Group Construction of Covering Arrays — Part 2

Karen Meagher
Department of Mathematics and Statistics
University of Ottawa
Ottawa ON K1N 6N5
kmeagher@site.uottawa.ca

Abstract

A covering array t-$CA(n, k, g)$ is a $k \times n$ array on a set of g symbols with the property that in each $t \times n$ subarray, every $t \times 1$ column appears at least once. This paper improves many of the best known upper bounds on n for covering arrays, t-$CA(n, k, g)$ with $g + 1 \leq k \leq 2g$, for $g = 11 \ldots 18$ by an algebraic construction developed by Chateauneuf, Colbourn and Kreher [1] and extended by Meagher and Stevens [6]. This construction uses a vector to build a circulant matrix and then a group action on the matrix to produce a covering array. In [6] the vectors are found by an exhaustive search, here they are found by a heurestic search.

Keywords
Covering array, Orthogonal array, Software testing, Transversal design.

1 Introduction

In [6] a construction for covering arrays is given. This construction is an application of the method developed by Chateauneuf, Colbourn and Kreher [1]. It uses a vector to build a circulant matrix and then a covering array is constructed by a group action on this matrix. The construction in [6] relies on an exhaustive search to find the vector. In this report we use the same construction, but the exhaustive search is replaced with a heurestic seach to find covering arrays for larger values of g. The goal of this report is not only to record the new bounds but also to show that the construction in [6] is effective for larger covering arrays.

A *covering array* with strength t and alphabet size g is an $k \times n$ array with entries from $\{0, 1, \ldots g - 1\}$ and the property that in each $t \times n$ subarray, every t-set of $\{0, 1, \ldots, g-1\}$ appears as a column at least once. A covering array with such parameters will be denoted as t-$CA(n, k, g)$. The construction used here was use in [1] to build covering arrays with strength 3. Here we only consider strength 2 covering arrays.

The are many applications for covering arrays; most notably software testing [2, 3, 4]. For most applications t, k and g are given and it is desirable to minimize n. The smallest possible value of n for a covering array with fixed parameters t, k and g will be denoted as

$$t\text{-}CAN(k, g) = \min_{l \in \mathbb{N}} \{l : \text{ there exists a } t\text{-}CA(l, k, g)\}.$$

In Section 2 we give a brief summary of known bounds for strength 2 covering arrays and we describe the construction used in [6]. In Section 3 we give a brief description of the heurestic search. The new bounds for $2 - CAN(k, g)$ with $g + 1 \leq k \leq 2g$ where $11 \leq g \leq 18$ are given in Table 1.

2 Background Results

In this section we state some well-known results. All of these results are also given in [6].

It is easy to see that deleting a row from a covering array produces a covering array with fewer rows, so

$$t\text{-}CAN(k-1, g) \leq t\text{-}CAN(k, g). \tag{1}$$

By Inequality 1 many of the bounds given in this report also give bounds for covering arrays with fewer rows. These bounds are not explicitly given here.

Standard finite field constructions for orthogonal arrays give

Theorem 1 *[5] For g a prime power, 2-$CAN(g+1, g) = g^2$.*

The block size recursive construction from [7] gives a method to construct larger covering arrays from smaller ones. Applying the block size recursive construction to the covering arrays from Theorem 1 gives the following lemma.

Lemma 1 *[7] For g a prime power and $k \leq 2(g+1)$,*

$$2\text{-}CAN(k, g) \leq 2g^2 - g.$$

The construction given in [6] builds, for many values of $g \leq 10$, covering arrays 2-$CA(k(g-1)+1, k, g)$ with $g+1 \leq k \leq 2g+1$.

Consider the action of the subgroup $G = \langle (1\ 2\ \dots\ g-1) \rangle < Sym_g$ on the pairs from $\mathbb{Z}_g$. There are $g+2$ orbits generated by this group action:

1. $(0, 0)$,
2. $\{(0, i) | i = 1, \dots, g-1\}$,
3. $\{(i, 0) | i = 1, \dots, g-1\}$,
4. $\{(a, b) | a, b \in \mathbb{Z}_g \backslash 0$ and $a - b \equiv j \bmod (g-1)\}$ for $j = 0, \dots g-1$.

For a vector $v = (0, v_1, v_2, \dots, v_{k-1})$ from $\mathbb{Z}_g^k$ consider the sets of pairs

$$d_i = \{(v_j, v_{j+i}) |\ j = 1 \dots k-1\} \quad i = 1 \dots k-1, \text{ and } i + j \neq 0$$

(where the subscripts are taken modulo k). If a vector $v \in \mathbb{Z}_g^k$ has the property that the set of differences $v_j - v_{j+i}$ for all $(v_j, v_{j+i}) \in d_i$ covers $\mathbb{Z}_g$ over $i = 1 \dots k-1$, then we say the v is a *starter vector*. This means for a starter vector the sets d_i for all $i = 1 \dots k$ contain at least one element from each of the orbits $\{(a, b) | a, b \in \mathbb{Z}_g \backslash 0$ and $a - b \equiv j \bmod (g-1)\}$ for $j = 0, \dots g-1$.

Set M to be the $k \times k$ circulant matrix generated from a starter vector v. Then, for any pair of rows in M, at least one element from each of the orbits of G (except $(0, 0)$) occurs in some column. For each $x \in G$ let M_x be the matrix formed by the action of x on the elements of M. Since any two rows in M have a representative from each orbit of G, for any two rows all pairs from $\mathbb{Z}_g$ (except $(0, 0)$) will occur in M_x for some $x \in G$. Finally let C be the $k \times 1$ all zero matrix. Then the array formed by concatenating C, M_x for all $x \in G$ is a covering array.

Example 1 *Let*

$$G = \{e, (12)\} < Sym_3 \text{ and } v = (01112) \in \mathbb{Z}_3^5.$$

The following circulant matrix generated by v is

$$M = \begin{pmatrix} 0 & 2 & 1 & 1 & 1 \\ 1 & 0 & 2 & 1 & 1 \\ 1 & 1 & 0 & 2 & 1 \\ 1 & 1 & 1 & 0 & 2 \\ 2 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

The elements of G acting on M produce the following two matrices

$$M_e = \begin{pmatrix} 0 & 2 & 1 & 1 & 1 \\ 1 & 0 & 2 & 1 & 1 \\ 1 & 1 & 0 & 2 & 1 \\ 1 & 1 & 1 & 0 & 2 \\ 2 & 1 & 1 & 1 & 0 \end{pmatrix} \quad \text{and} \quad M_{(12)} = \begin{pmatrix} 0 & 1 & 2 & 2 & 2 \\ 2 & 0 & 1 & 2 & 2 \\ 2 & 2 & 0 & 1 & 2 \\ 2 & 2 & 2 & 0 & 1 \\ 1 & 2 & 2 & 2 & 0 \end{pmatrix}.$$

Concatenating the matrices C, M_e and $M_{(12)}$ gives the following covering array with parameters $CA(11, 5, 3)$

$$\left(\begin{array}{c|ccccc|ccccc} 0 & 0 & 2 & 1 & 1 & 1 & 0 & 1 & 2 & 2 & 2 \\ 0 & 1 & 0 & 2 & 1 & 1 & 2 & 0 & 1 & 2 & 2 \\ 0 & 1 & 1 & 0 & 2 & 1 & 2 & 2 & 0 & 1 & 2 \\ 0 & 1 & 1 & 1 & 0 & 2 & 2 & 2 & 2 & 0 & 1 \\ 0 & 2 & 1 & 1 & 1 & 0 & 1 & 2 & 2 & 2 & 0 \end{array}\right).$$

3 New Results

In [6] the starter vectors v are found by an exhaustive search. For $g \geq 10$ an exhaustive search is impractical. Replacing the exhaustive search for the vector v by a heurestic search produced results for covering arrays with larger g.

The algorithm used was a simple hill-climbing algorithm. It starts with an arbitrary vector and counts the number of the orbits of G that occur in all the sets d_i. Then a single entry in v, randomly choosen, is changed to a random value. If the change increases the number of orbits in the sets d_i the change is accepted otherwise it is rejected. This is continued until either a starter vector is found or a set number of changes have been tried. The code is available at: http://www.site.uottawa.ca/ kmeagher/code.html.

This very simple hill climbing algorithm found many new bounds for covering arrays 2-$CA(n, k, g)$ with $11 \leq g \leq 18$ and $g+1 \leq k \leq 2g$. These are given in Table 1 with the starter vector that produces the covering array. By Equation 1 many of the new upper bounds on 2-$CAN(k, g)$ given in Table 1 give new bounds for smaller covering arrays as well. Many of the new bounds are substaintially smaller than the previously known bounds, but more importantly this gives an efficient way to build covering arrays for larger alphabet sizes.

Table 1: New upper bounds and the corresponding starter vectors.

	starter vector	new bound	old bound
$g = 11$			
$k = 17$	0 4 7 7 4 2 1 3 4 9 6 3 4 3 9 2 6	171	221
$g = 12$			
$k = 18$	0 6 5 10 7 3 9 1 3 2 3 7 6 6 3 1 4 8	199	255
$k = 19$	0 8 3 3 6 7 10 9 7 10 5 9 5 7 4 2 10 9 3	210	276
$g = 13$			
$k = 21$	0 6 9 5 7 6 2 8 8 9 6 2 3 10 5 3 5 9 8 1 8	253	325
$k = 22$	0 3 9 11 4 1 8 7 3 7 1 3 3 4 1 1 4 2 6 11 6 4	265	325
$k = 23$	0 6 7 10 2 5 4 11 8 2 4 1 9 3 5 3 3 1 10 10 11 4 6	277	325
$k = 24$	0 7 3 7 7 3 10 2 8 3 5 4 2 4 5 8 9 2 11 7 8 7 3 10	289	325
$k = 25$	0 1 2 4 10 11 5 1 10 2 7 6 4 3 6 8 6 7 10 3 10 1 2 8 8	301	325
$k = 26$	0 7 4 9 2 2 3 7 10 6 6 10 10 7 2 7 9 4 5 4 2 4 11 10 4 4	313	325
$k = 27$	0 10 1 4 5 1 5 7 1 8 9 10 2 3 3 10 8 1 3 1 1 10 5 10 9 9 1	325	325
$g = 14$			
$k = 23$	0 1 5 4 1 9 4 6 1 1 6 1 5 10 7 8 8 1 4 5 1 12 6	300	437
$k = 24$	0 5 4 12 9 2 7 10 10 10 2 1 6 2 8 12 9 7 8 8 1 8 10 7	313	437
$k = 25$	0 8 8 2 12 1 12 10 6 1 6 6 10 4 7 4 3 6 12 1 3 4 9 11 7	326	437
$k = 26$	0 7 2 9 10 7 6 3 11 1 10 2 6 8 3 5 10 5 5 6 12 5 3 2 6 5	339	437
$k = 27$	0 8 4 5 10 12 4 8 3 11 4 3 8 9 2 1 5 12 5 8 7 5 2 2 3 3 12	352	437
$k = 28$	0 5 3 11 6 5 5 10 3 10 6 1 4 11 9 4 12 6 8 1 8 1 10 1 11 12 4 4	365	437
$k = 29$	0 2 4 1 1 2 5 10 3 1 2 9 8 1 1 1 1 9 3 11 10 9 10 1 1 4 12 12 8	378	437
$g = 15$			
$k = 27$	0 11 8 9 11 1 7 12 4 4 11 9 4 2 6 12 10 6 7 6 6 5 1 8 11 4 12	379	450
$k = 28$	0 12 7 1 1 5 13 11 12 8 3 4 5 10 11 2 13 1 4 12 4 3 3 10 4 3 4 3	393	450
$k = 29$	0 2 13 8 12 8 1 13 1 8 11 3 8 2 9 8 13 5 7 7 13 10 11 6 11 6 5 1 12	407	450
$k = 30$	0 7 1 10 11 8 10 2 4 13 3 10 7 12 9 13 8 11 11 11 10 7 1 11 7 12 2 5 3 10	421	450
$k = 31$	0 9 8 1 7 1 5 2 5 3 1 4 3 12 12 1 2 6 1 2 1 13 1 6 1 11 7 7 10 3 10	435	450
$g = 16$			
$k = 29$	0 5 2 6 2 5 11 4 13 1 3 14 9 6 4 8 9 10 12 3 8 3 14 10 9 9 2 9 6	436	496
$k = 30$	0 11 8 9 13 7 3 8 2 10 4 14 6 3 2 12 4 8 6 4 6 13 3 6 6 12 12 4 2 1	451	496
$k = 31$	0 1 14 1 5 12 9 4 2 14 6 11 6 1 12 6 12 7 6 6 9 8 5 2 10 1 3 14 14 10 11	466	496
$k = 32$	0 7 7 3 7 5 1 7 12 3 5 10 11 11 8 10 1 13 6 2 10 11 5 12 5 8 7 13 14 11 9 4	481	496
$k = 33$	0 5 1 5 4 1 8 13 10 14 2 3 9 3 5 1 4 10 5 7 8 2 14 5 14 10 11 1 14 11 8 1 1	496	496
$g = 17$			
$k = 33$	0 13 4 10 9 5 8 9 4 14 8 13 2 11 6 14 13 10 14 2 4 9 13 5 5 5 7 10 4 7 13 11 4	529	561
$k = 34$	0 8 4 5 7 10 1 7 6 4 14 6 13 1 5 10 10 11 8 5 2 15 3 6 2 13 8 11 8 3 12 2 8 12	545	561
$g = 18$			
$k = 35$	0 2 13 12 8 4 5 12 9 12 4 8 8 15 10 10 16 5 5 4 6 13 15 13 1 11 8 13 14 5 13 10 7 10 8	596	666
$k = 36$	0 3 16 12 4 5 4 1 15 15 16 11 13 6 9 16 3 2 10 8 14 1 14 8 13 15 15 14 10 4 1 10 2 6 9 4	613	666
$k = 37$	0 6 14 3 5 8 12 15 4 15 5 2 11 7 2 3 2 7 15 5 12 12 10 8 4 1 5 5 7 7 10 16 10 16 9 2 4	630	666

References

[1] M. A. Chateauneuf, C. J. Colbourn, and D. L. Kreher. Covering arrays of strength three. 16(3):235–242, 1999.

[2] D. M. Cohen, S. R. Dalal, M. L. Fredman, and G. C. Patton. The AETG system: An approach to testing based on combinatorial design. 23:437–444, 1997.

[3] D. M. Cohen, S. R. Dalal, J. Parelius, and G. C. Patton. The combinatorial design approach to automatic test generation. *IEEE Software*, 13(5):83–88, Sep. 1996.

[4] S. R. Dalal and C. L. Mallows. Factor-covering designs for testing software. *Technometrics*, 40(3):234–243, 1998.

[5] A. S. Hedayat, N. J. A. Sloane, and J. Stufken. *Orthogonal arrays*. Springer Series in Statistics. Springer-Verlag, New York, 1999.

[6] K. Meagher and B. Stevens. Group Construction of Covering Arrays. to appear *Journal of Combinatorial Designs*.

[7] B. Stevens. *Transversal Covers and Packings*. PhD thesis, University of Toronto, Toronto, 1998.