Fall 10-28-2016

Covering Arrays and Fault Detection

Brooke LeFevre
Valparaiso University, brooke.lefevre@valpo.edu

Emily Anderson
Valparaiso University, emily.anderson@valpo.edu

Covering Arrays and Fault Detection

Emily Anderson and Brooke LeFevre

Advisor: Professor Jon Beagley

Abstract

Given their several applications, covering arrays have become a topic of significance over the last twenty years in both the mathematical and computer science fields. A covering array is a $N \times k$ array with strength t, k rows of length N, entries from the set $\{0, 1, ..., v-1\}$, and all v^t possible combinations occur between any t columns, where N, k, t, and v are positive integers. The focus of our research was to explore the different constructions of strength two and strength three covering arrays, to find better covering arrays (i.e. more cost and time efficient covering arrays), and to see if covering arrays can detect a fault in a system. Through analyzing the covering arrays that we constructed, we were able to successfully prove that in general, a covering array of strength $k+1$ can detect a single fault between any k or fewer variables in a system. Some areas of future research would include finding the location of a fault in a system or detecting two or more faults in a system.

Definitions

- A **covering array** is a $N \times k$ array with strength t, k rows of length N, entries from the set $\{0, 1, \ldots, v-1\}$, and all v^t possible combinations occur between any t columns, where N, k, t, and v are positive integers. It is often denoted as $CA(N; k, t, v)$. See Figure 1.
- A **binary covering array** is a $N \times k$ covering array with $v = 2$, where v can be 0 or 1 and N, k, t, and v are positive integers. See Figure 1.
- $\boldsymbol{CA(k, t, v) = N}$ denotes the minimum number of rows or tests that a covering array must have in order to be valid. See Figure 2.

$$\boldsymbol{CA}(4; 3, 2, 2)$$

$$\begin{matrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{matrix}$$

Figure 1: This is a 4×3 binary covering array of strength 2 with 4 tests and 3 variables.

$$\boldsymbol{CA}(8; 3, 3, 2)$$

$$\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{matrix}$$

Figure 2: This is a 8×3 binary covering array of strength 3 with 8 tests and 3 variables.

$s \backslash t$	1	2	3	4	5	6
0	2	4	8	16	32	64
1	2	4	8	16	32	64
2	2	5	10	21	42	85
3	2	6	12	24	48–52	96–108
4	2	6	12	24	48–54	96–116
5	2	6	12	24	48–56	96–118
6	2	6	12	24	48–64	96–128
7	2	6	12	24	48–64	96–128
8	2	6	12	24	48–64	96–128
9	2	7	15	30–32	60–64	120–128
10	2	7	15–16	30–35	60–79	120–179

Figure 3: This table represents the minimum number of tests needed for known binary covering arrays, where $s+t$ represents the number of columns and t represents the strength.

Constructions

Theorem 1 (Generalized Direct Product, Theorem 4.1,[2]). *When a $CA(N; k, 2, v)$ and a $CA(M; l, 2, v)$ both exist, a $CA(N+M, kl, 2, v)$ also exists.*

A=CA(4;3,2,2) B=CA(5;4,2,2) C=CA(9;12,2,2)

$$A = \begin{matrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{matrix} \qquad B = \begin{matrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{matrix}$$

$$C = \begin{matrix} 0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&1&1&0&1&1&0&1&1&0&1 \\ 0&1&1&0&1&1&0&1&1&0&1&1 \\ 1&1&0&1&1&0&1&1&0&1&1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&1&1&1&0&1&1&1&0&1&1 \\ 0&1&1&1&0&1&1&1&0&1&1&1 \\ 1&1&0&1&1&1&0&1&1&1&0&1 \\ 1&1&1&0&1&1&1&0&1&1&1&0 \end{matrix}$$

Figure 4: In this example, matrix A is written $l = 4$ times. Below that, matrix B is written $k = 3$ times. This construction then produces matrix C.

Theorem 2 (Roux Construction, Theorem 1, [1]). *When a $CA(N; k, 3, v)$, $CA(N; k, 3, v)$, and $CA(M; k, 2, v)$ exist, a $CA(N+M; 2k, 3, v)$ also exists.*

A=CA(8;3,3,2) B=CA(8;3,3,2) C=CA(4;3,2,2) D=CA(12;6,3,2)

$$A = \begin{matrix} 0&0&0 \\ 0&1&0 \\ 0&0&1 \\ 1&1&0 \\ 1&0&0 \\ 1&1&1 \\ 1&0&1 \\ 0&1&1 \end{matrix} \qquad B = \begin{matrix} 0&0&0 \\ 0&1&0 \\ 0&0&1 \\ 1&1&0 \\ 1&0&0 \\ 1&1&1 \\ 1&0&1 \\ 0&1&1 \end{matrix} \qquad C = \begin{matrix} 0&0&0 \\ 1&0&1 \\ 0&1&1 \\ 1&1&0 \end{matrix}$$

$$D = \begin{matrix} 0&0&0&0&0&0 \\ 0&1&0&0&1&0 \\ 0&0&1&0&0&1 \\ 1&1&0&1&1&0 \\ 1&0&0&1&0&0 \\ 1&1&1&1&1&1 \\ 1&0&1&1&0&1 \\ 0&1&1&0&1&1 \\ 0&0&0&1&1&1 \\ 1&0&1&0&1&0 \\ 0&1&1&1&0&0 \\ 1&1&0&0&0&1 \end{matrix}$$

Figure 5: The matrix D is produced as a result of a quad formation. It places A in the upper left corner, B in the upper right corner, C in the bottom left corner, and C complement in the bottom right corner.

Application

CA(5;4,2,2)

$$\begin{matrix} 0&0&0&0 \\ 1&0&1&1 \\ 0&1&1&1 \\ 1&1&0&1 \\ 1&1&1&0 \end{matrix}$$

Google Chat	
Wifi/Android	(0,0)
Wifi/iOS	(0,1)
Cellular/Android	(1,1)
Cellular/iOS	(1,0)

Figure 6: Here is an application of how covering arrays are used in the real world. Suppose Google wants to offer its consumers a way to communicate with others that has both visual and audio features. They have decided to call it Google Chat. Before releasing Google Chat, they want to make sure that it works between both Android and iOS phones as well as on both cellular network and wifi. The covering array to the left of the table codes the different combinations that are possible for those trying to communicate via Google Chat. Google can be pretty certain that their new application will work for these 5 combinations or "tests", but they cannot be 100% certain unless they test all 16 combinations of the 4 variables.

Fault Detection

Goal: To identify and characterize failures caused by specific combinations of option settings

Theorem 3. *If C is a covering array of strength $k+1$, then we can detect a single fault between any k variables in the system. Note that if C is a strength $k+1$ covering array, it is also a strength $k, k-1, \ldots, 1$ covering array.*

Proof. We show the result by contrapositive. Let s be a subset of size k of the columns of C, $p(s)$ be a set of values on these columns, and r be a subset of rows of C. Define $f : s \times p(s) \to r$ to be the set of rows that contain the values $p(s)$ on the subset s.

Now, take s_1, s_2 to be distinct subsets of size k of the columns of C with values $p(s_1), p(s_2)$, respectively. We know that $s_1 \cup s_2$ must contain at least $k+1$ elements. So, suppose $f(s_1, p(s_1)) = f(s_2, p(s_2))$. Let $r \in f(s_1, p(s_1))$, which means that column s_1 contains $p(s_1)$ and column s_2 contains $p(s_2)$. Let these contain distinct columns $c_1, c_2, \ldots, c_{k+1}$. In r, these values are fixed by our choice of $p(s_1), p(s_2)$. This means that $p(c_1), p(c_2), \ldots, p(c_{k+1})^{\complement}$, where $p(c_{k+1})^{\complement}$ represents a different value from the alphabet, does not appear in any row of C. Therefore, C is not a covering array of strength $k+1$, which contradicts our original assumption. So, f is injective meaning that it has a distinct set of rows for the subset of size k of the columns of C and the set of values on these columns. Thus, we can conclude that if C is a covering array of strength $k+1$, then we can detect a single fault between any k variables in the system. □

Future Work

- Detecting two faults in a system as opposed to just one fault
- Continuing to find "better" small covering arrays
- Improving the construction of both strength two and strength three covering arrays

Acknowledgments

- Valparaiso University Mathematics and Statistics Department
- MSEED Program (NSF Grant No. 1068346)
- Professor Jon Beagley

References

[1] Cohen, Myra B., Charles J. Colbourn, and Alan C. H. Ling. Constructing strength three covering arrays with augmented annealing. Discrete Mathematics, vol. 308, Elsevier B.V, 2008..doi:10.1016/j.disc.2006.06.036.

[2] Colbourn, Charles, and Jose Torres-Jimenez. Profiles of Covering Arrays of Strength Two. Journal of Algorithms and Computation 44.1 (2013): 31-59. Web. 29 June 2016.

[3] Lawrence, J., et al. A Survey of Binary Covering Arrays. Electronic Journal of Combinatorics 18.1 (2011) Web.

[4] Nie, Changhai, and Hareton Leung. A survey of combinatorial testing. ACM Computing Surveys (CSUR), vol. 43, ACM, Baltimore, 2011..doi:10.1145/1883612.1883618.

[5] Yilmaz, C., M. B. Cohen, and A. A. Porter. Covering Arrays for Efficient Fault Characterization in Complex Configuration Spaces. IEEE Transactions on Software Engineering 32.1 (2006): 20-34. Web.