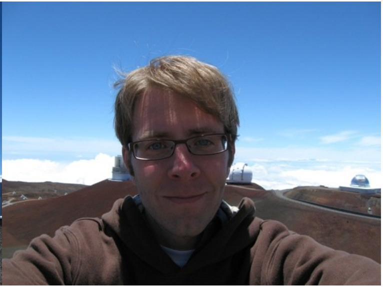
Variable Strength Covering Arrays

Lucia Moura
School of Electrical Engineering and Computer Science
University of Ottawa
lucia@eecs.uottawa.ca

ALCOMA 2015, Kloster Banz, March 2015

joint work with **Sebastian Raaphorst** and **Brett Stevens**


Orthogonal arrays

Strength $t = 2$; $v = 3$ symbols; $k = 4$ columns; 2^3 rows

$$\begin{bmatrix} 0000 \\ 0122 \\ 1220 \\ 2202 \\ 2021 \\ 0211 \\ 2110 \\ 1101 \\ 1012 \end{bmatrix}$$

Definition: Orthogonal Array

An *orthogonal array* of strength t, k columns, v symbols and index λ denoted by $OA_\lambda(t, k, v)$, is an $\lambda v^t \times k$ array with symbols from $\{0, 1, \ldots, v-1\}$ such that in every $t \times N$ subarray, every t-tuple of $\{0, 1, \ldots, v-1\}^t$ appears in exactly λ rows.

Covering arrays

Strength $t = 3$; $v = 2$ symbols; $k = 10$ columns; $N = 13$ rows

0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1
1	1	1	0	1	0	0	0	0	1
1	0	1	1	0	1	0	1	0	0
1	0	0	0	1	1	1	0	0	0
0	1	1	0	0	1	0	0	1	0
0	0	1	0	1	0	1	1	1	0
1	1	0	1	0	0	1	0	1	0
0	0	0	1	1	1	0	0	1	1
0	0	1	1	0	0	1	0	0	1
0	1	0	1	1	0	0	1	0	0
1	0	0	0	0	0	0	1	1	1
0	1	0	0	0	1	1	1	0	1

Definition: Covering Array

A *covering array* of strength t, k factors, v symbols, index λ and size N, denoted by $CA_\lambda(N; t, k, v)$, is an $N \times k$ array with symbols from $\{0, 1, \ldots, v-1\}$ such that in every $t \times N$ subarray, every t-tuple of $\{0, 1, \ldots, v-1\}^t$ appears in at least λ rows.

Covering arrays

Strength $t = 3$; $v = 2$ symbols; $k = 10$ columns; $N = 13$ rows

0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1
1	1	1	0	1	0	0	0	0	1
1	0	1	1	0	1	0	1	0	0
1	0	0	0	1	1	1	0	0	0
0	1	1	0	0	1	0	0	1	0
0	0	1	0	1	0	1	1	1	0
1	1	0	1	0	0	1	0	1	0
0	0	0	1	1	1	0	0	1	1
0	0	1	1	0	0	1	0	0	1
0	1	0	1	1	0	0	1	0	0
1	0	0	0	0	0	0	1	1	1
0	1	0	0	0	1	1	1	0	1

Definition: Covering Array

A *covering array* of strength t, k factors, v symbols, index λ and size N, denoted by $CA_\lambda(N; t, k, v)$, is an $N \times k$ array with symbols from $\{0, 1, \ldots, v-1\}$ such that in every $t \times N$ subarray, every t-tuple of $\{0, 1, \ldots, v-1\}^t$ appears in at least λ rows.

Covering arrays generalize orthogonal arrays

We are interested in the **covering array number**

$$CAN(t,k,v) = \min\{N : CA(N;t,k,v) \text{ exists}\}.$$

An obvious lower bound: $CAN(t,k,v) \geq v^t$.

$CAN(t,k,v) = v^t$ if and only if there exists an $OA_1(t,k,v)$.

For $t = 2$, if $k > v+1$, $CAN(2,k,v) > v^2$.

For $t = 3$, if $k > v+2$, $CAN(3,k,v) > v^3$.

Indeed, we know that for fixed v and t, letting $k \to \infty$,

$$CAN(t,k,v) = O(\log k)$$

Covering Arrays: constructions and bounds

$CAN(t,k,v) = \min\{N : CA(N;t,k,v) \text{ exists}\}$

Known **asymptotic bounds** on the covering array number:

- As $k \to \infty$, $CAN(2,k,v) = \frac{v}{2} \log k(1 + o(1))$.
  (Gargano, Korner and Vaccaro 1994)
- $CAN(t,k,v) \leq g^t(t-1) \ln k(1 + o(1))$
  (Godbole, Skipper and Sunley 1996)

For the finite case, we use specific **constructions**:

- direct constructions (algebraic, computer searches) - base ingredients.
- indirect constructions (recursive) - build "larger" arrays based on smaller ingredients.

Records of best upper bounds: Colbourn's CA tables (online).

Nice survey:
Colbourn (2004) "Combinatorial aspects of covering arrays".

CAs: applications in software and hardware testing

Component			
Web Browser	**Operating System**	**Connection Type**	**Printer Config**
Netscape	Windows	LAN	Local
IE	Macintosh	PPP	Networked
Mozilla	Linux	ISDN	Screen

Table 1. Four Factors, Each With Three Values

Test	Browser	OS	Connection	Printer
1	Netscape	Windows	LAN	Local
2	Netscape	Linux	ISDN	Networked
3	Netscape	Macintosh	PPP	Screen
4	IE	Windows	ISDN	Screen
5	IE	Macintosh	LAN	Networked
6	IE	Linux	PPP	Local
7	Mozilla	Windows	PPP	Networked
8	Mozilla	Linux	LAN	Screen
9	Mozilla	Macintosh	ISDN	Local

Table 2. Test Suite Covering All Pairs from Table 1

CAs: generalizations useful for applications

- **Mixed alphabets**: each column may have different alphabet sizes.
  - Moura, Stardom, Stevens, Williams (2003)
  - Colbourn, Martirosyan, Mullen, Shasha, Sherwood, Yucas (2005)

- **Variable strength**: different types of strength are required among different factors (hypergraph on columns)
  - Cheng, Dumitrescu, Schroeder (2003)
  - Meagher and Stevens (2005),
  - Meagher, Moura and Zekaoui (2007)
  - Cheng (2007)
  - Raaphorst, Moura, Stevens (2012).

Variable Strength Covering Arrays (Covering arrays on Hypergraphs)

Definition

Let Δ be an ASC over $\{0, \ldots, k-1\}$ with set of facets Λ, and let $t = rank(\Delta)$.
A $VCA_\lambda(N; \Lambda, g)$ **variable strength covering array**, where $\lambda = (\lambda_1, \ldots, \lambda_t)$, is an $N \times k$ array over $\{0, \ldots, g-1\}$ with columns $0, \ldots, k-1$ such that if $B = \{b_0, \ldots, b_{s-1}\} \in \Lambda$, then B is λ_s-covered. When $\lambda_i = 1$ for all $i \in \{|B| : B \in \Lambda\}$, the parameter λ is frequently omitted. We take $VCAN_\lambda(\Lambda, g)$ to be the smallest N such that a $VCA_\lambda(N; \Lambda, g)$ exists.

We take $\lambda = 1$.

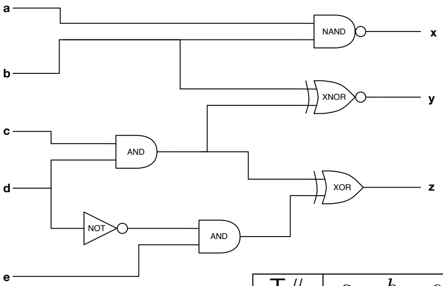
Variable Strength Covering Array: example

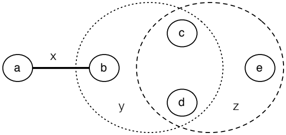
A $VCA(27; \Lambda, 3^5 9)$ for $\Lambda = \{\{0,1,2,3,4\} \times \{5\}\} \cup \left(\binom{\{0,1,2,3,4\}}{3} \setminus \{0,2,4\}\right)$:

f_0	f_1	f_2	f_3	f_4	f_5
0	0	0	2	1	0
0	0	1	0	2	1
0	0	2	1	0	2
0	1	0	1	1	3
0	1	1	2	2	4
0	1	2	0	0	5
0	2	0	0	1	6
0	2	1	1	2	7
0	2	2	2	0	8
1	0	0	0	0	4
1	0	1	1	1	8
1	0	2	2	2	6
1	1	0	2	0	7
1	1	1	0	1	2

f_0	f_1	f_2	f_3	f_4	f_5
2	0	0	1	2	5
2	0	1	2	0	3
2	0	2	0	1	7
2	1	0	0	2	8
2	1	1	1	0	6
2	1	2	2	1	1
2	2	0	2	2	2
2	2	1	0	0	0
2	2	2	1	1	4
1	2	0	1	0	1
1	2	1	2	1	5
1	2	2	0	2	3
1	1	2	1	2	0

Application example





T#	a	b	c	d	e
1	0	0	0	0	0
2	0	1	0	0	1
3	1	0	0	1	0
4	1	1	0	1	1
5	0	0	1	0	0
6	0	1	1	0	1
7	1	0	1	1	0
8	1	1	1	1	1

Constructions and bounds for VCA

- Constructions for specific hypergraphs/ASC: hyper trees,
- Construction with upper bound: density algorithm (greedy)
- Upper bound from the probabilistic method (non-constructive)

Density-based greedy algorithm

- For $t = 2$ Cohen, Dalal, Fredman and Patton (1997) propose a greedy algorithm with **logarithmic guarantee** on the size of the array (basis for their AETG software). It uses $O(\log k)$ steps but each step requires to solve an NP-complete problem which they approximate with a heuristic. So we either sacrifice the logarithmic guarantee or the polynomial time.
- Colbourn, Cohen and Turban (2004) introduced the concept of density and give a **polynomial time** algorithm with **logarithmic guarantee** for $t = 2$.
- Bryce and Coulbourn (2008) generalize this algorithm for **general** t. **(polytime; logarithmic guarantee)**
- Raaphorst, Moura and Stevens (2011) generalize this algorithm for **variable strength**. **(polytime; logarithmic guarantee)**

Density algorithm: main idea

An intermediate step of the algorithm:

	f_0	f_1	f_2	f_3	f_4
Test 1	0	2	4	6	8
Test 2	1	?	?	6	?

Close a factor, say f_1, and calculate densities:

$f_1 = 2$		$f_1 = 3$	
(1,2,4)	½	(1,3,4)	½
(1,2,5)	½	(1,3,5)	½
(1,2,6)	1	(1,3,6)	1
(1,2,8)	½	(1,3,8)	½
(1,2,9)	½	(1,3,9)	½
(2,5,6)	½	(3,4,6)	½
(2,6,9)	½	(3,5,6)	½
Total	**4**	(3,6,8)	½
		(3,6,9)	½
		Total	**5**

Density for f_1

$$\frac{(4+5)}{2} = 4.5$$

Choose a level for factor f_1. As shown above, $f_1 = 3$ has a larger density than that of $f_1 = 2$.

	f_0	f_1	f_2	f_3	f_4
Test 1	0	2	4	6	8
Test 2	1	3	?	6	?

(example extracted from Bryce and Colbourn (2008))

Density algorithm

Let $\mathcal{T} \leftarrow \emptyset$.
while there are interactions over Λ which are uncovered in $\mathcal{T}$ **do**
  Create a new row $S = \emptyset$
  Let $F \leftarrow \{0, \ldots, k-1\}$.
  while $F \neq \emptyset$ **do**
    Pick any $f \in F$.
    Choose $\sigma_f \in \{0, \ldots, g_f - 1\}$ such that $\delta_f(S \cup \{(f, \sigma_f)\})$ is maximized.
    $S \leftarrow S \cup \{(f, \sigma_f)\}$
    $F \leftarrow F \setminus \{f\}$
  end while
  $\mathcal{T} \leftarrow \mathcal{T} \cup \{S\}$
end while
return $\mathcal{T}$

Density concepts (SKIP DETAILS OR AUDIENCE DIES)

Let $W \in \Lambda$. Take $E(S,W)$ to be all possible interactions over W that respect S (i.e. the extensions of S to W), written:

$$E(S,W) = \{ \left(\bigcup_{f \in \phi(S) \cap W} \{(f, \sigma_f)\} \right) \cup \left(\bigcup_{f \in W \setminus \phi(S)} \{(f, a_f)\} \right) : a_f \in \{0, \ldots, g_f - 1\} \text{ for all } f \in W \setminus \phi(S) \}.$$

Define $r(S,W)$ to be the number of interactions in $E(S,W)$ that are not yet covered in some row, i.e. $r(S,W) = \sum_{I \in E(S,W)} \gamma(I)$.

Definition

The *density of a set W over an interaction S* is the ratio of uncovered interactions over W respecting S to the total number of interactions over W respecting S:

$$\delta(S,W) = \frac{\sum_{I \in E(S,W)} \gamma(I)}{|E(S,W)|} = \frac{r(S,W)}{|E(S,W)|} = \frac{r(S,W)}{\prod_{f \in W \setminus \phi(S)} g_f}. \quad (1)$$

(torture cont'd) (SKIP DETAILS OR SPEAKER DIES)

The *interaction density* of an interaction S is defined $\delta(S) = \sum_{W \in \Lambda} \delta(S, W)$, which can be rewritten:

$$\delta(S) = \sum_{\substack{W \in \Lambda \\ f \notin W}} \delta(S, W) + \sum_{\substack{W \in \Lambda \\ f \in W}} \delta(S, W).$$

The *factor density* of f with respect to S is defined as follows:

$$\delta_f(S) = \sum_{\substack{W \in \Lambda \\ f \in W}} \delta(S, W).$$

Proposition

For f and S, the factor density $\delta_f(S)$ is the average number of uncovered interactions extending S across all choices of levels for f, i.e.: $\delta_f(S) = \frac{1}{g_f} \sum_{\sigma \in \{0, \ldots, g_f - 1\}} \delta_f(S \cup \{(f, \sigma)\})$.

Density algorithm logarithmic guarantee

Theorem

Let Λ be an ASC over k factors with $g_1, \ldots, g_k$ levels respectively. Take $g = \max\{g_i : i \in \{1, \ldots, k\}\}$, $t = \max\{|W| : W \in \Lambda\}$, and $m = \max\{\prod_{i \in W} g_i : W \in \Lambda\}$. Then the density algorithm returns a VCA$(N; \Lambda, (g_1, \ldots, g_k))$ where:

$$N \leq \frac{\ln(m|\Lambda|)}{\ln \frac{m}{m-1}} \leq m \ln(m|\Lambda|) \leq g^t(\ln|\Lambda| + t \ln g) = O(\log|\Lambda|)$$

(as $k \to \infty$, for bounded g, t)

Corollary

If Λ is the t-uniform complete hypergraph on k vertices, fixed g:

$$N \leq \frac{\ln \binom{k}{t} + \ln g^t}{\ln \frac{g^t}{g^t - 1}} \leq g^t(\ln \binom{k}{t} + \ln g^t) = g^t t \ln k + o(1).$$

Density algorithm experimental example

Hypergraph = Steiner triple system of order k

k	$\#$	g	$CA(2,k,g)$	N_m	N_M	$CA(3,k,g)$
7	1	2	6	8	8	12
		3	12	31	31	40
		5	29	143	143	180
9	1	2	7	10	10	15
		3	15	35	35	45
		5	38	154	154	225
13	2	2	8	12	13	22
		3	17	40	40	78
		5	38	171	171	225
15	80	2	8	13	14	24
		3	19	41	43	90
		5	45	178	180	365

Probabilistic method bounds for variable strength arrays

Local Lemma bound for Steiner designs:

$$VCAN(S(t-1,t,k),g) \leq (t-2)g^t \ln k + O(1).$$

The (constructive) density method bound gives:

$$VCAN(S(t-1,t,k),g) \leq (t-1)g^t \ln k + O(1).$$

Theorem (Local Lemma bound for VCA over s-(k,t,λ) designs)

Let $\mathcal{B}$ be an s-(k,t,λ) design, and let d be an upper bound on the block intersection count of $\mathcal{B}$. Then:

$$VCAN(\mathcal{B},g) \leq \frac{\ln(d+1) + t\ln g + 1}{\ln \frac{g^t}{g^t-1}}.$$

For fixed s, t, g, and λ, as $k \to \infty$, we have that:

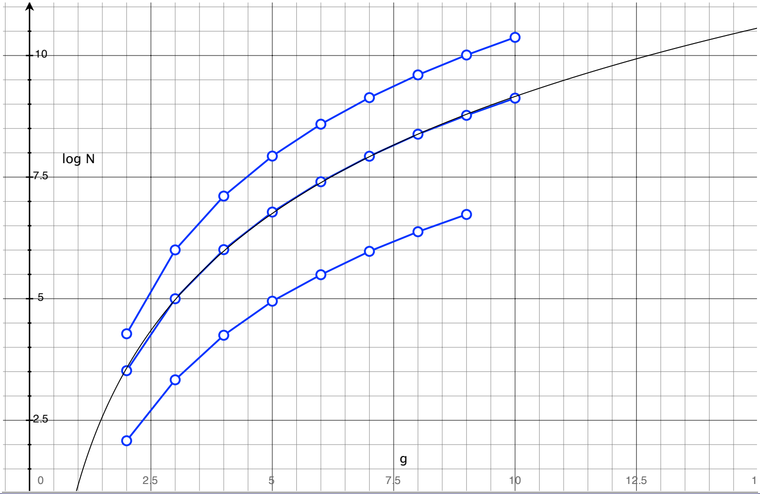
$$VCAN(\mathcal{B},g) \leq (s-1)g^t \ln k + O(1).$$

The (constructive) upper bound for the density method, gives, as $k \to \infty$:

$$VCAN(\mathcal{B},g) \leq s g^t \ln k + O(1).$$

Comparison between the bounds: density method vs probabilistic method

Hypergraph: $k = 15$, full strength 2 + strength 3 over 4 factors



Thank you!