Computing Coverage and Diversity Metrics for Constrained Covering Arrays

Joseph Morgan

SAS Institute Inc.

Abstract

Test engineers are often faced with the challenge of selecting test cases that maximize the chance of discovering faults while working with a limited budget. Combinatorial testing is an effective test case selection strategy to address this challenge. Currently, combinatorial testing usually implies a covering array as the underlying mathematical construct. Yet, despite their demonstrated utility, practitioners sometimes encounter challenges that impede their use. For example, given a covering array with constraints on allowed combinations of settings for some subset of inputs, it is often unclear how to assess the coverage and diversity [2] properties of the resulting covering array.

Introduction

Combinatorial testing is a method of selecting test cases that ensures that all possible combinations of settings from two (or more) inputs are accounted for, regardless of which subset of two (or more) inputs are selected. Combinatorial testing usually implies a covering array [3] as the underlying mathematical construct. Yet, despite their demonstrated utility, practitioners sometimes encounter challenges that impede their use. For example, when specifying a covering array to be used for combinatorial testing, constraints on allowed combinations of settings for some subset of inputs cannot always be easily accommodated. Furthermore, given a covering array with

constraints, assessing the coverage and diversity [2] properties of the resulting covering array is often unclear.

In this paper we address the coverage and diversity assessment challenge. However, we will limit our discussion to a particular subclass of constraints, namely "disallowed combinations."

Basic concepts

Definition 1: A covering array **CA**(N; t, ($v_1 \cdot v_2 \cdot \ldots \cdot v_k$)) is an $N \times k$ array such that the i^{th} column contains v_i distinct symbols. If for any t column projection, all $\prod_{k=1}^{t} v_{f(k)}$ combinations of symbols exist, where the k^{th} column of the t column projection is $f(k)$, then the array is a t covering array and is optimal if N is minimal for fixed t, k, and ($v_1 \cdot v_2 \cdot \ldots \cdot v_k$).

1	1	2	2	2
1	2	2	2	1
1	3	1	1	1
2	1	1	2	1
2	2	1	1	2
2	3	2	1	1
3	1	1	1	2
3	2	2	1	1
3	3	2	2	2

Figure 1: An optimal CA(9; 2, (3·3·2·2·2))

It is sometimes the case that the number of symbols is the same for all columns of the covering array, that is, $v_1=v_2=\ldots=v_k$.

1	1	1	1	1
1	1	2	1	1
1	2	1	1	2
1	2	2	1	2
2	1	1	2	1
2	1	2	2	1
2	2	1	2	2
2	2	2	2	2
1	1	1	2	2
2	2	1	1	1
2	1	2	1	2
1	2	2	2	1

Figure 2: CA(12; 3, (2·2·2·2·2))

It is usually more convenient to use exponential notation to more compactly express the symbol specification. That is, if $\mathcal{A}$ = **CA**(N; t, (v_1· v_2· ...· v_k)) then we express $\mathcal{A}$ thus,

$$\mathcal{A} = \mathbf{CA}(\mathrm{N}; \mathrm{t}, (v_1^{k_1} \cdot \ldots \cdot v_m^{k_m})), \mathrm{k} = \sum_1^m k_i.$$

Given a *t* covering array on k columns, a natural question that arises is: To what extent does a *t* covering array cover symbol combinations for projections involving *t*+1, *t*+2, …, *t*+i columns (where *t*+i ≤ k)? Dalal et al. [2] propose two metrics, *t*-Coverage and *t*-Diversity, to address this question.

Definition 2: For an $N \times K$ array where the i^{th} column contains v_i distinct symbols, its *t*-Coverage ($1 \leq t \leq K$) is the ratio of the number of **distinct** *t* tuples, to the number of **possible** *t* tuples, averaged over all *t* column projections. That is,

$$t\text{-Coverage} = \frac{\mathbf{1}}{\boldsymbol{M}} \sum_{i=1}^{M} n_i / p_i.$$

Here, n_i is the number of distinct t tuples in the covering array for the i^{th} projection and $p_i = \prod_{k=1}^{t} v_{f(k)}$, where $f(k)$ is the k^{th} column of the i^{th} projection, is the number of possible t tuples. Also, $M = {}_K\mathrm{C}_t$ is the number of t column projections that can be obtained from K columns.

Definition 3: For an $N \times K$ array where the i^{th} column contains v_i distinct symbols, its t-Diversity ($1 \leq t \leq K$) is the ratio of the number of **distinct** t tuples, to the **total** number of t tuples in the array, averaged over all t column projections. That is,

$$t\text{-Diversity} = \frac{\mathbf{1}}{\boldsymbol{M}} \sum_{i=1}^{M} n_i / r.$$

Here, r is the number of rows of the covering array.

Note that when t-Coverage is 100% the array is referred to as a t covering array. If t-Diversity is 100% then all t tuples in the array are different whereas a t-Diversity of 50% indicates that all t tuples, on average, occur twice. Whereas t-Coverage is a measure of coverage, t-Diversity is a measure of how well the covering array avoids replication.

Table 1 presents t-Coverage and t-Diversity for the covering arrays in Figures 1 and 2. The intent is to provide some sense of how those t covering arrays cover symbol combinations for projections involving t+1 and t+2 column projections. Although these values only hold for these examples, they nevertheless indicate that a t covering array could provide substantial coverage at the t+1 level.

	CA(12;3,5,2)		**CA**(9;2,($3^2 \cdot 2^3$))	
t	*t*-Cov	*t*-Div	*t*-Cov	*t*-Div
2	100	33	100	63
3	100	67	64	92
4	70	93	30	100
5	38	100	-	-

Table 1: *t*-Coverage %, *t*-Diversity % for Figures 1 and 2.

Constrained covering arrays

It is useful to further generalize covering arrays to account for constraints on combinations of symbols. In this paper, we will only address constraints that restrict the possible symbol combinations for one or more *p* column projections (where $2 \leq p \leq k$). Such constraints are typically referred to as "disallowed combinations" or "forbidden configurations" [1].

Definition 4: A constrained covering array **CCA**(N; t, ($v_1 \cdot v_2 \cdot \ldots \cdot v_k$), ϕ) is an $N \times K$ array such that the i^{th} column contains v_i distinct symbols and ϕ is a set of *p*-tuples ($2 \leq p \leq k$) such that each tuple is a set of two or more column/symbol pairs identifying a disallowed combination. If for any *t* column projection, all **possible** combinations of symbols exist at least once, then it is a *t* covering array and it is optimal if N is minimal for fixed t, k, ($v_1 \cdot v_2 \cdot \ldots \cdot v_k$), and ϕ.

For the example in Figure 3, there is a constraint on the possible symbol combinations for columns c_1 and c_3. That is, the symbol combinations (1, 2), (2, 1), (2, 2), (3, 1), and (3, 2) are allowed but the symbol combination (1, 1) is not allowed. We will denote disallowed combinations as a tuple of ordered pairs where each ordered pair represents a column, and its associated symbol from the disallowed combination. In this case, there are two columns involved, so we have the 2-tuple {(c_1, 1), (c_3, 1)}.

c_1	c_2	c_3	c_4	c_5
1	1	2	2	1
1	2	2	1	2
1	3	2	2	2
2	1	1	2	2
2	2	2	2	2
2	3	2	1	1
3	1	2	1	1
3	2	1	2	1
3	3	1	1	2

Figure 3: An optimal CCA(9; 2, ($3^2 \cdot 2^3$), φ={{(c_1,1), (c_3,1)}})

If you examine any two columns from the array in Figure 3 you will notice that all possible symbol combinations exist at least once. Since columns c_1 and c_2 each contain three symbols and they are not involved in the constraint, then we know that the construction must be optimal since the array contains nine rows.

Let us consider a slightly more complicated variant of the example above. As before, we have a constraint on the possible symbol combinations for columns c_1 and c_3. However, for this example the allowed symbol combinations are (2, 1), (2, 2), (3, 1), and (3, 2) and the disallowed symbol combinations are (1, 1) and (1, 2). That is, when column c_1 is set to 1 there are no allowable settings for column c_3. This poses a dilemma because, as we can see in Figure 4, these constraints lead to an array where the cells in column c_3 that correspond to cells in column c_1 that are set to 1 cannot be assigned a symbol from the allowed symbol set of column c_3.

The covering array in Figure 4 is a special subclass of constrained covering arrays in that unassigned cells are allowed. We will refer to such covering arrays as *unsatisfiable* constrained covering arrays.

c_1	c_2	c_3	c_4	c_5
1	1	.	1	1
1	2	.	2	2
1	3	.	2	1
2	1	1	2	1
2	2	2	1	1
2	3	2	1	2
3	1	2	2	2
3	2	1	1	1
3	3	1	2	2

Figure 4: An optimal CCA(9; 2, $(3^2 \cdot 2^3)$, $\varphi=\{\{(c_1,1), (c_3,1)\}, \{(c_1,1), (c_3,2)\}\}$)

The coverage and diversity definitions above cannot be applied to constrained covering arrays without making adjustments to account for how constraints affect the number of **possible** *t* column, symbol combinations. We generalize those definitions to accommodate constrained covering arrays and use the terms "*Adjusted t*-Coverage" and "*Adjusted t*-Diversity" to distinguish these generalized metrics from the previous definitions. Furthermore, to accommodate *unsatisfiable* covering arrays, rows that contain unassignable cells (i.e. cells that contain the symbol "·") will be excluded from the computation of n_i.

Definition 5: For an $N \times K$ array where the i^{th} column contains v_i distinct symbols, its *Adjusted t*-Coverage ($1 \leq t \leq k$) is the ratio of the number of **distinct** *t* tuples, to the number of **adjusted possible** *t* tuples, averaged over all *t* column projections that contain valid tuples. That is,

$$\textit{Adjusted } t\text{-Coverage} = \frac{\mathbf{1}}{\boldsymbol{M'}} \sum_{i=1}^{M'} n_i/(p_i - a_i).$$

Here, a_i is the number of invalid t tuples arising from columns in the i^{th} projection, and $M' = M - m$, where m is the number of projections where there are no valid t tuples.

Definition 6: For an $N \times K$ array where the i^{th} column contains v_i distinct symbols, its *Adjusted* t-Diversity ($1 \leq t \leq k$) is the ratio of the number of **distinct** t tuples, to the **adjusted total** number of t tuples in the array, averaged over all t column projections that contain valid tuples. That is,

$$\textit{Adjusted } t\text{-Diversity} = \frac{\mathbf{1}}{\boldsymbol{M'}} \sum_{i=1}^{M'} n_i / r_i.$$

Here, $r_i = r - q_i$ where q_i is the number of rows in the covering array with missing values (i.e. the symbol "·") for any column in the i^{th} projection. Notice that when $M' = M$ and $\forall\, i \in \{1,\ldots,M\}$, $r_i = r$, then *Adjusted* t-Coverage reduces to t-Coverage and *Adjusted* t-Diversity reduces to t-Diversity.

Table 2 presents *Adjusted* t-Coverage and *Adjusted* t-Diversity for the covering arrays in Figures 3 and 4. As for Table 1, the intent is to provide some sense of how those covering arrays cover t+1 and t+2 column projections. Notice that exponent notation is used in Table 2 for the constraint set (see Cohen et al. [1]). This shorthand notation indicates the number of p-tuples in the constraint set (i.e. p^k indicates k p-tuples)

	CCA(9;2,($3^2 \cdot 2^3$),ϕ={2^1})		**CCA**(9;2,($3^2 \cdot 2^3$),ϕ={2^2})	
t	t-Cov	t-Div	t-Cov	t-Div
2	100	62	100	71
3	64	87	60	96
4	32	96	28	100

Table 2: *Adjusted* t-Coverage %, *Adjusted* t-Diversity % for Figure 3, 4.

Conclusion

In this paper we propose generalizations to the *t*-Coverage and *t*-Diversity metrics presented in [2]. We show how these generalizations may be used to assess the coverage and diversity properties of constrained covering arrays, at least in the case of "disallowed combinations" constraints. Furthermore, we introduce the idea of *unsatisfiable* covering arrays and show that these generalized metrics also apply to this new class of covering arrays.

References

1. Cohen, M., Dwyer, M., & Shi, J., "Constructing interaction test suites for highly-configurable systems in the presence of constraints: A greedy approach," *IEEE Transactions on Software Engineering*, 34(5), 2008, pp. 633-650.
2. Dalal, S., & Mallows, C., "Factor-covering designs for testing software," *Technometrics*, 40(3), 1998, pp. 234-243.
3. Hartman, A., & Raskin, L., "Problems and algorithms for covering arrays," *Discrete Mathematics*, 284(1–3), 2004, pp. 149–156.