 (and therefore different columns of B_1), all pairs are covered in the $M - v$ rows except those of the form (i, i) for $1 \leq i \leq v$. Since both columns arise from the same column of A_2 (and therefore repeat the same column of X in the latter v rows as well), every pair of the form (i, i) is covered among the first $N - v$ and last v rows.

If both columns arise from the last $k_1\ell_2$ columns of R, when the columns arise from different columns of B_2 all pairs are covered in the last M rows since these form columns of the SCA. If they arise from the same column of B_2, all pairs are covered except those of the form (i, i) for $1 \leq i \leq v$; these are covered in the $M - v$ rows.

If one column arises from the first $k_1\ell_1$ columns and another from the next $k_2\ell_1$ columns, the first $N - v$ and last v rows form distinct columns of the PCA.

If one column arises from the first $k_1\ell_1$ columns and another from the last $k_1\ell_2$ columns, the last M rows form distinct columns of the SCA.

If one column arises from the second group of $k_2\ell_1$ columns and another from the last $k_1\ell_2$ columns, the $M - v$ rows cover all pairs except possibly those of the form $(x, 1)$ for $1 \leq x \leq v$, and these are covered in the first $N - v$ and last v rows.

This treats all six cases for the choice of two columns, and hence R is a covering array. That R is a $PCA(N + M - v; 2, (k_1\ell_1, k_1\ell_2 + k_2\ell_1), v)$ follows from the fact that R has a $v \times k_1\ell_1$ subarray consisting of (column) identity permutations, occurring in the last v rows. ■

When the PCA in this construction is in fact an SCA, the result is also an SCA.

A substantial further improvement is possible. One can, on occasion, find a larger submatrix in the result R that contains column identity permutations, and hence provide a better ingredient for the next iteration of the recursion. We explore this next. Consider the SCA used. Were it to contain within B_1 an $v \times \eta$ subarray in which every column is a permutation of $\{1, \ldots, v\}$, then let us examine the impact on the covering array R constructed in Theorem 3.2. Each column of B_1 is replicated $k_1 + k_2$ times in total, and hence R contains a $v \times \eta(k_1 + k_2)$ subarray in which every column is a permutation. If $\eta(k_1 + k_2) > k_1\ell_1$, we can permute symbols within each column, and permute rows:

Theorem 3.3 *When the SCA in Theorem 3.2 contains within B_1 a $v \times \eta$ subarray whose columns are permutations, the result is a $PCA(N + M - v; 2, (\eta(k_1 + k_2), (\ell_1 - \eta)(k_1 + k_2) + k_1\ell_2), v)$.*

A similar observation applies to subarrays consisting of column permutations in the PCA.

We illustrate the product constructions by establishing some applications leading to new covering array numbers. We focus on the case when every factor has the same number of levels, since tables are available in the literature [14, 23].

Lemma 3.4 *When q is a prime power, and $r \geq 0$ is any integer, there is an $SCA((r+1)q^2 - rq; 2, (q^{r+1}, (r+1)q^r), q)$ and hence a $CA((r+1)q^2 - rq; 2, q^{r+1} + (r+1)q^r, q)$.*

Proof. Apply Theorem 3.2 inductively r times using an $SCA(q^2; 2, (q, 1), q)$. ■

Some small examples are

$$\begin{array}{lll} CA(15;2,15,3) & CA(21;2,54,3) & CA(27;2,189,3) \\ CA(28;2,24,4) & CA(40;2,112,4) & CA(52;2,516,4) \\ CA(45;2,35,5) & CA(65;2,200,5) & CA(85;2,1125,5) \\ CA(91;2,63,7) & CA(133;2,490,7) & CA(175;2,3673,7) \end{array}$$

These improve upon the construction in [23]. However, further improvements result from Theorem 3.3. For a non-negative integer r we define $D_{r,t}$ by

$$D_{r,t} = \sum_{i=1}^{s} \binom{r-i}{i-1} t^{r-i},$$

where $s = \lfloor \frac{r+1}{2} \rfloor$.

The numbers $D_{r,t}$ satisfy the recurrence

$$D_{0,t} = 0;$$

$$D_{1,t} = 1;$$

$$D_{r+1,t} = tD_{r,t} + tD_{r-1,t}.$$

Lemma 3.5 *When q is a prime power, and $r \geq 0$ is any integer, there is an $SCA((r+1)q^2 - rq; 2, ((q+1)D_{r+1,q}, qD_{r,q}), q)$ and hence a $CA((r+1)q^2 - rq; 2, (q+1)D_{r+1,q} + qD_{r,q}, q)$.*

Proof. The $SCA(q^2; 2, (q, 1), q)$ has $\eta = q$ column permutations within B_1. Apply Theorem 3.3 to obtain an $SCA(2q^2 - q; 2, ((q+1)q, q), q)$. This has $(q+1)D_{2,q} + qD_{1,q}$ columns. Apply Theorem 3.3 using the $SCA(q^2; 2, (q, 1), q)$ at each iteration. ■

Some small examples are

$$\begin{array}{llll} CA(15;2,15,3) & CA(21;2,57,3) & CA(27;2,216,3) & CA(33;2,819,3) \\ CA(28;2,24,4) & CA(40;2,116,4) & CA(52;2,560,4) & CA(64;2,2704,4) \\ CA(45;2,35,5) & CA(65;2,205,5) & CA(85;2,1200,5) & CA(105;2,7025,5) \\ CA(91;2,63,7) & CA(133;2,497,7) & CA(175;2,3920,7) & CA(217;2,30919,7) \end{array}.$$

Using tabu search, Nurmela [17] provides better results for some of the small cases, and explicitly presents a $CA(15; 2, 20, 3)$. We examined his solution, and found that three rows can be chosen so that 14 (and no more) columns form permutations. We can therefore rewrite his solution as a

$$\begin{array}{|cccccccccccccc|cccccc|}
\hline
0&0&0&0&1&1&1&1&1&1&1&1&1&1&0&0&0&1&1&1\\
0&0&1&1&0&1&2&2&2&2&2&2&2&2&0&0&0&2&2&2\\
0&2&2&2&0&1&0&0&0&1&2&0&1&1&2&2&2&0&2&1\\
1&1&1&0&1&0&1&1&0&0&2&2&2&1&2&2&2&2&1&0\\
1&2&0&2&1&0&0&2&2&1&0&1&0&2&0&1&2&2&2&1\\
1&2&2&1&2&0&2&1&1&0&1&0&2&0&1&0&2&1&2&1\\
2&0&0&1&2&2&2&2&0&2&2&1&0&1&2&2&2&1&0&2\\
2&1&2&0&2&2&1&2&2&0&1&0&1&2&0&2&1&2&2&0\\
2&1&1&2&0&2&0&1&1&2&1&2&0&2&2&0&1&1&0&1\\
2&1&1&2&2&1&2&0&2&1&0&1&2&0&2&1&0&1&0&0\\
1&2&2&1&1&2&1&0&1&2&0&2&1&0&1&2&0&2&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
\hline
2&2&1&0&1&0&2&2&1&2&2&0&1&0&1&1&1&0&1&2\\
1&0&2&2&0&1&1&1&2&0&0&1&0&1&1&1&1&0&1&2\\
0&1&0&1&2&2&0&0&0&1&1&2&2&2&1&1&1&0&1&2\\
\hline
\end{array}$$

Figure 5: A $PCA(15; 2, (14, 6), 3)$

$PCA(15; 2, (14, 6), 3)$, as shown in Figure 5. One can verify that none of the entries can be changed to a “don’t care” position $\star$ and obtain a covering array.

Theorem 3.2 using an $SCA(9; 2, (3, 1), 3)$ then gives a $CA(21; 2, 74, 3)$, and indeed Theorem 3.3 gives a $PCA(21; 2, (60, 14), 3)$. This provides the sequence

$$CA(15; 2, 20, 3) \quad CA(21; 2, 74, 3) \quad CA(27; 2, 282, 3) \quad CA(33; 2, 1002, 3) \ .$$

Instead since the $PCA(15; 2, (14, 6), 3)$ can be written as an SCA by suitable permutations of the symbols in each column, by Theorem 3.2 we can produce an $SCA(27; 2, (196, 168), 3)$ and hence a $CA(27; 2, 364, 3)$. The PCA in Figure 5 has the property that the upper left contains within the last three rows eleven columns that are permutations, and hence by Theorem 3.3 we can rewrite the solution as an $SCA(27; 2, (220, 144), 3)$. Applying Theorem 3.3 with an $SCA(9; 2, (3, 1), 3)$ then yields an $SCA(33; 2, (1092, 220), 3)$ and hence a $CA(33; 2, 1312, 3)$. In summary, we have found:

$$CA(15; 2, 20, 3) \quad CA(21; 2, 74, 3) \quad CA(27; 2, 364, 3) \quad CA(33; 2, 1312, 3) \ .$$

By contrast, using in his words “a few months of CPU time”, Nurmela [17] produces by tabu search

$$CA(15; 2, 20, 3) \quad CA(21; 2, 60, 3) \quad CA(27; 2, 191, 3) \quad CA(33; 2, 462, 3) \ .$$

Evidently heuristic search is a valuable tool. However, these few results demonstrate that employing the $CA(15; 2, 20, 3)$ (found by tabu search) in recursions outperforms the direct application of tabu search for larger orders. We explore this further next. In Figure 6, we present PCAs found by the simulated annealing method from [6]; the sizes of the covering arrays match those found (also by simulated annealing) in [22].

These improvements are not restricted to $q = 3$. We provide some useful small ingredients for $q = 4$ in Figure 7. These match bounds from simulated annealing from [22], and reported in [17]

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

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

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

Figure 6: $PCA(11; 2, (4, 1), 3)$, $PCA(12; 2, (4, 3), 3)$, $PCA(13; 2, (6, 3), 3)$

as the best current bounds. In this presentation, however, they are shown as PCAs, and indeed have "don't care" positions. We leave to the interested reader relevant application of the product constructions.

4 A direct construction

We develop a direct construction along the lines of Theorem 3.2, that leads to a different improvement. Sherwood [20] earlier gave a direct construction, upon which the one presented improves.

Let $\mathsf{A} = \{a_{i,j}\}$ be a $SCA(N; 2, (k_1, k_2), v)$ where $j = 1, \cdots, (k_1 + k_2)$, and $i = 1, \cdots, N$. Denote by $\mathsf{B} = \{b_{i,j}\}$ the $(N - v) \times k_1$ array where $b_{i,j} = a_{i,j}$ for $j = 1, \cdots, k_1$, and $i = 1, \cdots, N - v$. Denote by $\mathsf{C} = \{c_{i,j}\}$ the $(N - v) \times k_2$ array where $c_{i,j-k_1} = a_{i,j}$ for $j = k_1 + 1, \cdots, (k_1 + k_2)$, and $i = 1, \cdots, N - v$. Denote by b_j the jth column of B and by c_j the jth column of C. Let v be the v-tuple $(1, 2, \cdots, v)^T$ and I be the v-tuple $(1, 1, \cdots, 1)^T$.

Further, define D to be a binary code having M codewords each with length r; every two codewords (columns) have a position where the first one is 0 and the second one is 1 and a position where the first one is 1 and the second one is 0. Denote such a code by $\mathsf{D}(r, M)$.

First construct a $r \times k_1^r$ array E_r whose columns are all different r-tuples of the columns $\{b_1, b_2, \cdots, b_{k_1}\}$. Next, suppose that the jth column of $\mathsf{D}(r, M)$ has x_j zeroes and $y_j = r - x_j$ ones. Construct an array $\mathbf{L}_j^r$ with ${k_1}^{x_j}{k_2}^{y_j}$ columns and r rows. Suppose the positions where the jth column of $\mathsf{D}(r, M)$ has the symbol 0 are $\{z_1, z_2 \cdots z_{x_j}\}$. Then in the rows $\{z_1, z_2 \cdots z_{x_j}\}$ of $\mathbf{L}_j^r$ the columns are all possible x_j-tuples of the columns $\{b_1, b_2, \cdots, b_{k_1}\}$. Each of these x_j-tuples is repeated ${k_2}^{y_j}$ times. For each repetition, in the remaining rows of $\mathbf{L}_j^r$ the columns are all possible y_j-tuples of columns $\{c_1, c_2, \cdots, c_{k_2}\}$. Thus the set of ${k_2}^{y_j}$ y_j-tuples is repeated ${k_1}^{x_j}$ times in total.

The array $\mathbf{F}_r$ is constructed as follows:

$\mathbf{E}_r$	$\mathbf{L}_1^r$	$\mathbf{L}_2^r$	$\cdots$	$\mathbf{L}_M^r$
v	I	I	$\cdots$	I

0	2	2	2	3	2
0	0	0	3	1	1
3	0	2	1	2	2
0	3	1	1	0	3
1	1	2	3	0	3
2	0	3	2	3	3
2	1	1	2	1	2
1	3	0	0	3	2
3	1	0	1	3	1
2	0	2	0	0	1
1	3	1	2	2	1
1	2	3	1	1	1
3	3	3	3	0	2
3	2	1	0	1	3
2	2	0	3	2	3
3	2	0	2	0	0
0	1	3	0	2	0
2	3	2	1	1	0
1	0	1	3	3	0

2	3	2	3	3	0	3
3	0	0	3	0	0	2
1	2	1	0	3	2	2
0	3	0	1	1	2	0
1	3	3	0	2	0	0
3	3	3	1	3	3	2
3	2	3	3	1	1	0
2	1	3	1	0	2	3
3	1	2	0	1	1	2
0	2	0	0	0	3	3
2	1	1	2	1	0	0
3	0	2	2	2	2	3
0	0	3	2	3	1	0
1	0	1	1	1	1	3
0	1	1	3	2	3	2
1	$\star$	2	2	0	3	0
2	2	0	2	2	1	2
1	1	0	3	3	2	1
0	2	2	1	2	0	1
2	0	3	0	1	3	1
3	3	1	2	0	1	1

2	0	1	1	0	0	1	1
1	2	2	1	3	3	0	1
2	3	0	0	2	1	2	1
2	1	1	2	1	3	3	2
3	2	1	0	1	1	0	3
3	0	2	2	0	1	2	2
0	3	0	1	1	2	0	2
2	1	3	3	0	2	0	1
1	1	2	0	2	2	1	3
3	3	3	2	1	3	1	1
0	0	3	2	2	0	0	0
0	1	3	1	3	1	2	3
1	3	1	3	3	1	1	0
2	3	2	3	1	0	2	3
0	2	1	3	2	3	1	2
3	0	0	3	3	2	3	3
1	2	0	2	0	3	1	3
0	3	2	0	0	1	3	1
1	2	3	0	3	0	3	2
2	2	1	2	3	2	2	0
1	0	2	0	1	3	2	0
3	1	0	1	2	0	3	0
$\star$	$\star$	$\star$	$\star$	0	$\star$	$\star$	0

Figure 7: $PCA(19; 2, (5,1), 4)$, $PCA(21; 2, (6,1), 4)$, $PCA(23; 2, (6,2), 4)$

Theorem 4.1 *For any $r \geq 0$, $\mathbf{F}_r$ is a $CA(r(N-v)+v; 2, k, v)$ where $k = {k_1}^r + ({k_1}^{x_1}{k_2}^{y_1} + {k_1}^{x_2}{k_2}^{y_2} + \cdots + {k_1}^{x_M}{k_2}^{y_M})$.*

Proof. $\mathbf{F}_r$ consists of $M+1$ blocks. The first block, with array $\mathbf{E}_r$ in the first r rows, is denoted by $\mathbf{F}_r^0$. The ith block is denoted by $\mathbf{F}_r^i$ for $i = 1, 2, \cdots, M$. Consider any two columns f^1 and f^2 of $\mathbf{F}_r$.

1. $f^1, f^2 \in \mathbf{F}_r^0$. All pairs are covered since two different columns of B appear in the same row of $\mathbf{E}_r$. Together with the vector V they cover all pairs.

2. $f^1, f^2 \in \mathbf{F}_r^i$ for $i = 1, 2, \cdots, M$. Suppose that, for two columns of $\mathbf{L}_j^r$, there exists a row (c_h, c_j) where $h \neq j$. Then all pairs are covered except possibly the pair $(1,1)$. This pair is also covered, since both f^1 and f^2 end with vector I. Otherwise, it follows from the construction of $\mathbf{L}_j^r$ that f^1, f^2 have at least one row (b_h, b_j) where $h \neq j$ and another row (c_t, c_t) for some t. Hence all pairs are covered.

3. $f^1 \in \mathbf{F}_r^0$ and $f^2 \in \mathbf{F}_r^i$ where $1 \leq i \leq M$. All pairs are covered since there exist two rows that contain

$$\begin{pmatrix} b_h & c_j \\ \mathrm{V} & I \end{pmatrix}$$

for some $1 \leq h \leq k_1$ and $1 \leq j \leq k_2$.

4. $f^1 \in \mathbf{F}_r^j$ and $f^2 \in \mathbf{F}_r^i$ where $1 \leq i, j \leq M$ and $i \neq j$. All pairs are covered since there exist two rows that contain

$$\left(\begin{array}{cc} b_h & c_j \\ c_t & b_s \end{array} \right)$$

for some $1 \leq h, s \leq k_1$ and $1 \leq t, s \leq k_2$. This is assured from the structure of $\mathsf{D}(r, M)$. All pairs are covered but $(1,1)$ in these rows. The pair $(1,1)$ is also covered as f^1 and f^2 have the vector I in the last v positions.

■

This construction generalizes Theorem 3.2. Using a $SCA(q^2; 2, (q, 1), q)$ and a $D(r, r)$, where codewords have weight one, from Theorem 4.1 we obtain covering array $CA((r+1)q^2 - rq; 2, q^{r+1} + (r+1)q^r, q)$, but this construction is direct. Further, taking a different $\mathsf{D}(r, M)$ the result can be improved for some parameters. For example, taking $\mathsf{D}(r, \frac{r(r-1)}{2})$ where the codewords have weight two, we get a $CA((r+1)q^2 - rq; 2, q^{r+1} + \frac{(r+1)r}{2} q^{r-1}, q)$. In general, we have:

Theorem 4.2 *For $r \geq s \geq 1$ and q a prime power, an $SCA((r+1)q^2 - rq; 2, (q^{r+1}, \binom{r+1}{s} q^{r+1-s}), q)$ exists.*

A direct analog of Theorem 3.3 can also be written, but does not appear to form any simplification of the recursive method provided.

5 A Construction for SCAs

In order to exploit the power of Theorems 3.2 and 3.3 more fully, further ingredients are needed. As a first step, we adapt an approach of Meagher and Stevens [14]. Choose two parameters, ℓ and g. We form a vector $(v_0, \ldots, v_{\ell-1})$ with entries from $\mathbb{Z}_{g-1} \cup \{\infty\}$. The set $D_s = \{(v_j - v_i) \pmod{g-1} : j - i \equiv s \pmod{\ell}, v_i \neq \infty, v_j \neq \infty\}$ consists of the s-apart differences. Consider vectors in which $v_0 = \infty$ and $v_i \in \mathbb{Z}_{g-1}$ for $1 \leq i < \ell$. When $D_s = \mathbb{Z}_{g-1}$ for $1 \leq s < \ell$, such a vector is a *(g, ℓ)-cover starter*. When $\mathbb{Z}_{g-1} \setminus \{0\} \subseteq D_s$ for each $1 \leq s < \ell$, and $\{v_1, \ldots, v_{\ell-1}\} = \mathbb{Z}_{g-1}$, such a vector is a *$(g, \ell)$-distinct cover starter*.

When a (g, ℓ)-cover starter exists, Meagher and Stevens [14] note that a $CA(\ell(g-1)+1; 2, \ell, g)$ exists, as follows. Form all ℓ cyclic shifts of the cover starter. For each, form $g-1$ vectors by developing each modulo $g-1$ (keeping ∞ fixed). The resulting $\ell(g-1)$ vectors form the rows of an array, so that for any two columns all pairs are covered except for (∞, ∞). Adding one constant row consisting only of ∞ completes the covering array. Meagher and Stevens [14] give numerous examples of cover starters, proving

Lemma 5.1 *A (g, ℓ)-cover starter exists when*

1. *$g = 3$ and $\ell \in \{5, 8\}$;*

2. $g = 4$ *and* $\ell \in \{5, 6, 7, 8, 9, 10\}$*;*

3. $g = 5$ *and* $\ell \in \{7, 8, 9, 10, 11, 12\}$*;*

4. $g = 6$ *and* $\ell \in \{9, 10, 11, 12, 13, 14\}$*;*

5. $g = 7$ *and* $\ell \in \{10, 11, 12, 13, 14, 15, 16\}$*;*

6. $g = 8$ *and* $\ell \in \{9, 11, 12, 13, 14, 15, 16, 17, 18\}$*;*

7. $g = 9$ *and* $\ell \in \{13, 14, 15, 16, 17, 18, 19, 20\}$*.*

We instead employ distinct cover starters. Start with a (g, ℓ)-distinct cover starter. Treat this as a row, and form the ℓ cyclic shifts of this, obtaining an $\ell \times \ell$ array. Then add a new column with all entries equal to 0. Develop this $\ell \times (\ell + 1)$ array modulo $g - 1$ to form an $\ell(g - 1) \times (\ell + 1)$ array. Adding the g constant rows, we obtain a $SCA(\ell(g - 1) + g; 2, (\ell, 1), g)$, and hence a $CA((\ell + 1)(g - 1) + 1; 2, \ell + 1, g)$. For example, a (5,6)-distinct cover starter yields a $CA(29; 2, 7, 5)$.

However, distinct cover starters have many more effective applications, via Theorem 3.2. For purposes of illustration, consider the case when $g = 5$. Here are some distinct cover starters:

(5,5)	(∞,0,1,3,2)	(5,6)	(∞,0,0,1,3,2)
(5,7)	(∞,0,0,0,1,3,2)	(5,8)	(∞,0,0,0,0,1,3,2)
(5,9)	(∞,0,0,0,0,0,1,3,2)	(5,10)	(∞,0,0,0,0,0,0,1,3,2)

Each $(5, \ell)$-distinct cover starter gives an $SCA(4\ell + 5; 2, (\ell, 1), 5)$. Apply Theorem 3.2 to obtain $SCA(4(\ell_1 + \ell_2) + 5; 2, (\ell_1\ell_2, \ell_1 + \ell_2), 5)$ for $5 \leq \ell_1, \ell_2 \leq 10$. Choosing ℓ_1 and ℓ_2 as nearly equal as possible, we obtain the following covering arrays: $CA(45; 2, 35, 5)$, $CA(49; 2, 41, 5)$, $CA(53; 2, 48, 5)$, $CA(57; 2, 55, 5)$, $CA(61; 2, 63, 5)$, $CA(65; 2, 71, 5)$, $CA(69; 2, 80, 5)$, and so on. Comparing with the bounds in [23], these produce improvements whenever the number of factors exceeds 30. Further, since an SCA is produced as an intermediate step, the construction can be applied recursively. Indeed, as we have seen before, with 65 tests we can treat 200 factors with five values each, obtaining a substantial improvement by applying Theorem 3.2 twice rather than once. With Theorem 3.3 we improve this to 205 factors.

Next we give further examples of distinct cover starters. When $g = 3$ and $\ell \geq 3$, setting $v_0 = \infty$, $v_{\ell-1} = 1$ and $v_i = 0$ for $1 \leq i < \ell - 1$ gives a $(3, \ell)$-distinct cover starter. When $g = 4$ and $\ell \geq 6$, setting $v_0 = \infty$, $v_{\ell-3} = v_{\ell-1} = 1$, $v_{\ell-2} = 2$, and $v_i = 0$ for $1 \leq i < \ell - 3$ gives a $(4, \ell)$-distinct cover starter. Other examples follow:

(6,8)	(∞,0,1,3,0,2,1,4)	(6,9)	(∞,0,0,1,0,0,3,2,4)
(6,10)	(∞,0,0,0,1,1,4,3,0,2)	(6,11)	(∞,0,0,0,0,1,0,4,2,3,0)
(6,12)	(∞,0,0,0,0,0,1,0,4,2,3,0)	(7,7)	(∞,0,2,1,4,5,3)
(7,9)	(∞,0,0,2,1,4,5,3,3)	(7,10)	(∞,0,0,0,1,0,3,5,4,2)
(7,11)	(∞,0,0,0,0,1,0,3,5,4,2)	(7,12)	(∞,0,0,0,0,0,1,0,3,5,4,2)
(7,13)	(∞,0,0,0,0,0,0,1,0,3,5,4,2)	(8,11)	(∞,0,1,0,0,0,3,5,4,2,6)
(8,12)	(∞,0,0,0,1,0,6,4,6,3,2,5)	(8,13)	(∞,0,0,0,0,1,2,6,5,3,6,2,4)
(8,14)	(∞,0,0,0,0,0,1,0,4,6,4,3,2,5)	(9,12)	(∞,0,1,6,4,5,0,7,6,2,1,3)
(9,13)	(∞,0,0,0,0,2,6,1,7,6,3,4,5)	(9,14)	(∞,0,0,0,0,0,2,6,1,7,6,3,4,5)
(9,15)	(∞,0,0,0,0,0,0,2,6,1,7,6,3,4,5)		

In general, using the log function sequence [16] we establish:

Lemma 5.2 *For $p \geq 3$ a prime, there exists a (p,p) distinct cover starter.*

Proof. Let ω be a primitive element of $\mathbb{Z}_p$. Define $a_0 = \infty$ and $a_{\omega^j} = j$ for $0 \leq j < p-1$. Then $(a_0, a_1, \ldots, a_{p-1})$ is a (p,p) distinct cover starter. ■

We do not explore the applications of these in detail here. Rather they illustrate that the direct product construction can avail itself of ingredients far beyond the $SCA(q^2; 2, (q,1), q)$s arising from the finite field.

6 Mixed Covering Arrays

We now combine the ideas of Theorem 3.1 and Theorem 3.2 to obtain an improved product construction for mixed level covering arrays. In Theorem 3.2, the repeated use of a single array B is used to ensure that certain pairs are covered in the product C. In order to generalize, our task is to specify the relationship required among the ingredients $\{B_i\}$. We first define an $SMCA(N; 2, (k_1, k_2), (v_1, \ldots, v_{k_1+k_2}))$ to be an $N \times (k_1 + k_2)$ array in which the ith column has entries from a v_i-set (which we take to be $\{1, \ldots, v_i\}$), and for which selecting any two columns, we find all pairs covered except possibly

- those of the form (x,x) if both columns are among the first k_1;
- those of the form $(x,1)$ if the first column is among the first k_1, the second among the latter k_2; and
- (1,1) if both columns are among the latter k_2.

Theorem 6.1 *Suppose that there exist*

1. *an $SMCA(N; 2, (k_1, k_2), (v_1, \ldots, v_{k_1+k_2}))$, A, with profile $(d_1, \ldots, d_{k_1+k_2})$;*
2. *for each $1 \leq i \leq k_1$, an $SMCA(M_i; 2, (\ell_{i1}, \ell_{i2}), (w_{i1}, \ldots, w_{1,\ell_{i1}+\ell_{i2}}))$, B_i, for which $w_{ij} \leq v_i$ for $1 \leq j \leq \ell_{i1} + \ell_{i2}$; and*
3. *for each $k_1 < i \leq k_1 + k_2$, an $SMCA(M_i; 2, (\ell_{i1}, 0), (w_{i1}, \ldots, w_{1,\ell_{i1}}))$, B_i, for which $w_{ij} \leq v_i$ for $1 \leq j \leq \ell_{i1}$.*

Write $T = N + max_{i=1}^{k_1+k_2}(M_i - d_i)$. Suppose further that

1. *For $1 \leq i_1 < i_2 \leq k_1$, for each $\ell_{i_1,1} < j_1 \leq \ell_{i_1,1} + \ell_{i_1,2}$, and each $\ell_{i_2,1} < j_2 \leq \ell_{i_2,1} + \ell_{i_2,2}$, the $(T-N) \times 2$ matrix whose first column contains the entries in the first $T-N$ rows of column j_1 of B_{i_1}, and whose second column contains the entries in the first $T-N$ rows of column j_2 of B_{i_2}, contains every pair of the form (x,x) with $1 \leq x \leq min(w_{i_1,j_1}, w_{i_2,j_2})$; and*
2. *For $1 \leq i_1 \leq k_1$ and $k_1 < i_2 \leq k_1+k_2$, for each $\ell_{i_1,1} < j_1 \leq \ell_{i_1,1}+\ell_{i_1,2}$, and each $1 \leq j_2 \leq \ell_{i_2,1}$, the $(T-N) \times 2$ matrix whose first column contains the entries in the first $T-N$ rows of column j_1 of B_{i_1}, and whose second column contains the entries in the first $T-N$ rows of column j_2 of B_{i_2}, contains every pair of the form $(x,1)$, for $1 \leq x \leq w_{i_1,j_1}$.*

Then there exists an

$$SMCA(T;2,(\sum_{i=1}^{k_1}\ell_{i1},\sum_{i=k_1+1}^{k_1+k_2}\ell_{i1}+\sum_{i=1}^{k_1}\ell_{i2}),$$

$$(w_{11},\dots,w_{1,\ell_{11}},\cdots,w_{k_1,1},\dots,w_{k_1,\ell_{k_1,1}},$$

$$\cdots,w_{k_1+1,1},\dots,w_{k_1+1,\ell_{k_1+1,1}},\cdots,w_{k_1+k_2,1},\dots,w_{k_1+k_2,\ell_{k_1+k_2,1}},$$

$$\cdots,w_{1,\ell_{11}+1},\dots,w_{1,\ell_{12}},\cdots,w_{k_1,\ell_{k_1,1}+1},\dots,w_{k_1,\ell_{k_1,2}})).$$

Proof. The construction parallels that of Theorem 3.1. When placing the rows of B_i, the first $T-N$ rows of B_i are always placed on the last $T-N$ rows, in the same order, in C. Columns are then permuted so that those indexed by (i,j) with $1 \le i \le k_1$ and $1 \le j \le \ell_{i1}$ are placed in the first $\sum_{i=1}^{k_1}\ell_{i1}$ positions.

The verification is similar. ■

Such an SMCA can be completed to an MCA by the addition of at most m rows, where $m = \max\{w_{ij} : 1 \le i \le k_1, 1 \le j \le \ell_{i1}\}$. This is done by placing, in the ith additional row, the symbol "1" in columns indexed by (i,j) with $i > k_1$ or $j > \ell_{i,1}$. In the remaining columns, we place the symbol i if $i \le w_{ij}$, and $\star$ otherwise.

We expect the most effective applications of Theorems 3.3 and 6.1 to arise when heuristic search techniques are tailored to find suitable ingredients; this appears to be a challenging problem.

Acknowledgments

Research of the first two authors (CJC,SSM) is supported by the Consortium for Embedded and Inter-Networking Technologies. Research of the fourth author (DES) is supported by NSF grants IIS-9988636, 0115586, and MCB-0209754.

References

[1] M.A. Chateauneuf, C.J. Colbourn, and D.L. Kreher, Covering arrays of strength three, *Designs, Codes and Cryptography* 16 (1999), 235-242.

[2] M. A. Chateauneuf and D. L. Kreher. On the state of strength-three covering arrays. *Journal of Combinatorial Designs*, 10(4):217–238, 2002

[3] 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 Transactions on Software Engineering*, 23(7):437–44, 1997.

[4] D. M. Cohen and M. L. Fredman. New techniques for designing qualitatively independent systems. *Journal of Combinatorial Designs*, 6(6):411–16, 1998.

[5] M. B. Cohen, C. J. Colbourn, J.S. Collofello, P. B. Gibbons and W. B. Mugridge. Variable Strength Interaction Testing of Components. *Proc. 27th Intl. Computer Software and Applications Conference (COMPSAC 2003)*, Dallas TX, November 2003, pp. 413–418.

[6] M. B. Cohen, C. J. Colbourn, P. B. Gibbons and W. B. Mugridge. Constructing test suites for interaction testing. In *Proc. Intl. Conf. on Software Engineering (ICSE 2003)*, Portland, Oregon, May 2003, pp 38-49.

[7] M. B. Cohen, C. J. Colbourn, and A. C. H. Ling, Augmented simulated annealing to build interaction test suites, *Proc. IEEE Int Symp Software Reliability Eng (ISSRE 2003)*, Denver CO, November 2003, pp. 394-405.

[8] C.J. Colbourn. Combinatorial Aspects of Covering Arrays. *Le Matematiche (Catania)*, to appear.

[9] C. J. Colbourn and J. H. Dinitz (editors), *The CRC Handbook of Combinatorial Designs*, CRC Press, Boca Raton, 1996.

[10] S. R. Dalal, A. J. N. Karunanithi, J. M. Leaton, G. C. Patton, and B. M. Horowitz. Model-based testing in practice. In *Proc. Intl. Conf. on Software Engineering,(ICSE '99)*, 1999, pp. 285-94, New York.

[11] I. S. Dunietz, W. K. Ehrlich, B. D. Szablak, C. L. Mallows, and A. Iannino. Applying design of experiments to software testing. In *Proc. Intl. Conf. on Software Engineering, (ICSE '97)*, 1997, pp. 205-215, New York.

[12] A. S. Hedayat, N. J. A. Sloane, and J. Stufken. *Orthogonal Arrays.* Springer-Verlag, New York, 1999.

[13] S. Martirosyan and Tran Van Trung. On t-covering arrays. *Designs, Codes and Cryptography* 32 (2004), 323–339..

[14] K. Meagher and B. Stevens. Group construction of covering arrays. *Journal of Combinatorial Designs*, to appear.

[15] L. Moura, J. Stardom, B. Stevens, and A. Williams. Covering arrays with mixed alphabet sizes. *Journal of Combinatorial Designs*, 11(6):113-132, 2003.

[16] G.L. Mullen and D. White. A polynomial representation for logarithms in GF(q). *Acta Arithmetica* 47 (1986), 255-261.

[17] K. Nurmela. Upper bounds for covering arrays by tabu search. *Discrete Applied Math.*, 138 (2004), 143-152.

[18] S. Poljak and Z. Tuza, On the maximum number of qualitatively independent partitions, *Journal of Combinatorial Theory (A)* 51 (1989), 111-116.

[19] D.E. Shasha, A.Y. Kouranov, L.V. Lejay, M.F. Chou, and G.M. Coruzzi, Using combinatorial design to study regulation by multiple input signals. A tool for parsimony in the post-genomics era, *Plant Physiology* 127(2001), 1590-1594.

[20] G. Sherwood. On the Construction of Orthogonal Arrays and Covering Arrays Using Permutation Groups. Online at `http://home.att.net/ gsherwood/cover.htm`

[21] N. J. A. Sloane. Covering arrays and intersecting codes. *Journal of Combinatorial Designs*, 1(1):51–63, 1993.

[22] B. Stevens, *Transversal Covers and Packings*, Ph.D. Thesis, Mathematics, University of Toronto, 1998.

[23] B. Stevens and E. Mendelsohn. New recursive methods for transversal covers. *Journal of Combinatorial Designs*, 7(3):185–203, 1999.

[24] B. Stevens, L. Moura, and E. Mendelsohn. Lower bounds for transversal covers. *Designs Codes and Cryptography*, 15(3):279–299, 1998.

[25] K. C. Tai and L. Yu. A test generation strategy for pairwise testing. *IEEE Transactions on Software Engineering*, 28(1):109-111, 2002.

[26] Y.-W. Tung and W. S. Aldiwan. Automating test case generation for the new generation mission software system. In *Proc. IEEE Aerospace Conf.*, 2000, pp. 431-437.

[27] A. W. Williams and R. L. Probert. A practical strategy for testing pair-wise coverage of network interfaces. In *Proc. Seventh Intl. Symp. on Software Reliability Engineering*, 1996, pp. 246-54.

[28] A. W. Williams and R. L. Probert. A measure for component interaction test coverage. In *Proc. ACS/IEEE Intl. Conf. on Computer Systems and Applications*, 2001, pp. 301-311.