Electronic Research Archive

https://www.aimspress.com/journal/era

ERA, 33(5): 3328–3347.
DOI: 10.3934/era.2025147
Received: 31 December 2024
Revised: 13 May 2025
Accepted: 23 May 2025
Published: 28 May 2025

Research article

Detecting arrays on graphs

Chengmin Wang[1], Ce Shi[2,*], Tatsuhiro Tsuchiya[3] and Quanrui Zhang[4]

[1] Department of Mathematics, Taizhou University, Taizhou 225300, China
[2] School of Statistics and Mathematics, Shanghai Lixin University of Accounting and Finance, Shanghai 201209, China
[3] Graduate School of Information Science and Technology, Osaka University, Suita 565-0871, Japan
[4] School of Information Science and Technology, Yunnan Normal University, Kunming 650500, China

* **Correspondence:** Email: shice060@lixin.edu.cn.

Abstract: Covering arrays on graphs can be used to generate test suites in component-based systems but they cannot identify and determine faulty interactions from the outcome of the test. To address this problem, the notion of detecting arrays on graphs (DAGs) was proposed in this paper. Then, the equivalence and the existence of DAGs were intensively studied. We established a general criterion for measuring the optimality of DAGs in terms of their size. Based on this optimality criterion, the equivalence between optimal DAGs and orthogonal arrays with prescribed properties was established. With this equivalence property, a great number of optimal detecting arrays on cycles were produced by constructing the equivalent combinatorial configurations. In particular, the existence of optimal detecting arrays on cycles with few vertices was almost completely determined.

Keywords: combinatorial testing; interaction faults; detecting arrays on graphs; orthogonal arrays on graphs; optimality; M-sequence

1. Introduction

In the past one hundred years, combinatorial design has developed very rapidly as a branch of mathematics. There are both rich theoretical achievements and many application fields. By now, it is well-known that the combination design is a useful study subject, which has wide applications in different areas, including computer networks, cryptography, experiment designs, optical communications, and so on [1, 2]. For example, combinatorial testing is an efficient testing strategy for software systems. It has the capability of detecting failures resulting from interactions among components [3]. This paper will explore a set of combinatorial configurations with significant applications in combinatorial testing.

Throughout this paper, the set $\{1, 2, \cdots, n\}$ is denoted by I_n. A covering array (CA) or orthogonal array (OA) is an $N \times k$ array with entries from the set $V = \{0, 1, \cdots, v-1\}$ that satisfies the condition where each $N \times t$ subarray covers each t-tuple in V^t at least (or exactly) once among its rows. We represent it as $\mathrm{CA}(N; t, k, v)$ (or $\mathrm{OA}(N; t, k, v)$). The notation $\mathrm{CA}(N; k, v)$ (or $\mathrm{OA}(k, v)$) is commonly used when $t = 2$.

CAs are mathematical objects that can be used as the test suites in combinatorial testing. Typically, an input of the integer t is required to ensure the testing of all interactions on t parameter values (i.e., t-way interactions) at least once, as opposed to considering all possible combinations of system parameters. We frequently employ CAs to generate test suites. Many empirical results demonstrate a substantial reduction in testing workload in practical scenarios. CAs have undergone extensive research, resulting in the documentation of a diverse range of methods and results. For more information, readers are encouraged to refer to the following references: Colbourn [4] on covering array tables of strength two, Chateauneuf et al. [5–7] on covering arrays of strength three and four, Hartman et al. [8, 9] on algorithms for covering arrays, and Tzanakis et al. [10] on special constructions from m-sequence.

Example 1.1. *A system under test (SUT) model for electronic commerce systems (ECS) is shown in Table 1. The ECS has five components: Client, OS, Web Server, Payment, and Database, each of which can assume one of three distinct values. An exhaustive testing would necessitate conducting* $3^5 = 243$ *tests. A* $CA(11; 2, 5, 3)$ *is available on Colbourn's web site [11], which has minimum size and provides a test suite for the ECS. The CA signifies a test suite comprising* 11 *tests, ensuring that every pair of components appears at least once in the array.*

Table 1. Configuration Parameters for ECS.

	Parameter		Values	
F_1	Client	Opera (0)	IE (1)	Firefox (2)
F_2	OS	Win (0)	Linux (1)	Mac (2)
F_3	Web Server	WebSphere (0)	Apache (1)	Nginx (2)
F_4	Payment	MasterCard (0)	Visa (1)	UnionPay (2)
F_5	Database	Db2 (0)	Oracle (1)	MySQL (2)

As illustrated above, the purpose of using CAs as test suites is to systematically test the interactions between every subset of t components. As stated in [12], if certain pairs of components don't have interactions, we may test the ECS with fewer than 11 tests. Specific graph structures, where each component corresponds to a vertex and interacting components are connected by edges, can be employed to enhance efficiency in various applications. To address this, the notion of covering arrays on graphs was introduced. Actually, there are a lot of studies on the cross-structures and their applications of combinatorial designs and graphs, such as mutually orthogonal graph squares [13], graph-orthogonal arrays, and so on [14].

Definition 1.1. *(Covering arrays on graphs) A covering array on a graph G is an* $N \times k$ *array with entries from* $\mathbb{Z}_v$*, where* $k = |V(G)|$*. Each column in the array corresponds to a vertex in the graph G.*

The array has the property that all the pairs of the two columns, which correspond to adjacent vertices in the graph, cover each 2-tuple in $\mathbb{Z}_v^2$ at least λ times. It is denoted by $CA_\lambda(N; G, v)$. When $\lambda = 1$, we omit the subscript.

Example 1.2. *It is known that, in an ECS, some components do not affect each other. Figure 1 displays an interactions graph G where some vertices are not adjacent. For example, there are no edge between F_4 and F_5, which means that the component (Payment) and the component (Database) have no interaction. Table 2 gives a 2-way CA on G. Clearly, the size 9 is strictly smaller than the size 11 of optimal $CA(11; 2, 5, 3)$.*

A $\text{CA}(N; G, v)$ improves efficiency in many applications over a $\text{CA}(N; k, v)$. Therefore, it is valuable to explore covering arrays within the context of graphs, a concept initially introduced by Meagher and Stevens [12]. Subsequently, mixed covering arrays on graphs were also defined in [15]. A thorough examination of covering arrays on product graphs was also conducted. For the configuration of different graphs and related topics, the reader may refer to [16].

Just like $\text{CA}(N; k, v)$, a $\text{CA}(N; G, v)$ has the capability of producing test suites that reveal the existence or absence of faulty interactions for combinatorial testing of interaction factors. However, it lacks the ability to identify and precisely determine these interactions solely based on test outcomes. For instance, if all the tests in Table 2 were executed, and all except the fourth one passed, the failed test would involve six 2-way interactions: (IE, Win), (IE, Apache), (IE, Visa), (Win, Apache), (Win, Visa) and (Apache, MySQL). Each of these interactions cannot be safely excluded from consideration. Therefore, it becomes impossible to determine which specific interaction or interactions triggered the failure.

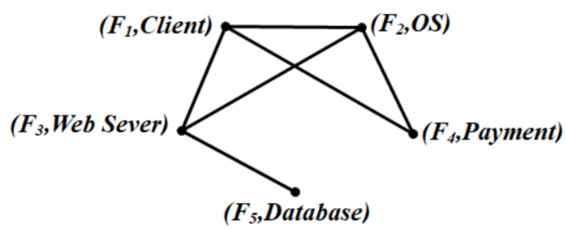

Figure 1. An interaction graph G for ECS.

At this point, conducting tests to pinpoint the location of interaction faults becomes both intriguing and meaningful. To address this issue, we will introduce the concept of detecting arrays on graphs, which is a generalization of detecting arrays (DA) with strength 2. The notion of DAs was introduced by Colbourn and McClary in their investigation of identifying faults among factors in a component-based system [17]. However, the absence of mutual interactions among components is not considered in DAs. Here, a graph structure is considered to capture the relationships between factors. We will formally present the concept of detecting arrays on graphs (DAGs) in this paper. To assess the optimality of DAGs, a general criterion is introduced based on their size. Through this optimality criterion, a connection is established between optimal DAGs and orthogonal arrays possessing special properties. Leveraging this equivalence property, a considerable number of optimal DAs on cycles are constructed by generating equivalent combinatorial configurations. Notably, the existence of optimal DAs on cycles with few vertices is almost entirely determined.

The subsequent sections of this paper are organized as follows. In Section 2, the concept of DAGs is formally introduced. This section also includes an application example. Section 3 delves into a general criterion for assessing the optimality of DAGs, emphasizing their size. Notably, this section establishes the equivalence between optimal DAGs and orthogonal arrays with specified properties. Building upon this equivalence, Section 4 presents various constructions and results regarding the existence of DAs on cycles. The paper concludes with Section 5, where final remarks are provided.

Table 2. A CA$(9; G, 3)$.

Test Case	F_1	F_2	F_3	F_4	F_5
1	Opera	Win	WebSphere	MasterCard	Oracle
2	Opera	Linux	Apache	UnionPay	Oracle
3	Opera	Mac	Nginx	Visa	Oracle
4	IE	Win	Apache	Visa	MySQL
5	IE	Linux	Nginx	MasterCard	MySQL
6	IE	Mac	WebSphere	UnionPay	MySQL
7	Firefox	Win	Nginx	UnionPay	Db2
8	Firefox	Linux	WebSphere	Visa	Db2
9	Firefox	Mac	Apache	MasterCard	Db2

2. Preliminaries

2.1. Definitions and notations

Consider an $N \times k$ array $A = (a_{ij})$ with entries from $\mathbb{Z}_v$ where $i \in I_N$ and $j \in I_k$. For any subset $\{j_1, j_2, \cdots, j_t\}$ of column indices in A with $j_1 < j_2 < \cdots < j_t$, the set $T = \{(j_r, x_r) : x_r \in V, 1 \leq r \leq t\}$ is defined as a *t-way interaction*. The set $\rho(A, T)$ represents the indices of rows in A where T is covered, expressed as: $\rho(A, T) = \{i : a_{ij_r} = x_r, 1 \leq r \leq t\}$.

For an arbitrary set $\mathcal{T}$ of interactions, the notation $\rho(A, \mathcal{T}) = \cup_{T \in \mathcal{T}} \rho(A, T)$ is used. Let $\mathcal{I}_t$ denote the set of all t-way interactions of A. For any $\mathcal{T} \subseteq \mathcal{I}_t$ with $|\mathcal{T}| = d$ and any $T \in \mathcal{I}_t$, the condition

$$\rho(A, T) \subseteq \rho(A, \mathcal{T}) \Leftrightarrow T \in \mathcal{T},$$

holds, then the array A is termed a (d, t)-DA, denoted as (d, t)-DA$(N; k, v)$.

Let $G = (V, E)$ be a simple graph in what follows. Suppose that $A = (a_{ij})_{N \times k}$ is a CA$(N; G, v)$. A 2-way interaction $T = \{(i, x_i), (j, x_j) : x_i, x_j \in \mathbb{Z}_v\}$ of A is called *valid*, if vertices i and j, which correspond to the factors i and j, are adjacent in G. Write $V\mathcal{I}_2$ for the set of all valid 2-way interactions of A. It is easy to know $|V\mathcal{I}_2| = |E|v^2$, where $|E|$ is the number of edges in G. To locate and detect interaction faults between the factors, it is only necessary to identify and determine the valid interaction faults from the outcome of the tests. Hence, the notion of DAGs is ready to be proposed as follows.

Definition 2.1. *(DAGs) Let G be a simple graph. An $N \times k$ array $A = (a_{ij})$ $(i \in I_N, j \in I_k)$ with entries from a set with v symbols is called a DAG G, if $\rho(A, T) \subseteq \rho(A, \mathcal{T}) \Leftrightarrow T \in \mathcal{T}$, for any $\mathcal{T} \subseteq V\mathcal{I}_2$ with $|\mathcal{T}| = d$ and any $T \in V\mathcal{I}_2$. It is denoted by DA$(N; d, G, v)$.*

It is noteworthy that an array with empty rows is of no practical value in software testing applications. Henceforth, in our subsequent definitions and usages, we will implicitly exclude such instances without additional explanation. Obviously, $T \in \mathcal{T}$ implies $\rho(A, T) \subseteq \rho(A, \mathcal{T})$. Hence, it is straightforward that $\rho(A, T) \subseteq \rho(A, \mathcal{T}) \Leftrightarrow T \in \mathcal{T}$ can be deduced from $T \notin \mathcal{T} \Rightarrow \rho(A, T) \nsubseteq \rho(A, \mathcal{T})$. We will use this simple fact without mentioning in what follows. When $G = K_k$, a DA$(N; d, G, v)$ is actually a $(d, 2)$-DA$(N; k, v)$. Thus, DAGs can be viewed as a generalization of DAs with strength 2. Setting $d = 0$ in the definition, since $\rho(A, \mathcal{T}) = \emptyset, \rho(A, T) \neq \emptyset$ for any $T \in \mathcal{VI}_2$. Then a DA$(N; 0, G, v)$ is an array in which each valid 2-way interaction is covered in at least one row. This leads to the notion of covering arrays on graphs, which was mentioned earlier. This simple fact tells us that a CA$(N; G, v)$ is equivalent to a DA$(N; 0, G, v)$.

As a special case, consecutive DAs of strength 2 were studied by the second author and coauthors [18]. They are equivalent to DAs on paths and can be used to locate and determine the adjacent interactions among factors. In this paper, we present a general definition and will then systematically study its optimality criterion, equivalence, and more constructions.

Similar to DAs, there are some admissible parameters for the existence of DAs on $G = (V, E)$. When $|V| = 1$ and $d > 0$, or $d < 0$, no DA$(N; d, G, v)$'s exist. If $|V| = 2$, or $V > 2$ and $|E| = 1$, we can form an array consisting of all valid 2-tuples. Hence, we only treat the cases with $|V| > 2, |E| \geq 2$, and $d > 0$. The following results for DAs were given in [17].

Lemma 2.1. *Suppose that A is a (d, t)-DA$(N; k, v)$. Then*

1) $d < v$.
2) *A is also an (s, t)-DA$(N; k, v)$, where* $0 \leq s \leq d - 1$.

As an immediate outcome of the characteristics of DAs, the following lemma can be easily derived. We provide it for future reference.

Lemma 2.2. *Assuming A is a DA$(N; d, G, v)$, the following observations hold:*

1) $d < v$.
2) *A is also a DA$(N; s, G, v)$ with* $0 \leq s \leq d - 1$.

According to the definition, a DA$(N; d, G, v)$ essentially is a distinctive category of covering arrays on graphs. The significance of utilizing DAs on graphs to generate test suites lies in their ability to precisely identify any set of d valid 2-way interaction faults from the test outcomes. In addition, if the number of faults caused by the valid 2-way interactions is more than d, this fact can be detected. For more detailed information, the reader may refer to the practical application of DAs outlined in [17]. Since the rows of a DA on a graph correspond to the number of tests, the DAs on graphs with the smallest size, while keeping other parameters constant, are of significant interest. The minimum N for which a DA$(N; d, G, v)$ exists is named as *detecting arrays number on graphs* (DAN on graphs), denoted by DAN(d, G, v). A DA$(N; d, G, v)$ with $N =$ DAN(d, G, v) is called *optimal*. In the upcoming section, we establish a lower bound for the function DAN(d, G, v) with a given G and provide a combinatorial illustration for optimal DAGs.

2.2. An application example

A key application of DAs on graphs is in the testing of information systems. For instance, consider the scenario of testing a network game software system, and assume we have successfully extracted factors and their values using *test parameter analysis* [3] for testing as shown in Table 3. A test in this example is a tuple of size $k = 4$, yielding a total of $2 \times 2 \times 2 \times 2 = 16$ possible tests.

Table 3. Factors and values of a network game software system.

Factor	Values
F_1 Web browser	IE(0), Firefox(1)
F_2 OS	Win(0), Linux(1)
F_3 Access	5G(0), WiFi(1)
F_4 Audio	Surround(0), Stereo(1)

Table 4 presents a test suite comprising eight selected tests from the total possibilities. This test set corresponds to a DA$(8; 1, G, 2)$, where G is defined as $G = (V, E)$ such that $V = \{F_1, F_2, F_3, F_4\}$ and $E = \{\{F_1, F_2\}, \{F_1, F_3\}, \{F_1, F_4\}, \{F_2, F_3\}, \{F_3, F_4\}\}$.

Table 4. Test sets corresponding to DAGs.

	F_1: **Browser**	F_2: **OS**	F_3: **Access**	F_4: **Audio**
1	IE	Win	5G	Surround
2	IE	Win	WiFi	Stereo
3	IE	Linux	5G	Stereo
4	IE	Linux	WiFi	Surround
5	Firefox	Win	5G	Stereo
6	Firefox	Win	WiFi	Surround
7	Firefox	Linux	5G	Surround
8	Firefox	Linux	WiFi	Stereo

To provide a more concrete illustration of the practical applicability of DAGs, we consider a simulated fault localization scenario based on the test suite in Table 4. Suppose that only tests 1 and 5 result in failures, while all other tests yield passing outcomes. By inspecting the interactions covered by these two tests, we observe that the interaction $(F_2, F_3) = (\text{Win}, \text{5G})$, corresponding to the tuple $T = \{(2, 0), (3, 0)\}$, is the only strength-2 interaction common to both failing tests. Let $\mathcal{T} = \{T\}$ denote the (hypothetical) set of faulty interactions. Then, the set of rows covering $\mathcal{T}$ is $\rho(A, \mathcal{T}) = \{1, 5\}$. Due to the defining property of the DA on graph G, there exists no other subset $\mathcal{T}' \subseteq V\mathcal{I}_2$ with $|\mathcal{T}'| = 1$ such that $\rho(A, \mathcal{T}') = \{1, 5\}$. Hence, T is uniquely identifiable as the underlying fault. This demonstrates the fault localization capability inherent in the proposed DA$(8; 1, G, 2)$. It is worth noting that this level of diagnostic precision is not generally guaranteed by traditional covering arrays, which may suffice for fault detection but not localization. Moreover, the use of graph-structured interaction constraints, encoded in G, enables a reduction in test suite size without compromising identifiability. In the given example, the full factorial design would require $2^4 = 16$ tests, while the graph-based DA achieves localization with only 8 tests. This example serves as an empirical validation of the proposed framework, demonstrating both its efficiency in test suite size and its effectiveness in interaction fault localization. Such properties are particularly valuable in practical testing environments, such as software systems with heterogeneous configurations, where reducing test cost while ensuring diagnostic accuracy is essential.

Another prevalent application of *screening experiments* involves identifying interactions that exert the most significant impact on the response of a complex system, distinct from information system testing. In contrast to exhaustive full-factorial designs, utilizing DAs (on graphs) as experimental designs is able to greatly decrease the number of design points; thus, the cost of experiments is reduced [17].

3. Lower bounds and combinatorial description

Let us consider a simple graph $G = (V, E)$ with $|V| = k > 2$ and $|E| \geq 2$, ensuring the absence of isolated vertices in G. The main objective of this section is to establish a lower bound for the function $\mathrm{DAN}(d, G, v)$ and investigate the combinatorial features of optimal DAs on graphs that achieve this lower bound. To begin with, we set a standard for evaluating the optimality of a $\mathrm{DA}(N; d, G, v)$. The following result can be derived through a proof akin to Lemma 2.1 in [19].

Lemma 3.1. *Suppose that A is a $DA(N; 1, G, v)$. Then, $|\rho(A, T)| \geq 2$ for any valid 2-way interaction T.*

Proof. A $\mathrm{DA}(N; 1, G, v)$ is also a $\mathrm{DA}(N; 0, G, v)$ by Lemma 2.2. As mentioned in the subsequent paragraph of Definition 2.1, a $\mathrm{DA}(N; 0, G, v)$ is a $\mathrm{CA}(N; G, v)$. Thus, we have $|\rho(A, T)| \geq 1$ for any valid 2-way interaction T. Therefore, it suffices to show $|\rho(A, T)| \neq 1$. Suppose that $|\rho(A, T)| = 1$ for some T and $(x_1, x_2, \ldots, x_k)$ is the unique row of A that covers T. This row also covers at least one valid 2-way interaction T' other than T under the assumption $|V| > 2, |E| \geq 2$ and there are no isolated vertices in G. Therefore, A is not a $\mathrm{DA}(N; 1, G, v)$ since $\rho(A, T) \subseteq \rho(A, T')$. □

The following lemma can be viewed as a generalization of Lemma 3.1. The proof is similar to that for Lemma 2.2 in [20]. We only substitute t-way interactions with valid 2-way interactions, leveraging the information provided in Lemma 2.2 of this paper.

Lemma 3.2. *Suppose that A is a $DA(N; d, G, v)$. Then, $|\rho(A, T)| \geq d + 1$ for any valid 2-way interaction T.*

Through the utilization of Lemma 3.2, we establish a lower bound for the function $\mathrm{DAN}(d, G, v)$. This lower bound acts as our reference point for evaluating the optimality of DAs on graphs.

Theorem 3.3. *Let $G = (V, E)$ be a simple graph with $|V| = k > 2$ and $|E| \geq 2$. Then,*

$$DAN\,(d, G, v) \geq (d + 1)v^2.$$

Proof. Suppose that A is a $\mathrm{DA}(N; d, G, v)$ over V with $N = \mathrm{DAN}(d, G, v)$. Then for any fixed 2 columns $\{j_1, j_2\}$, satisfying that j_1 and j_2 as vertices of G are adjacent, there exist exactly v^2 2-way interactions of A: $\{(j_r, x_r) : 1 \leq r \leq 2\}$ with (x_1, x_2) runs over v^2 2-tuples from V^2. According to Lemma 3.2, it is established that $|\rho(A, T)| \geq d + 1$ for any 2-way interaction T within A. Therefore, we can deduce that $N = \mathrm{DAN}(d, G, v) \geq (d + 1)v^2$. □

We try to produce a $\mathrm{DA}(N; d, G, v)$ with $N = (d + 1)v^2$ as *optimal*. It is noteworthy that optimal DAs on graphs find practical applications in software testing due to their minimal row count. To delve into the combinatorial characteristics of optimal DAs on graphs, the concept of simple OAs on graphs will be introduced. An orthogonal array on a graph G is a $\mathrm{CA}(N; G, v)$ with $N = \lambda v^2$, denoted by $\mathrm{OA}_\lambda(G, v)$. An $\mathrm{OA}_\lambda(G, v)$ is considered *simple* if, for any $\lambda v^2 \times (4-i)$ subarray consisting of the columns corresponding to the vertices of any two edges E and E' with $|V(E) \cap V(E')| = i$ (where $i = 0, 1$), each $(4 - i)$-tuple appears at most once. It is evident from this definition that a simple $\mathrm{OA}_\lambda(G, v)$ can only exist when $\lambda \leq v$.

Example 3.1. *The array obtained by transposing the following is a simple $OA_2(G, 3)$ over $\mathbb{Z}_3$, where G is the same as the graph in Figure* 1.

F_1	0 0 0 0 0 0 1 1 1 1 1 1 2 2 2 2 2 2
F_2	0 0 1 1 2 2 1 1 2 2 0 0 2 2 0 0 1 1
F_3	0 1 2 0 1 2 1 2 0 1 2 0 2 0 1 2 0 1
F_4	0 2 2 1 0 1 1 0 0 2 1 2 2 1 1 0 2 0
F_5	1 0 0 2 1 2 2 1 1 0 2 0 0 2 2 1 0 1

It is easy to check that it is an $OA_2(G, 3)$ over $\mathbb{Z}_3$. Any 4-tuple over $\mathbb{Z}_3$ from two disjoint edges occurs at most once. In addition, for any two edges with one vertex in common, each 3-tuple appears at most once.

Table 5. A DA$(18; 1, G, 3)$ for the ECS in Figure 1.

Test Case	Client	OS	Web Server	Payment	Database
1	Opera	Win	WebSphere	MasterCard	Oracle
2	Opera	Win	Apache	UnionPay	Db2
3	Opera	Linux	Nginx	UnionPay	Db2
4	Opera	Linux	WebSphere	Visa	MySQL
5	Opera	Mac	Apache	MasterCard	Oracle
6	Opera	Mac	Nginx	Visa	MySQL
7	IE	Linux	Apache	Visa	MySQL
8	IE	Linux	Nginx	MasterCard	Oracle
9	IE	Mac	WebSphere	MasterCard	Oracle
10	IE	Mac	Apache	UnionPay	Db2
11	IE	Win	Nginx	Visa	MySQL
12	IE	Win	WebSphere	UnionPay	Db2
13	Firefox	Mac	Nginx	UnionPay	Db2
14	Firefox	Mac	WebSphere	Visa	MySQL
15	Firefox	Win	Apache	Visa	MySQL
16	Firefox	Win	Nginx	MasterCard	Oracle
17	Firefox	Linux	WebSphere	UnionPay	Db2
18	Firefox	Linux	Apache	MasterCard	Oracle

The DA$(18; 1, G, 3)$ in Example 3.1 provides test cases for the ECS in Example 1.1, as illustrated in Table 5.

We observe that Theorem 3.3 provides a comprehensive criterion for assessing the optimality of DAGs based on their size. This criterion enables us to establish an equivalence between optimal DAGs and a specific type of orthogonal arrays with predefined properties. Consequently, the challenge of constructing optimal DAs is effectively transformed into the task of constructing optimal orthogonal arrays.

Theorem 3.4. *Suppose that G is a simple graph. Then, a simple $OA_{d+1}(G, v)$ is equivalent to an optimal $DA((d + 1)v^2; d, G, v)$.*

Proof. (⇒) Assuming A is a simple $\mathrm{OA}_{d+1}(G, v)$ on G, consider an arbitrary valid 2-way interaction T in A with $T \notin \mathcal{T}$, where $\mathcal{T} = \{T_1, T_2, \ldots, T_d\}$. The objective is to demonstrate that $\rho(A, T) \nsubseteq \rho(A, \mathcal{T})$. Given that A is an $\mathrm{OA}_{d+1}(G, v)$, it follows that $|\rho(A, T)| = d + 1$. If $\rho(A, T) \subseteq \rho(A, \mathcal{T})$, there must be at least one $T_j \in \mathcal{T}$ such that $|\rho(A, T_j) \cap \rho(A, T)| \geq 2$. Let us assume the column indices of T and T_j have i columns in common, where $i = 0, 1$. Since $T \neq T_j$, there exists a certain $(4 - i)$-tuple that occurs at least twice. This contradicts the fundamental simplicity property of A.

(⇐) Let B be a $\mathrm{DA}((d + 1)v^2; d, G, v)$. By Lemma 3.2, it is known that $|\rho(A, T)| = d + 1$ for any valid 2-way interaction T, as B contains precisely $(d + 1)v^2$ rows. This implies that each 2-tuple occurs as a row exactly $(d + 1)$ times in any two columns of B corresponding to the edges of G. Thus, B is an $\mathrm{OA}_{d+1}(G, v)$. In addition, B is simple. This can be deduced from the definition of a DA on a graph. □

Theorem 3.4 establishes a one-to-one correspondence between simple orthogonal arrays on graphs and optimal DAs constrained by the same graph structure. This result generalizes the well-known equivalence between classical orthogonal arrays and DAs in the unconstrained setting. In particular, when the underlying graph G is the complete graph on k vertices, Theorem 3.4 reduces to the classical result stating that a simple orthogonal array $\mathrm{OA}_{d+1}(k, v)$ yields an optimal $\mathrm{DA}((d + 1)v^2; d, k, v)$, as noted in previous literature [20]. Therefore, our result can be seen as a natural generalization to the case where the factor interactions are restricted by an arbitrary simple graph G. The significance of Theorem 3.4 lies in its theoretical and practical implications. From a theoretical perspective, it bridges two important combinatorial structures under a unified graph-based framework, thereby extending the applicability of classical design theory to more structured systems. From a practical viewpoint, the equivalence provides a constructive method to obtain optimal graph-constrained DAs using known constructions of orthogonal arrays. This is particularly valuable in application domains where interactions among system parameters are not fully pairwise, but governed by a sparse dependency structure.

4. Optimal DAs on cycles

In this section, we will construct a large number of optimal DAs on cycles in terms of simple OAs on cycles using the equivalent characterization described in Theorem 3.4. The combinatorial features of the simple OAs on cycles will be explored precisely. Cyclic consecutive orthogonal arrays (CCOAs)), denoted as a $\mathrm{CCOA}_\lambda(k, v)$, are $N \times k$ arrays with entries from a set with v symbols such that two consecutive columns on the cycle contain each 2-tuple exactly λ times among its rows. Generally, when N is relatively large, the $N \times k$ array is written in transposed form; otherwise, it is not. The notion of CCOAs with $\lambda = 1$ and arbitrary strength was proposed by the second author's research on neighbor combinatorial testing [21]. Such CCOAs can only be used to detect interaction faults but cannot locate or determine them. For faults localization, we require CCOA with $\lambda > 2$ and distinctive properties. Also, $t = 2$ is only required here. A $\mathrm{CCOA}_\lambda(k, v)$ is called *simple*, if any $\lambda v^2 \times (4 - i)$ subarray consisting of two cyclically consecutive columns, sharing i columns in common, contains each $(4 - i)$-tuple at most once, where $i = 0, 1$.

Example 4.1. *The following array represents a simple $CCOA_2(5, 2)$ over $\mathbb{Z}_2$.*

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

Demonstrating that a simple $\mathrm{CCOA}_\lambda(k, v)$ is equivalent to a simple $\mathrm{OA}_\lambda(G, v)$, where G is a cycle with k vertices, is straightforward. This equivalence is formally stated in the following theorem.

Theorem 4.1. *Let $G = (V, E)$ be a cycle with $|V| = k > 2$. Then a simple $CCOA_{d+1}(k, v)$ is equivalent to an optimal $DA((d + 1)v^2; d, G, v)$.*

To construct simple CCOAs, we leverage the configuration of super-simple orthogonal arrays (SSOA). An $\mathrm{OA}_\lambda(t, k, v)$ is deemed *super-simple* if any $(t + 1)$ columns of the array contain every $(t + 1)$-tuple of symbols as a row at most once. This specific orthogonal array is denoted as $\mathrm{SSOA}_\lambda(t, k, v)$. It is noteworthy that an $\mathrm{SSOA}_\lambda(2, k, v)$ qualifies as a simple $\mathrm{CCOA}_\lambda(k, v)$, but not vice versa.

4.1. Optimal DAs on cycles from orthogonal arrays

OA is an important research object first introduced by Rao [22]. OAs find broad applications in statistics, computer science, coding theory, and cryptography. Over the last five decades, there has been extensive research on orthogonal arrays in the literature, and a few methods and results are available in the monograph by Hedayat, Sloane, and Stufken [23]. In the following, we introduce some OAs of strength 3 for later use.

Lemma 4.2. [24] *If $q > 3$ is a prime power, the existence of an orthogonal array $OA(3, q + 1, q)$ is guaranteed. Additionally, if $q \geq 4$ is a power of 2, an orthogonal array $OA(3, q + 2, q)$ exists.*

Lemma 4.3. *Assume that $v = q_1 q_2 \cdots q_s$ represents the standard factorization of v into distinct prime powers. If $q_i > 3$ for any i, then there exists an $OA(3, k + 1, v)$, where $k = \min\{q_i : 1 \leq i \leq s\}$.*

If $k = 6$, Ji and Yin provided the following insightful result.

Lemma 4.4. [6] *Suppose v is a positive integer such that $gcd(v, 4) \neq 2$ and $gcd(v, 18) \neq 3$. Then, there exists an $OA(3, 6, v)$, and an $OA(3, 6, 3u)$ with $u \in \{5, 7\}$ also exists.*

Subsequently, we will employ OAs to create simple CCOAs. This procedure can be conceptualized as a modification of Construction 3.7 outlined in [20].

Construction 4.5. *Let $k \geq 5$ be an odd integer. If there exists an $OA(3, k + 1, v)$, then a simple $CCOA_\lambda(ki, v)$ exists for $i = 1, 2, \cdots, \frac{\varphi(k)}{2}$ and any integer $\lambda \leq v$, where $\varphi(k)$ is the Euler function.*

Proof. If an OA$(3, k+1, v)$ exists, then an SSOA$_\lambda(2, k, v)$ exists for any integer $\lambda \le v$ by Construction 3.7 in [20]. We denote it by $A = (A_0, A_1, \cdots, A_{k-1})$. It is obvious that there are $\varphi(k)$ multiple inverse elements in $\mathbb{Z}_k$. We arrange such inverse elements from small to large and denote them by $a_1 = 1, a_2, \cdots, a_{\frac{\varphi(k)}{2}}, \cdots, a_{\varphi(k)}$. Clearly, the elements $a_1 = 1, a_2, \cdots, a_{\frac{\varphi(k)}{2}}$ have the property $a_i + a_j \not\equiv 0 \pmod{k}$, where $1 \le i \ne j \le \frac{\varphi(k)}{2}$.

Write $A' = (A_0, A_{a_1}, \cdots, A_{a_1\cdot(k-2)}, A_{a_1\cdot(k-1)}, \cdots, A_0, A_{a_i}, \cdots, A_{a_i\cdot(k-2)}, A_{a_i\cdot(k-1)})$, where the subscripts are under the operation $\pmod{k}$ and $i = 1, 2, \cdots, \frac{\varphi(k)}{2}$. It can be easily confirmed that A' satisfies the conditions of a simple CCOA. □

Construction 4.6. *Let k be an even integer. If an OA$(3, k+1, v)$ with $k \ge 6$ exists, then a simple CCOA$_\lambda(ki + \frac{k}{2}, v)$ exists for $i = 1, 2, \cdots, \frac{\varphi(k)}{2}$ and any integer $\lambda \le v$.*

Proof. The proof is similar to Construction 4.5. We only need to concatenate A' with $(A_0, A_2, \cdots, A_{k-2})$. □

Combining Theorem 3.4 with Constructions 4.5 and 4.6, along with the previously established OAs from the lemmas, we can derive an infinite series of optimal DAs on cycles.

Theorem 4.7. *Let $q \ge 5$ be a prime power and k be an odd integer with $3 \le k \le q$. Then, an optimal DA$((d+1)v^2; d, G, q)$ exists for any positive integers $d < q$, where G is a cycle with kk' vertices and $1 \le k' \le \frac{\varphi(k)}{2}$.*

Proof. By assumption and Lemma 4.2, we know that an OA$(3, q+1, q)$ exists. This implies that an SSOA$_\lambda(2, q, q)$ with $\lambda \le q$ exists. Thus, an SSOA$_\lambda(2, k, q)$ exists for any odd integer k with $3 \le k \le q$. Applying Theorem 4.1 and Constructions 4.5 produces the required DAs on cycles. □

Theorem 4.8. *Let $q \ge 5$ be a prime power and k be an even integer with $4 \le k \le q$. Then, an optimal DA$((d+1)v^2; d, G, q)$ exists for any positive integers $d < q$, where G is a cycle with $kk' + \frac{k}{2}$ vertices and $1 \le k' \le \frac{\varphi(k)}{2}$.*

Proof. The proof is similar to Theorem 4.7. We only replace Constructions 4.5 by Construction 4.6. □

Similarly, if q is a power of 2, we have the following DAs on cycles.

Theorem 4.9. *Let $q \ge 4$ be a power of 2 and k be an odd integer with $3 \le k \le q+1$. Then, an optimal DA$((d+1)v^2; d, G, q)$ exists for any positive integers $d < q$, where G is a cycle with kk' vertices and $1 \le k' \le \frac{\varphi(k)}{2}$.*

Theorem 4.10. *Let $q \ge 4$ be a power of 2 and k be an even integer with $4 \le k \le q+1$. Then, an optimal DA$((d+1)v^2; d, G, q)$ exists for any positive integers $d < q$, where G is a cycle with $kk' + \frac{k}{2}$ vertices and $1 \le k' \le \frac{\varphi(k)}{2}$.*

Theorem 4.11. *Suppose that $v = q_1 q_2 \cdots q_s$ is a standard factorization of v into distinct prime powers and $k = \min\{q_i : 1 \le i \le s\} \ge 3$. If $q_i > 3$ and $d + 1 \le v$, then there exists an optimal DA$((d+1)v^2; d, G, v)$, where G is a cycle with ii' vertices, $1 \le i' \le \frac{\varphi(i)}{2}$, and i is an odd integer with $3 \le i \le k$.*

Proof. By Lemma 4.3, an OA$(3, k+1, v)$ exists. Thus, it can produce an SSOA$_\lambda(2, i, v)$ for $3 \le i \le k$. Applying Construction 4.5 produces a simple CCOA$_{d+1}(2, ii', v)$ under the assumption $d + 1 \le v$. The desired DAs on cycles are obtained by Theorem 4.1. □

Theorem 4.12. *Suppose that $v = q_1q_2\cdots q_s$ is a standard factorization of v into distinct prime powers and $k = min\{q_i : 1 \le i \le s\} \ge 4$. If $q_i > 3$ and $d+1 \le v$, then there exists an optimal $DA((d+1)v^2; d, G, v)$, where G is a cycle with $ii' + \frac{\varphi(i)}{2}$ vertices, $1 \le i' \le \frac{\varphi(i)}{2}$, and i is an even integer with $4 \le i \le k$.*

Theorem 4.13. *Suppose that v is a positive integer, satisfying $gcd(v,4) \ne 2$ and $gcd(v,18) \ne 3$. Then, there exists an optimal $DA((d+1)v^2; d, G, v)$ for any positive integer d with $d+1 \le v$, where G is a cycle with 5 or 10 vertices.*

Proof. Apply Theorem 4.1. The required simple CCOAs are provided by Construction 4.5 and Lemma 4.4. □

4.2. A class of optimal DAs on cycles from m-sequences

Let $f = c_0 + c_1x + c_2x^2 + \cdots + c_{t-1}x^{t-1} + x^t \in F_q[x]$ and $I = (b_0, b_1, \cdots, b_{t-1}) \in F_q^t$. A *linear feedback shift register (LFSR)* sequence with characteristic polynomial f is defined as $S(f,I) = (a_0, a_1, \cdots,)$, where

$$a_i = \begin{cases} b_i, 0 \le i < t, \\ -c_{t-1}a_{i-1} - c_{t-2}a_{i-2} - \cdots - c_1a_{i-(t-1)} - c_oa_{i-t}, i \ge t. \end{cases}$$

A sequence produced by an LFSR over F_q through a primitive polynomial is known as a maximal (period) sequence. Such sequences are commonly denoted as m-sequences. Maximal sequences exhibit a complex algebraic structure and possess diverse statistical properties, making them well-suited for applications in areas like radar, sonar, and wireless communications [25]. Throughout this paper, the primitive polynomial over F_q with $deg f = 3$, i.e., $f(x) = c_0 + c_1x + c_2x^2 + x^3 \in F_q[x]$ is considered. For a given sequence $S = (a_0, a_1, \cdots)$ and a positive integer n, the subinterval of length n starting at position i is denoted as $C_i^n(S) = (a_i, a_{i+1}, \cdots, a_{i+n-1})$.

Cyclic shifts of maximal sequences have been employed in previous studies to generate (ordered) orthogonal and covering arrays [26,27]. Construction of simple consecutive orthogonal arrays (COAs) from m-sequences was given in [28]. In this subsection, we will construct simple CCOAs from m-sequences, which produce optimal DAs on cycles by Theorem 4.1. For the given m-sequence $S(f,I)$, consider the following $q^3 \times k$ arrays M, where $k = \frac{q^3-1}{q-1} = q^2 + q + 1$.

$$M = \begin{bmatrix} 0, 0, \cdots, 0 \\ C_0^k(S(f,I)) \\ C_1^k(S(f,I)) \\ \vdots \\ C_{q^3-2}^k(S(f,I)) \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ a_0 & a_1 & \cdots & a_{k-1} \\ a_1 & a_2 & \cdots & a_k \\ \vdots & \vdots & & \vdots \\ a_{q^3-2} & a_{q^3-1} & \cdots & a_{q^3-2+k-1} \end{bmatrix}.$$

Similar to the argument in Theorem 3 from [28], we can establish the following theorem.

Theorem 4.14. *If f is a primitive polynomial over F_q with $\deg f = 3$ and $I = (b_0, b_1, b_2) \in F_q^3 \backslash (0,0,0)$, then M forms a simple $CCOA_q(2, q^2 + q + 1, q)$.*

It is remarkable that m-sequence can be used to construct COA, which was presented in [28]. Despite using similar tools and proof methods, the definitions of CCOA and COA are distinct. In fact, CCOA must be a COA with strength 2, but the reverse may not hold true.

Example 4.2. *Let* $f = x^3 + 0x^2 + 2x + 1$ *be a primitive polynomial over* F_3 *and initial values* $I = (0, 0, 1)$*. An m-sequence generated by* f *and* T *is* $(00101211201110020212210222\cdots)$ *with the period* $3^3 - 1 = 26$*. An* 27×13 *array* M *is arranged below.*

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 2 & 1 & 1 & 2 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 2 & 1 & 1 & 2 & 0 & 1 & 1 & 1 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & 0 & 0 & 2 & 0 & 2 & 1 & 2 & 2 & 1 & 0 & 2 & 2 \\ 0 & 0 & 2 & 0 & 2 & 1 & 2 & 2 & 1 & 0 & 2 & 2 & 2 \end{bmatrix}.$$

It is evident that each 2*-tuple, considering any two cyclic consecutive columns, appears exactly* $q = 3$ *times. Therefore,* M *constitutes a* $CCOA_3(2, 13, 3)$*. By the properties of the m-sequence, every* 3*-tuple within any three cyclic consecutive columns occurs precisely once. It can be verified that each* 4*-tuple occurs at most once for any two disjoint cyclic consecutive columns. Thus, the proof is concluded.* □

Theorem 4.15. Suppose that G forms a cycle with $q^2 + q + 1$ vertices, and that f is a primitive polynomial of degree 3 over F_q with nonzero initial values $I = (b_0, b_1, \cdots, b_{t-1}) \in F_q^3$. Then the array M is an optimal DA$(q^3; q - 1, G, q)$.

Proof. The conclusion follows from Theorem 3.4 and 4.14. □

Examples of optimal DAs on cycles over F_p from Theorem 4.15 with $p = 2, 3, 5, 7$ are provided in Table 6. The first column indicates the finite field F_p, while the second and third columns are primitive polynomial and the number of factors k of optimal DAs on cycles, respectively.

Table 6. Optimal DAs on cycles over F_p with $p = 2, 3, 5, 7$.

F_p	$f(x)$	k
$p = 2$	$x^3 + x^2 + 1$	7
$p = 3$	$x^3 + 2x^2 + 1$	13
$p = 5$	$x^3 + 3x + 3$	31
$p = 7$	$x^3 + 5x + 2$	57

4.3. Existence of optimal DAs on cycles with few vertices

The existence of optimal DAs on cycles with few vertices is almost determined in this subsection. To this end, we will present some recursive constructions.

Let A be a CCOA$_\lambda(k, v)$ over the symbol set V. If the rows of A can be partitioned into μ subarrays, each possessing the specified simple property, then we label A as a *μ-row-divisible CCOA$_\lambda(k, v)$*. Two distinct simple CCOAs sharing the same symbol set are deemed *compatible* if their superimposition results in a simple CCOA. A collection comprising w simple CCOAs over the same symbol set is termed *compatible* if every pair of elements within the set is compatible. The notion of μ-row-divisible and compatible OAs was first introduced in [20], where they were utilized for constructing SSOAs. In this context, we adapt and modify these concepts for constructing simple CCOAs. Analogous to the proof of Theorem 11 in [18], we present the following construction.

Theorem 4.16. *Suppose that v_1 and v_2 are two positive integers for which there exist μ compatible simple $CCOA_\eta$ $(k, v_2)'s$. Further, let r denote a non-negative integer, and let $m_1, m_2, \cdots, m_r$ be nonnegative integers, along with $2r$ positive integers $\mu_1, \mu_2, \cdots, \mu_r$, $\lambda_1, \lambda_2, \cdots, \lambda_r$ such that the following conditions are met:*

1) $m_1\mu_1 + m_2\mu_2 + \cdots + m_r\mu_r \leq \mu$;
2) a μ_i-row-divisible $CCOA_{\lambda_i}(k, v_1)$ is available for $1 \leq i \leq r$.

Then, there exists a simple $CCOA_{\eta(m_1\lambda_1+m_2\lambda_2+\cdots+m_r\lambda_r)}(k, v_1v_2)$.

According to Theorem 4.16, selecting $r = 1$, $m_1 = 1$, $\lambda_1 = \lambda$, $v_1 = v$, $\mu_1 = \mu$, and $v_2 = m$, we can derive the subsequent outcome.

Corollary 4.17. *Let v, k, and t be integers satisfying $k \geq t \geq 2$. If a μ-row-divisible $CCOA_\lambda(k, v)$ and μ compatible simple $CCOA_\eta(k, m)'s$ all exist, then so does a simple $CCOA_{\lambda\eta}(k, mv)$. In particular, if a simple $CCOA_\lambda(k, v)$ and a simple $CCOA_\eta(k, m)$ both exist, then so does a simple $CCOA_{\lambda\eta}(k, mv)$.*

It is clear that a CCOA(k, m) is a simple CCOA(k, m). By taking $\eta = 1$ in Corollary 4.17, we have the following corollary.

Corollary 4.18. *Suppose that a simple $CCOA_\lambda(k, v)$ and a $CCOA(k, m)$ exist. Then, a simple $CCOA_\lambda(k, mv)$ exists.*

It is known that an OA$(2, k, v)$ is also an OA$(2, k', v)$ by deleting a $k - k'$ column, where $2 \leq k' < k$. However, this simple fact is not always true for CCOAs. Thus, we have to use other ways to construct simple CCOAs for each set of values k, v.

Theorem 4.19. *An optimal $DA((d + 1)v^2; d, G, v)$ exists for any integer d with $(d + 1) \leq v$, where G is a cycle with 3 or 4 vertices.*

Proof. The existence of SSOA$_{d+1}(2, k, v)'$s with $k = 3, 4$ is given in [20, 29]. Clearly, an SSOA is also a simple CCOA. Applying Theorem 4.1 produces optimal DAs on cycles as desired. □

By Theorem 4.13, for the completeness of existence for a simple CCOA$_\lambda(5, v)$, we consider the case $v = 2, 3, 6$ and $\gcd(v, 4) = 2$ or $\gcd(v, 18) = 3$. For $v \in \{2, 3, 6\}$, we have the following results.

Lemma 4.20. *A simple $CCOA_\lambda(5, 2)$ over $\mathbb{Z}_2$ exists for $\lambda = 1, 2$.*

Proof. A is the required simple CCOA$(2, 5, 2)$ as follows.

$$A = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

A simple CCOA$_2(5, 2)$ over $\mathbb{Z}_2$ is given in Example 4.1. □

Lemma 4.21. *A simple $CCOA_\lambda(5, 3)$ over $\mathbb{Z}_3$ exists for $\lambda = 1, 2, 3$.*

Proof. A simple CCOA$_\lambda(5, 3)$ over $\mathbb{Z}_3$ with $\lambda = 1, 3$ is presented in Appendix. An SSOA$_2(2, 5, 3)$ is given in [30]. It is also a simple CCOA$_2(5, 3)$. □

Lemma 4.22. *A simple CCOA*$_\lambda(5,6)$ *over* $\mathbb{Z}_6$ *exists for* $\lambda = 1,2,3,4,6$.

Proof. By Corollaries 4.17 and 4.18, we can obtain the required simple CCOAs for $\lambda \neq 3$. The ingredients are given by Lemmas 4.20 and 4.21. We show the existence of a simple CCOA$_3(5,6)$ over $\mathbb{Z}_6$ given in the Appendix. □

Lemma 4.23. *A* 2*-row-divisible CCOA*$_5(5,6)$ *over* $\mathbb{Z}_6$ *exists.*

Proof. Combining a simple CCOA$_4(5,6)$ and a simple CCOA$(5,6)$ produces a 2-row-divisible CCOA$_5(5,6)$. □

We proceed to establish the existence of a simple CCOA$_\lambda(5,v)$ with $\gcd(v,4) = 2$ or $\gcd(v,18) = 3$. Let u be any positive integer satisfying $\gcd(u,6) = 1$. Consequently, any integer $v \geq 4$ can be expressed as $v = 2^\alpha 3^\beta \cdot u$, where α and β are nonnegative integers. We consider the following three cases.

1) $v = 2u$ or $v = 2 \cdot w$, where $w = 3^\beta \cdot u \geq 9u$;
2) $v = 3u$;
3) $v = 6u$.

To provide the results for the above cases, we first prove the existence of some compatible CCOAs as follows.

Lemma 4.24. *Let* $u > 1$ *and* w *be an integer as mentioned above. Then, both* u *compatible simple CCOA*$(2,5,v)'$*s and* w *compatible simple CCOA*$(2,5,w)'$*s exist.*

Proof. Combining the assumption and Lemma 4.3, it follows that an OA$(3,6,u)$ or an OA$(3,6,w)$ exists. By Construction 3.7 in [20], u compatible OA$(2,5,v)'$s and w compatible OA$(2,5,w)'$s exist. Clearly, they are also compatible simple CCOAs. □

Lemma 4.25. *Let* $u > 1$ *be an integer with* $\gcd(u,6) = 1$*. Then, a simple CCOA*$_\lambda(2,5,2u)$ *exists for any* $\lambda \leq 2u$ *and a simple CCOA*$_\lambda(2,5,2w)$ *with* $w = 3^\beta u \geq 9u$ *exists for any* $\lambda \leq 2w$*, respectively.*

Proof. For any given λ with $\lambda \leq 2u$, we write $h = \lfloor \lambda/2 \rfloor$ and $\varepsilon = \lambda - 2h \in \{0,1\}$. Then $h \leq u$, if $\varepsilon = 0$; $h \leq u-1$, if $\varepsilon = 1$. Apply Theorem 4.16 with $v_1 = 2$, $v_2 = u$, $(\mu_1,\lambda_1) = (1,1)$, $(\mu_2,\lambda_2) = (1,2)$ and $(m_1,m_2) = (\varepsilon,h)$. A simple CCOA$_\lambda(5,2)$ with $\lambda = 1,2$ was given in Lemma 4.20. u compatible simple CCOA$(2,5,u)'$s are given by Lemma 4.24. For $w = 3^\beta u \geq 9u$, a similar argument can be applied to draw the conclusion. □

Lemma 4.26. Let $u > 1$ be an integer with $gcd(u,6) = 1$. Then, a simple CCOA$_\lambda(5,3u)$ exists for any positive integer $\lambda \leq 3u$.

Proof. By assumption, there exist u compatible simple CCOA$(2,5,u)'$s by Lemma 4.24. Apply Theorem 4.16 with $v_1 = 3$, $v_2 = u$, $(\mu_1,\lambda_1) = (1,3)$, and

$$(\mu_2,\lambda_2) = \begin{cases} (1,2), & \text{if } \lambda \equiv 2 \pmod 3, \\ (1,1), & \text{if } \lambda \equiv 1 \pmod 3. \end{cases}$$

By Lemma 4.21, a simple CCOA$_\lambda(5,3)$ exists for $\lambda = 1,2,3$. It is left to show that the system of equations

$$\begin{cases} 3m_1 + \lambda_2 m_2 = \lambda, \\ m_1 + \mu_2 m_2 \leq u. \end{cases}$$

is solvable in nonnegative integers m_1 and m_2 for any given λ with $2 \le \lambda \le 3u$.

Let $h = \lfloor \lambda/3 \rfloor$ and $\varepsilon = \lambda - 3h \in \{0, 1, 2\}$, then we have

$$\begin{cases} h \le u, \text{ if } \varepsilon = 0, \\ h \le u - 1, \text{ if } \varepsilon = 1, 2. \end{cases}$$

It turns out that

$$(m_1, m_2) = \begin{cases} (h, 0), \text{ if } \varepsilon = 0; \\ (h, 1), \text{ if } \varepsilon = 1, 2. \end{cases}$$

is one solution of the above system of equations. The proof is complete. □

Lemma 4.27. *Let $v = 6u$ be an integer with $u \ne 1$ and $gcd(u, 6) = 1$. Then, a simple $CCOA_\lambda(5, v)$ exists for any integer $\lambda \le v$ except for $\lambda = 6u - 1$.*

Proof. From Lemmas 4.22 and 4.23, we know that both a simple $\mathrm{CCOA}_\lambda(5, 6)$ with $\lambda \in \{1, 2, 3, 4, 6\}$ and a 2-row-divisible $\mathrm{CCOA}_5(5, 6)$ exists. For any given λ with $\lambda \le 6u$ and $\lambda \ne 6u - 1$, let $h = \lfloor \lambda/6 \rfloor$ and $\varepsilon = \lambda - 6h \in \{0, 1, \cdots, 5\}$. Then,

$$\begin{cases} h \le u, \text{ if } \varepsilon = 0, \\ h \le u - 1, \text{ if } \varepsilon = 1, 2, 3, 4, \\ h \le u - 2, \text{ if } \varepsilon = 5\,. \end{cases}$$

For $\lambda = 6h$, a simple $\mathrm{CCOA}_\lambda(5, v)$ can be obtained by taking $v_1 = 6$, $v_2 = u$, $(\mu_1, \lambda_1) = (1, 6)$ and $(m_1, m_2) = (h, 0)$ in Theorem 4.16. For $\lambda \ne 6u - 1$ and $\lambda \ne 6h$, apply Theorem 4.16 with $v_1 = 6$, $v_2 = u$, $(\mu_1, \lambda_1) = (1, 6)$, $(m_1, m_2) = (h, 1)$, and (μ_2, λ_2) for the pairs of nonnegative integers as described below to obtain the required simple CCOAs.

$$(\mu_2, \lambda_2) = \begin{cases} (1, \varepsilon), \text{ if } \varepsilon = 1, 2, 3, 4, \\ (2, 5), \text{ if } \varepsilon = 5. \end{cases}$$

□

Combining Theorem 4.1 with the results in Lemmas 4.13, 4.25–4.27, the subsequent conclusions are drawn.

Theorem 4.28. *An optimal $DA((d + 1)v^2; d, G, v)$ exists for any positive integer $d + 1 \le v$, where G is a cycle with 5 vertices, except possibly in cases where*

1) $(d, v) = (4, 6)$;
2) $v = 6u$ and $d = v - 2$, where $u \ne 1$ and $gcd(u, 6) = 1$.

The remaining unresolved case corresponds to the existence of a simple $\mathrm{CCOA}_5(5, 6)$, which is required in the recursive construction applied in the proof. The exhaustive search for such an array is computationally intensive and has not succeeded within acceptable time bounds under our current implementation. We leave the resolution of this case to future work, where improved algorithms or alternative construction techniques may yield progress.

5. Conclusions

Utilizing DAs with graph structures, where a vertex represents each component and an edge signifies interaction between components, can enhance efficiency in various applications, especially software testing. This paper introduced the concept of DAGs, serving as a generalization of DAs with strength 2. The key advantage of employing DAGs to generate test suites lies in the ability to identify any set of d valid 2-way interaction faults from the test outcomes. Moreover, for a DA($(d+1)v^2; d, G, v$), if there are more than d valid 2-way interactions causing faults, they can also be detected. The paper established a general lower bound on the size of DA($N; d, G, v$). The equivalence between an optimal DA($(d+1)v^2; d, G, v$) and a combinatorial configuration (simple $\mathrm{OA}_{d+1}(G, v)$) was established in Theorem 3.4. With this equivalence, numerous optimal DAs on cycles were obtained by constructing simple CCOAs. Notably, the existence of DA($(d+1)v^2; d, G, v$) with few vertices, where G is a cycle, was nearly completely determined.

Nevertheless, this paper is only the beginning of the study of this issue. Different practical problems correspond to different graph structures. In the future, it is worthwhile to concentrate on exploring new methods for constructing simple orthogonal arrays on various types of graphs, for example, Eulerian graphs as stated in [31] and Polygon Networks in [32]. Meanwhile, DAs on graphs are limited to locate and detect the faulty interactions triggered by only two components. Hence, generating a new notion of DAs on graphs to eliminate the restriction is also a valuable direction for further study.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by NSFC No. 11301342 and Natural Science Foundation of Shanghai No. 17ZR1419900 and Natural Science Foundation of Jiangsu Grant No. BK20171318.

Conflict of interest

The authors declare there is no conflicts of interest.

References

1. C. J. Colbourn, *CRC Handbook of Combinatorial Designs*, CRC Press, 2010. https://doi.org/10.1201/9781003040897
2. D. R. Stinson, *Combinatorial Designs: Constructions and Analysis*, Springer, New York, 2004. https://doi.org/10.1007/b97564
3. D. R. Kuhn, R. N. Kacker, Y. Lei, *Introduction to Combinatorial Testing*, CRC Press, 2013. https://dl.acm.org/doi/book/10.5555/2500930
4. C. J. Colbourn, Strength two covering arrays: Existence tables and projection, *Discrete Math.*, **308** (2008), 772–786. https://doi.org/10.1016/j.disc.2007.07.050

5. C. J. Colbourn, S. S. Martirosyan, T. V. Trung, R. A. Walker, Roux-type constructions for covering arrays of strengths three and four, *Des. Codes Cryptogr.*, **41** (2006), 33–57. https://doi.org/10.1007/s10623-006-0020-8

6. L. Ji, J. Yin, Constructions of new orthogonal arrays and covering arrays of strength three, *J. Comb. Theory Ser. A*, **117** (2010), 236–247. https://doi.org/10.1016/j.jcta.2009.06.002

7. J. T. Jimenez, I. I. Marquez, Covering arrays of strength three from extended permutation vectors, *Des. Codes Cryptogr.*, **86** (2018), 2629–2643. https://doi.org/10.1007/s10623-018-0465-6

8. A. Hartman, L. Raskin, Problems and algorithms for covering arrays, *Discrete Math.*, **284** (2004), 149–156. https://doi.org/10.1016/j.disc.2003.11.029

9. K. Sarkar, C. J. Colbourn, Two-stage algorithms for covering array construction, *J. Comb. Des.*, **27** (2019), 475–505. https://doi.org/10.1002/jcd.21657

10. G. Tzanakis, L. Moura, D. Panario, B. Stevens, Covering arrays from m-sequences and character sums, *Des. Codes Cryptogr.*, **85** (2017), 437–456. https://doi.org/10.1007/s10623-016-0316-2

11. *Covering array tables for* $t = 2, 3, 4, 5, 6$, 2020. Available from: https://www.public.asu.edu/ ccolbou/src/tabby/2-3-ca.html

12. K. Meagher, B. Stevens, Covering arrays on graphs, *J. Comb. Theory Ser. B*, **95** (2005), 134–151. https://doi.org/10.1016/j.jctb.2005.03.005

13. A. El-Mesady, O. Bazighifan, H. M. Shabana, On graph transversal designs and graph-authentication codes based on mutually orthogonal graph squares, *J. Math.*, (2022), 8992934. https://doi.org/10.1155/2022/8992934

14. M. Higazy, A. El-Mesady, M. S. Mohamed, On graph-orthogonal arrays by mutually orthogonal graph squares, *Symmetry*, **12** (2020), 1895. https://doi.org/10.3390/sym12111895

15. K. Meagher, L. Moura, L. Zekaoui, Mixed covering arrays on graphs, *J. Comb. Des.*, **15** (2007), 393–404. https://doi.org/10.1002/jcd.20149

16. T. Mahapatra, M. Pal, An investigation on m-polar fuzzy threshold graph and its application on resource power controlling system, *J. Ambient Intell. Humaniz. Comput.*, **13** (2022), 501–514. https://doi.org/10.1007/s12652-021-02914-6

17. C. J. Colbourn, D. W. McClary, Locating and detecting arrays for interaction faults, *J. Comb. Optim.*, **15** (2008), 17–48. https://doi.org/10.1007/s10878-007-9082-4

18. C. Shi, L. Jiang, A. Y. Tao, Consecutive detecting arrays for interaction faults, *Graphs Combin.*, **36** (2020), 1203–1218. https://doi.org/10.1007/s00373-020-02176-7

19. Y. Tang, J. X. Yin, Detecting arrays and their optimality, *Acta Math. Sin. Engl. Ser.*, **27** (2011), 2309–2318. https://doi.org/10.1007/s10114-011-0184-7

20. C. Shi, Y. Tang, J. X. Yin, The equivalence between optimal detecting arrays and super-simple OAs, *Des. Codes Cryptogr.*, **62** (2012), 131–142. https://doi.org/10.1007/s10623-011-9498-9

21. C. Shi, B. Wen, J. Lin, Cyclic consecutive orthogonal arrays, *Acta Math. Appl. Sin.*, **39** (2016), 786–800.

22. C. R. Rao, Factorial experiments derivable from combinatorial arrangements of arrays, *Suppl. J. Roy. Stat. Soc.*, **9** (1947), 128–139. https://doi.org/10.2307/2983576

23. A. S. Hedayat, N. J. A. Sloane,J. Stufken, *Orthogonal Arrays: Theory and Applications*, Springer-Verlag, New York, 1999. http://dx.doi.org/10.1007/978-1-4612-1478-6

24. K. A. Bush, Orthogonal arrays of index unity, *Ann. Math. Stat.*, **23** (1952), 426–434. https://doi.org/10.1214/aoms/1177729387

25. S. W. Golomb, G. Gong, *Signal Design for Good Correlation for Wireless Communication, Cryptography, Radar*, Cambridge Univ. Press, Cambridge, UK, 2005. https://doi.org/10.1017/CBO9780511546907

26. A. G. Castoldi, L. Moura, D. Panario, B. Stevens, Ordered orthogonal array construction using LFSR sequences, *IEEE Trans. Inf. Theory*, **63** (2017), 1336–1346. https://doi.org/10.1109/tit.2016.2634010

27. 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. https://doi.org/10.1007/s10623-013-9835-2

28. C. Shi, A. Tao, Consecutive detecting arrays from *m*-sequence, *IAENG Int. J. Appl. Math.*, **50** (2020), 80–86.

29. S. Hartman, On simple and supersimple transversal designs, *J. Comb. Des.*, **8** (2000), 311–322. https://doi.org/10.1002/1520-6610(2000)8:5%3C311::AID-JCD1%3E3.0.CO;2-1

30. Y. H. Chen, *Constructions of Optimal Detecting Arrays of Degree* 5 *and Strength* 2, Master thesis, Soochow University in Suzhou, 2011.

31. J. B. Liu, C. Wang, S. Wang, B. Wei, Zagreb indices and multiplicative zagreb indices of eulerian graphs, *Bull. Malays. Math. Sci. Soc.*, **42** (2019), 67–78. https://doi.org/10.1007/s40840-017-0463-2

32. J. B. Liu, L. Guan, J. Cao, L. Chen, Coherence analysis for a class of polygon networks with the noise disturbance, *IEEE Trans. Syst. Man Cyber. Syst.*, **2025** (2025). https://doi.org/10.1109/TSMC.2025.3559326

33. Y. Akhtar, S. Maity, Covering arrays on product graphs, *Graphs Combin.*, **33** (2017), 635–652. https://doi.org/10.1007/s00373-017-1800-9

34. A. El-Mesady, Y. S. Hamed, K. M. Abualnaja, A novel application on mutually orthogonal graph squares and graph-orthogonal arrays, *Mathematics*, **7** (2022), 7349C7373. https://doi.org/10.3934/math.2022410

35. C. Shi, Y. Tang, J. X. Yin, Optimum mixed level detecting arrays, *Ann. Statist.*, **42** (2014), 1546–1563. https://doi.org/10.1214/14-AOS1228

36. C. Shi, J. Yin, Existence of super-simple $\mathrm{OA}_\lambda(3,5,v)$'s, *Des. Codes Cryptogr.*, **72** (2014), 369–380. https://doi.org/10.1007/s10623-012-9771-6

Appendix

A simple $\mathrm{CCOA}_\lambda(5,3)$ with $\lambda = 1$ or 3 is given below.
$\lambda = 1$, $A = A_1^T$, where A_1 is as follows.

$$A_1 = \begin{pmatrix} 000111222 \\ 012012012 \\ 012120201 \\ 012201120 \\ 120120120 \end{pmatrix}$$

$\lambda = 3$, $A = A_2^T$, where A_2 is as follows.

$$A_2 = \begin{pmatrix} 000000000111111111222222222 \\ 000111222000111222000111222 \\ 012012012012012012012012012 \\ 000111222111222000222000111 \\ 012012012120120120201201201 \end{pmatrix}$$

A simple $\mathrm{CCOA}_3(5, 6)$ A is given below. $A = (A_1, A_2)^T$, where A_1 and A_2 are as follows.

$$A_1 = \begin{pmatrix} 333113220312441034003343441152300110042550501441242024 \\ 242451344503052441233344424232311035205131453050141042 \\ 300445035044005142355351240024003043112444252243533411 \\ 112121102553203544504430503401314000043552102431423243 \\ 340535311011424052143022330152402353034545420500113225 \end{pmatrix}$$

$$A_2 = \begin{pmatrix} 535355332455020035411421312410215524523021013402245555 \\ 110122250534350002252121534313310051100525440321034555 \\ 511221224513155055534422154313133231031132252000520014 \\ 053541204534025335253401110022212131252432315455430451 \\ 133145110521554150022401224453442001332454334052510321 \end{pmatrix}$$

AIMS Press