J. Algebra Comb. Discrete Appl.
3(3) • 127–134

Received: 31 October 2015
Accepted: 16 February 2016

Journal of Algebra Combinatorics Discrete Structures and Applications

Locating one pairwise interaction: Three recursive constructions*

Research Article

Charles J. Colbourn, Bingli Fan

Abstract: In a complex component-based system, choices (levels) for components (factors) may interact to cause faults in the system behaviour. When faults may be caused by interactions among few factors at specific levels, covering arrays provide a combinatorial test suite for discovering the presence of faults. While well studied, covering arrays do not enable one to determine the specific levels of factors causing the faults; locating arrays ensure that the results from test suite execution suffice to determine the precise levels and factors causing faults, when the number of such causes is small. Constructions for locating arrays are at present limited to heuristic computational methods and quite specific direct constructions. In this paper three recursive constructions are developed for locating arrays to locate one pairwise interaction causing a fault.

2010 MSC: 05B30, 05A18, 05D99, 62K05, 68P10

Keywords: Locating array, Covering array, Detecting array

1. Introduction

Although covering arrays have been explored as a method to reveal the presence of faults caused by interactions among components in a complex system [4, 8], they are inadequate to determine which interaction(s) account for the faulty behaviour. Colbourn and McClary [7] extend covering arrays to provide sufficient information to identify all faults when few faults each involving few factors are present. To set the stage, there are k factors $F_1, \dots, F_k$. Each factor F_i has a set $S_i = \{v_{i1}, \dots, v_{is_i}\}$ of s_i possible values (*levels*). A *test* is an assignment, for each i with $1 \le i \le k$, of a level from $v_{i1}, \dots, v_{is_i}$ to F_i. A test, when executed, can *pass* or *fail*. For any subset $\{i_1, \dots, i_t\} \subseteq \{1, \dots, k\}$ and levels $\sigma_{i_j} \in S_{i_j}$, the set $\{(i_j, \sigma_{i_j}) : 1 \le j \le t\}$ is a *t-way interaction*, or an interaction of *strength t*. Thus a test on k factors

* *The first author's research was supported in part by the National Science Foundation under Grant No. 1421058. The second author's research was supported by the China Scholarship Council.*

Charles J. Colbourn (Corresponding Author); School of CIDSE, Arizona State University, Tempe AZ 85287-8809, U.S.A. (email: colbourn@asu.edu).
Bingli Fan; Department of Mathematics, Beijing Jiaotong University, Beijing, China (email: blfan@bjtu.edu.cn).

http://dx.doi.org/10.13069/jacodesmath.17551

ISSN 2148-838X

contains (*covers*) $\binom{k}{t}$ interactions of strength t. A *test suite* is a collection of tests; the *outcomes* are the corresponding set of pass/fail results. A fault is evidenced by a failure outcome for a test; however the fault is rarely due to a complete k-way interaction; rather it is the result of one or more faulty interactions of strength smaller than k covered in the test. Tests are executed concurrently, so that testing is *nonadaptive*.

We employ a matrix representation. An array A with N rows, k columns, and symbols in the ith column chosen from an alphabet S_i of size s_i is denoted as an $N \times k$ array of type $(s_1, \ldots, s_k)$. A *t-way interaction* in A is a choice of t columns $i_1, \ldots, i_t$, and the selection of a level $\sigma_{i_j} \in S_{i_j}$ for $1 \leq j \leq t$, represented as $T = \{(i_j, \sigma_{i_j}) : 1 \leq j \leq t\}$. For such an array $A = (a_{xy})$ and interaction T, define $\rho_A(T) = \{r : a_{ri_j} = \sigma_{i_j}, 1 \leq j \leq t\}$, the set of rows of A in which the interaction is covered. For a set of interactions $\mathcal{T}$, $\rho_A(\mathcal{T}) = \bigcup_{T \in \mathcal{T}} \rho_A(T)$.

Let $\mathcal{I}_t$ be the set of *all* t-way interactions for an array of type $(s_1, \ldots, s_k)$, and let $\overline{\mathcal{I}_t}$ be the set of all interactions of strength *at most* t. Consider an interaction $T \in \overline{\mathcal{I}_t}$ of strength less than t. Any interaction T' of strength t that contains T necessarily has $\rho_A(T') \subseteq \rho_A(T)$; a subset $\mathcal{T}'$ of interactions in $\mathcal{I}_t$ is *independent* if there do not exist $T, T' \in \mathcal{T}'$ with $T \subset T'$. Some interactions may cause faults. To formulate arrays for testing, we limit both the number of interactions causing faults and their strengths.

As in [7], this leads to a variety of types of array A for testing a system with N tests and k factors having $(s_1, \ldots, s_k)$ as the numbers of levels:

Array	Definition
Covering Arrays:	
$\mathrm{MCA}(N; t, k, (s_1, \ldots, s_k))$	$\rho_A(T) \neq \emptyset$ for all $T \in \overline{\mathcal{I}_t}$
$\mathrm{CA}(N; t, k, v)$	$\rho_A(T) \neq \emptyset$ for all $T \in \overline{\mathcal{I}_t}$ and $v = s_1 = \cdots = s_k$
Locating Arrays:	
(d, t)-$\mathrm{LA}(N; t, k, (s_1, \ldots, s_k))$	$\rho_A(\mathcal{T}_1) = \rho_A(\mathcal{T}_2) \Leftrightarrow \mathcal{T}_1 = \mathcal{T}_2$ whenever $\mathcal{T}_1, \mathcal{T}_2 \subseteq \mathcal{I}_t$, $\lvert\mathcal{T}_1\rvert = d$, and $\lvert\mathcal{T}_2\rvert = d$
$(\overline{d}, t)$-$\mathrm{LA}(N; t, k, (s_1, \ldots, s_k))$	$\rho_A(\mathcal{T}_1) = \rho_A(\mathcal{T}_2) \Leftrightarrow \mathcal{T}_1 = \mathcal{T}_2$ whenever $\mathcal{T}_1, \mathcal{T}_2 \subseteq \mathcal{I}_t$, $\lvert\mathcal{T}_1\rvert \leq d$, and $\lvert\mathcal{T}_2\rvert \leq d$
$(d, \overline{t})$-$\mathrm{LA}(N; t, k, (s_1, \ldots, s_k))$	$\rho_A(\mathcal{T}_1) = \rho_A(\mathcal{T}_2) \Leftrightarrow \mathcal{T}_1 = \mathcal{T}_2$ whenever $\mathcal{T}_1, \mathcal{T}_2 \subseteq \overline{\mathcal{I}_t}$, $\lvert\mathcal{T}_1\rvert = d$, $\lvert\mathcal{T}_2\rvert = d$, and $\mathcal{T}_1$ and $\mathcal{T}_2$ are independent
$(\overline{d}, \overline{t})$-$\mathrm{LA}(N; t, k, (s_1, \ldots, s_k))$	$\rho_A(\mathcal{T}_1) = \rho_A(\mathcal{T}_2) \Leftrightarrow \mathcal{T}_1 = \mathcal{T}_2$ whenever $\mathcal{T}_1, \mathcal{T}_2 \subseteq \overline{\mathcal{I}_t}$, $\lvert\mathcal{T}_1\rvert \leq d$, $\lvert\mathcal{T}_2\rvert \leq d$, and $\mathcal{T}_1$ and $\mathcal{T}_2$ are independent

When all factors have the same number of levels v, the notation replaces v for $(s_1, \ldots, s_k)$.

Colbourn and McClary [7] also examine detecting arrays, which permit faster recovery than do locating arrays but in general require more tests. Here we focus on locating arrays. Although locating arrays have been successfully applied in applications to measurement and testing [1], few constructions are known. Martínez *et al.* [9] develop adaptive analogues and establish feasibility conditions for a locating array to exist. In [11] and [12] the minimum number of rows in a locating array is determined when the number of factors is quite small. Few direct, and no recursive, constructions are known. Indeed at present unless the number of factors is small, the observation in [7] that covering arrays of higher strength provide examples of locating arrays serves as the main device for their construction.

In this paper, we explore a different avenue, extending recursive constructions from covering arrays to locating arrays. We extend a covering array construction pioneered by Roux [10], extended by Chateauneuf and Kreher [2], and further generalized in [3]. The basic strategy in each of these constructions is a "cut–and–paste" approach using covering arrays with fewer factors and the same or smaller strengths. Each operates by repeating subarrays; because we want different interactions to appear in different sets of rows, such recursions for locating arrays necessitates more ingredients than for covering arrays.

We focus on constructions for $(1, \overline{2})$-locating arrays in which all factors have the same number v of levels. In other words, we treat the case of locating one interaction of strength at most two.

We require one further class of ingredients. Let $(\Gamma, \odot)$ be a group of order k. A $(k, n; \lambda)$-*difference matrix over* Γ is an $n \times k\lambda$ matrix $D = (d_{ij})$ with entries from Γ, so that for each $1 \leq i < j \leq n$, the multiset

$$\{d_{i\ell} \odot d_{j\ell}^{-1} : 1 \leq \ell \leq k\lambda\}$$

(the *difference list*) contains every element of Γ λ times.

2. A doubling construcion

We develop a doubling construction that is reminscent of one for covering arrays of strength three in [2] generalizing that in [10].

Theorem 2.1. *If there exist a $(1, \overline{2})$-LA$(N; 2, k, v)$ and a $(1, \overline{1})$-LA$(M; 2, k, v)$ in which the set of differences modulo v between entries in two distinct columns contains all symbols, then a $(1, \overline{2})$-LA$(N + (v-1)M; 2, 2k, v)$ exists.*

Proof. Let $A = (a_{ij})$ be a $(1, \overline{2})$-LA$(N; 2, k, v)$ on symbols $\{0, \ldots v-1\}$ with columns indexed by $\{1, \ldots, k\}$. Let $B = (b_{ij})$ be a $(1, \overline{1})$-LA$(M; 2, k, v)$ on symbols $\{0, \ldots v-1\}$ with columns indexed by $\{1, \ldots, k\}$ with the additional property that for every $1 \leq c < c' \leq k$ and every $1 \leq d < v$, there exists a ρ for which $b_{\rho c} - b_{\rho c'} \equiv d \pmod{v}$.

We form an $(N + (v-1)M) \times 2k$ array C with columns indexed by $\{1, \ldots, k\} \times \{0, 1\}$ by juxtaposing v arrays $C_0, \ldots, C_{v-1}$. C_0 is obtained by setting the entry in position $(\rho, (c, \alpha))$ to $a_{\rho c}$. For $1 \leq \ell < v$, the entry of C_ℓ in position $(\rho, (c, 0))$ is $b_{\rho c}$ and the entry in position $(\rho, (c, 1))$ is $b_{\rho c} + \ell \bmod v$.

Let $T = \{((c_1, \alpha_1), \sigma_1), ((c_2, \alpha_2), \sigma_2)\}$ and $T' = \{((c_1', \alpha_1'), \sigma_1'), ((c_2', \alpha_2'), \sigma_2')\}$ be interactions of C with $\rho_C(T) = \rho_C(T')$.

Because A is a $(1, \overline{2})$-locating array and $\rho_A(\{(c_1, \sigma_1), (c_2, \sigma_2)\}) = \rho_A(\{(c_1', \sigma_1'), (c_2', \sigma_2')\})$, there are two cases to treat.

$c_1 = c_2$, $c_1' = c_2'$, $\sigma_1 \neq \sigma_2$, and $\sigma_1' \neq \sigma_2'$: Then

$$T = \{((c_1, 0), \sigma_1), ((c_1, 1), \sigma_2)\} \text{ and } T' = \{((c_1', 0), \sigma_1'), ((c_1', 1), \sigma_2')\}.$$

Now $\rho(T)$ contains at least one row index from $C_{\sigma_2 - \sigma_1 \bmod v}$, and $\rho(T')$ contains at least one row index from $C_{\sigma_2' - \sigma_1' \bmod v}$. These agree only when $\sigma_1 = \sigma_1'$, $\sigma_2 = \sigma_2'$, and $c_1 = c_1'$ because B is a $(1, \overline{1})$-locating array. But then $T = T'$.

$c_1 = c_1'$, $c_2 = c_2'$, $\sigma_1 = \sigma_1'$, and $\sigma_2 = \sigma_2'$: Then

$$T = \{((c_1, \alpha_1), \sigma_1), ((c_2, \alpha_2), \sigma_2)\} \text{ and } T' = \{((c_1, \alpha_1'), \sigma_1), ((c_2, \alpha_2'), \sigma_2)\}.$$

We treat subcases.

$c_1 = c_2$: For T and T' to be interactions, either $\sigma_1 = \sigma_2$, or both $\alpha_1 \neq \alpha_2$ and $\alpha_1' \neq \alpha_2'$.

First suppose that $\sigma_1 = \sigma_2$. For $1 \leq x < v$,

$$\rho_B(\{(c_1, \sigma_1 - x\alpha_1), (c_1, \sigma_1 - x\alpha_2)\}) \;=\; \rho_B(\{(c_1, \sigma_1 - x\alpha_1'), (c_1, \sigma_1 - x\alpha_2')\})$$

Now if $\alpha_1 = \alpha_2$, then $\rho_B(\{(c_1, \sigma_1 - x\alpha_1)\}) \neq \emptyset$, but this is not equal to $\rho_B(\{(c_1, \sigma_1 - x\alpha_1'), (c_1, \sigma_1 - x\alpha_2')\})$ unless $\alpha_1' = \alpha_2' = \alpha_1$, in which case $T = T'$. Similarly when if $\alpha_1' = \alpha_2'$, $T = T'$. But when $\alpha_1 \neq \alpha_2$ and $\alpha_1' \neq \alpha_2'$, again $T = T'$.

So suppose that $\sigma_1 \neq \sigma_2$. Without loss of generality, $(\alpha_1, \alpha_2) = (0, 1)$. For $1 \leq \ell < v$,

$$\rho_{C_\ell}(T) = \begin{cases} \rho_B(\{(c_1, \sigma_1)\}) & \text{if } \ell \equiv \sigma_2 - \sigma_1 \pmod{v} \\ \emptyset & \text{otherwise} \end{cases}$$

If $(\alpha'_1, \alpha'_2) = (0, 1)$, then $T = T'$. So suppose that $(\alpha'_1, \alpha'_2) = (1, 0)$. Then $\rho_{C_{\sigma_1 - \sigma_2}}(T') = \rho_B(\{(c_1, \sigma_2)\})$, and hence $\sigma_1 = \sigma_2$, which cannot be.

$c'_1 = c'_2$: This is symmetric to the previous case.

$c_1 \neq c_2$ and $c'_1 \neq c'_2$: If $(\alpha_1, \alpha_2) = (\alpha'_1, \alpha'_2)$, then $T = T'$.
First suppose that $(\alpha_1, \alpha_2) = (0, 0) \neq (\alpha'_1, \alpha'_2)$. If (σ_1, σ_2) appears in columns (c_1, c_2) of B,

$$\rho_{C_1}(\{((c_1, 0), \sigma_1), ((c_2, 0), \sigma_2)\}) \neq \rho_{C_1}(\{((c_1, \alpha'_1), \sigma_1 + \alpha'_1), ((c_2.\alpha'_2), \sigma_2 + \alpha'_2)\}),$$

but then $\rho_C(T) \neq \rho_C(T')$. If (σ_1, σ_2) does not appear in columns (c_1, c_2) of B, choose x so that $(\sigma_1 - x, \sigma_2 - x)$ does appear; choose y and z so that (σ_1, y) and (z, σ_2) appear. Then

$$\begin{aligned} \rho_{C_x}(\{((c_1, 0), \sigma_1), ((c_2, 0), \sigma_2)\}) &\neq \rho_{C_x}(\{((c_1, 1), \sigma_1), ((c_2, 1), \sigma_2)\}) \\ \rho_{C_{\sigma_2 - y}}(\{((c_1, 0), \sigma_1), ((c_2, 0), \sigma_2)\}) &\neq \rho_{C_{\sigma_2 - y}}(\{((c_1, 0), \sigma_1), ((c_2, 1), \sigma_2)\}) \\ \rho_{C_{\sigma_1 - z}}(\{((c_1, 0), \sigma_1), ((c_2, 0), \sigma_2)\}) &\neq \rho_{C_{\sigma_1 - z}}(\{((c_1, 1), \sigma_1), ((c_2, 0), \sigma_2)\}) \end{aligned}$$

But then $\rho_C(T) \neq \rho_C(T')$. Hence $(\alpha_1, \alpha_2) \neq (0, 0)$ and, in the same way, $(\alpha'_1, \alpha'_2) \neq (0, 0)$.
Next suppose that $(\alpha_1, \alpha_2) = (1, 1) \neq (\alpha'_1, \alpha'_2) \neq (0, 0)$. Choose $y \neq \sigma_2$ and $z \neq \sigma_1$ so that (σ_1, y) and (z, σ_2) appear in columns (c_1, c_2) of B. Then

$$\begin{aligned} \rho_{C_{\sigma_2 - y}}(\{((c_1, 1), \sigma_1), ((c_2, 1), \sigma_2)\}) &\neq \rho_{C_{\sigma_2 - y}}(\{((c_1, 0), \sigma_1), ((c_2, 1), \sigma_2)\}) \\ \rho_{C_{\sigma_1 - z}}(\{((c_1, 1), \sigma_1), ((c_2, 1), \sigma_2)\}) &\neq \rho_{C_{\sigma_1 - z}}(\{((c_1, 1), \sigma_1), ((c_2, 0), \sigma_2)\}) \end{aligned}$$

But then $\rho_C(T) \neq \rho_C(T')$.
Finally suppose without loss of generality that $(\alpha_1, \alpha_2) = (1, 0)$ and $(\alpha'_1, \alpha'_2) = (0, 1)$. Choose a pair (z, σ_2) that appears in columns (c_1, c_2) of B Then

$$\rho_{C_{\sigma_1 - z}}(\{((c_1, 1), \sigma_1), ((c_2, 0), \sigma_2)\}) \neq \rho_{C_{\sigma_1 - z}}(\{((c_1, 0), z), ((c_2, 1), \sigma_2 + \sigma_1 - z)\})$$

But then $\rho_C(T) \neq \rho_C(T')$.

Hence C is a $(1, \overline{2})$-LA$(N + (v-1)M; 2, 2k, v)$. □

3. A product construction permuting symbols

Theorem 2.1 permutes symbols in one ingredient in order to double the number of factors. In the next construction, we also permute the symbols, but we require further ingredients in order to multiply the number of factors by v.

Theorem 3.1. *If a $(1, \overline{2})$-LA$(N; 2, k, v)$, a $(1, \overline{2})$-LA$(R; 2, v, v)$, and a CA$(M; 2, k, v)$ all exist, then a $(1, \overline{2})$-LA$(N + M + R; 2, kv, v)$ exists.*

Proof. Let $V = \{0, \ldots, v-1\}$. Let

$A = (a_{ij})$ be a $(1, \overline{2})$-LA$(N; 2, k, v)$ on symbols V.

$B = (b_{ij})$ be a CA$(M; 2, k, v)$ on symbols V.

$C = (c_{ij})$ be a $(1, \overline{2})$-LA$(R; 2, v, v)$ on symbols V with columns indexed by V.

D be the $N \times kv$ array with columns indexed by $\{1, \dots, k\} \times \{0, \dots, v-1\}$ by placing $a_{\rho,\gamma}$ in entry $(\rho, (\gamma, s))$ whenever $1 \le \rho \le N$, $1 \le \gamma \le k$, and $0 \le s < v$.

E be the $M \times kv$ array with columns indexed by $\{1, \dots, k\} \times \{0, \dots, v-1\}$ by placing $(b_{\rho,\gamma} + s) \bmod v$ in entry $(\rho, (\gamma, s))$ whenever $1 \le \rho \le M$, $1 \le c \le k$, and $0 \le s < v$.

F be the $R \times kv$ array with columns indexed by $\{1, \dots, k\} \times \{0, \dots, v-1\}$ by placing $(c_{\rho,s} + s) \bmod v$ in entry $(\rho, (\gamma, s))$ whenever $1 \le \rho \le R$, $1 \le \gamma \le k$, and $0 \le s < v$.

L be the $(N + M + R) \times kv$ array obtained by vertically juxtaposing D, E, and F.

We show that L is a $(1, \overline{2})$-locating array. Let $T = \{((c_1, \alpha_1), \sigma_1), ((c_2, \alpha_2), \sigma_2)\}$ and $T' = \{((c'_1, \alpha'_1), \sigma'_1), ((c'_2, \alpha'_2), \sigma'_2)\}$ be interactions of L with $\rho_L(T) = \rho_L(T')$. It follows that

$$
\begin{aligned}
\rho_A(\{(c_1, \sigma_1), (c_2, \sigma_2)\}) &= \rho_A(\{(c'_1, \sigma'_1), (c'_2, \sigma'_2)\}) \\
\rho_B(\{(c_1, \sigma_1 - \alpha_1), (c_2, \sigma_2 - \alpha_2)\}) &= \rho_B(\{(c'_1, \sigma'_1 - \alpha'_1), (c'_2, \sigma'_2 - \alpha'_2)\}) \\
\rho_C(\{(\alpha_1, \sigma_1 - \alpha_1), (\alpha_2, \sigma_2 - \alpha_2)\}) &= \rho_C(\{(\alpha'_1, \sigma'_1 - \alpha'_1), (\alpha'_2, \sigma'_2 - \alpha'_2)\}),
\end{aligned}
$$

with entries of B and C computed modulo v.

Because A is a $(1, \overline{2})$-locating array and $\rho_A(\{(c_1, \sigma_1), (c_2, \sigma_2)\}) = \rho_A(\{(c'_1, \sigma'_1), (c'_2, \sigma'_2)\})$, there are two cases to treat.

$c_1 = c_2$, $c'_1 = c'_2$, $\sigma_1 \neq \sigma_2$, and $\sigma'_1 \neq \sigma'_2$: Hence

$$\rho_C(\{(\alpha_1, \sigma_1 - \alpha_1), (\alpha_2, \sigma_2 - \alpha_2)\}) = \rho_C(\{(\alpha'_1, \sigma'_1 - \alpha'_1), (\alpha'_2, \sigma'_2 - \alpha'_2)\}),$$

Two subcases arise:

$\alpha_1 = \alpha_2$, $\alpha'_1 = \alpha'_2$, $\sigma_1 - \alpha_1 \neq \sigma_2 - \alpha_2$, and $\sigma'_1 - \alpha'_1 \neq \sigma_2 - \alpha'_2$: This cannot arise because then $T = \{((c_1, \alpha_1), \sigma_1), ((c_1, \alpha_1), \sigma_2)\}$ is not a 2-way interaction.

$\alpha_1 = \alpha'_1$, $\alpha_2 = \alpha'_2$, $\sigma_1 - \alpha_1 = \sigma'_1 - \alpha'_1$, and $\sigma_2 - \alpha_2 = \sigma'_2 - \alpha'_2$: Then $\sigma_1 = \sigma'_1$ and $\sigma_2 = \sigma'_2$ and

$$\rho_B(\{(c_1, \sigma_1 - \alpha_1), (c_1, \sigma_2 - \alpha_2)\}) = \rho_B(\{(c'_1, \sigma_1 - \alpha_1), (c'_1, \sigma_2 - \alpha_2)\}).$$

But then $c_1 = c'_1$ and $T = T'$.

$c_1 = c'_1$, $c_2 = c'_2$, $\sigma_1 = \sigma'_1$, and $\sigma_2 = \sigma'_2$: Hence

$$\rho_B(\{(c_1, \sigma_1 - \alpha_1), (c_2, \sigma_2 - \alpha_2)\}) = \rho_B(\{(c_1, \sigma_1 - \alpha'_1), (c_2, \sigma_2 - \alpha'_2)\}),$$

When $c_1 \neq c_2$, the rows of B are partitioned into v^2 nonempty sets by examining the ordered pair of symbols appearing, because B is a covering array of strength 2. Therefore when $c_1 \neq c_2$, $\alpha_1 = \alpha'_1$ and $\alpha_2 = \alpha'_2$, and hence $T = T'$.

It remains to treat the case when $c_1 = c_2$. If $\alpha_1 = \alpha_2$ or $\alpha'_1 = \alpha'_2$ and both T and T' are interactions, we have $T = T'$. So $\alpha_1 \neq \alpha_2$, $\alpha'_1 \neq \alpha'_2$, and

$$\rho_C(\{(\alpha_1, \sigma_1 - \alpha_1), (\alpha_2, \sigma_2 - \alpha_2)\}) = \rho_C(\{(\alpha'_1, \sigma_1 - \alpha'_1), (\alpha'_2, \sigma_2 - \alpha'_2)\}).$$

Then because C is a $(1, \overline{2})$-locating array, without loss of generality $\alpha_1 = \alpha'_1$, $\alpha_2 = \alpha'_2$, and hence $T = T'$.

Consequently L is a $(1, \overline{2})$-locating array. $\square$

4. A product construction permuting columns

In the next construction, we combine the "cut–and–paste" approach with ideas from another main type of recursive construction, the so-called column replacement methods (see [4], for example). To do this, we permute columns in some of the ingredients, using a difference matrix to determine the column permutations.

Theorem 4.1. *If a $(1,\overline{2})$-LA$(N;2,k,v)$ exists and $k \equiv 0,1,3 \pmod 4$, a $(1,\overline{2})$-LA$(3N;2,k^2,v)$ exists.*

Proof. Because $k \equiv 0,1,3 \pmod 4$ and $k \geq 3$, there is a $(k,3,1)$-difference matrix $D = (d_{ij})$ over a group Γ (see, for example, [13]). Suppose that Γ has elements $\{g_1, \cdots, g_k\}$ and that g_1 is the group identity. Assume without loss of generality that $d_{1j} = g_1$ and $d_{2j} = g_j$ for $1 \leq j \leq k$. Let $A = (a_{ij})$ be a $(1,\overline{2})$-LA$(N;2,k,v)$ on symbols $\{0,\dots v-1\}$ with columns indexed by Γ.

We form three arrays C_1, C_2, C_3 with columns indexed by $\Gamma \times \Gamma$. For $s \in \{1,2,3\}$, C_s has N rows, and the entry in row i and column (j,ℓ) is $a_{i,jd_{s\ell}^{-1}}$. C is the $3N \times k^2$ array obtained by vertical juxtaposition of C_1, C_2, and C_3.

Let $T = \{((c_1,\alpha_1),\sigma_1),((c_2,\alpha_2),\sigma_2)\}$ and $T' = \{((c'_1,\alpha'_1),\sigma'_1),((c'_2,\alpha'_2),\sigma'_2)\}$ be interactions of C with $\rho_C(T) = \rho_C(T')$, We permit $((c_1,\alpha_1),\sigma_1) = ((c_2,\alpha_2),\sigma_2)$ so T or T' may be 1-way interactions. However, we do not permit that $T = ((c_1,\alpha_1),\sigma_1),((c_1,\alpha_1),\sigma_2)$ but $\sigma_1 \neq \sigma_2$, for then T is not an interaction at all. We must show that $T = T'$. Because $\rho_C(T) = \rho_C(T')$ $\rho_{C_s}(T) = \rho_{C_s}(T')$ for each $1 \leq s \leq 3$.

Then for each $1 \leq s \leq 3$,

$$\rho_A(\{(c_1 d_{s\alpha_1}^{-1},\sigma_1),(c_2 d_{s\alpha_2}^{-1},\sigma_2)\}) = \rho_{C_s}(T) = \rho_{C_s}(T') = \rho_A(\{(c'_1 d_{s\alpha'_1}^{-1},\sigma'_1),(c'_2 d_{s\alpha'_2}^{-1},\sigma'_2)\})$$

Because A is a $(1,\overline{2})$-locating array, for each $1 \leq s \leq 3$,

$$\{(c_1 d_{s\alpha_1}^{-1},\sigma_1),(c_2 d_{s\alpha_2}^{-1},\sigma_2)\} = \{(c'_1 d_{s\alpha'_1}^{-1},\sigma'_1),(c'_2 d_{s\alpha'_2}^{-1},\sigma'_2)\}$$

unless $c_1 d_{s\alpha_1}^{-1} = c_2 d_{s\alpha_2}^{-1}$, $c'_1 d_{s\alpha'_1}^{-1} = c'_2 d_{s\alpha'_2}^{-1}$, $\sigma_1 \neq \sigma_2$, and $\sigma'_1 \neq \sigma'_2$.

Employing this equality for $s = 1$, without loss of generality two cases remain.

$c_1 = c_2$, $c'_1 = c'_2$, $\sigma_1 \neq \sigma_2$, and $\sigma'_1 \neq \sigma'_2$: Consider the equalities when $s \in \{2,3\}$. Now $c_1 d_{s\alpha_1}^{-1} = c_1 d_{s\alpha_2}^{-1}$ only when $\alpha_1 = \alpha_2$, but then T is an interaction only when $\sigma_1 = \sigma_2$, which cannot be. Similarly $c'_1 d_{s\alpha'_1}^{-1} = c'_1 d_{s\alpha'_2}^{-1}$ only when $\alpha'_1 = \alpha'_2$, but then T' is an interaction only when $\sigma'_1 = \sigma'_2$, which cannot be. So because A is a locating array,

$$\{(c_1 d_{s\alpha_1}^{-1},\sigma_1),(c_1 d_{s\alpha_2}^{-1},\sigma_2)\} = \{(c'_1 d_{s\alpha'_1}^{-1},\sigma'_1),(c'_1 d_{s\alpha'_2}^{-1},\sigma'_2)\}.$$

Without loss of generality, $\sigma_1 = \sigma'_1$ and $\sigma_2 = \sigma'_2$ so $c_1 d_{2\alpha_1}^{-1} = c'_1 d_{2\alpha'_1}^{-1}$ and $c_1 d_{2\alpha_2}^{-1} = c'_1 d_{2\alpha'_2}^{-1}$. Then

$$c_1^{-1} c'_1 = d_{2\alpha_1}^{-1} d_{2\alpha'_1} = d_{2\alpha_2}^{-1} d_{2\alpha'_2} = d_{3\alpha_1}^{-1} d_{3\alpha'_1} = d_{3\alpha_2}^{-1} d_{3\alpha'_2}.$$

Then $d_{3\alpha'_1} d_{2\alpha'_1}^{-1} = d_{3\alpha_1} d_{2\alpha_1}^{-1}$ and hence $\alpha_1 = \alpha'_1$ because D is a difference matrix. Similarly $d_{3\alpha_2} d_{2\alpha_2}^{-1} = d_{3\alpha'_2} d_{2\alpha'_2}^{-1}$ and hence $\alpha_2 = \alpha'_2$. But then $T = T'$.

$c_1 = c'_1$, $c_2 = c'_2$, $\sigma_1 = \sigma'_1$, and $\sigma_2 = \sigma'_2$: Then for each $s \in \{2,3\}$,

$$\{(c_1 d_{s\alpha_1}^{-1},\sigma_1),(c_2 d_{s\alpha_2}^{-1},\sigma_2)\} = \{(c_1 d_{s\alpha'_1}^{-1},\sigma_1),(c_2 d_{s\alpha'_2}^{-1},\sigma_2)\}$$

If $c_1 d_{s\alpha_1}^{-1} = c_1 d_{s\alpha'_1}^{-1}$, then $d_{s,\alpha_1}^{-1} = d_{s,\alpha'_1}^{-1}$ and hence $\alpha_1 = \alpha'_1$. But then $T = T'$.

Hence $\sigma_1 = \sigma_2$ and for $s \in \{2,3\}$, $c_1 d^{-1}_{s,\alpha_1} = c_2 d^{-1}_{s,\alpha'_2}$ and $c_1 d^{-1}_{s,\alpha'_1} = c_2 d^{-1}_{s,\alpha_2}$. Hence

$$c_1^{-1} c_2 = d_{2,\alpha'_2} d^{-1}_{2,\alpha_1} = d_{2,\alpha_2} d^{-1}_{2,\alpha'_1} = d_{3,\alpha'_2} d^{-1}_{3,\alpha_1} = d_{3,\alpha_2} d^{-1}_{3,\alpha'_1}$$

Then

$$d^{-1}_{2,\alpha_1} d_{3,\alpha_1} = d^{-1}_{2,\alpha'_2} d_{3,\alpha'_2} \text{ and } d^{-1}_{2,\alpha_2} d_{3,\alpha_2} = d^{-1}_{2,\alpha'_1} d_{3,\alpha'_1}.$$

Because D is a difference matrix, $\alpha_1 = \alpha'_2$ and $\alpha_2 = \alpha'_1$. Then $T = T'$.

Hence C is the required locating array. □

5. Concluding remarks

Theorems 2.1, 3.1, and 4.1 establish that cut–and–paste constructions provide viable methods for generating locating arrays. Although the repetition inherent in methods of this type initially result in many interactions appearing in the same sets of rows, at least in the case for one interaction of strength at most two, we have shown that the symmetry from the repetition can be interrupted by adjoining further ingredients. The methods here to make locating arrays of strength two are loosely patterned on recursive constructions for covering arrays of strength three. One might hope to obtain more powerful recursive constructions by adapting the product construction for covering arrays of strength two [5], but the methods we have used do not appear to be sufficient for this.

On the other hand, the theorems established here can be generalized to certain “mixed” locating arrays in which different factors have different numbers of levels. Although we have not pursued it here, we also expect that the methods can generalize to the location of more than one faulty interactions at the cost of further ingredients and more cases to verify. Finally further recursive constructions that exploit the methods developed for covering arrays in [4, 6] appear to be promising.

Acknowledgment: The authors thank Violet Syrotiuk for helpful discussions about this work, and thank Vladimir Tonchev for the opportunity to present it at the First Annual Kliekhandler Conference. This research was completed while the second author was visiting Arizona State University. He expresses his sincere thanks to the School of Computing, Informatics, and Decision Systems Engineering at Arizona State University for kind hospitality.

References

[1] A. N. Aldaco, C. J. Colbourn, V. R. Syrotiuk, Locating arrays: A new experimental design for screening complex engineered systems, SIGOPS Oper. Syst. Rev. 49(1) (2015) 31–40.
[2] M. Chateauneuf, D. L. Kreher, On the state of strength-three covering arrays, J. Combin. Des. 10(4) (2002) 217–238.
[3] M. B. Cohen, C. J. Colbourn, A. C. H. Ling, Constructing strength three covering arrays with augmented annealing, Discrete Math. 308(13) (2008) 2709–2722.
[4] C. J. Colbourn, Covering arrays and hash families, Information Security and Related Combinatorics, NATO Peace and Information Security, IOS Press (2011), 99–136.
[5] C. J. Colbourn, S. S. Martirosyan, G. L. Mullen, D. Shasha, G. B. Sherwood, J. L. Yucas, Products of mixed covering arrays of strength two, J. Combin. Des. 14(2) (2006) 124–138.
[6] C. J. Colbourn, S. S. Martirosyan, T. van Trung, R. A. Walker II, Roux-type constructions for covering arrays of strengths three and four, Des. Codes Cryptogr. 41(1) (2006) 33–57.

[7] C. J. Colbourn, D. W. McClary, Locating and detecting arrays for interaction faults, J. Comb. Optim. 15(1) (2008) 17–48.
[8] A. Hartman, Software and hardware testing using combinatorial covering suites, Graph Theory, Combinatorics and Algorithms, Springer, 2005, 237–266.
[9] C. Martínez, L. Moura, D. Panario, B. Stevens, Locating errors using ELAs, covering arrays, and adaptive testing algorithms, SIAM J. Discrete Math. 23(4) (2009/10) 1776–1799.
[10] G. Roux, k-Propriétés dans les tableaux de n colonnes: cas particulier de la k-surjectivité et de la k-permutivité, Ph.D. Dissertation, University of Paris 6, 1987.
[11] C. Shi, Y. Tang, J. Yin, Optimal locating arrays for at most two faults, Sci. China Math. 55(1) (2012) 197–206.
[12] Y. Tang, C. J. Colbourn, J. Yin, Optimality and constructions of locating arrays, J. Stat. Theory Pract. 6(1) (2012) 20–29.
[13] J. Yin, Cyclic difference packing and covering arrays, Des. Codes Cryptogr. 37(2) (2005) 281–292.