

Strength-three covering arrays from a coloured added column

Peter Seelman, Institute for Recursive Self Improvement (IRSI)

Draft, 13 October 2026

Abstract

A standard way to add one factor to a strength-three covering array built on an orthogonal array is to give the new column a constant value on the orthogonal array and to append one strength-two covering array for every other value of the new column. Colbourn, Kéri, Rivas Soriano and Schlage-Puchta observed that this should be improvable by arranging for the orthogonal array to cover more of the required triples itself. We carry this out for the Bush orthogonal array $OA(q^3; 3, m, q)$ of an arc in $PG(2, q)$. The added column takes a value determined by the line through a fixed point of an affine quotient plane. For $2 \leq v \leq q$ and $1 \leq c \leq \min(v, \lfloor (q+1)/2 \rfloor)$ we obtain

$$\text{CAN}(3, m+1, v) \leq q^3 + (c-1)q + (v-c) \text{CAN}(2, m, v),$$

which at $c = 1$ is the known bound. For odd q this gives $\text{CAN}(3, q+2, q) \leq q^3 + q(q^2 - 1)/2$. The construction improves the November 2024 covering array tables, their last public state, in twelve cells with $11 \leq k \leq 21$ and $8 \leq v \leq 19$, by between 2.5% and 13.9% in ten of them; for example $\text{CAN}(3, 15, 13) \leq 3289$ (previously 3798) and $\text{CAN}(3, 21, 18) \leq 9902$ (previously 11500). The arrays, the generator and an exact checker are supplied as ancillary files.

1 Introduction

A *covering array* $\text{CA}(N; t, k, v)$ is an $N \times k$ array over a v -set in which every $N \times t$ subarray contains each t -tuple at least once as a row. The *covering array number* $\text{CAN}(t, k, v)$ is the least N for which a $\text{CA}(N; t, k, v)$ exists. An orthogonal array $OA(v^t; t, k, v)$ of index one is a covering array in which every t -tuple occurs exactly once. Covering arrays are used to design test suites [4, 10]. Upper bounds on CAN for $t \leq 6$ were collected in tables maintained by Colbourn until 2025, when they were withdrawn from public access. Their November 2024 state, preserved by Grömping [9, 16], is the last public state and is the reference used here. A provenance-tracked successor table [17], which adds independently verified later arrays, changes none of the cells below (checked 13 October 2026).

Let q be a prime power. The Bush construction [1] gives $\text{CAN}(3, q+1, q) = q^3$, and $\text{CAN}(3, q+2, q) = q^3$ for even q by using a hyperoval [2, 14]. One column further, the usual recursive step is the “add a factor” theorem of Colbourn, Kéri, Rivas Soriano and Schlage-Puchta [7, Theorem 3.2], whose second inequality reads

$$\text{CAN}(s+1, n+1, q) \leq \text{CAN}(s+1, n, q) + (q-1)\text{CAN}(s, n, q) \quad (1)$$

([7, (9)], stated there for radius-covering arrays; radius 0 is the case used here). It is proved by giving the new column the constant value 0 on a $\text{CA}(s+1)$ and appending, for each other value i , a $\text{CA}(s)$ with the constant new column i . Immediately after the proof the authors write [7, p. 1160]:

“One should be able to improve Theorem 3.2 by choosing B in such a way that large parts of A are already covered, and can be removed. However, simply knowing the parameters of A and B does not ensure that they have any overlap at all.”

They then carry this out once, for Bush arrays, in [7, Lemma 3.3]. That lemma starts from the duplicated-column inequality [7, (8)] and removes q^{s-2} redundant rows, giving $\text{CAN}(3, q+2, q) \leq 2q^3 - q^2 - q$ at $s = 3$.

This note carries out the same idea from the other side. Instead of choosing the strength-two part so that it overlaps the orthogonal array, we give the new column a non-constant value on the orthogonal array, so that the orthogonal array itself covers about half of the values of the new column. The new column is read off from the geometry of the arc that defines the Bush array (Section 2). At $c = 1$ the construction is inequality (1) applied to a fused Bush array. Each further colour costs q rows and saves one strength-two array, so the saving over (1) is $(c-1)(\text{CAN}(2, m, v) - q)$, about $q^3/2$ at $v = q$.

2 The construction

Let $W = \text{GF}(q)^3$ with the standard bilinear form $\langle u, w \rangle = \sum_i u_i w_i$, and identify the points of $\text{PG}(2, q)$ with the one-dimensional subspaces of W . A set S of points is an *arc* if no three of them are collinear. The conic $\{(1, x, x^2) : x \in \text{GF}(q)\} \cup \{(0, 0, 1)\}$ is an arc of size $q+1$, and for even q adding the nucleus $(0, 1, 0)$ gives a hyperoval of size $q+2$ [11].

Bush array. For an arc $S = \{p_1, \dots, p_m\}$ (representative vectors) let $B(S)$ be the $q^3 \times m$ array with rows indexed by $w \in W$ and entry $\langle p_j, w \rangle$ in column j . For any three columns i, j, l the map $w \mapsto (\langle p_i, w \rangle, \langle p_j, w \rangle, \langle p_l, w \rangle)$ is a linear bijection, because p_i, p_j, p_l are linearly independent. Hence $B(S)$ is an $\text{OA}(q^3; 3, m, q)$.

Fusion. For $2 \leq v \leq q$ fix a surjection $\varphi : \text{GF}(q) \rightarrow \{0, \dots, v-1\}$ and let $B_v = \varphi(B(S))$ entrywise.

The coloured column. Choose $\kappa \in W \setminus \{0\}$ such that $\langle p_j, \kappa \rangle = 0$ for at most one j , that is, the line $\kappa^\perp$ of $\text{PG}(2, q)$ contains at most one point of S . Let $\pi : W \rightarrow W/\langle \kappa \rangle$ be the quotient map. The quotient is an affine plane $\text{AG}(2, q)$ with origin $O = \pi(\langle \kappa \rangle)$. Its $q+1$ lines through O are the images of the $q+1$ planes of W that contain κ . Fix $1 \leq c \leq \min(v, \lfloor (q+1)/2 \rfloor)$ and partition these $q+1$ lines into classes $C_0, \dots, C_{c-1}$, each of size at least two. This is possible exactly when $c \leq \lfloor (q+1)/2 \rfloor$. Define $g : W \rightarrow \{0, \dots, c-1\}$ by $g(w) = 0$ for $w \in \langle \kappa \rangle$, and otherwise $g(w) = y$ where the line through O and $\pi(w)$ lies in C_y .

The array. Let D be a $\text{CA}(N_2; 2, m, v)$. Let A be the array with $m+1$ columns obtained by stacking:

- (B1) the q^3 rows $(B_v(w) \mid g(w))$, $w \in W$;
- (B2) for each $y = 1, \dots, c-1$, the q rows $(B_v(w) \mid y)$, $w \in \langle \kappa \rangle$;
- (B3) for each $y = c, \dots, v-1$, the N_2 rows of D with the constant entry y appended.

Theorem 1. *Let q be a prime power, S an arc of size m in $\text{PG}(2, q)$, $2 \leq v \leq q$ and $1 \leq c \leq \min(v, \lfloor (q+1)/2 \rfloor)$. Then a vector κ as above exists, and A is a $\text{CA}(q^3 + (c-1)q + (v-c)N_2; 3, m+1, v)$. Consequently*

$$\text{CAN}(3, m+1, v) \leq q^3 + (c-1)q + (v-c) \text{CAN}(2, m, v).$$

Proof. Existence of κ . A line of $\text{PG}(2, q)$ meets an arc in at most two points, and at most $\binom{m}{2}$ lines meet S in two points. If $m \leq q+1$ then $\binom{m}{2} \leq q(q+1)/2 < q^2 + q + 1$. If $m = q+2$ (q even, S a hyperoval), the $\binom{q+2}{2}$ secants leave $q^2 + q + 1 - \binom{q+2}{2} = q(q-1)/2 > 0$ external lines. In both cases some line ℓ meets S in at most one point, and we take κ with $\kappa^\perp = \ell$.

Coverage. Write $1, \dots, m$ for the columns coming from B and $*$ for the new column. Since φ is onto, every target symbol s has a preimage $\varphi^{-1}(s) \neq \emptyset$.

Case 1: columns $i < j < l \leq m$. Given (s, t, u) , choose preimages α, β, γ under φ . Since $B(S)$ is an orthogonal array of index one, some w has $(\langle p_i, w \rangle, \langle p_j, w \rangle, \langle p_l, w \rangle) = (\alpha, \beta, \gamma)$, and row w of (B1) covers (s, t, u) .

Case 2: columns $i < j \leq m$ and $*$, target (s, t, y) with $c \leq y \leq v-1$. The copy of D in (B3) for the value y covers (s, t) in columns i, j , and its entry in column $*$ is y .

Case 3: columns $i < j \leq m$ and $*$, target (s, t, y) with $0 \leq y \leq c-1$. The map $\psi : w \mapsto (\langle p_i, w \rangle, \langle p_j, w \rangle)$ is onto $\text{GF}(q)^2$, as p_i, p_j are independent. Its kernel is the one-dimensional space $L = \langle p_i, p_j \rangle^\perp$. Choose preimages α, β of s, t and let $F = \psi^{-1}(\alpha, \beta)$, a coset of L with q elements. Every row of (B1) indexed by $w \in F$ has (s, t) in columns i, j . If $L = \langle \kappa \rangle$, then κ would be orthogonal to both p_i and p_j , contrary to the choice of κ . Hence $L \cap \langle \kappa \rangle = 0$, π is injective on F , and $M = \pi(F)$ is a line of the affine plane.

- If $O \in M$, then F contains some $w \in \langle \kappa \rangle$. Row w of (B1) has $g(w) = 0$, and for each $y = 1, \dots, c-1$ block (B2) contains $(B_v(w) \mid y)$. All c values occur with (s, t) .
- If $O \notin M$, then M is parallel to exactly one line through O and meets each of the other q lines through O in one point. So at most one of the $q+1$ lines through O is missed by M . Since every class C_y has at least two lines, M meets some line of C_y in a point $\pi(w)$ with $w \in F$. As $w \notin \langle \kappa \rangle$, $g(w) = y$.

In every case the target is covered. The row count is $q^3 + (c-1)q + (v-c)N_2$. $\square$

Remark 2. At $c = 1$ the class partition is trivial, (B2) is empty and the array is inequality (1) applied to B_v . Increasing c by one adds q rows and removes N_2 rows, so the bound decreases in c exactly when $\text{CAN}(2, m, v) > q$. That holds for $m \geq 3$ and $v \geq 2$ whenever $v^2 > q$, and in particular in every cell below. Then the best choice is $c = \min(v, \lfloor (q+1)/2 \rfloor)$, realised by pairs of lines for odd q and by $q/2 - 1$ pairs and one triple for even q . When v is smaller, merge classes until $c = v$.

Corollary 3. *For an odd prime power q , $\text{CAN}(3, q+2, q) \leq q^3 + q(q^2 - 1)/2$.*

Proof. Take the conic ($m = q+1$), $v = q$, $c = (q+1)/2$ and $D = \text{OA}(q^2; 2, q+1, q)$. Then $q^3 + \frac{q-1}{2}q + \frac{q-1}{2}q^2 = q^3 + q(q^2 - 1)/2$. $\square$

For comparison, [7, Lemma 3.3] gives $2q^3 - q^2 - q$ for the same parameters, and the lower bound [7, Theorem 8.1] is $q^3 + 12$ for odd $q \geq 11$ ($q^3 + 11$ for $q = 7, 9$).

Corollary 4. *For an even prime power $q \geq 4$, $\text{CAN}(3, q+3, q) \leq q^3 + q(q-2)/2 + (q/2) \text{CAN}(2, q+2, q)$.*

Proof. Take the hyperoval ($m = q+2$), $v = q$ and $c = q/2$. $\square$

q	Corollary 3	[7, Lemma 3.3]	lower bound [7, Thm. 8.1]
9	1089	1368	740
11	1991	2530	1343
13	3289	4212	2209
19	10279	13338	6871

Table 1: Bounds on $\text{CAN}(3, q + 2, q)$ for odd q .

3 Twelve improved table entries

Table 2 lists the cells in which Theorem 1 improves on the November 2024 tables. These are improvements on the tables, on [17] and on [21]; the tables of results of [15] were not read (Section 4). The strength-two ingredient D is of three kinds.

- (o) For $v = q$, $D = \text{OA}(q^2; 2, q + 1, q)$.
- (f) Fused: the orthogonal array $\text{OA}(Q^2; 2, Q + 1, Q)$, with Q the least prime power such that $Q \geq v$ and $Q + 1 \geq m$, is fused onto v symbols, and rows whose pairs are all covered elsewhere are then removed greedily. This gives $Q^2 - 2(Q - v)$ rows: $\text{CA}(79; 2, 10, 8)$ from $Q = 9$, and $\text{CA}(357; 2, 20, 17)$ and $\text{CA}(359; 2, 20, 18)$ from $Q = 19$ (compare the fusion bound [7, (4)]).
- (s) Supplied: for the triples $(N_2, m, v) = (117, 12, 10), (165, 14, 12), (266, 18, 14), (278, 18, 15)$ and $(285, 18, 16)$, a $\text{CA}(N_2; 2, m, v)$. These were found by a local search that fuses an orthogonal array over $\text{GF}(11)$, $\text{GF}(13)$ or $\text{GF}(17)$ and then deletes and repairs rows. We do not describe that search as an algorithm. The five arrays are given as ancillary files and each is checked exactly at strength two; Theorem 1 needs nothing else about D .

With the fused ingredient (f) in place of the supplied ones, the five cells of kind (s) give 1862, 3110, 5906, 6203 and 6504 rows. All but the first still improve on the tables. The smallest published values of $\text{CAN}(2, m, v)$ are smaller than ours in six cells (72, 116, 247, 255, 352 and 357 for $v = 8, 10, 14, 15, 17, 18$); with those arrays Theorem 1 would give 824, 1850, 5690, 5993, 9494 and 9886. For $v = 8$ the value is the bound $\text{CAN}(2, 10, 8) \leq 72$ printed in [7, §8, Table 7] (a 2007 private communication, no public array); the tables list 76, which would give 840. We do not hold those arrays and report only the arrays we built and checked. The supplied $\text{CA}(285; 2, 18, 16)$ has one row fewer than the tables' $\text{CAN}(2, 18, 16) \leq 286$.

The improvements in $\text{CAN}(3, 13, 10)$ (4 rows, 0.22%) and $\text{CAN}(3, 21, 17)$ (8 rows, 0.08%) are thin; the first also depends on the supplied D . The other ten improvements are between 2.5% and 13.9%. The cells share one shape: v is a prime power or just below one, and k is one column past the largest Bush array over $\text{GF}(q)$. There the tables fall back on a fused orthogonal array over a larger field or on doubling. In the sweep described at the end of this note, the tables already do better than Theorem 1 in every cell with $q \leq 7$ or $q \in \{17, 23, 25\}$, and with $q = 16$ and $v \leq 13$.

Remark 5. Local search started from arrays of this shape can go below Theorem 1. The program search described at the end of this note produced a $\text{CA}(1698; 3, 13, 10)$ and a $\text{CA}(2987; 3, 15, 12)$. Both pass the exact checkers and are included in the ancillary files, but the row deletions and repairs that produced them are not reproduced there, so we do not count them as consequences of Theorem 1. A non-LLM local-search control subsequently produced a $\text{CA}(2891; 3, 15, 12)$, checked by an independent exact checker and included in the ancillary files; this search certificate is also not a consequence of Theorem 1.

cell	q	arc	c	N_2	N (this note)	tables Nov. 2024	gain (rows)	gain (%)	best from [7]
CAN(3, 11, 8)	8	hyperoval	4	79 (f)	852	960	108	11.25	960
CAN(3, 11, 9)	9	conic	5	81 (o)	1089	1175	86	7.32	1368
CAN(3, 13, 10)	11	conic	6	117 (s)	1854	1858	4	0.22	2199
CAN(3, 13, 11)	11	conic	6	121 (o)	1991	2064	73	3.54	2530
CAN(3, 15, 12)	13	conic	7	165 (s)	3100	3486	386	11.07	3754
CAN(3, 15, 13)	13	conic	7	169 (o)	3289	3798	509	13.40	4212
CAN(3, 19, 14)	16	hyperoval	8	266 (s)	5804	6006	202	3.36	6562
CAN(3, 19, 15)	16	hyperoval	8	278 (s)	6154	6415	261	4.07	7237
CAN(3, 19, 16)	16	hyperoval	8	285 (s)	6488	6652	164	2.47	7936
CAN(3, 21, 17)	19	conic	10	357 (f)	9529	9537	8	0.08	11411
CAN(3, 21, 18)	19	conic	10	359 (f)	9902	11500	1598	13.90	12364
CAN(3, 21, 19)	19	conic	10	361 (o)	10279	11791	1512	12.82	13338

Table 2: Cells improved by Theorem 1, with $N = q^3 + (c - 1)q + (v - c)N_2$ and the kind of D in brackets. The table values are those of the November 2024 tables [6] as preserved in [16], and agree with [17] as of 13 October 2026; for $(k, v) = (11, 8)$, $(19, 14)$ and $(21, 17)$ the entry is inherited from a larger k . The gain is relative to the table value. The last column is the least of [7, (8), (9), Lemma 3.3] evaluated on the tables’ values of $\text{CAN}(3, k - 1, v)$ and $\text{CAN}(2, k - 1, v)$.

Ancillary files and checking. The directory `arrays` contains the twelve arrays of Table 2 (one row per line). Each is the exact output of `construct.py`, which builds the array of Theorem 1 for given (q, v, m, c) and, for the five cells of kind (s), reads D from the directory `blocks`. Before using a supplied D , `construct.py` checks that every pair of its columns shows all v^2 pairs. The directory `arrays/outreach` holds eight arrays: the seven arrays of the earlier program search for $v = 10, \dots, 16$, unchanged, and the $\text{CA}(2891; 3, 15, 12)$ of the non-LLM local-search control. `check.ca.py` is an exact checker. It takes t, k, v, N on the command line, checks the row count, the row lengths and the symbol range, and checks that every t -set of columns shows all v^t tuples. Both scripts use only the Python standard library. The checker rejects six planted faulty inputs and accepts the control (`--selftest`). Checking the largest array takes about six seconds on one core.

4 Relation to other constructions

The constructions below extend arrays by a constant or duplicated column, augment the symbol set, multiply the number of columns, or use layers of linear columns. Adding a non-constant column is not new in itself: the duplicated-column inequality [7, (8)] does so. What we have not found elsewhere is a column coloured by directions of a quotient plane, with the repair cost of Theorem 1. Four works below were read only in part or not at all; we say which.

- Colbourn, Kéri, Rivas Soriano, Schlage-Puchta [7]: inequality (1) is the case $c = 1$; the passage quoted in the introduction states the improvement we carry out; Lemma 3.3 is their own instance of it for Bush arrays.
- Chateauneuf and Kreher [2]: Construction D adds a *symbol* at the cost of k strength-two arrays and $k(v - 1)$ further rows; Theorem 4.5 doubles the number of columns. Their table gives $\text{CAN}(3, 11, 9) \leq 1331$ from the Bush array over $\text{GF}(11)$.

- Chateauneuf, Colbourn and Kreher [3]: arrays developed under $\text{PGL}(2, q)$ and a squaring construction; no single-column extension.
- Sherwood, Martirosyan and Colbourn [14]: the Bush and hyperoval arrays as permutation vectors, and arrays built from several layers of linear columns. Going past the Bush width costs a second layer of $q^3 - q$ rows; there is no non-linear column.
- Colbourn, Martirosyan, Tran Van Trung and Walker [8]: Roux-type constructions that multiply the number of columns.
- Shokri and Moura [12]: Roux-type multiplication of LFSR arrays. The independent review inspected the introduction, Theorem 4.6 and Tables 6–7. The displayed same-alphabet examples have more columns and larger row counts, including $N = 7969$, $k = 366$ for $q = 13$ and $N = 25669$, $k = 762$ for $q = 19$. Projection alone of those displayed examples does not improve the corresponding cells here. This bounded comparison does not rule out improvements by specialization or post-optimization.
- Shokri’s thesis [13] on covering arrays from finite geometry and finite fields could not be obtained (the repository did not respond on 13 and 14 October 2026); we read only two conference abstracts by its author. By those abstracts its strength-three constructions are Roux-type concatenations of the arrays of [20], with at least $2(q^2 + q + 1)$ columns, and its geometric part treats strength four; neither adds a coloured column to a Bush array. We cannot exclude a relevant remark in the unread text.
- Colbourn [5]: augmentation by quilting arrays, which increases the number of symbols; its abstract states results for strengths four to six. We have read only the abstract of this paper.
- Torres-Jiménez and Izquierdo-Márquez [15] build covering perfect hash families from extended permutation vectors, giving $\text{CA}(2v^3 - v; 3, v^2 - v + 3, v)$ for prime powers v , and eighteen further arrays by simulated annealing. We could read only the abstract and the reference list; the full text, including its tables of results, was not openly available to us and has not been read. None of its ten references concerns orthogonal arrays, Bush arrays or adding a factor. The tables list results attributed to these authors for $8 \leq v \leq 19$ only at $k \geq 75$ and $N \geq 1447$, above every array in Table 2, but we have not checked the paper’s own tables against them. We therefore make no claim to improve on [15] itself: for any of our cells that its results may cover, the comparison in this note is only with the tables [6], with [17] and with [21]. Structurally, every column of a permutation-vector array is a linear function on $\text{GF}(q)^3$, so past the Bush width an unrestricted hash family needs a second layer and $2q^3 - q$ rows, and any union of two linear blocks has at least $2q^3 - q^2$ distinct rows; restricted families [18] and post-optimisation can remove rows, and for those our comparison is with the tabulated values only. We cannot exclude a remark in the unread text. Their later paper [21] treats $v = 7$ only; its restricted hash families do go below the unrestricted count, for example $\text{CA}(973; 3, 64, 7)$ from three layers against $3(7^3 - 7) + 7 = 1015$.
- Raaphorst, Moura and Stevens [20] construct $\text{CA}(2q^3 - 1; 3, q^2 + q + 1, q)$ from linear feedback shift register sequences; projected to $q + 2$ columns this is far above Corollary 3.
- Das and Mészáros [19] also combine linear-algebraic partial coverage with completion: blocks of v^3 rows that each cover a large fraction of all column triples, used in a semi-random construction. Their Corollary 1.8 needs three or four such blocks at the seven cells of Table 2 with a prime-power alphabet v , between 1.55 and 2.68 times our N .

- Small q : the general families above are not the whole story. For example [7] records a CA(472; 3, 9, 7), below the value 511 of Corollary 3 at $q = 7$, and the tables beat Theorem 1 for every $q \leq 7$ (Section 3).

5 Open questions

Is the restriction to classes of size at least two optimal? A colouring adapted to the parallel classes of each pair of columns might cover more than $\lfloor (q+1)/2 \rfloor$ values inside the orthogonal array. We do not know whether Corollary 3 is tight; the gap to the lower bound of [7] is about $q^3/2$.

Disclosure: how these results were produced

- *Origin.* The step that separates Theorem 1 from inequality (1), giving the added column several values on the orthogonal array and paying strength-two arrays only for the values left over, first appeared on 24 September 2026 in arrays produced by an LLM-driven program search run by the Institute for Recursive Self Improvement (IRSI). In that search, language models write generator programs, an exact checker scores their output, and the best programs are kept and modified. Its brief supplied the frame (orthogonal array, one added column, patch rows) but asked for an added column that takes as many values as possible; the step was not in the brief. The repository does not record which model wrote the program in question. The search’s proposers at that time were Claude Opus 5.5 (two slots), Kimi K3 (two), GPT-6 Sol (two), GPT-6 Luna and gpt-oss-120b (one each). The same day a Claude Opus 5.5 agent read a closed form off those arrays and produced the supplied strength-two arrays of Section 3. On 30 September another Claude Opus 5.5 agent stated and proved Theorem 1 and evaluated it for every prime power $q \leq 25$, every $2 \leq v \leq q$ and every admissible k (2,314 parameter triples, against the November 2024 tables). The twelve cells of Table 2 are the cells in which it improved on the tables.
- *Choice of problem and failures.* Covering-array tables were chosen because they are public and every entry can be checked exactly. The program search ran on 34 target instances in five families (covering arrays 5, covering designs 11, degree-diameter graphs 6, binary linear codes 7, van der Waerden numbers 5) and improved on the reference value in 7 of them: the five covering-array cells it targeted and two van der Waerden numbers. The reporting period was 23 September to 1 October 2026. A retrospective, non-exhaustive triage grouped 21 problem families, including mathematical optimization, as 9 with reported candidate improvements, 3 partial or contested, and 9 without an improvement. These are reported candidates, most not yet accepted by an independent check, rather than nine independently accepted discoveries. The list and the counting method are available from the authors on request.
- *Models, inputs, time and cost.* The program search ran for 10 hours 9 minutes (02:15 to 12:24 UTC, 24 September); the step first appeared after 6 hours 40 minutes. The proof and the sweep took about 1.5 hours (Claude Opus 5.5, high reasoning effort). GPT-6 Sol agents and a Claude agent recomputed earlier copies of the arrays with their own code. This note was drafted and revised by Claude Opus 5.5 agents at high reasoning effort. GPT-6 Astra agents applied review edits and reran the checkers; the 13 October revision had no metered cost. The search’s metered model calls cost US\$35.82 for its whole covering-array run; drafting used about US\$6 of prepaid cloud-session credit; all other agent work ran on flat-rate subscriptions with no metered cost. All computation ran on CPUs.

- *Checking.* Every array in the ancillary files passes two exact checkers written independently of each other (`check_ca.py` and a second checker in the lab’s repository), and each array of Table 2 is the output of `construct.py`. All twelve arrays of Table 2, the seven arrays of the earlier program search and the five supplied strength-two arrays were rechecked on 3 October 2026 by an independent GPT-6 agent with its own exact checker, which also rejected five planted faulty inputs, and again on 13 October 2026. The proof of Theorem 1 was checked line by line by that agent, which found one gap in generality (now Remark 2’s last sentence) and no gap in the coverage argument. The CA(2891; 3, 15, 12), added on 14 October 2026, passes `check_ca.py` and a separate standard-library checker written for it, which rejected three planted faulty inputs. The proof has not been checked by a human referee and has not been formalised in Lean.
- *Literature.* The papers [7, 2, 3, 14, 8] were read in full ([2, 3] as author preprints). Only the abstracts of [5] and [15] (and the reference list of the latter) were read; neither full text was openly available to us, so the tables of results of [15] were not compared cell by cell with Table 2. Of [13] only two conference abstracts were read. [12] was read in part (introduction, Theorem 4.6, Tables 6 and 7). The literature and the tables were rechecked on 13 October 2026, and a reviewer from a different model provider (OpenAI GPT-6.1 Sol) searched independently for prior work and found none that gives this construction or reaches these cells.

References

- [1] K. A. Bush, Orthogonal arrays of index unity, *Ann. Math. Statist.* 23 (1952) 426–434.
- [2] M. A. Chateauneuf, D. L. Kreher, On the state of strength-three covering arrays, *J. Combin. Des.* 10 (2002) 217–238.
- [3] M. A. Chateauneuf, C. J. Colbourn, D. L. Kreher, Covering arrays of strength three, *Des. Codes Cryptogr.* 16 (1999) 235–242.
- [4] C. J. Colbourn, Combinatorial aspects of covering arrays, *Le Matematiche* 59 (2004) 125–172.
- [5] C. J. Colbourn, Covering arrays, augmentation, and quilting arrays, *Discrete Math. Algorithms Appl.* 6 (2014) 1450034.
- [6] C. J. Colbourn, Covering array tables for $t = 2, \dots, 6$, Arizona State University; withdrawn in 2025, state of November 2024.
- [7] C. J. Colbourn, G. Kéri, P. P. Rivas Soriano, J.-C. Schlage-Puchta, Covering and radius-covering arrays: constructions and classification, *Discrete Appl. Math.* 158 (2010) 1158–1180.
- [8] C. J. Colbourn, S. S. Martirosyan, Tran Van Trung, R. A. Walker II, Roux-type constructions for covering arrays of strengths three and four, *Des. Codes Cryptogr.* 41 (2006) 33–57.
- [9] U. Grömping, Implementing uniform covering arrays based on the Colbourn tables, Reports in Mathematics, Physics and Chemistry 01/2026, Berliner Hochschule für Technik, 2026; and the R package `CAs`.
- [10] A. S. Hedayat, N. J. A. Sloane, J. Stufken, *Orthogonal Arrays: Theory and Applications*, Springer, 1999.

- [11] J. W. P. Hirschfeld, *Projective Geometries over Finite Fields*, 2nd ed., Oxford University Press, 1998.
- [12] K. Shokri, L. Moura, New families of strength-3 covering arrays using linear feedback shift register sequences, *J. Combin. Des.* 33 (2025) 156–171.
- [13] K. Shokri, Constructing new families of covering arrays using finite geometry and finite fields, PhD thesis, University of Ottawa, 2026.
- [14] G. B. Sherwood, S. S. Martirosyan, C. J. Colbourn, Covering arrays of higher strength from permutation vectors, *J. Combin. Des.* 14 (2006) 202–213.
- [15] J. Torres-Jiménez, I. Izquierdo-Márquez, Covering arrays of strength three from extended permutation vectors, *Des. Codes Cryptogr.* 86 (2018) 2629–2643.
- [16] U. Grömping, Colbourn tables (November 2024 status), <https://github.com/ugroempi/CAs/blob/main/ColbournTables.md>, 25 July 2025.
- [17] ca-tools, provenance-tracked covering array table, <https://github.com/chrispbismolen/ca-tools>, commit 7699e6d (20 September 2026).
- [18] C. J. Colbourn, E. Lanus, Subspace restrictions and affine composition for covering perfect hash families, *Art Discrete Appl. Math.* 1 (2018) #P2.03.
- [19] S. Das, T. Mészáros, A semi-random construction of small covering arrays, arXiv:1703.05252 (2017).
- [20] S. Raaphorst, L. Moura, B. Stevens, A construction for strength-3 covering arrays from linear feedback shift register sequences, *Des. Codes Cryptogr.* 73 (2014) 949–968.
- [21] J. Torres-Jiménez, I. Izquierdo-Márquez, New upper bounds for covering arrays of order seven, *Mathematics* 12 (2024) 1908.