A Branch and Bound Algorithm for Ternary Covering Arrays Construction using Trinomial Coefficients

Jorge Martinez-Pena and Jose Torres-Jimenez

CINVESTAV-Tamaulipas, Information Technology Laboratory, Km. 5.5 Carretera Cd. Victoria-Soto la Marina, 87130, Cd. Victoria Tamps., Mexico
{jmartinez,jtj}@tamps.cinvestav.mx

Abstract. A ternary covering array (CA($N; t, k, 3$)) is an $N \times k$ array with entries from the set $\{0, 1, 2\}$ where every $N \times t$ subarray contains each of the 3^t combinations at least once. Here, t is called the strength, k the degree or number of factors, and the alphabet is set to 3. Covering arrays (CAs) are combinatorial designs that have applications in experimental designs and they have been proved to be useful in software and hardware testing. This paper presents a Branch & Bound algorithm for constructing small instances of ternary CAs using trinomial coefficients. As a result, we derived direct constructions for some ternary CAs.

Keywords: Design of Experiments, Ternary Covering Arrays, Branch & Bound, Trinomial Coefficients.

1 Introduction

Combinatorial designs have a wide use in the design of experiments. An orthogonal array (OA) is an example of a combinatorial object that has been applied successfully in this area [1]. An OA, denoted by OA(t, k, v) is an $N \times k$ array with entries from a set of v symbols such that each $N \times t$ subarray contains exactly once the v^t different combinations. The parameter t is called the strength, k is the number of factors, and v is called the alphabet. However, the orthogonal arrays are very restricted objects because they do not exist for many configurations of the parameters t, k and v. Suppose that an experiment requires an interaction of size 6 among 8 factors, with 3 symbols per factor, it is not possible to construct an OA($6, 8, 3$). Without the required OA, an exhaustive experiment should be performed and would require 6561 tests.

A less restricted combinatorial object is a covering array (CA). A CA exists for any possible combination of parameter values t, k and v. A CA [2] is defined as follows:

Definition 1. *Let t be the strength, k the degree, and v the order (or alphabet), with $2 \leq t \leq k$ and $v \geq 2$. A covering array, denoted by CA($N; t, k, v$), is an $N \times k$ array that has entries from the set $X = \{0, .., v-1\}$ such that for every $N \times t$ subarray each of the v^t combinations are contained at least once.*

In this way, a ternary CA is a CA with v set to three. For instance, a CA($N; 6, 8, 3$) can be constructed with 1177 experiments only (result obtained from our algorithm and taken from Table 1) which means savings around 82% off in experimental tests. The same problem arises in software and hardware testing. Recent experimental results have shown that CAs are an efficient solution for generating tests suites for interaction testing problems [3]. Due to the importance of constructing CAs with the minimum number of tests we begin to describe the CA construction (CAC) problem.

The covering array number (CAN(t, k, v)) represents the minimum number of rows for which a CA($N; t, k, v$) exists. A CA($N; t, k, v$) whose parameter $N =$ CAN(t, k, v) is called optimal [4].

The CAC problem consists in obtaining optimal covering arrays. Despite the existence of some special cases where the solution is obtained by polynomial order algorithms, the general case of determining the CAN is a hard combinatorial optimization problem [5]. Therefore, it is desirable to obtain covering arrays with minimum value of N, even when the CAN is not reached. Some of the best known solutions are presented in [6]. For viewing explicit covering arrays, there is a covering array repository [7] that maintains some optimal and near optimal CAs.

Branch and bound (B&B) algorithms allows exploring the whole search space of the problems using an intelligent approach to avoid looking for solutions that do not improve the current best solution [8]. B&B techniques [9] have been applied to the problem of constructing binary CAs. This paper presents a B&B algorithm for the construction of ternary CAs that uses the trinomial coefficients to represent the search space.

This paper continues as follows. Section 2 presents the relevant related work in the construction of ternary CAs. Section 3 describes the trinomial coefficients [10] and the way we used them to construct ternary CAs. Section 4 describes our B&B algorithm that uses trinomial coefficients to represent the search space for constructing ternary CAs. Section 5 exposes the performed experimentation, a comparison of the results obtained with our B&B algorithm to the results obtained with a state-of-the-art greedy method and a set of direct constructions derived from the analysis of trinomial coefficients usage in the formation of the ternary CAs produced by the B&B algorthm. Finally, Section 6 gives important conclusions of this research work.

2 Relevant Related Work

The CAC problem consists in obtaining CAs with minimum value N. Most configurations of parameters t, k and v make the CAC problem a hard combinatorial optimization problem. Due to the importance of obtaining optimal or near optimal CAs, several strategies have been applied to the CAC problem. In this section we give a brief state-of-the-art in the construction of ternary covering arrays including direct, greedy, metaheuristic, recursive, algebraic and exact methods.

Direct methods construct CAs in polynomial time. In 1952, Bush [11] reported a direct method for constructing optimal CAs that uses Galois finite fields obtaining all CA$(v^2; 2, v+1, v)$ where v is a prime or a prime power. Then, he generalizes the algorithm for any strength $t \geq 2$. Another direct method that can construct some optimal CAs is named zero-sum [12]. Zero-sum leads to CA$(v^t; t, t+1, v)$ for any $t > 2$. Note that the value of degree is in function of the value of strength.

Different greedy methods have been designed for constructing CAs. For instance, IPOG-F [13] has been applied for small values of v and t and as large as possible value of k. This algorithm has been able to find some of the best known values of CAN for large k. However, it did not obtain the optimal CAs reported by the previous direct methods. The IPO family are some of the first computational strategies used for constructing CAs and they have been updated since the first version of the IPO algorithm. Although the greedy approach has the guarantee of constructing a CA, values of N are generally higher in comparison to some non-deterministic search algorithms.

Several versions of metaheuristic algorithms have been implemented for the construction of CAs including works using Simulated Annealing (SA) [14, 15], Tabu Search [16] and those in the cathegory of artificial life algorithms such as Genetic Algorithms and Ant Colony Optimization [17]. SA is the best metaheuristic reported for constructing CAs from small to medium large values of k. For larger values of k, metaheuristics require exponential time for attempting to obtain optimal or near optimal results.

Other strategies for obtaining ternary CAs include recursive methods [18, 19]. These algorithms makes use of small CAs called ingredients in order to construct CAs with larger k. Most of best known values of CAN for large k values are constructed using recursive methods.

There are algebraic strategies that have been applied to the CAC problem. The *group construction* approach and *cyclotomic vectors* have in common the use of a particular vector which is submited to a set of operations, usually rotations and permutations, to generate all rows necessary for CA [20, 21]. However, algebraic methods have the premise that we have the requiered vector for that specific construction. In this sense, a previous search for the correct vector is required.

Finally, exact algorithms have not been exploited as other strategies due to the capacity of generating only small instances of CAs in reasonable time. Up to now, we could find only two papers related to exact algorithms, both of them using a B&B approach [22, 9]. These works only construct binary CAs and are based on symmetric breaks and isomorphism avoidance.

In this paper we use the results reported by IPOG-F as a benchmark.

3 Proposed Representation

This section describes the trinomial coefficients and how we use them to represent the search space in the construction of a ternary CA.

3.1 Trinomial Coefficients

Given a vector of k positions and three available values per position x, y and z, we can see a trinomial coefficient as the number of combinations with repetition obtained by using the k positions vector with a times the value x, b times the value y and c times the value z ($x^a y^b z^c$), where the sum of a, b and c equals k. A formal definition of trinomial coefficients [10] is:

Definition 2. *Let $0 \leq a, b, c \leq k$ and $k = a + b + c$. A trinomial coefficient, denoted by $\binom{k}{a,b,c}$, is the coefficient obtained by the next equation*

$$\binom{k}{a, b, c} = \frac{(a + b + c)!}{a!b!c!} \tag{1}$$

The trinomial coefficients arise in the expansion of a trinomial $(x+y+z)^k$. The solution of a trinomial expansion is the trinomial theorem [23] and is determined by the following identity.

$$(x + y + z)^k = \sum_{\substack{0 \leq a,b,c \leq k \\ a+b+c=k}} \binom{k}{a, b, c} x^a y^b z^c \tag{2}$$

The equation in 2 is very important. For any k, it describes all the possible trinomial coefficients. Moreover, the number of trinomial coefficients for a given k is $\binom{k+2}{2}$ and the sum of those trinomial coefficients is 3^k.

3.2 Representation

We use the trinomial coefficients for the representation of the search space in the construction of ternary CAs. It is clear that any CA is formed by a row set. In this sense, a trinomial coefficient represents a particular subset of rows which may belong to a ternary CA. First, we give some preliminar statements that lead to our proposed representation. Then, we describe our proposed representation using a clear example.

Definition 3. *Let $0 \leq a, b, c \leq k$, $k = a+b+c$ and $k \geq 2$, where k is the ternary CA degree. A candidate row subset $\mathscr{R}^k_{a,b,c}$ is a collection of rows obtained by the trinomial coefficient $\binom{k}{a,b,c}$ and its cardinality is equal to that coefficient. The candidate row subset is generated by evaluating all combinations using $0^a 1^b 2^c$ symbols, i.e. symbol 0 is used a times, symbol 1 is used b times and symbol 2 is used c times over a k-column row.*

The previous definition leads to the next theorem.

Theorem 1. *Let $\mathscr{A}$ be a set of k-th degree trinomial coefficients. For any strength $2 \leq t \leq k$, a vertical concatenation of the row subsets generated by each trinomial coefficient in $\mathscr{A}$ may construct any ternary CA.*

Proof. Let the strength and the degree of a required CA equal q. Let $\mathscr{C}$ be a $3^q \times q$ array. Adjoin the $\binom{q+2}{2}$ trinomial coefficients of q-th degree in the set $\mathscr{A}$. For each element in $\mathscr{A}$ generate its candidate row subset and append it vertically to $\mathscr{C}$. Then $\mathscr{C}$ is an optimal $\mathrm{CA}(3^q; q, q, 3)$.

We verify the result is a ternary CA by looking into the definitions of the trinomial theorem and of the candidate row subsets. The trinomial theorem generates all possible trinomial coefficients of q-th degree. Remark that the sum of all q-th degree trinomial coefficients is 3^q. Hence if we transform each trinomial coefficient of q-th degree into candidate row subsets we will be producing 3^q different rows. These rows represent all the possible combinations that any $N \times q$ subarray must contain. By definition, any $\mathrm{CA}(q, q, v)$ has only one subarray of size q. Therefore, we have constructed an optimal ternary CA of strength q and degree q.

Now, consider the constructed $\mathrm{CA}(3^q; q, q, 3)$. The strength value in a CA is upper bounded by degree value. For any $2 \leq s < q$ we automatically derive a $\mathrm{CA}(3^q; s, q, 3)$. Then we can construct a ternary CA of any strength and any degree. Thus the theorem is correct.

Figure 1 summarizes our proposed representation by giving an explicit example of construction of a ternary CA. In this example, we construct a $\mathrm{CA}(9; 2, 3, 3)$ by using a set $\mathscr{A}$ with 4 trinomial coefficients. In the middle, we can observe a table describing each trinomial coefficient in $\mathscr{A}$ and their corresponding candidate row subsets. The table to the right displays the array $\mathscr{C}$ (the ternary CA), which is composed of every row generated by $\mathscr{A}$.

$$\mathscr{A} = \{\tbinom{3}{3,0,0}, \tbinom{3}{0,3,0}, \tbinom{3}{0,0,3}, \tbinom{3}{1,1,1}\}$$

$$\mathscr{C} = (\mathscr{R}^3_{3,0,0} + \mathscr{R}^3_{0,3,0} + \mathscr{R}^3_{0,0,3} + \mathscr{R}^3_{1,1,1})$$

$$N = \sum \mathscr{A} = 9$$

a)

$\mathscr{A}$	$\mathscr{R}^k_{a,b,c}$
$\binom{3}{3,0,0}$	0 0 0
$\binom{3}{0,3,0}$	1 1 1
$\binom{3}{0,0,3}$	2 2 2
	0 1 2
	1 2 0
$\binom{3}{1,1,1}$	2 0 1
	0 2 1
	1 0 2
	2 1 0

b)

$\mathscr{C}$
0 0 0
1 1 1
2 2 2
0 1 2
1 2 0
2 0 1
0 2 1
1 0 2
2 1 0

c)

Fig. 1. Representation of a $\mathrm{CA}(9; 2, 3, 3)$ in trinomial coefficients. a) The mathematical formulation. b) The candidate row subsets from trinomial coefficients. c) The produced ternary CA.

4 Branch and Bound Algorithm

We designed an exhaustive search algorithm that finds a ternary CA in the trinomial coefficient domain. The number of rows obtained is the minimum rows for which a ternary CA exists using our trinomial coefficients representation. The algorithm takes as input the strength t and the degree k for the construction of the ternary CA and it is structured in three steps.

In the first step we generate the set of all trinomial coefficients of k-th order and their representing row subsets in $\mathscr{T}$. The second step is a backtracking algorithm that implements a B&B technique for performance improvement. The backtracking step incrementally constructs a ternary CA by adding *feasible* elements from $\mathscr{T}$ into $\mathscr{A}$. We call *feasible* to a trinomial coefficient that is able to construct the ternary CA without exceeding the number of rows required by the current best ternary CA. Each time a solution improves the current best solution it is stored in $\mathscr{S}$ and it is used as an upper bound for the backtracking. Finally, the third step transforms $\mathscr{S}$ into the ternary CA and store it into an output file.

Figure 2 is a flow diagram that describes our B&B algorithm. The first step is only the first state in the diagram, while the third step is the last state where we output the ternary CA. As observed, the second step of the algorithm is the main part of this flow diagram. Note that each time a better solution is found, we update the upper bound and the current best solution. This upper bound determines which is a next feasible trinomial coefficient based on the sum of that coefficient and the current number of rows in $\mathscr{A}$. If this sum is longer than the current best upper bound, the trinomial coefficient is ignored and the algorithm goes to the next trinomial coefficient. Each time the algorithm joins the next feasible trinomial coefficient to $\mathscr{A}$, a partial test CA is performed.

A partial test CA requires a $\binom{k}{t} \times 3^t$ matrix. For each combination of all subarrays, this matrix allows to decide how many combinations are missing to complete the ternary CA. We perform the partial test CA in $\binom{k}{a,b,c}\binom{k}{t}$. In this sense, each row of a candidate row subset is verified by this partial test.

5 Experimentation and Results

We implemented the B&B algorithm described in section 4. The algorithm was coded in C and compiled with *gcc*. This section presents the comparison between the results from IPOG-F and the results obtained with our B&B algorithm that uses trinomial coefficients. As a result of the experimentation, some direct constructions for ternary CAs are described using specific trinomial coefficients.

5.1 Results from B&B Algorithm

We run the experimentation for different values of strength and degree. Strength ranged in $2 \leq t \leq 6$ and degree ranged in $t + 1 \leq k \leq 15$. Results for strength 5

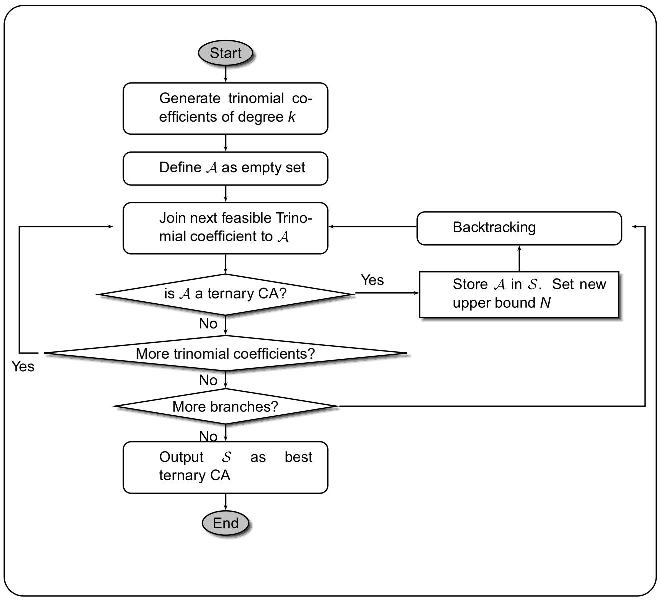
Start
Generate trinomial coefficients of degree k
Define $\mathcal{A}$ as empty set
Join next feasible Trinomial coefficient to $\mathcal{A}$
Backtracking
is $\mathcal{A}$ a ternary CA?
Yes
Store $\mathcal{A}$ in $\mathcal{S}$. Set new upper bound N
No
More trinomial coefficients?
Yes
No
More branches?
No
Output $\mathcal{S}$ as best ternary CA
End

Fig. 2. Flow diagram of our B&B algorithm.

and strength 6 are not complete due to the long time the algorithm requires to finish that experimentation.

Figure 3 displays the behaviour of results obtained with our *B&B* algorithm. We can observe that the growth of N using strength 2 is lineal in k. For strengths 3 and 4, the growth of N is quadratic in k. Finally, the growth of N for strength 5 and 6 is cubic in k.

5.2 Comparison Between IPOG-F and our B&B Algorithm

We compared the results obtained from the greedy algorithm IPOG-F [13] to the results from our B&B algorithm. The comparison is done using the vales of N from the constructed CAs.

Table 1 concentrates the results using strength values in range $2 \leq t \leq 6$ and degree values in range $t+1 \leq k \leq 15$. We can observe that our B&B algorithm outputs better results for any $k \leq t+2$. However, the rest of cases IPOG-F offers better results than those obtained with our B&B. Notice that any result obtained with our *B&B* algorithm when $k = t+1$ equals the CANs. Results for strength 5 and strength 6 are not complete due to the long time the algorithm requires to finish that experimentation.

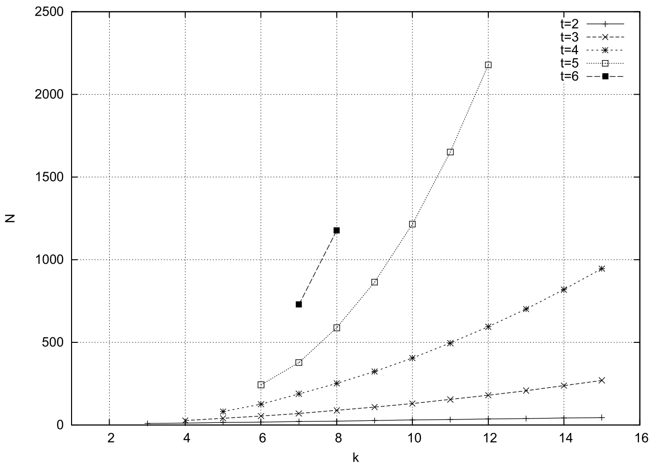

Fig. 3. Growth behaviour of N using our B&B algorithm. Abscissa values are the degree values for $2 \le k \le 15$ while ordinate values are the number of rows for the produced ternary CAs.

Table 1. Solution comparison between the IPOG-F and our B&B algorithm. Values of IPOG-F are in columns marked γ while the results obtained with our B&B algorithm are marked β. Bold values mean improved results.

k / t	2		3		4		5		6	
	γ	β	γ	β	γ	β	γ	β	γ	β
3	10	**9**								
4	12	**12**	34	**27**						
5	13	15	42	**40**	98	**81**				
6	15	18	49	54	140	**126**	318	**243**		
7	15	21	52	70	164	189	467	**378**	990	**729**
8	15	24	56	88	188	252	557	588	1490	**1177**
9	17	27	62	108	211	324	652	864		
10	19	30	66	130	228	405	738	1215		
11	19	33	68	154	248	495	815	1650		
12	20	36	71	180	262	594	885	2178		
13	20	39	76	208	277	702				
16	20	42	77	238	288	819				
15	20	45	80	270	302	945				

5.3 New Ternary Covering Array Direct Constructions

We derived some direct constructions by analyzing the trinomial coefficients formation for the ternary CAs found by our B&B algorithm.

These direct constructions match the results obtained in Table 1 and are able to construct more ternary CAs than those reported in our experimentation. In this sense, these constructions leads to ternary CAs that have the minimum possible value of N in our trinomial coefficient representation. Note that this value of N is not the CAN.

The following 4 theorems can construct ternary CAs by using a polynomial direct method. Each theorem focus the construction based in the strength of the generated ternary CAs.

Theorem 2. *$CA(3k; 2, k, 3)$ is directly derived using trinomial coefficients for any $k \geq 3$.*

Proof. Let $\mathscr{A} = \{\binom{k}{k-1,0,1}, \binom{k}{1,k-1,0}, \binom{k}{0,1,k-1}\}$ and any $k \geq 3$. Construct an array of size $3k \times k$ called $\mathscr{C}$. Concatenate vertically into $\mathscr{C}$ each of the candidate row subsets produced by $\mathscr{A}$. The result is a strength two ternary CA of degree k with optimal N in our trinomial coefficients representation.

Theorem 3. *$CA(k^2 + 3k; 3, k, 3)$ is directly derived using trinomial coefficients for any $k \geq 5$.*

Proof. Let $\mathscr{A} = \{\binom{k}{k-1,0,1}, \binom{k}{k-1,1,0}, \binom{k}{1,0,k-1}, \binom{k}{0,1,k-1}, \binom{k}{1,k-2,1}\}$ and any $k \geq 5$. Construct an array of size $(k^2 + 3k) \times k$ called $\mathscr{C}$. Concatenate vertically into $\mathscr{C}$ each of the candidate row subsets produced by $\mathscr{A}$. The result is a strength three ternary CA of degree k with optimal N in our trinomial coefficients representation.

Theorem 4. *$CA(4.5k^2 - 4.5k; 4, k, 3)$ is directly derived using trinomial coefficients for any $k \geq 7$.*

Proof. Let $\mathscr{A} = \{\binom{k}{k-2,0,2}, \binom{k}{0,2,k-2}, \binom{k}{k-2,2,0}, \binom{k}{k-2,1,1}, \binom{k}{1,1,k-2}, \binom{k}{1,k-2,1}\}$ and any $k \geq 7$. Construct an array of size $(4.5k^2 - 4.5k) \times k$ called $\mathscr{C}$. Concatenate vertically into $\mathscr{C}$ each of the candidate row subsets produced by $\mathscr{A}$. The result is a strength four ternary CA of degree k with optimal N in our trinomial coefficients representation.

Theorem 5. *$CA(\frac{4.5(k-1)^3 - 4.5(k-1)^2}{3}; 5, k, 3)$ is directly derived using trinomial coefficients for any $k \geq 7$.*

Proof. Let $\mathscr{A} = \{\binom{k}{k-2,0,2}, \binom{k}{0,2,k-2}, \binom{k}{2,k-2,0}, \binom{k}{k-3,2,1}, \binom{k}{2,1,k-3}, \binom{k}{1,k-3,2}\}$ and any $k \geq 7$. Construct an array of size $(\frac{4.5(k-1)^3 - 4.5(k-1)^2}{3}) \times k$ called $\mathscr{C}$. Concatenate vertically into $\mathscr{C}$ each of the candidate row subsets produced by $\mathscr{A}$. The result is a strength five ternary CA of degree k with optimal N in our trinomial coefficients representation.

6 Conclusions

In this paper we presented a backtracking algorithm for the ternary CAC problem that makes use of trinomial coefficients as a representation of the search space. A B&B technique and partial testing CA were implemented for improving the performance in our algorithm.

Although the proposed algorithm in this research work takes much time using longer values of k and t, it proposes a new approach for modeling the search space. The proposed representation of search space based on trinomial coefficients was proved to be complete for constructing any ternary CA.

The results of our B&B algorithm were compared with reported results of the IPOG-F algorithm. Although most of the cases IPOG-F offered better N values, we can obtain better ternary CAs for any $2 \leq t \leq 6$ and any $k \leq t + 2$.

According to our experimentation, we conclude that the use of our representation of trinomial coefficients can lead to good ternary CAs for any strength t and any $k \leq t + 2$.

The analysis over the trinomial coefficient usage for the performed constructions using our B&B algorithm derived new direct constructions that uses specific trinomial coefficients for strengths 2 to 5. These direct constructions requires polynomial time and space thus there is no need to search those ternary CAs with our B&B algorithm.

We believe that our representation of trinomial coefficients for the construction of ternary CAs can lead to more general direct constructions than those reported in this paper. It is necessary to do more experimentation for higher strengths and a deeper analysis in the structure of the trinomial coefficients requiered for the generated ternary CAs.

Acknowledgements. This research work was partially funded by the following projects: CONACyT 58554-Cálculo de Covering Arrays and 51623-Fondo Mixto CONACyT y Gobierno del Estado de Tamaulipas.

References

1. Hedayat, A., Sloane, N.J.A., Stufken, J.: Orthogonal Arrays, Theory and Applications. Springer-Verlag, New York, US (1999)
2. Sloane, N.: Covering arrays and intersecting codes. Journal of Combinatorial Designs **1** (1993) 51–63
3. Kuhn, R., Lei, Y., Kacker, R.: Practical combinatorial testing: Beyond pairwise. IT Professional **10** (2008) 19–23
4. Ji, L., Yin, J.: Constructions of new orthogonal arrays and covering arrays of strength three. Journal of Combinatorial Theory, Series A **117** (2010) 236–247
5. Colbourn, C.: Combinatorial aspects of covering arrays. Le Matematiche (Catania) **58** (2004) 121–167
6. Colbourn, C.: Covering array tables. `http://www.public.asu.edu/~ccolbou/src/tabby/catable.html` (accessed Jun 1, 2010)

7. Torres-Jimenez, J.: Covering arrays repository. `http://www.tamps.cinvestav.mx/\~jtj/CA.php` (accessed Jun 1, 2010)
8. Kreher, D.L., Stinson, D.R.: Combinatorial algorithms: generation, enumeration, and search. SIGACT News **30** (1999) 33–35
9. Bracho-Rios, J., Torres-Jimenez, J., Rodriguez-Tello, E.: A new backtracking algorithm for constructing binary covering arrays of variable strength. In: MICAI. (2009) 397–407
10. Keeney, R.: On the trinomial coefficients. Mathematics Magazine **42** (1969) 210–212
11. Bush, K.: Orthogonal arrays of index unity. Annals of Mathematical Statistics **23(3)** (1952) 426–434
12. Colbourn, C., Dinitz, J.: The CRC Handbook of Combinatorial Designs. CRC press, Boca Raton (1996)
13. Forbes, M., Lawrence, J., Lei, Y., Kacker, R.N., Kuhn, D.R.: Refining the in-parameter-order strategy for constructing covering arrays. Journal of Research of the National Institute of Standards and Technology **213** (2008) 287–297
14. Cohen, M.B., Colbourn, C.J., Ling, A.C.H.: Augmenting simulated annealing to build interaction test suites. In: ISSRE. (2003) 394–405
15. Lopez-Escogido, D., Torres-Jimenez, J., Rodriguez-Tello, E., Rangel-Valdez, N.: Strength two covering arrays construction using a sat representation. In: MICAI. (2008) 44–53
16. Nurmela, K.J.: Upper bounds for covering arrays by tabu search. Discrete Appl. Math. **138** (2004) 143–152
17. Shiba, T., Tsuchiya, T., Kikuno, T.: Using artificial life techniques to generate test cases for combinatorial testing. In: COMPSAC '04: Proceedings of the 28th Annual International Computer Software and Applications Conference, Washington, DC, USA, IEEE Computer Society (2004) 72–77
18. Martirosyan, S., van Trung, T.: On t-covering arrays. Des. Codes Cryptography **32** (2004) 323–339
19. Hartman, A., Raskin, L.: Problems and algorithms for covering arrays. Discrete Mathematics **284** (2004) 149–156
20. Meagher, K., Stevens, B.: Group construction of covering arrays. Journal of Combinatorial Designs **13** (2005) 70–77
21. Colbourn, C.J.: Covering arrays from cyclotomy. Designs, Codes and Cryptography **55** (2010) 201–219
22. Yan, J., Zhang, J.: A backtracking search tool for constructing combinatorial test suites. J. Syst. Softw. **81** (2008) 1681–1693
23. Graham, R.L., Knuth, D.E., Patashnik, O.: Concrete Mathematics: A Foundation for Computer Science. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1994)