Perfect Sequence Covering Arrays

by

Jingzhou Na

B.Sc., Capital Normal University, 2019
B.Sc., University of Cincinnati, 2019

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
Department of Mathematics
Faculty of Science

SIMON FRASER UNIVERSITY
Summer 2021

Declaration of Committee

Name:	**Jingzhou Na**
Degree:	**Master of Science**
Thesis title:	**Perfect Sequence Covering Arrays**
Committee:	**Chair:** Imin Chen Professor, Mathematics
	Jonathan Jedwab Supervisor Professor, Mathematics
	Petr Lisonek Committee Member Professor, Mathematics
	Jake Levinson Examiner Assistant Professor, Mathematics

Abstract

An (n, k)-perfect sequence covering array with multiplicity λ is a subset of the $n!$ permutations of the sequence $(1, 2, \cdots, n)$ whose elements collectively contain each ordered length k subsequence exactly λ times. The primary objective is to determine, for given n and k, the smallest value of λ (denoted $g(n, k)$) for which such a configuration exists. In 2020, Yuster determined the first known value of $g(n, k)$ greater than 1, namely $g(5, 3) = 2$, and suggested that finding other such values would be challenging. We determine that $g(6, 3) = g(7, 3) = g(7, 4) = 2$ and $g(8, 3) \in \{2, 3\}$ by modifying an old search algorithm due to Mathon and van Trung, and by restricting a perfect sequence covering array to be a union of cosets of a prescribed nontrivial subgroup of the symmetric group S_n. This prescribed structure provides a deeper understanding of the existence pattern for perfect sequence covering arrays by combining combinatorial and algebraic viewpoints.

Keywords: combinatorial design theory; perfect sequence covering array; group theory; search algorithm

Acknowledgements

It should be a joyful moment to appreciate all the people who supported me. However, it is also near the submission deadline and my flight's departure time, which does not give me enough time to share all the enjoyable and memorable stories I had.

As a master's student, I am confident to say that "Jonathan Jedwab is the best supervisor I have ever had during my student life include the Ph.D. program." It is a great pleasure to collaborate with him and growing under his thoughtful kindness. For detail, please check the acknowledgement section of my Ph.D. thesis.

It is wonderful to be a member of Jonathan's research group together with Federico Firoozi, Ivan Lau, Samuel Simon, Shuxing Li, and Thaís Bardini Idalino. It is a pity that no director invites us to film our discussion as a documentary named "Academic Life: beyond research". Appreciation for all their comments and suggestions for me to affirm and improve myself. Especially for Shuxing, who is a research member of this project and provides various suggestions and directions.

I want to thank Jake Levinson, Imin Chen, and Petr Lisonek for spending time reading my thesis, holding my thesis defence, and giving constructive comments and direction of improvements.

I want to thank the research group of Dan Gentle, Daniel Horsley, and Ian Wanless at Monash University, who have researched the same topic independently and decided to wait for the completion of my thesis.

Appreciation to Ian for reminding us of "the reversal construction". Appreciation also to Karen Meagher at the University of Regina for the suggestion of "searching over conjugacy classes".

I want to thank my high school classmates Boxiang Zhao and Chengxi Li for providing programming suggestions during the early stage of the project.

I would also like to thank my friends from these two years who learnt together, traveled together, and supported me during my thesis defence, including but not limited to Alireza Yazdani, Benjamin Chase, Cassidy Tam, Jie Jian, Ivan Lau, Pengyu Liu, Peter Bradshaw, Qinghong Xu, Shuxing Li, Tian Chen, and Xu Zhe.

Omitting all the fun experiences we had, it is my pleasure to sell my buddy Ivan's master thesis to you: "Nonuniform Compressed Sensing Schemes with Sublinear Measurements, Sublinear Time, and Low Entropy", which can be found from the SFU library.

Special thanks to my wholehearted girl Jingwen Zhang. I am so glad we can spend six years together studying at the same universities. Without her, it is impossible for me to start the master program application myself, so I would miss all the kindness I received during these two years. To me, the biggest challenge while studying abroad is to bear solitude. Thank you for your company when we are far away from home. All the joys and sorrows are for you.

Last but not least, I want to thank my father, Qiang Yuan, who planted my interest in mathematics in my childhood. I want to thank my mother, Erjin Na, who raised me with her whole love.

Look at what I have done to my acknowledgement that could take as long as this thesis! If it is not the case that I am running out of time to catch up on the flight, emmmmmmmm, more tears would come out.

Table of Contents

List of Tables

Chapter 1

Introduction and Background

1.1 Definitions

1.1.1 Perfect sequence covering arrays

Throughout, we regard n and k as integers satisfying $2 \leq k \leq n$. We write $[n] := \{1, \ldots, n\}$.

Definition 1.1 (n-sequence and k-subsequence)**.** An *n-sequence* is an ordering of the elements of $[n]$. A *k-subsequence* of $[n]$ is an ordered k-subset of $[n]$.

We sometimes wish to regard an n-sequence as a permutation, namely as an element of the symmetric group S_n on n elements. We write the set of all k-subsequences of $[n]$ as $S_{n,k}$.

Definition 1.2 (cover)**.** Let x be an n-sequence and y be a k-subsequence. We say *x covers y* (or equivalently *y is covered by x*) if y can be obtained by removing $n-k$ symbols from x. Let S be a set of n-sequences. We say *S covers y* if at least one element of S covers y, and *S covers y with multiplicity λ* if exactly λ elements of S cover y.

We give some elementary counting results for n-sequences and k-subsequences.

Result 1.3 (n-sequence and k-subsequence)*.*

(1) $|S_n| = n!$

(2) $|S_{n,k}| = \binom{n}{k} k!$

(3) Each n-sequence covers exactly $\binom{n}{k}$ elements of $S_{n,k}$.

(4) Each element of $S_{n,k}$ is covered by exactly $\binom{n}{k}(n-k)!$ n-sequences.

We now introduce the main object of study.

Definition 1.4 (sequence covering array, perfect sequence covering array)**.** An *(n,k)-sequence covering array* is a set of n-sequences that covers each element of $S_{n,k}$ at least once. An *(n,k)-perfect sequence covering array* with multiplicity λ, denoted $\mathrm{PSCA}(n,k,\lambda)$, is a set of n-sequences that covers each element of $S_{n,k}$ with multiplicity λ.

Example 1.5. The set S_4 of all 4-sequences is

$$\begin{array}{cccccc} 1234 & 1243 & 1324 & 1342 & 1423 & 1432 \\ 2134 & 2143 & 2314 & 2341 & 2413 & 2431 \\ 3124 & 3142 & 3214 & 3241 & 3412 & 3421 \\ 4123 & 4132 & 4213 & 4231 & 4312 & 4321. \end{array}$$

The set $S_{4,3}$ of all 3-subsequences is

$$\begin{array}{cccccc} 123 & 124 & 132 & 134 & 142 & 143 \\ 213 & 214 & 231 & 234 & 241 & 243 \\ 312 & 314 & 321 & 324 & 341 & 342 \\ 412 & 413 & 421 & 423 & 431 & 432. \end{array}$$

The set

$$P = \{1234,\ 1432,\ 2413,\ 3412,\ 3214,\ 4231\}$$

is a PSCA$(4, 3, 1)$: it covers each element of $S_{4,3}$ exactly once. Each subset of S_4 containing P is a $(4, 3)$-sequence covering array: it covers each element of $S_{4,3}$ at least once.

Remark 1.6.

1. The definition of sequence covering array follows that of [CCHZ13], for example. There is a natural generalization of sequence covering array to include a parameter $\lambda > 1$. However, this does not seem to have been widely studied.

2. The fundamental question in the study of sequence covering arrays is to determine the smallest possible size of an (n, k)-sequence covering array. If a PSCA$(n, k, 1)$ exists, then it achieves the smallest possible size of an (n, k)-sequence covering array; this extremal case often imposes additional structure on the array.

3. Yuster [Yus20] introduced the terminology PSCA(n, k, λ) in 2020, although equivalent objects were studied earlier by other authors (see Section 1.1.2). Yuster [Yus20, p. 586] allows a PSCA(n, k, λ) to be a multiset of n-sequences, whereas we have chosen to restrict the definition to a set. Certain results in [Yus20] do not apply under this more restrictive definition (see Remark 1.12).

Result 1.7. Suppose P is a PSCA(n, k, λ). Then the result of applying one or both of the following operations is also a PSCA(n, k, λ):

(1) reverse each element of P

(2) relabel symbols by applying a permutation $\sigma \in S_n$ to each of the symbols of each of the sequences of P.

Definition 1.8 (equivalence)**.** Suppose that each of P and Q is a PSCA(n, k, λ). Then P and Q are *equivalent* if Q can be obtained from P by one or both of the operations of Result 1.7.

Result 1.9 (size of PSCA). Let P be a PSCA(n, k, λ). Then $|P| = k!\lambda$.

Proof. Since P covers each element of $S_{n,k}$ with multiplicity λ, the size of the multiset of k-subsequences covered by P is $|S_{n,k}|\lambda = \binom{n}{k}k!\lambda$ by Result 1.3 (2). Since each n-sequence covers exactly $\binom{n}{k}$ k-subsequences, the size of this multiset is also given by $\binom{n}{k}|P|$. Equate these two expressions to obtain the result. ■

Definition 1.10 (g-value)**.** The value $g(n, k)$ is the smallest λ for which a PSCA(n, k, λ) exists.

The value $g(n, k)$ is well-defined: it is at most $n!/k!$, because S_n is a trivial PSCA$(n, k, \frac{n!}{k!})$ for all $k \leq n$ by Result 1.3 (4). By Example 1.5, we have $g(4, 3) = 1$. The central objective in the study of perfect sequence covering arrays is to determine the value of $g(n, k)$ for all n and k.

We next consider the result of deleting one of the n symbols from a PSCA$(n, k, 1)$.

Result 1.11 (symbol deletion). Let $k \leq n - 1$. If $g(n, k) = 1$, then $g(n - 1, k) = 1$.

Proof. Suppose that $g(n, k) = 1$, and let P be a PSCA$(n, k, 1)$. Delete the symbol n from each sequence in P to obtain a multiset P'. Since $S_{n-1,k} \subset S_{n,k}$ and P covers each element of $S_{n,k}$ exactly once, the $(n-1)$-sequences of P' collectively cover each element of $S_{n-1,k}$ exactly once. We shall show that the multiset P' is actually a set, so that P' is a PSCA$(n - 1, k, 1)$ and therefore $g(n - 1, k) = 1$.

Suppose, for a contradiction, that P' is not a set. Then P' contains some $(n-1)$-sequence x at least twice. Let y be a k-subsequence covered by x. Then y is covered by at least two $(n - 1)$-sequences in P', and therefore by at least two n-sequences in P. This contradicts that P is a PSCA$(n, k, 1)$. ■

Remark 1.12. As noted in Remark 1.6 (3), the paper [Yus20] allows a PSCA(n, k, λ) to be a multiset. According to that definition, deletion of symbol n from all sequences of a PSCA(n, k, λ) leaves a PSCA$(n-1, k, \lambda)$ and so we would conclude that $g(n-1, k) \leq g(n, k)$. We do not assume this conclusion since we do not allow a PSCA(n, k, λ) to be a multiset. For example, the set

$$Q = \{12345,\ 13254,\ 14523,\ 15432,\ 24315,\ 25413,\ 34512,\ 35214,\ 42513,\ 43215,\ 52314,\ 53412\}$$

is a PSCA(5,3,2). Deletion of the symbol 5 from all sequences of Q leaves the set

$$\{1234,\ 1324,\ 1423,\ 1432,\ 2431,\ 2413,\ 3412,\ 3214,\ 4213,\ 4321,\ 2314,\ 3412\},$$

which is not a PSCA$(4, 3, 2)$ because it contains the element 3412 (shown in red) twice.

1.1.2 Related combinatorial objects

We now describe how perfect sequence covering arrays are related to directed designs and to completely scrambling sets.

Definition 1.13 (k-(v, n, λ) directed packing/design)**.** A k-(v, n, λ) *directed packing* (X, B) comprises a v-set X of elements (points) and a collection B of ordered n-subsets (blocks) of X, such that each ordered k-subset of points occurs in at most λ blocks. A k-(v, n, λ) *directed design* is a k-(v, n, λ)-directed packing (X, B) in which each ordered k-subset of points occurs in exactly λ blocks. A k-$(n, n, 1)$ directed design is also known as a *directed Steiner system*.

The main focus in the study of directed packings is to determine the largest possible size $|B|$ of a k-(v, n, λ) directed packing (X, B). In the case $v = n$, we can write $X = [n]$ so that B is a set of n-sequences that covers each element of $S_{n,k}$ at most λ times. For example, each subset P' of the PSCA$(4, 3, 1)$ P given in Example 1.5 satisfies that $([4], P')$ is a 3-$(4, 4, 1)$ directed packing. Comparison with Definition 1.4 shows that a PSCA(n, k, λ) P is equivalent to a k-(n, n, λ) directed design $([n], P)$; and if a PSCA(n, k, λ) exists, then it achieves the largest possible size of a k-(n, n, λ) directed packing. See Mathon and van Trung [MvT99] or Dawson, Seberry, Skillicorn [DSS84], for example, for further details of directed packings. See Bennett and Mahmoodi [BM07], for example, for further details of directed designs.

Definition 1.14 (completely scrambling set)**.** A set of permutations $\{\pi_1, \ldots, \pi_N\}$ of an n-set X is *completely k-scrambling* if for each ordered k-subset $(x_1, \ldots, x_k)$ of X there is some ℓ such that

$$\pi_\ell(x_i) < \pi_\ell(x_j) \quad \text{if and only if} \quad i < j.$$

The principal objective in the study of completely scrambling sets is to determine the smallest possible size N of a completely k-scrambling set of an n-set X. It is straightforward to show that a completely k-scrambling set of $X = [n]$ is equivalent to an (n, k)-sequence covering array [CCHZ13, Lemma 1.1], and so the principal objectives in the study of these two related objects are identical. See Spencer [Spe72], Füredi [Für96, Section 5], and Ishigami [Ish96], for example, for further details of completely scrambling sets.

1.2 Motivation

In this section, we describe the theoretical and practical motivation for studying perfect sequence covering arrays.

Perfect sequence covering arrays are a type of combinatorial design: by Definition 1.13, a PSCA(n, k, λ) is equivalent to a k-(n, n, λ) directed design, and is the extremal case of a

k-(n, n, λ) directed packing. We wish to determine $g(n, k)$, the smallest value of λ for which a PSCA(n, k, λ) exists. We are primarily concerned with determining exact values of $g(n, k)$ rather than asymptotic bounds. We are particularly interested in determining exact values of $g(n, k)$ greater than 1; only one such value was previously known (see Section 2.6).

Sequence covering arrays are useful in various applications in which faults can arise when certain events occur in a particular order [WSLK08, WLS$^+$09, ARA09, HCM10, YM10, YCM11, KHL$^+$12, CCHZ13]. In order to determine whether faults arise under all possible ordered subsets of at most k out of n events, we require a set of tests in which each ordering of each subset of k events occurs: this is given by an (n, k)-sequence covering array. For example, the faults might be adverse reactions when multiple medications are taken in a certain order. By Definition 1.4, if a PSCA$(n, k, 1)$ exists then it is the extremal case of an (n, k)-sequence covering array and so represents the most cost-efficient method of carrying out the required set of tests. Furthermore, a sequence covering array that is perfect can be used to remove certain types of bias from statistical experiments.

1.3 Overview

Chapter 2 describes previous results. Section 2.1 introduces some trivial constructions. Section 2.2 summarizes the best known asymptotic bounds on the growth rate of $g(n, k)$. Section 2.3 describes the construction due to Levenshtein [Lev91] of a PSCA$(n, n-1, 1)$ using coding theory. Section 2.4 presents combinatorial nonexistence results for certain parameter sets (n, k, λ). Section 2.5 describes a recursive algorithm due to Mathon and van Trung [MvT99] for finding spreads of an incidence structure, and recasts the algorithm to refer specifically to finding all possible examples of a PSCA$(n, k, 1)$. Section 2.6 describes the method used by Yuster [Yus20] to determine the first known exact g-value greater than 1. Section 2.7 summarizes previous results on the exact value of $g(n, k)$ for small n and k. Section 2.8 connects the combinatorial properties of a perfect sequence covering array with its group properties, by regarding a set of n-sequences as a set of permutations in S_n.

Chapter 3 contains new results. Section 3.1 modifies the search algorithm of Section 2.5 to find all possible examples of a PSCA(n, k, λ) for arbitrary $\lambda \geq 1$, thereby determining two new g-values. Section 3.2 reinterprets the Yuster construction of Section 2.6. This motivates Section 3.3, which identifies a structure for a perfect sequence covering array based on cosets of a subgroup of S_n and then adapts the search algorithm by prescribing this structure. Section 3.4 presents the results of applying this algorithm, including the determination of a third new g-value and the restriction of a fourth g-value to one of two possibilities.

We conclude in Chapter 4 by reviewing our findings and posing several open problems.

Chapter 2

Previous Results

2.1 Trivial constructions

In this section, we introduce some trivial constructions of perfect sequence covering arrays, and conditions on the corresponding value of $g(n,k)$. As noted in Chapter 1, we always assume that $2 \leq k \leq n$.

Result 2.1. We have $g(n,k) \leq \frac{n!}{k!}$, and $g(n,n) = 1$.

Proof. As noted after Definition 1.10, the sequences of S_n form a trivial PSCA$(n,k,\frac{n!}{k!})$ by Result 1.3 (4) and so $g(n,k) \leq \frac{n!}{k!}$. In particular, when $k = n$, we obtain $g(n,n) = 1$. ■

Result 2.2. We have $g(n,2) = 1$.

Proof. The set $\{12\cdots n,\ n\cdots 21\}$ is a trivial PSCA$(n,2,1)$: all ascending 2-subsequences are covered exactly once by $12\cdots n$, and all descending 2-subsequences are covered exactly once by $n\cdots 21$. Therefore $g(n,2) = 1$. ■

Example 2.3. Each of $\{12345, 54321\}$ and $\{53241, 14235\}$ is a PSCA$(5,2,1)$.

2.2 Asymptotic bounds on $g(n,k)$

In this section, we summarize the best known asymptotic bounds on the growth rate of $g(n,k)$ as n and k grow. These results are due to Yuster [Yus20], who improved on previous asymptotic results [Spe72, Ish96, Für96, Rad03]) for completely scrambling sets (or equivalently, for sequence covering arrays: see Definition 1.14). The first of these results holds for general k, and the second for the case $k = 3$. Yuster's results hold under the convention that a PSCA(n,k,λ) can be a multiset, and so in particular hold under our convention that a PSCA(n,k,λ) is a set (see Remark 1.6 (3)).

Theorem 2.4 ([Yus20, Thm. 1])**.** *Let $k \geq 4$ be an integer.*

(1) If $k/2$ is a prime, then for all $n \geq k$ we have

$$g(n,k) \geq \frac{\binom{n}{k/2} - \binom{n}{k/2-1}}{k!}.$$

(2) Let n and k grow such that $n \gg k$. Then $g(n,k) > n^{k/2-o_k(1)}$ (where $o_k(1)$ represents a function that approaches 0 as $k \to \infty$).

The proof of Theorem 2.4 combines combinatorial arguments with a result due to Wilson [Wil90, Thm. 1] on the rank of a set inclusion matrix over a finite field. This set inclusion matrix is derived from a structure we shall later call the (n,k)-incidence matrix (see Definition 2.33).

Theorem 2.5 ([Yus20, Thm. 2])**.** *Let $n \geq 3$. Then $n/6 \leq g(n,3) \leq Cn(\log_2 n)^{\log_2 7}$ for some absolute constant C.*

The proof of the upper bound in Theorem 2.5 arises from a recursive construction that builds a PSCA$(n^2, 3, 2(n+1)\lambda)$ from a PSCA$(n,3,\lambda)$, using a finite affine plane of order n where n is a power of 3. We note that this proof [Yus20, proof of Lemma 3.2] implicitly constructs a PSCA$(3^{2^t}, 3, 2^{t-1}(3^{2^t}-1))$ for each non-negative integer t.

In the rest of this thesis, we shall be concerned with exact values of $g(n,k)$ rather than asymptotic bounds.

2.3 Levenshtein construction of a PSCA$(n, n-1, 1)$

In this section, we describe a construction for a PSCA$(n, n-1, 1)$ due to Levenshtein [Lev91] that uses perfect codes capable of correcting single deletions. The construction uses a derivative function from n-sequences to binary words to partition the set of all possible length $n-1$ binary words into n binary codes. We now review the Levenshtein construction and show how to interpret it as partitioning the $n!$ sequences of S_n into n disjoint sets of sequences, each of which is a PSCA$(n, n-1, 1)$. This implies that $g(n, n-1) = 1$.

We first introduce the binary code space.

Definition 2.6 (binary word, binary code)**.** A *length m binary word* is an element of the set $B^m := \{0,1\}^m$. A *length m binary code* is a subset of B^m.

Example 2.7. $\{1011, 1100, 0000\}$ is a length 4 binary code.

Definition 2.8 (subsequence of a binary word)**.** Let $\ell \leq m$. A *length ℓ subsequence* of a length m binary word z is a length ℓ binary word that can be obtained from z by removing $m - \ell$ entries.

Definition 2.9 (perfect in B^m code capable of correcting single deletions)**.** A *perfect in B^m code capable of correcting single deletions* is a length m binary code C such that each

element of B^{m-1} occurs (possibly more than once) as a length $m-1$ subsequence of exactly one word in C.

Example 2.10. The length 3 code $\{001, 110\}$ is a perfect in B^3 code capable of correcting single deletions. Each element of B^2 occurs as a length 2 subsequence of exactly one word in $\{001, 110\}$: the elements 00 and 01 in the word 001, and the elements 10, 11 in the word 110. We can therefore uniquely recover each length 3 word of the code from the result of deleting one of its symbols, so the length 3 code is "capable of correcting single deletions".

Definition 2.11 (norm and weight of a binary word)**.** Let $z = z_1 \cdots z_m \in B^m$. The *norm* of z is $||z|| \coloneqq \sum_{i=1}^{m} z_i$, and the *weight* of z is $w(z) \coloneqq \sum_{i=1}^{m} i z_i$.

The norm of z is the number of ones in z. The weight of z is the sum of the positions of the ones in z (counting upwards from 1 from left to right). These definitions are taken from [Lev91].

Example 2.12. Let $z = 1011$. Then $||z|| = 3$ and $w(z) = 1 + 3 + 4 = 8$.

Definition 2.13 (binary code $W^{m,a}$)**.** Let a, m be integers satisfying $0 \leq a \leq m$. Write $W^{m,a}$ for the length m binary code $\{z \in B^m : w(z) \equiv a \pmod{m+1}\}$.

From the definition, the $m+1$ binary codes $\{W^{m,a}\}_{0 \leq a \leq m}$ form a partition of B^m.

Example 2.14. We have $101 \in W^{3,0}$ because $w(101) = 4 \equiv 0 \pmod{3+1}$. Table 2.1 shows the partition of B^3 into the four binary codes $W^{3,a}$ for $a = 0, 1, 2, 3$.

Table 2.1: Partition of B^3

B^3							
$W^{3,0}$		$W^{3,1}$		$W^{3,2}$		$W^{3,3}$	
000	101	100	011	010	111	001	110

Theorem 2.15 (Levenshtein [Lev91, Thm. 2.1])**.** *Let a, m be integers satisfying $0 \leq a \leq m$. Then the binary code $W^{m,a}$ is a perfect in B^m code capable of correcting single deletions.*

Proof. Let $z = z_1 \cdots z_{m-1} \in B^{m-1}$, and suppose that $x \in W^{m,a}$ contains the subsequence z. We shall establish the result by deriving necessary conditions on x, and then showing that they are satisfied by a unique $x \in W^{m,a}$.

Since x contains z, it has the form

$$x = z_1 \cdots z_{j-1}\, \sigma\, z_j \cdots z_{m-1} \tag{2.1}$$

for some $j \in \{1, \cdots, m\}$ and some $\sigma \in \{0, 1\}$. Since $x \in W^{m,a}$,

$$a \equiv w(x) \pmod{m+1}$$

$$\equiv \sum_{i=1}^{j-1} iz_i + j\sigma + \sum_{i=j}^{m-1} (i+1)z_i \pmod{m+1}$$
$$\equiv w(z) + j\sigma + \sum_{i=j}^{m-1} z_i \pmod{m+1}.$$

Rewrite this as

$$a - w(z) \equiv \begin{cases} \sum_{i=j}^{m-1} z_i \pmod{m+1} & \text{if } \sigma = 0 \\ j + \sum_{i=j}^{m-1} z_i \pmod{m+1} & \text{if } \sigma = 1 \end{cases}$$

and define

$$f(z) := \Big(a - w(z)\Big) \bmod (m+1).$$

Using that both $\sum_{i=j}^{m-1} z_i$ and $j + \sum_{i=j}^{m-1} z_i$ lie in $\{0, \ldots, m\}$, we then have

$$f(z) = \begin{cases} \sum_{i=j}^{m-1} z_i & \text{if } \sigma = 0 \\ j + \sum_{i=j}^{m-1} z_i & \text{if } \sigma = 1. \end{cases} \tag{2.2}$$

Since $\sum_{i=j}^{m+1} z_i \le ||z||$ and $j + \sum_{i=j}^{m+1} z_i \ge 1 + ||z||$, we conclude that σ must be chosen as

$$\sigma = \begin{cases} 0 & \text{if } f(z) \le ||z|| \\ 1 & \text{if } f(z) > ||z||. \end{cases} \tag{2.3}$$

We now show that there is at least one solution to (2.2) for $j \in \{1, \ldots, m\}$, and that all such solutions lead to the same value of x in (2.1).

Case 1: $f(z) \in \{0, \ldots, ||z||\}$**.** In this case, we have $\sigma = 0$ by (2.3). Therefore

$$x = z_1 \cdots z_{j-1}\, 0\, z_j \cdots z_{m-1} \tag{2.4}$$

by (2.1), and we wish to solve

$$f(z) = \sum_{i=j}^{m-1} z_i \tag{2.5}$$

for some $j \in \{1, \ldots, m\}$. Now the sum $\sum_{i=j}^{m-1} z_i$ takes the value 0 for $j = m$, the value $||z||$ for $j = 1$, and each value between 0 and $||z||$ at least once as j decreases from m to 1 because each z_i lies in $\{0, 1\}$. We can therefore define t to be the largest $j \in \{1, \ldots, m\}$ for which (2.5) is satisfied (so that $z_t = 1$), and then (2.4) gives a

solution for x when $j = t$. It is possible that (2.5) is also satisfied for $j < t$ when $z_{t-1} = z_{t-2} = \cdots = z_j = 0$, but the resulting solution (2.4) for x will not change.

Case 2: $f(z) \in \{1 + ||z||, \ldots, m\}$**.** In this case, we have $\sigma = 1$ by (2.3). Therefore

$$x = z_1 \cdots z_{j-1}\, 1\, z_j \cdots z_{m-1} \tag{2.6}$$

by (2.1), and we wish to solve

$$f(z) = j + \sum_{i=j}^{m-1} z_i$$

for some $j \in \{1, \ldots, m\}$. Rewrite this as

$$f(z) = m - \sum_{i=j}^{m-1} (1 - z_i). \tag{2.7}$$

Now $m - \sum_{i=j}^{m-1}(1 - z_i)$ takes the value m for $j = m$, the value $1 + ||z||$ for $j = 1$, and each value between m and $1+||z||$ at least once as j decreases from m to 1 because each $1 - z_i$ lies in $\{0, 1\}$. We can therefore define t to be the largest $j \in \{1, \ldots, m\}$ for which (2.7) is satisfied (so that $z_t = 0$), and then (2.6) gives a solution for x when $j = t$. It is possible that (2.7) is also satisfied for $j < t$ when $z_{t-1} = z_{t-2} = \cdots = z_j = 1$, but the resulting solution (2.6) for x will not change.

■

We next introduce the derivative of a sequence.

Definition 2.16 (derivative)**.** Let $v = v_1 \cdots v_k$ be a sequence of k distinct integers. The derivative of v is the element of B^{k-1} given by

$$D(v) = z_1 \cdots z_{k-1} \text{ where } z_i = \begin{cases} 0 \text{ if } v_i < v_{i+1} \\ 1 \text{ if } v_i > v_{i+1}. \end{cases}$$

$D(v)$ describes the "shape" of the sequence v according to the ascending or descending relationship between consecutive pairs of symbols: $(-1)^{z_i}$ is the sign of $v_{i+1} - v_i$.

Example 2.17. We have $D(2143) = 101$, because $2 > 1$ and $1 < 4$ and $4 > 3$.

Theorem 2.18 (Levenshtein [Lev91, Thm. 3.1])**.** *Let $n \geq 2$, and let*

$$P_a \coloneqq \{x \in S_n : D(x) \in W^{n-1,a}\} \quad \textit{for } a = 0, 1, \ldots, n-1.$$

Then the sets $P_0, P_1, \ldots, P_{n-1}$ partition S_n, and each P_a is a PSCA$(n, n-1, 1)$*. Consequently, $g(n, n-1) = 1$.*

Example 2.19. To illustrate Theorem 2.18, consider again the partition of B^3 into the four binary codes $W^{3,a}$ for $a = 0, 1, 2, 3$. Table 2.2 shows the partition of S_4 according to the eight possible values of the derivative (shown in the third row), and associates these values with the partition of B^3. This gives four sets P_a for $a = 0, 1, 2, 3$, each of which is a PSCA$(4, 3, 1)$. For example, $P_0 = \{1234, 2143, 3142, 3241, 4132, 4231\}$.

Table 2.2: The case $n = 4$ of Theorem 2.18

B^3							
$W^{3,0}$		$W^{3,1}$		$W^{3,2}$		$W^{3,3}$	
000	101	100	011	010	111	001	110
1234	2143	2134	1432	1324	4321	1243	3214
	3142	3124	2431	1423		1342	4213
	3241	4123	3421	2314		2341	4312
	4132			2413			
	4231			3412			
P_0		P_1		P_2		P_3	
S_4							

Theorem 2.18 is a special case of a more general result given in [Lev91, Thm. 3.1]. The original proof of this more general result is rather involved. We now provide a simple and direct proof of Theorem 2.18 using the framework of perfect sequence covering arrays, based on the following result on derivatives.

Lemma 2.20. *Let $x \in S_n$ and $y \in S_{n,n-1}$, where x covers y. Then $D(y)$ is a length $n-2$ subsequence of $D(x)$.*

Proof. Let $y = y_1 \cdots y_{n-1}$ and let $D(y) = z_1 \cdots z_{n-2}$. Let s be the symbol that must be inserted into y in order to obtain x. If s is inserted before y_1 or after y_{n-1}, then the result holds immediately. Otherwise, s is inserted between y_i and y_{i+1} for some i and then $D(x)$ is obtained from $D(y)$ by replacing z_i by symbols $u, v \in \{0, 1\}$. Since at least one of $y_{i+1} - s$ and $s - y_i$ has the same sign as $y_{i+1} - y_i$, at least one of u, v equals z_i and therefore $D(y)$ is a subsequence of $D(x)$. ■

Proof of Theorem 2.18. The $\{W^{n-1,a}\}_{0 \le a \le n-1}$ form a partition of B^{n-1}, so by the definition of P_a we have that the $\{P_a\}_{0 \le a \le n-1}$ form a partition of S_n.

Now let $a \in \{0, 1, \ldots, n-1\}$. We claim firstly that P_a covers each element of $S_{n,n-1}$ at most once, and secondly that P_a covers each element of $S_{n,n-1}$ at least once. This shows that P_a is a PSCA$(n, n-1, 1)$ and establishes the result.

To prove the first claim, suppose that n-sequences x_1 and x_2 in P_a cover the same element y of $S_{n,n-1}$. By Lemma 2.20, $D(y) \in B^{n-2}$ is then a length $n-2$ subsequence of both $D(x_1)$ and $D(x_2)$. Since $D(x_1), D(x_2) \in W^{n-1,a}$ by definition of P_a, and $W^{n-1,a}$ is

perfect in B^{n-1} code capable of correcting single deletions by Theorem 2.15, this shows that $x_1 = x_2$ and so proves the claim.

To prove the second claim, since each sequence in P_a covers exactly n elements of $S_{n,n-1}$ by Result 1.3 (3), the multiset of elements of $S_{n,n-1}$ collectively covered by P_a has size $|P_a|n$. The first claim shows that this multiset is actually a set, and that

$$|P_a|n \leq |S_{n,n-1}| \tag{2.8}$$

with equality if and only if the second claim holds. We have $n! = |S_n| = \sum_{a=0}^{n-1} |P_a|$ because the P_a partition S_n, and we have $|S_{n,n-1}| = n!$ by Result 1.3 (2). Sum (2.8) over a from 0 to $n-1$ and substitute to obtain the required equality. ■

From Results 2.1 and 2.2 and Theorem 2.18, we have $g(n,2) = g(n,n-1) = g(n,n) = 1$. Levenshtein conjectured there are no other values of k for which $g(n,k) = 1$.

Conjecture 2.21 (Levenshtein [Lev90, p. 140])**.** *If $k \notin \{2, n-1, n\}$, then $g(n,k) > 1$.*

In 1999, Mathon and van Trung [MvT99, p. 191] reported that van Trung had found a PSCA$(6,4,1)$ eight years earlier, implying that $g(6,4) = 1$ and so disproving Conjecture 2.21. They gave two inequivalent examples of a PSCA$(6,4,1)$, one of which we reproduce here. We shall discuss their search algorithm in Section 2.5.

Proposition 2.22. *[MvT99, p. 191] The following 24 sequences form a* PSCA$(6,4,1)$.

123456	612543	514623	415263
153624	163542	143265	564132
652134	216453	246135	425136
256314	236541	524316	642315
351264	361452	341625	465312
321546	632451	534261	435621

Mathon and van Trung [MvT99, Sect. 6] also used their search algorithm to show that $g(7,5) > 1$ and $g(8,6) > 1$. They concluded that $k = 4$ might be the only value of k for which Conjecture 2.21 fails.

Conjecture 2.23 (revised version of Conjecture 2.21 [MvT99, p. 198])**.** *If $k \notin \{2, 4, n-1, n\}$, then $g(n,k) > 1$.*

Conjecture 2.23 can be rephrased as: if $k \notin \{2,4\}$ and $n \notin \{k, k+1\}$, then $g(n,k) > 1$. Using the contrapositive of Result 1.11 on symbol deletion, we obtain the following equivalent form of Conjecture 2.23.

Conjecture 2.24 (equivalent form of Conjecture 2.23)**.** *Let $k \notin \{2,4\}$. Then $g(k+2,k) > 1$.*

More than twenty years after publication of [MvT99], the smallest open case for Conjecture 2.24 remains $k = 7$.

2.4 Combinatorial nonexistence results

In this section, we present combinatorial proofs for the nonexistence of a PSCA$(5, 3, 1)$ and a PSCA$(7, 4, 1)$, and state the nonexistence of a PSCA$(2k, k, 1)$ for $k \geq 3$.

2.4.1 Nonexistence of PSCA(5,3,1)

Theorem 2.25 (Mathon and van Trung [MvT99, Thm 3.2])**.** *There is no* PSCA$(5, 3, 1)$.

Proof. Suppose, for a contradiction, that P is a PSCA$(5, 3, 1)$. Since P contains $3! = 6$ sequences (by Result 1.9) that use only 5 symbols, some two sequences of P agree in their last symbol. We may assume after relabelling (see Result 1.7 (2)) that one of these sequences is 12345. Then the other sequence has the last symbol 5 and contains no 3-subsequence covered by 12345. The only choice is 43215, because no ascending 2-subsequence can occur prior to the symbol 5.

Now consider extending $\{12345, 43215\}$ to P by adding four further 5-sequences. One such sequence s must contain the 3-subsequence 354 not yet covered, but contain no 3-subsequence already covered. Regard s as being formed from 354 by adding symbols 1 and 2. By considering the possible positions of each of symbols 1 and 2 in turn, we see that there are only two choices for s, namely 35142 and 35412.

Finally consider extending $\{12345, 43215, s\}$ to P by adding three further 5-sequences. For each of the two choices of s, a similar argument shows that it is not possible to add even one 5-sequence to $\{12345, 43215, s\}$ that contains the 3-subsequence 534 not yet covered but contains no 3-subsequence already covered. This gives the required contradiction. ■

Remark 2.26.

1. By Results 1.11 and 2.1 and Theorem 2.18, Theorem 2.25 implies that $g(n, 3) = 1$ if and only if $n \in \{3, 4\}$.

2. The original proof of [MvT99, Thm. 3.2] (phrased using the symbol set $\{a, b, c, d, x\}$ rather than $\{1, 2, 3, 4, 5\}$) overlooked the possibility that $s = 35412$.

3. The proof strategy in Theorem 2.25 is to demonstrate the nonexistence of a PSCA$(n, k, 1)$ by attempting to build the perfect sequence covering array one sequence at a time, keeping track of all possible search branches. For $n = k + 2$ and $k > 3$, this procedure is difficult to manage by hand, but can be implemented as an efficient computer search. Mathon and van Trung [MvT99, Sect. 6] used this strategy to determine nonexistence of a PSCA$(k + 2, k, 1)$ for $k = 5$ and $k = 6$ (as mentioned after Conjecture 2.21). We shall discuss their algorithm in Section 2.5.

2.4.2 Nonexistence of PSCA(7,4,1)

Theorem 2.27 (Klein [Kle04, Thm 3.2])**.** *There is no* PSCA$(7, 4, 1)$.

Proof. Suppose, for a contradiction, that P is a PSCA$(7, 4, 1)$. We may assume after relabelling that P contains the sequence 1234567. Let T be the set of elements in $P \setminus \{1234567\}$ that contain one of the 3-subsequences in the set

$$U = \{124, 134, 234\},$$

and let T' be the set of elements in $P \setminus \{1234567\}$ that contain one of the 4-subsequences in the set

$$U' = \{3124, 1324, 1243, 2134, 1342, 2314, 2341\}.$$

It is easy to check that $T = T'$. By the PSCA property, the set $P \setminus \{1234567\}$ covers each element of U', so there are at least $|U'| = 7$ elements in $T' = T$.

Now in the sequence $1234567 \in P$, each of the symbols 5, 6, 7 occurs after each of the 3-subsequences in U. Therefore in every element of T, each symbol $5, 6, 7$ occurs before the symbol 4 (otherwise P would cover some 4-subsequence more than once). Since there are at least 7 elements of T, but there are only $3! < 7$ ways to order the symbols $5, 6, 7$ to occur before the symbol 4, we conclude that there are two distinct elements of T covering the same 4-subsequence (formed from some permutation of symbols $5, 6, 7$ followed by the symbol 4). This gives the required contradiction. ■

Remark 2.28.

1. By Results 1.11 and 2.1 and Proposition 2.22, Theorem 2.27 implies that $g(n, 4) = 1$ if and only if $n \in \{4, 5, 6\}$.

2. Theorem 2.27 was established by hand in 2004 [Kle04] using the proof rephrased here, although the result was known earlier from computer search [MvT99, Thm. 4.2].

3. The final step in the proof of Theorem 2.27 uses the pigeonhole principle: there are at least 7 elements of T, but only $3! < 7$ ways to order the symbols 5,6,7 to occur before the symbol 4. This method does not generalize to other parameter sets in an obvious way. For example, the corresponding step for analyzing a PSCA$(9, 5, 1)$ would be that there are at least 13 elements of $P \setminus \{123456789\}$ that contain one of the 4-subsequences in the set $U = \{1235, 1245, 1345, 2345\}$, but there are $4! > 13$ ways to order the symbols 6,7,8,9 to occur before the symbol 5.

2.4.3 Nonexistence of PSCA$(2k, k, 1)$

Theorem 2.29 (Chee, Colbourn, Horsley, Zhou [CCHZ13, Thm 2.3])**.** *For each* $k \geq 3$*, there is no* PSCA$(2k, k, 1)$.

We omit the proof of Theorem 2.29. It involves matrix rank arguments, and appeals to several previous results on covering arrays such as [Col11].

Remark 2.30.

1. Theorem 2.29 improves on an earlier result due to Mathon and van Trung [MvT99, Thm 2.1], that for each $k \geq 7$ there is no PSCA$(\frac{k(k+1)}{2}, k, 1)$.

2. Chee el al. remark [CCHZ13, p. 1849] that "it appears plausible" that Theorem 2.29 can be strengthened to: for each $k \geq 3$, there is no PSCA$(k+3, k, 1)$.

2.5 Mathon and van Trung search algorithm for PSCA$(n, k, 1)$

In this section, we describe a recursive algorithm due to Mathon and van Trung [MvT99] that finds all possible examples of a PSCA$(n, k, 1)$. The algorithm is a tree search that attempts to build a PSCA$(n, k, 1)$ one n-sequence at a time without covering any k-subsequence more than once, backtracking when this is not possible. If it terminates without output, then we can conclude that no PSCA$(n, k, 1)$ exists and so $g(n, k) > 1$.

This algorithm was originally implemented to search for a directed t-packing (see Definition 1.13), and gave the following results.

Result 2.31 (Search results in [MvT99, Sect. 4 & 6]). We have $g(6, 4) = 1$, $g(7, 4) > 1$, $g(7, 5) > 1$, and $g(8, 6) > 1$.

An example of a PSCA$(6, 4, 1)$ found using the algorithm of [MvT99] was given in Proposition 2.22. A theoretical proof of the nonexistence of a PSCA$(7, 4, 1)$ was given in 2004 by Klein [Kle04]: see Theorem 2.27. Mathon and van Trung reported in 1999 that, using an Ultra SPARCstation 5, the time required to establish $g(7, 5) > 1$ and $g(8, 6) > 1$ was 5 minutes and 100 hours, respectively.

The search method of Mathon and van Trung [MvT99] relies on an algorithm for finding spreads of an incidence structure given earlier by Mathon [Mat97]. We shall recast the spread algorithm of [Mat97] in Algorithm 1 to refer specifically to the search for a PSCA$(n, k, 1)$, and to reference directly the (n, k)-incidence matrix defined below. We shall extend Algorithm 1 in Section 3.1 to search for a PSCA(n, k, λ) for $\lambda > 1$.

2.5.1 Repetition vector and incidence matrix

We first introduce the concept of a repetition vector to describe the k-subsequences covered by a set of n-sequences. We create a one-to-one correspondence between the elements of $S_{n,k}$ taken in lexicographic order and the set $[\binom{n}{k}k!]$, and between the elements of S_n taken in lexicographic order and the set $[n!]$. For example, for $(n, k) = (3, 2)$, the one-to-one correspondence between $S_{3,2}$ and $[6]$ is

$$12 \leftrightarrow 1, \quad 13 \leftrightarrow 2, \quad 21 \leftrightarrow 3, \quad 23 \leftrightarrow 4, \quad 31 \leftrightarrow 5, \quad 32 \leftrightarrow 6,$$

and the one-to-one correspondence between S_3 and [6] is

$$123 \leftrightarrow 1, \quad 132 \leftrightarrow 2, \quad 213 \leftrightarrow 3, \quad 231 \leftrightarrow 4, \quad 312 \leftrightarrow 5, \quad 321 \leftrightarrow 6.$$

Definition 2.32 (repetition vector)**.** Let I be a subset of $[n!]$. The *repetition vector of* I with respect to n and k, denoted $\mathrm{rv}_{n,k}(I)$, is a length $\binom{n}{k}k!$ vector of non-negative integers whose j^{th} entry is λ when the set of n-sequences corresponding to I covers the k-subsequence corresponding to j with multiplicity λ (see Definition 1.2).

By Result 1.3 (3), the sum of the entries of $\mathrm{rv}_{n,k}(I)$ is $\binom{n}{k}|I|$. A subset I of $[n!]$ corresponds to a PSCA(n, k, λ) if and only if each entry of $\mathrm{rv}_{n,k}(I)$ is λ.

We now introduce the (n, k)-incidence matrix as the fundamental object on which Algorithm 1 operates.

Definition 2.33 (incidence matrix)**.** The (n, k)-*incidence matrix* is the $n! \times \binom{n}{k}k!$ array over $\{0, 1\}$ whose i^{th} row is $\mathrm{rv}_{n,k}(\{i\})$.

Remark 2.34. The (n, k)-incidence matrix is the adjacency matrix of the bipartite graph with vertex sets S_n and $S_{n,k}$, where vertices $x \in S_n$ and $y \in S_{n,k}$ are adjacent when x covers y.

For a subset $I \subset [n!]$, the vector $\mathrm{rv}_{n,k}(I)$ is the sum (over $\mathbb{Z}$ rather than GF(2)) of the rows of the (n, k)-incidence matrix indexed by I.

Example 2.35. The $(4, 3)$-incidence matrix is given in the next page (see Table 2.3). It has $4! = 24$ rows and $\binom{4}{3}3! = 24$ columns. To assist in verifying the entries of this matrix, we have displayed the n-sequence corresponding to each row index, and the k-subsequence (read vertically from top to bottom) corresponding to each column index. The vector

$$\mathrm{rv}_{4,3}(\{8, 11, 12\}) = (0, 0, 0, 0, 0, 1, 2, 1, 1, 0, 2, 3, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0)$$

is the sum of rows $8, 11, 12$ of the matrix.

Table 2.3: The (4,3)-incidence matrix

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
		1	1	1	1	1	1	2	2	2	2	2	2	3	3	3	3	3	3	4	4	4	4	4	4
		2	2	3	3	4	4	1	1	3	3	4	4	1	1	2	2	4	4	1	1	2	2	3	3
		3	4	2	4	2	3	3	4	1	4	1	3	2	4	1	4	1	2	2	3	1	3	1	2
1	1234	1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1243	1	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
3	1324	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
4	1342	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
5	1423	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
6	1432	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
7	2134	0	0	0	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2143	0	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
9	2314	0	0	0	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
10	2341	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0
11	2413	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0
12	2431	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0
13	3124	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0
14	3142	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0
15	3214	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
16	3241	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0
17	3412	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0
18	3421	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0	0	0
19	4123	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
20	4132	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1
21	4213	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
22	4231	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
23	4312	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	1
24	4321	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	1

2.5.2 Idea of algorithm

Let $A = (a_{i,j})$ be the (n,k)-incidence matrix. We wish to construct a PSCA$(n,k,1)$ by finding a $k!$-subset $I \subset [n!]$ for which each entry of $\mathrm{rv}_{n,k}(I)$ is 1. Since $\mathrm{rv}_{n,k}(I)$ is the sum of the rows of the (n,k)-incidence matrix indexed by I, we therefore wish to choose a $k!$-subset $I \subset [n!]$ such that

$$\sum_{i \in I} a_{i,j} = 1 \quad \text{for each } j.$$

We initialize I to be empty. We then add one index to I at a time, subject to the condition that

$$\sum_{i \in I} a_{i,j} \leq 1 \quad \text{for each } j. \tag{2.9}$$

The algorithm succeeds in finding a PSCA$(n, k, 1)$ if $|I|$ reaches $k!$. If it is not possible to add an index to I subject to (2.9), we backtrack.

We also keep track of two sets, J and L. The set J contains the indices of k-subsequences not yet covered (by the n-sequences indexed by I), namely the values j for which $\sum_{i \in I} a_{i,j} = 0$. The set L contains the candidates for enlarging I, namely the indices of n-sequences that do not cover a k-subsequence already covered.

Suppose that I currently contains fewer than $k!$ indices. For each $j \in J$, let $L(j)$ be the set of indices $\ell \in L$ for which $a_{l,j} = 1$ (that is, whose addition to the set I will cause the j^{th} k-subsequence to be covered). At the next iteration of the algorithm, we choose one $j \in J$ and recurse by attempting to add to I each of the elements of $L(j)$ in turn. In order to reduce the number of tree branches that must be searched, we choose a value of $j \in J$ that minimizes $|L(j)|$: call this value j^*.

For each $i \in L(j^*)$ in turn, we update the sets I, J, L and then recurse. We update I by adding the index i. We update J by removing the indices of all k-subsequences newly covered by the addition of i to I. We update L by removing the indices of all n-sequences having a k-subsequence in common with the i^{th} n-sequence, whose inclusion in I would violate (2.9).

2.5.3 Pseudocode

Algorithm 1 Tree search for PSCA$(n, k, 1)$ by backtracking

Input: PSCA parameters (n, k, λ) and $(a_{i,j}) = (n, k)$-incidence matrix

procedure SEARCH(I, J, L)
    % I = indices of sequences of partial PSCA$(n, k, 1)$ P
    % J = indices of k-subsequences not yet covered by P
    % L = candidates for enlarging I, namely $\{\ell \notin I : a_{\ell,j} = 0 \text{ for all } j \notin J\}$
    if $|I| = k!$ **then**
        record I as the index set of a PSCA$(n, k, 1)$.
        return
    end if
    for each $j \in J$ **do**
        record $|L(j)|$, where $L(j) \coloneqq \{\ell \in L : a_{\ell,j} = 1\}$.
    end for
    choose an arbitrary $j = j^*$ for which $|L(j^*)| = \min_{j \in J} |L(j)|$.
    for each $i \in L(j^*)$ **do**
        $I_{\text{new}} \coloneqq I \cup \{i\}$
        $J_{\text{new}} \coloneqq J \setminus \{j : a_{i,j} = 1\}$
        $L_{\text{new}} \coloneqq L \setminus \{\ell : a_{\ell,j} = a_{i,j} = 1 \text{ for some } j\}$
        SEARCH$(I_{\text{new}}, J_{\text{new}}, L_{\text{new}})$
    end for
end procedure

By calling SEARCH with $I = \emptyset$ and $J = [\binom{n}{k}k!]$ and $L = [n!]$, we obtain every possible PSCA$(n, k, 1)$. Alternatively, we may instead assume after relabelling (see Result 1.7 (2)) that the perfect sequence covering array contains the sequence $12\cdots n$ and so call SEARCH with $I = \{1\}$ and $J = [\binom{n}{k}k!] \setminus \{j : a_{1,j} = 1\}$ and $L = [n!] \setminus \{\ell : a_{\ell,j} = 1 \text{ and } a_{1,j} = 1 \text{ for some } j\}$.

2.5.4 Example: search for PSCA(4,3,1)

We illustrate Algorithm 1 by applying it to search for all possible examples of a PSCA$(4, 3, 1)$. We provide a 2-step visualization of each iteration of this recursive algorithm. Step 1 of each iteration corresponds to finding j^* and $L(j^*)$ (Lines 9–12), and Step 2 corresponds to updating the sets I, J, and L (Lines 14–16).

We begin with the (4,3)-incidence matrix (see Table 2.3). We call the search procedure at Line 1 with $I = \emptyset$, and $J = [24]$, and $L = [24]$.

Iteration 1. In Step 1, at Line 10 we calculate $L(1) = \{1, 2, 5, 19\}$, $L(2) = \{1, 2, 3, 13\}$, $\ldots$, $L(24) = \{6, 20, 23, 24\}$ which gives that $|L(j)|$ (the sum of the j^{th} column of the incidence matrix) equals 4 for each j. We may therefore choose j^* freely at Line 12. Suppose we choose $j^* = 1$.

At Line 13, we recurse over $i \in L(1) = \{1, 2, 5, 19\}$. We shall illustrate the branch $i = 1$.

At Step 2, we set I_{new} to be $\{1\}$, which we represent by colouring row 1 red in Table 2.4. We then determine J_{new} by removing from $J = [24]$ the set $\{1, 2, 4, 10\}$ of column positions (shown in pink in Table 2.4) at which row 1 contains the entry 1 (shown in red): we represent this by deleting these columns from the matrix. We then determine L_{new} by removing from $L = [24]$ the set $\{1, 2, 3, 4, 5, 7, 9, 10, 13, 19\}$ of row positions (shown in orange in Table 2.4) which contain an entry 1 in at least one of the pink columns $\{1, 2, 4, 10\}$: we represent this by deleting these rows from the matrix. (In particular, we remove the row position $i = 1$ from L.) This leaves a 14×20 matrix for the next iteration.

Table 2.4: Algorithm 1, Iteration 1, Step 2

L \ J	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
3	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
4	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
5	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
6	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
7	0	0	0	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0
11	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0
12	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0
13	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0
14	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0
15	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0	0	0
19	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0
20	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1
21	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
22	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
23	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	1
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	1

Iteration 2. In Step 1, we calculate the $|L(j)|$ as the column sums of the updated matrix, displayed here as an additional row. Then $|L(j)|$ attains its minimum value of 2 (shown in green in Table 2.5) at columns $j \in \{3, 5, 6, 8, 9, 14, 16, 22\}$ (shown in light blue in Table 2.5). We may choose j^* arbitrarily in this set. Suppose we choose $j^* = 6$.

At Line 13, we recurse over $i \in L(6) = \{6, 8\}$ (shown in dark blue in Table 2.5); we shall illustrate the branch $i = 8$.

Table 2.5: Algorithm 1, Iteration 2, Step 1

L \ J	3	5	6	7	8	9	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1																				
6	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
8	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0
12	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0
14	0	1	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0
15	0	0	0	0	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0	0	0
20	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1
21	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
22	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0
23	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	1
24	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	1
$\|L(j)\|$	2	2	2	3	2	2	3	3	3	2	4	2	3	3	3	3	4	2	4	4

In Step 2, we include 8 in set I (shown in red). We remove from J the set {6, 7, 8, 12} of pink column positions at which row 8 contains the entry 1 (shown in red), and remove from L the set $\{6, 8, 11, 12, 15, 21\}$ of orange row positions containing an entry 1 in at least one of the pink columns. This leaves an 8×16 matrix for the next iteration. The remaining iterations proceed similarly.

Table 2.6: Algorithm 1, Iteration 2, Step 2

L \ J	3	5	6	7	8	9	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1																				
6	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
8	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0
12	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0
14	0	1	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0
15	0	0	0	0	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0	0	0
20	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1
21	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
22	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
23	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	1
24	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	1

Iteration 3. In Step 1, the green minimum of the column sums $|L(j)|$ is 1. Suppose we choose $j^* = 14$.

Table 2.7: Algorithm 1, Iteration 3, Step 1

L \ J	3	5	9	11	13	14	15	16	17	18	19	20	21	22	23	24
1																
8																
14	0	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0
16	0	0	0	1	0	0	1	1	1	0	0	0	0	0	0	0
17	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0
18	0	0	0	0	0	0	1	0	1	1	0	0	1	0	0	0
20	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1
22	0	0	1	0	0	0	0	0	0	0	0	0	1	1	1	0
23	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	1
24	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	1
$\|L(j)\|$	1	1	1	1	3	1	3	1	3	3	3	1	3	1	3	3

In Step 2, we recurse over the single choice $i \in \{14\}$ at Line 13. We remove four pink column positions and four orange row positions from the matrix. This leaves a 4×12 matrix for the next iteration.

Table 2.8: Algorithm 1, Iteration 3, Step 2

L \ J	3	5	9	11	13	14	15	16	17	18	19	20	21	22	23	24
1																
8																
14	0	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0
16	0	0	0	1	0	0	1	1	1	0	0	0	0	0	0	0
17	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0
18	0	0	0	0	0	0	1	0	1	1	0	0	1	0	0	0
20	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1
22	0	0	1	0	0	0	0	0	0	0	0	0	1	1	1	0
23	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	1
24	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	1

Iteration 4. In Step 1, the green minimum of the column sums $|L(j)|$ is 1. Suppose we choose $j^* = 9$. At Line 13, we recurse over the single choice $i \in \{22\}$.

Table 2.9: Algorithm 1, Iteration 4, Step 1

L \ J	3	9	11	15	16	17	19	20	21	22	23	24
1												
8												
14												
16	0	0	1	1	1	1	0	0	0	0	0	0
20	1	0	0	0	0	0	1	1	0	0	0	1
22	0	1	0	0	0	0	0	0	1	1	1	0
24	0	0	0	1	0	0	0	0	1	0	1	1
$\|L(j)\|$	1	1	1	2	1	1	1	1	2	1	2	2

In Step 2, we include 22 in set I and remove four pink column positions and two orange row positions from the matrix. This leaves a 2×8 matrix for the next iteration.

Table 2.10: Algorithm 1, Iteration 4, Step 2

L \ J	3	9	11	15	16	17	19	20	21	22	23	24
1												
8												
14												
16	0	0	1	1	1	1	0	0	0	0	0	0
20	1	0	0	0	0	0	1	1	0	0	0	1
22	0	1	0	0	0	0	0	0	1	1	1	0
24	0	0	0	1	0	0	0	0	1	0	1	1

Iterations 5 and 6. We repeat this process until the number of red rows reaches $3! = 6$. If successful, we record these rows at Line 6 as indexing the 4-sequences of a PSCA$(4, 3, 1)$. If at any stage $|L(j)| = 0$ for some j, then $|L(j^*)| = 0$ at Line 12 and the loop at Lines 13–18 is empty and we backtrack.

In the present example, it is easy to check that both of the remaining rows $16, 20$ will be included in the set I. The resulting index set $I = \{1, 8, 14, 16, 20, 22\}$ corresponds to the subset $P = \{1234, 2143, 3142, 3241, 4132, 4231\}$ of S_4, which is a PSCA$(4, 3, 1)$. The repetition vector $\mathrm{rv}_{4,3}(I)$ is the sum of the six rows of the $(4, 3)$-incidence matrix indexed by I, and each of its entries is 1 as shown below.

I	P	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	1234	1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2143	0	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
14	3142	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0
16	3241	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0
20	4132	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1
22	4231	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
$\mathrm{rv}_{4,3}(I)$		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

2.5.5 Comments

1. Algorithm 1 is derived from an algorithm due to Mathon [Mat97] for finding spreads in an incidence structure. We believe this algorithm from 1997 still has considerable value. Mathon observed [Mat97, Sec. 2, p.164] that

   > "An actual implementation of this algorithm requires good data structures to facilitate fast and efficient computations in, and updating of the various point and line sets."

However, [Mat97] does not explicitly describe suitable data structures for an efficient implementation.

2. The removal of rows and columns of the incidence matrix at successive iterations of Algorithm 1 substantially reduces the search time. The time complexity of the algorithm is then dominated by the determination at Line 10 of $|L(j)|$ for each $j \in J$, which requires the calculation of $|J|$ column sums of the remaining matrix. The only value of j for which $L(j)$ is explicitly required is $j = j^*$, at Line 13.

3. In Section 3.1, we shall obtain a significant improvement in the time complexity of Algorithm 1 by passing the vector

$$M := (|L(j)| : j \in J_{\text{new}})$$

as an additional recursion parameter at Line 17. This improvement is not described in [Mat97] or [MvT99].

4. The space complexity of Algorithm 1 is determined by the representation of the (n, k)-incidence matrix. There is no need to store this matrix explicitly as an $n! \times \binom{n}{k}k!$ matrix over $\{0, 1\}$: it is sufficient to store the positions of the 1 entries in each row, and the positions of the 1 entries in each column.

5. Despite the improvements described here and later, there is still a combinatorial explosion in both time and space complexity of Algorithm 1 as n or k increases.

6. When Algorithm 1 is called with $I = \emptyset$ and $J = [\binom{n}{k}k!]$ and $L = [n!]$, the initial determination at Line 10 of $|L(j)|$ for each $j \in J$ can be bypassed because, by Result 1.3 (4), each such $|L(j)|$ is equal to $\binom{n}{k}(n-k)!$. In the example of Section 2.5.4, this corresponds to setting $|L(j)| = 4$ for each $j \in [24]$ at Step 1 of Iteration 1 without explicitly determining the $L(j)$.

2.6 Yuster construction method for PSCA(5,3,2)

In 2020, Yuster [Yus20] determined the first known exact g-value greater than 1.

Result 2.36 ([Yus20, Prop. 3.4]). We have $g(5, 3) = 2$.

Yuster established Result 2.36 by exhibiting a PSCA$(5, 3, 2)$ and combining with Theorem 2.25. In this section, we describe Yuster's method for constructing a PSCA$(5, 3, 2)$. We shall recast this method in Section 3.3.

We first extend Definition 2.32 for a repetition vector, from a subset of $[n!]$ to a subset of S_n.

Definition 2.37 (repetition vector of a set of n-sequences)**.** Let X be a subset of S_n whose corresponding index set (see Section 2.5.1) is I. The *repetition vector of* X with respect to k is $\mathrm{rv}_k(X) \coloneqq \mathrm{rv}_{n,k}(I)$.

For example, $\mathrm{rv}_3(\{1342, 2134\}$ is the sum of rows 4 and 7 of the $(4, 3)$-incidence matrix given in Table 2.3.

Remark 2.38.

1. We write the repetition vector $\mathrm{rv}_{n,k}(I)$ for a subset I of $[n!]$ with both n and k as subscripts (for example, $\mathrm{rv}_{5,3}(\{1, 6, 43\})$) because their values cannot be determined from the subset. However, we write the repetition vector $\mathrm{rv}_k(X)$ for a subset X of S_n with only k as a subscript (for example, $\mathrm{rv}_3(\{12345, 12543, 25134\})$ because the value of n is apparent from the subset.

2. We previously used the repetition vector $\mathrm{rv}_{n,k}(I)$ to describe Algorithm 1, which relies on the structure of the (n, k)-incidence matrix but not the row labels from S_n nor the column labels from $S_{n,k}$. We shall now use the repetition vector $\mathrm{rv}_k(X)$ to describe the Yuster construction method, and later in Section 2.8 in connection with the action of a permutation on a set of sequences.

Yuster's method was to use computer search to find a set X of six 5-sequences for which the length 60 vector $\mathrm{rv}_3(X)$ contains the largest possible number of nonzero entries (namely 56), and a set Y of six 5-sequences (depending on X and disjoint from X) such that each entry of $\mathrm{rv}_3(X) + \mathrm{rv}_3(Y)$ is 2: then $X \cup Y$ is a PSCA$(5, 3, 2)$.

Yuster obtained the set Y by applying a permutation to each of the sequences of X (regarded as elements of S_5). Following [Yus20] (as well as [MvT99, GAP20]), we shall use the following convention for the composition of permutations.

Definition 2.39 (Composition of permutations)**.** Let $\pi, \sigma \in S_n$ be permutations. We write $\pi \cdot \sigma$ (or simply $\pi\sigma$) for π followed by σ. For a set X of permutations, we write $X\sigma \coloneqq \{x\sigma : x \in X\}$ and $\pi X \coloneqq \{\pi x : x \in X\}$.

Example 2.40. In S_5, we have $13452 \cdot 53124 = 51243$ and $53124 \cdot 13452 = 24135$.

Proposition 2.41 (Yuster [Yus20])**.** *Let* $\sigma = 13254$ *and*

$$X = \{12345, 43215, 35214, 14523, 25413, 53412\},$$
$$X\sigma = \{13254, 52314, 24315, 15432, 34512, 42513\}.$$

Then $X \cup X\sigma$ *is a* PSCA$(5, 3, 2)$.

For the set X in Proposition 2.41, the repetition vector $\mathrm{rv}_3(X)$ has four entries 0 (corresponding to the 3-subsequences 132, 231, 154, 451), four entries 2 (corresponding

to the 3-subsequences 123, 321, 145, 541), and the remaining 52 entries 1 (for a total of 56 nonzero entries). The set $X\sigma$ has the same property, but the positions in $\mathrm{rv}_3(X\sigma)$ of the 0 and 2 entries are interchanged with those in $\mathrm{rv}_3(X)$. This ensures that every entry of $\mathrm{rv}_3(X) + \mathrm{rv}_3(X\sigma)$ is 2, and therefore that $X \cup X\sigma$ is a PSCA$(5, 3, 2)$.

Yuster [Yus20, Sec. 4, p.592] concluded:

> "Proving additional exact values of $g(n, k)$ which are not of unit multiplicity in addition to $g(5, 3)$ also seems challenging".

2.7 Table of known g-values

We summarize in Table 2.11 all previous results on the exact value of $g(n, k)$ for small n and k.

Table 2.11: Previously known g-values

n \ k	2	3	4	5	6	7
2	1					
3	1	1				
4	1	1	1			
5	1	2*	1	1		
6	1	>1	1	1	1	
7	1		> 1◇	>1	1	1
8	1		>1		>1	1
9	1					
10	1			>1		

$k = 2$ and $n = k$ (trivial constructions of Results 2.1 and 2.2)
Theorem 2.18 (1991 construction by Levenshtein [Lev91])
Result 2.31 (1999 computational results by Mathon and van Trung [MvT99])
◇ Theorem 2.27 (2004 theoretical result by Klein [Kle04])
Theorem 2.29 (2013 theoretical result by Chee, Colbourn, Horsley, Zhou [CCHZ13])
* Result 2.36 (2020 example constructed by Yuster [Yus20])

2.8 Action of a permutation on a set of sequences

In this section, we regard a set of n-sequences as a set of permutations in S_n, and thereby connect combinatorial properties to group properties (see Section 1.2).

We firstly note that the right action of a permutation on a set X of n-sequences relabels the symbols of each of the elements of X, whereas the left action permutes the positions of each of the symbols (see the convention of Definition 2.39). This allows us to rephrase Definition 1.8 as: a PSCA(n, k, λ) P is equivalent to a PSCA(n, k, λ) Q if, for some $\pi \in$

$\{12\cdots n,\ n\cdots 21\}$ and some $\sigma \in S_n$, we have $Q = \pi P\sigma$. We can likewise rephrase Result 1.7 as follows.

Result 2.42. Suppose P is a PSCA(n, k, λ), and let $\sigma \in S_n$. Then $P\sigma$ is also a PSCA(n, k, λ).

We next note that the right action of a permutation on a set X of n-sequences permutes the entries of the repetition vector $\mathrm{rv}_k(X)$.

Lemma 2.43. *Let X be a set of n-sequences and let $\sigma \in S_n$. Then the vector $\mathrm{rv}_k(X\sigma)$ is obtained by permuting the entries of the vector $\mathrm{rv}_k(X)$.*

Proof. The right action of σ on the set X permutes the symbols in $[n]$ and so permutes the elements of $S_{n,k}$. The result follows by Definitions 2.32 and 2.37. ■

Example 2.44. Let $X = \{1234, 1324, 3142\}$ be a subset of S_4 and $\sigma = 3142$, so that $X\sigma = \{3142, 3412, 4321\}$. Then the repetition vectors $\mathrm{rv}_3(X)$ and $\mathrm{rv}_3(X\sigma)$ are shown in the table below, which displays the column indices in the first row and their corresponding 3-subsequences in the next three rows. We can calculate $\mathrm{rv}_3(X\sigma)$ directly from $\mathrm{rv}_3(X)$ by applying σ to each of the 3-subsequence column labels: for example, the entry of $\mathrm{rv}_3(X)$ corresponding to column label 134 (in position 4) maps to the entry of $\mathrm{rv}_3(X\sigma)$ corresponding to column label $134 \cdot \sigma = 342$ (in position 18).

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
	1	1	1	1	1	1	2	2	2	2	2	2	3	3	3	3	3	3	4	4	4	4	4	4
	2	2	3	3	4	4	1	1	3	3	4	4	1	1	2	2	4	4	1	1	2	2	3	3
	3	4	2	4	2	3	3	4	1	4	1	3	2	4	1	4	1	2	2	3	1	3	1	2
$\mathrm{rv}_3(X)$	=(1,	2,	1,	2,	1,	0,	0,	0,	0,	1,	0,	0,	1,	1,	0,	1,	0,	1,	0,	0,	0,	0,	0,	0)
$\mathrm{rv}_3(X\sigma)$	=(0,	0,	0,	0,	1,	0,	0,	0,	0,	0,	0,	0,	2,	1,	1,	0,	1,	2,	1,	0,	1,	0,	1,	1)

Mathon and van Trung [MvT99, Sect. 5] considered the action of a permutation $\sigma \in S_n$ on the elements of a directed k-packing (see Definition 1.13). They noted [MvT99, Thm. 5.1] that the right action of σ maps a directed k-packing to another directed k-packing, whereas the left action does not. More generally, the left action of $\sigma \in S_n \setminus \{n\cdots 21\}$ on a set X of n-sequences does not necessarily permute the entries of $\mathrm{rv}_k(X)$. In particular, the left action of σ on a PSCA(n, k, λ) does not necessarily give another PSCA(n, k, λ).

Example 2.45. Let $X = \{1234, 1432\} \subset S_4$ and $\sigma = 1324 \in S_4$, so that $\sigma X = \{1324, 1342\}$. Then the vector

$$\mathrm{rv}_3(\sigma X) = (0, 1, 2, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0)$$

is not obtained by permuting the entries of the vector

$$\mathrm{rv}_3(X) = (1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1).$$

As well as proving that $g(6,4) = 1$ (see Result 2.31), Mathon and van Trung classified all possible examples of a PSCA$(6,4,1)$.

Theorem 2.46 ([MvT99, Thm 4.1])**.** *Up to equivalence, there are exactly two examples P_1, P_2 of a* PSCA$(6,4,1)$*:*

(1) the 24 *sequences of P_1 form a subgroup of S_6 isomorphic to S_4; the automorphism group of P_1 is isomorphic to S_4.*

(2) the 24 *sequences of P_2 form three right cosets of a subgroup G of S_6 isomorphic to D_8 (the dihedral group of order* 8*); the automorphism group of P_2 is isomorphic to D_8.*

We note a presentation error in the explicit listing of the subgroup $G = D$ and its two right cosets D_1, D_2 illustrating Theorem 2.46 (2) in [MvT99, p. 192 and p. 195]: the listed coset representatives $x_1 = 0\ 3\ 2\ 5\ 1\ 4$ and $x_2 = 3\ 0\ 4\ 5\ 1\ 2$ give identical cosets D_1, D_2 (as sets). We can correct this error by replacing x_2 with $x_2' = 1\ 3\ 5\ 0\ 2\ 4$ and then replacing the given set D_2 by the set

$$
\begin{array}{c}
D_2' \\
1\ 3\ 5\ 0\ 2\ 4 \\
1\ 4\ 5\ 2\ 0\ 3 \\
3\ 1\ 4\ 0\ 2\ 5 \\
3\ 5\ 4\ 2\ 0\ 1 \\
4\ 1\ 3\ 2\ 0\ 5 \\
4\ 5\ 3\ 0\ 2\ 1 \\
5\ 3\ 1\ 2\ 0\ 4 \\
5\ 4\ 1\ 0\ 2\ 3.
\end{array}
$$

Mathon and van Trung [MvT99, p. 196] also found that, although a PSCA$(8,6,1)$ does not exist (see Result 2.31), there is a 304-subset P of S_8 for which $\mathrm{rv}_6(P)$ has each entry at most 1, and P is the union of 38 left cosets a subgroup $G \cong C_2^3$ of S_8.

In Chapter 3, we shall seek a PSCA(n,k,λ) as a union of cosets of a subgroup of S_n.

Chapter 3

New Results

3.1 Revised Mathon and van Trung search algorithm for general λ

In Section 2.5, we described Algorithm 1 for finding all possible examples of a PSCA$(n, k, 1)$. In this section, we modify this algorithm to Algorithm 2 for finding all possible examples of a PSCA(n, k, λ) for arbitrary $\lambda \geq 1$. The modified algorithm attempts to build a PSCA(n, k, λ) one n-sequence at a time without covering any k-subsequence more than λ times, backtracking when this is not possible. If it terminates without output, then we can conclude that no PSCA(n, k, λ) exists and so $g(n, k) \neq \lambda$.

By using Algorithm 2, we find the following examples of a PSCA$(6, 3, 2)$ and a PSCA$(7, 3, 2)$. Since $g(5, 3) > 1$ by Theorem 2.25, this gives the new g-values $g(6, 3) = g(7, 3) = 2$ via Result 1.11.

Proposition 3.1.

(1) The following 12 *sequences form a* PSCA$(6, 3, 2)$

123456	154326	216543	245613	354162	361452
423165	461325	516234	532614	632541	645231

and therefore $g(6, 3) = 2$.

(2) The following 12 *sequences form a* PSCA$(7, 3, 2)$

1234567	1573426	3275641	3617524	4261735	4756123
5243176	5164327	6257314	6345721	7216453	7431625

and therefore $g(7, 3) = 2$.

3.1.1 Idea of algorithm

Let $A = (a_{i,j})$ be the (n,k)-incidence matrix. We wish to construct a PSCA(n,k,λ) by finding a $k!\lambda$-subset $I \subset [n!]$ for which each entry of $\mathrm{rv}_{n,k}(I)$ is λ, so that

$$\sum_{i \in I} a_{i,j} = \lambda \quad \text{for each } j.$$

We initialize I to be empty. We then add one index to I at a time, subject to the condition that

$$\sum_{i \in I} a_{i,j} \leq \lambda \quad \text{for each } j. \tag{3.1}$$

The algorithm succeeds in finding a PSCA(n,k,λ) if $|I|$ reaches $k!\lambda$. If it is not possible to add an index to I subject to (3.1), we backtrack.

We keep track of sets J and L. The set J contains the indices of k-subsequences not yet covered λ times, namely the values j for which $\sum_{i \in I} a_{i,j} < \lambda$. The set L contains the candidates for enlarging I, namely the indices of n-sequences that do not cover a k-subsequence already covered λ times. We also keep track of the repetition vectors $R = \mathrm{rv}_{n,k}(I) = (r_j) = (\sum_{i \in I} a_{i,j})$ and $M = \mathrm{rv}_{n,k}(L) = (m_j) = (\sum_{\ell \in L} a_{l,j})$, although only at the positions indexed by J.

Suppose that I currently contains fewer than $k!\lambda$ indices. If there is a k-subsequence that cannot be covered λ times, even by adding every candidate in L to I (that is, $r_j + m_j < \lambda$ for some $j \in J$), then we terminate the branch early and backtrack. Otherwise, we find the set J' of indices $j \in J$ at which r_j attains its maximum value (that is, the indices of k-subsequences that are not yet covered λ times but are closest to being so). At the next iteration of the algorithm, we choose one $j \in J'$ and recurse by attempting to add to I each of the elements of $\{\ell \in L : a_{l,j} = 1\}$ in turn (each such addition causing the j^{th} k-subsequence to be covered one more time). In order to reduce the number of tree branches that must be searched, we choose a value of $j \in J'$ at which $|\{\ell \in L : a_{l,j} = 1\}| = m_j$ is minimized: call this value j^*.

For each $\ell \in L$ in turn for which $a_{l,j^*} = 1$, we update the sets I, J, L and the vectors R, M and then recurse. We update I by adding the index i. We update J by removing the indices of all k-subsequences newly covered λ times. We update R by adding the i^{th} row of A to it. We update L by removing both i and the indices of all n-sequences covering a k-subsequence that is newly covered λ times, whose inclusion in I would violate (3.1). We update M by subtracting the rows of A whose indices have just been removed from L.

3.1.2 Pseudocode

Algorithm 2 Tree search for PSCA(n, k, λ) by backtracking

Input: PSCA parameters (n, k, λ) and $A = (a_{i,j}) = (n, k)$-incidence matrix

1: **procedure** SEARCH(I, J, R, L, M)
2: % I = indices of sequences of partial PSCA(n, k, λ) P
3: % write $(r_j) = \mathrm{rv}_{n,k}(I) = (\sum_{i \in I}\ a_{i,j})$ = sum of rows of A indexed by I
4: % $J = \{j : r_j < \lambda\}$ = indices of k-subsequences not yet covered λ times by P
5: % $R = (r_j : j \in J))$ = entries of $\mathrm{rv}_{n,k}(I)$ indexed by J
6: % $L = \{\ell \notin I : a_{\ell,j} = 0 \text{ for all } j \notin J\}$ = candidates for enlarging I
7: % write $(m_j) = \mathrm{rv}_{n,k}(L) = (\sum_{\ell \in L} a_{\ell,j})$ = sum of rows of A indexed by L
8: % $M = (m_j : j \in J)$ = entries of $\mathrm{rv}_{n,k}(L)$ indexed by J
9: **if** $|I| = k!\lambda$ **then**
10: record I as the index set of a PSCA(n, k, λ).
11: **return**
12: **end if**
13: **if** $r_j + m_j < \lambda$ for some $j \in J$ **then**
14: **return** % terminate branch early
15: **end if**
16: let J' be the set of indices for which r_j attains its maximum value over $j \in J$.
17: choose an arbitrary $j = j^*$ for which m_j attains its minimum value over $j \in J'$.
18: $L(j^*) \coloneqq \{\ell \in L : a_{\ell,j^*} = 1\}$
19: **for each** $i \in L(j^*)$ **do**
20: $I_{\mathrm{new}} \coloneqq I \cup \{i\}$
21: $J_{\mathrm{new}} \coloneqq J \setminus \{j : r_j + a_{i,j} = \lambda\}$
22: $R_{\mathrm{new}} \coloneqq (r_j + a_{i,j} : j \in J_{\mathrm{new}})$
23: $B \coloneqq \{i\} \cup \{\ell \in L : a_{\ell,j} = 1 \text{ for at least one } j \in J \setminus J_{\mathrm{new}}\}$
24: $L_{\mathrm{new}} \coloneqq L \setminus B$
25: $M_{\mathrm{new}} \coloneqq (m_j - \sum_{\ell \in B} a_{\ell,j} : j \in J_{\mathrm{new}})$
26: SEARCH$(I_{\mathrm{new}}, J_{\mathrm{new}}, R_{\mathrm{new}}, L_{\mathrm{new}}, M_{\mathrm{new}})$
27: **end for**
28: **end procedure**

By calling SEARCH with $I = \emptyset$ and $J = [\binom{n}{k}k!]$ and $R = (0)^{\binom{n}{k}k!}$ and $L = [n!]$ and $M = (\frac{n!}{k!})^{\binom{n}{k}k!}$, we obtain every possible PSCA(n, k, λ).

3.1.3 Example: partial search for PSCA(5,3,2)

We illustrate Algorithm 2 by presenting part of a search for all possible examples of a PSCA$(5, 3, 2)$. We visualize each iteration as comprising 3 steps. Step 1 corresponds to checking the early termination condition (Line 13). Step 2 corresponds to finding j^* and $L(j^*)$ (Lines 16–18). Step 3 corresponds to updating the sets I, J, R, L, M (Lines 20–25).

We begin with the (5,3)-incidence matrix, having $5! = 120$ rows and $\binom{5}{3}3! = 60$ columns. As previously, it is not necessary to store the row and column labels of the (5,3)-incidence

matrix as elements of S_5 and $S_{5,3}$, respectively: we instead index the rows using [120] and the columns using [60].

Rather than describe the complete search procedure, we suppose that the first six iterations added indices $1, 6, 43, 54, 78, 27$ to I in that order, and that we are now entering SEARCH at Line 1 with $|L| = 24$ and $|J| = 42$. Table 3.1 displays the remaining 24×42 submatrix of the $(5, 3)$-incidence matrix, whose rows and columns are indexed by L and J, respectively. The vector M is the sum of all rows of the remaining 24×42 matrix. The six additional rows shown in red correspond to the six indices previously added to I, and the vector R is their sum.

Iteration 7. In Step 1, we test the early termination condition of Line 13 by adding the vectors R and M. Since no entry of the sum is less than $\lambda = 2$, we continue.

We now refer to Table 3.2 for Step 2. At Line 16 we calculate the set J' of indices (shown in green) at which R attains its maximum value of 1. At Line 17 we find the set of positions (shown in light blue) at which M attains its minimum value of 2 over the green positions, and choose j^* arbitrarily in this set. Suppose we choose $j^* = 4$. Then at Line 18 we determine the set of row positions containing an entry 1 in column j^* (shown in dark blue) to be $\{92, 116\}$. At Line 19, we recurse over $i \in \{92, 116\}$. We shall illustrate the branch $i = 116$.

We now refer to Table 3.3 for Step 3. At Line 20 we include 116 (shown in red) in the set I. At Line 21 we remove from J the set of column positions (shown in pink) at which the sum of R and row 116 equals $\lambda = 2$; we represent this updating of J by deleting these columns from the matrix. After deleting the pink columns, the updated vector R is given directly at Line 22 by the sum of R and row 116. At Line 23, we calculate the set B as the union of $\{116\}$ with the set of row positions containing an entry 1 in at least one of the pink columns (shown in orange background). We also use orange to colour the row positions of B and each of their 1 entries in the columns that remain after deleting the pink columns. At Line 24 we remove the orange row positions from L; we represent this updating of L by deleting these rows from the matrix. At Line 25, we wish to update M so that it is the sum of the rows of the remaining 5×34 matrix. Rather than calculate this sum directly, we instead determine M_{new} by subtracting from M (in column positions that are not pink) the sum Σ of the orange rows.

Iteration 8. We refer to Table 3.4 for Step 1, in which we begin with the 5×34 matrix remaining from Iteration 7. We add the vectors R and M at Line 13. Since their sum $R+M$ is less than $\lambda = 2$ in some positions (in fact at the 15 positions shown in gray), the early termination condition is satisfied and we backtrack.

Table 3.1: The submatrix of the (5,3)-incidence matrix relative to L and J

	J	4	7	15	16	19	22	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
L		1	1	2	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3	4	4	4	4	4	4	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5
		3	4	1	3	4	5	1	1	1	2	2	2	4	4	4	5	5	5	1	1	1	2	2	2	3	3	3	5	5	5	1	1	1	2	2	2	3	3	3	4	4	4
		2	2	5	1	1	1	2	4	5	1	4	5	1	2	5	1	2	4	2	3	5	1	3	5	1	2	5	1	2	3	2	3	4	1	3	4	1	2	4	1	2	3
1	12345															1																											
6	12543																																										1
43	25134						1																										1	1						1			
54	31542		1					1	1	1					1			1	1																						1		
78	41532	1																		1	1	1					1			1	1								1				
27	21435			1																								1															
63	34215	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
64	34251	0	0	0	0	0	1	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
65	34512	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0
66	34521	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0
68	35142	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0
70	35241	0	0	0	0	1	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0
71	35412	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0
72	35421	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0
87	43215	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
88	43251	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
89	43512	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0
90	43521	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0
92	45132	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	1	0	0	0	0
94	45231	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	1	1	0	1	0	0	0	0	0
95	45312	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	1	1	0	0	0	0
96	45321	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	0	0	1	1	0	0	0	0
110	53142	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	1	1	0	1	0
112	53241	0	0	0	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	0	0
113	53412	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0
114	53421	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	0
116	54132	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0	1	1	1
118	54231	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	1	1
119	54312	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	1
120	54321	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1
R	1	1	1	0	0	1	1	1	1	0	0	0	0	1	1	0	1	1	1	1	1	0	0	0	0	1	1	0	1	1	0	1	1	0	0	0	0	1	1	0	1	1	
M	2	2	2	2	2	2	8	2	2	12	2	4	10	10	4	10	8	4	8	2	2	12	2	4	10	10	4	10	8	4	10	2	2	10	2	2	10	10	4	10	10	4	

Table 3.2: Algorithm 2, Iteration 7, Steps 1 and 2

L \ J	4	7	15	16	19	22	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
1																																										
6																																										
43																																										
54																																										
78																																										
27																																										
63	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
64	0	0	0	0	0	1	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
65	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0
66	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0
68	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0
70	0	0	0	0	1	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0
71	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0
72	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0
87	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
88	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
89	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0
92	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	1	0	0	0	0
94	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	1	1	0	1	0	0	0	0	0
95	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	1	1	0	0	0	0
96	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	0	0	1	1	0	0	0	0
110	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	1	1	0	1	0
112	0	0	0	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	0	0
113	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0
114	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	0
116	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0	1	1	1
118	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	1	1
119	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	1
120	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1
R	1	1	1	0	0	1	1	1	1	0	0	0	0	1	1	0	1	1	1	1	1	0	0	0	0	1	1	0	1	1	0	1	1	0	0	0	0	1	1	0	1	1
M	2	2	2	2	2	2	8	2	2	12	2	4	10	10	4	10	8	4	8	2	2	12	2	4	10	10	4	10	8	4	10	2	2	10	2	2	10	10	4	10	10	4

Table 3.3: Algorithm 2, Iteration 7, Step 3

L \ J	4	7	15	16	19	22	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
1																																										
6																																										
43																																										
54																																										
78																																										
27																																										
63	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
64	0	0	0	0	0	1	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
65	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0
66	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0
68	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0
70	0	0	0	0	1	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0
71	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0
72	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0
87	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
88	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
89	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0
92	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	1	0	0	0	0
94	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	1	1	0	1	0	0	0	0	0
95	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0	1	1	0	0	0	0
96	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	0	0	1	1	0	0	0	0
110	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	1	1	0	1	0
112	0	0	0	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	0	0
113	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0
114	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	0
116	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0	1	1	1
118	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	1	1
119	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0	1	1	1
120	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1
R	1	1	1	0	0	1	1	1	1	0	0	0	0	1	1	0	1	1	1	1	1	0	0	0	0	1	1	0	1	1	0	1	1	0	0	0	0	1	1	0	1	1
R + row 116	2	1	1	0	0	1	1	1	1	0	0	0	0	1	1	0	1	1	2	2	1	0	0	0	0	2	1	0	1	1	1	2	1	0	0	0	0	2	1	1	2	2
M	2	2	2	2	2	2	8	2	2	12	2	4	10	10	4	10	8	4	8	2	2	12	2	4	10	10	4	10	8	4	10	2	2	10	2	2	10	10	4	10	10	4
Σ		2	1	1	1	1	8	2	1	8	1	2	6	7	1	7	6	3			1	8	1	2	9		4	7	6	3	10		2	7	1	1	9		4	9		
M_{new}		0	1	1	1	1	0	0	1	4	1	2	4	3	3	3	2	1			1	4	1	2	1		0	3	2	1	0		0	3	1	1	1		0	1		

Table 3.4: Algorithm 2, Iteration 8, Step 1

L \ J	7	15	16	19	22	25	26	27	28	29	30	31	32	33	34	35	36	39	40	41	42	43	45	46	47	48	49	51	52	53	54	55	57	58
1																																		
6																																		
43																																		
54																																		
78																																		
27																																		
116																																		
63	0	1	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
64	0	0	0	0	1	0	0	0	1	0	1	1	1	1	1	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
66	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	0	0	1	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0
70	0	0	0	1	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1
94	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	1	1	1	0	0	1	1	0	1	0	0
R	1	1	0	0	1	1	1	1	0	0	0	0	1	1	0	1	1	1	0	0	0	0	1	0	1	1	1	1	0	0	0	0	1	1
M	0	1	1	1	1	0	0	1	4	1	2	4	3	3	3	2	1	1	4	1	2	1	0	3	2	1	0	0	3	1	1	1	0	1
$R+M$	1	2	1	1	2	1	1	2	4	1	2	4	4	4	3	3	2	2	4	1	2	1	1	3	3	2	1	1	3	1	1	1	1	2

3.1.4 Comments

1. The removal of rows and columns of the incidence matrix at successive iterations of Algorithm 2 substantially reduces the search time, as it does for Algorithm 1.

2. There is an important distinction between Algorithms 1 and 2. In Algorithm 1, we calculate $|L(j)| = \sum_{\ell \in L} a_{\ell,j}$ for each $j \in J$ directly. In contrast, in Algorithm 2 we pass the vector $M = (\sum_{\ell \in L} a_{\ell,j} : j \in J)$ as a recursion parameter, and can then update M more efficiently at line 25 by subtracting rows of the incidence matrix that have just been removed from L. This gives a further significant speed advantage.

3. The early termination condition of Line 13 in Algorithm 2 was not needed previously because it is embedded in Algorithm 1: if $r_j + m_j < \lambda$ for some $j \in J$ and $\lambda = 1$, then Line 12 of Algorithm 1 will calculate $|L(j^*)| = 0$ and so the for-loop of Lines 13–18 will operate on an empty set.

4. The calculations in Lines 13, 16, 17 can be accomplished in linear time with a single pass through the indices $j \in J$.

5. To demonstrate why the index i must be included in the set B at Line 23, suppose we use Algorithm 2 to search for a PSCA$(5, 3, 2)$ and that in the first iteration we choose $j^* = 1$ at Line 17. Then at Lines 20–22 we calculate $I_{\text{new}} = \{1\}$, and $J_{\text{new}} = J = [60]$, and $R_{\text{new}} = R +$ row 1 as shown in the table below. Since $J \setminus J_{\text{new}} = \emptyset$, we calculate $B = \{1\}$ at Line 23 and so remove the index 1 from L at Line 24. If we had not included $\{i\}$ in the definition of B, then we would instead have obtained $B = \emptyset$ at Line 23 and $L_{\text{new}} = L$ at Line 24 and $M_{\text{new}} = M$ at Line 25. Then the next iteration could again choose $j^* = 1$, but we disallowed this when defining a perfect sequence covering array to be a set rather than a multiset (see Remark 1.6 (3)).

	J	1	2	3	4	5	6	7	8	9	...	17	18	...	21	...	33	...	60
		1	1	1	1	1	1	1	1	1	...	2	2	...	2	...	3	...	5
L		2	2	2	3	3	3	4	4	4	...	3	3	...	4	...	4	...	4
		3	4	5	2	4	5	2	3	5	...	4	5	...	5	...	5	...	3
1	12345	1	1	1	0	1	1	0	0	1	...	1	1	...	1	...	1	...	0
2	12354	1	1	1	0	1	1	0	0	0	...	1	1	...	0	...	0	...	0
3	12435	1	1	1	0	0	1	0	1	1	...	0	1	...	1	...	0	...	0
...	...																		
R		0	0	0	0	0	0	0	0	0	...	0	0	...	0	...	0	...	0
$R +$ row 1		1	1	1	0	1	1	0	0	1	...	1	1	...	1	...	1	...	0
M		20	20	20	20	20	20	20	20	20	...	20	20	...	20	...	20	...	20

3.2 Yuster construction method revisited

In this section, we consider again the Yuster construction method of Section 2.6. We show firstly that we can regard the examples of a PSCA$(6,3,2)$ and PSCA$(7,3,2)$ given in Proposition 3.1 as arising from a modification of this construction method. We then show that we can reinterpret these two examples, as well as the example of a PSCA$(5,3,2)$ from Proposition 2.41, using left cosets of a subgroup of S_n.

3.2.1 Modification of the method

A direct application of the Yuster construction method fails to find a PSCA$(6,3,2)$. For every 6-subset X of S_6, we find by a straightforward recursive computer search (attempting to build X one 6-sequence at a time) that the largest possible number of nonzero entries in the repetition vector $\mathrm{rv}_3(X)$ is 112. For example, for the set

$$X = \{123456, 164352, 326154, 462513, 524316, 563412\}$$

we find that $\mathrm{rv}_3(X)$ has eight entries 0 (corresponding to the 3-subsequences 153, 263, 351, 362, 415, 426, 514, 624), eight entries 2 (corresponding to the 3-subsequences 135, 145, 236, 246, 531, 541, 632, 642), and the remaining 104 entries 1 (for a total of 112 nonzero entries). However, we find by computer search that, for every 6-subset X of S_5 for which $\mathrm{rv}_3(X)$ has 112 nonzero entries, there is no $\sigma \in S_6$ such that each entry of $\mathrm{rv}_3(X) + \mathrm{rv}_3(X\sigma)$ is 2.

Nonetheless, we can regard the example of a PSCA$(6,3,2)$ given in Proposition 3.1 (1) as arising from the Yuster construction method, provided we drop the assumption that $\mathrm{rv}_3(X)$ should attain the largest possible number of nonzero entries.

Example 3.2. Let $\sigma = 154326$ and

$$X = \{123456, 216543, 354162, 461325, 532614, 645231\},$$
$$X\sigma = \{154326, 516234, 423165, 361452, 245613, 632541\}.$$

Then $\mathrm{rv}_3(X)$ has twelve entries 0 (corresponding to the 3-subsequences 142, 152, 241, 251, 315, 365, 426, 436, 513, 563, 624, 634), twelve entries 2 (corresponding to the 3-subsequences 125, 135, 214, 264, 346, 356, 412, 462, 521, 531, 643, 653), and the remaining 96 entries 1 (for a total of 108 nonzero entries). The set $X\sigma$ has the same property, but with the positions in $\mathrm{rv}_3(X\sigma)$ of the 0 and 2 entries interchanged. Therefore $X \cup X\sigma$ is a PSCA$(6,3,2)$.

We can likewise regard the example of a PSCA$(7,3,2)$ given in Proposition 3.1 (2) as arising from the same modification of the Yuster construction method.

Example 3.3. Let $\sigma = 4261735$ and

$$X = \{1234567, 1573426, 3275641, 3617524, 5243176, 5164327\},$$

$$X\sigma = \{4261735, 4756123, 6257314, 6345721, 7216453, 7431625\}.$$

Then $\mathrm{rv}_3(X)$ has 56 entries 0, and 56 entries 2, and the remaining 98 entries 1 (for a total of 154 nonzero entries). The set $X\sigma$ has the same property, but with the positions in $\mathrm{rv}_3(X\sigma)$ of the 0 and 2 entries interchanged. Therefore $X \cup X\sigma$ is a PSCA$(7, 3, 2)$.

The perfect sequence covering arrays given in Examples 3.2 and 3.3 were originally found using Algorithm 2. It turns out by instead using the modified Yuster construction method to search for perfect sequence covering arrays with the same parameters, we can easily find examples of a PSCA$(6, 3, 2)$. However, we did not find a single example of a PSCA$(7, 3, 2)$ in this way because of memory and time constraints in searching for the set X.

It is possible to combine the modified Yuster construction method with Algorithm 2 to seek a PSCA$(n, k, 2)$, by recursively searching for a set X of size $\frac{k!\lambda}{2}$ with the required properties and then searching for a corresponding permutation σ. We did not implement this combination because the improvements to Algorithm 2 that we shall describe in Section 3.3.3 appear to be much more powerful.

3.2.2 Reinterpretation using left cosets

We have shown that the examples of a PSCA$(n, 3, 2)$ for $n \in \{5, 6, 7\}$ given in Proposition 2.41 and Examples 3.2 and 3.3 can each be represented in the form $X \cup X\sigma$, for some 6-set X and some permutation $\sigma \in S_n$. In each of these examples, we find that the subgroup $\langle\sigma\rangle$ of S_n has order 2 and so we can reinterpret the perfect sequence covering array as six left cosets of $\langle\sigma\rangle$ in S_n, namely as $\bigcup_{x\in X} x\langle\sigma\rangle$. Under this interpretation, X is a set of left coset representatives for $\langle\sigma\rangle$.

This is shown in the tables below, in which the rows give the representation $X \cup X\sigma$ and the columns give the representation $\bigcup_{x\in X} x\langle\sigma\rangle$.

PSCA$(5, 3, 2)$ with $\sigma = 13254$

	$\langle\sigma\rangle$	$43215\,\langle\sigma\rangle$	$35214\,\langle\sigma\rangle$	$14523\,\langle\sigma\rangle$	$25413\,\langle\sigma\rangle$	$53412\,\langle\sigma\rangle$
X	12345	43215	35214	14523	25413	53412
$X\sigma$	13254	52314	24315	15432	34512	42513

PSCA$(6, 3, 2)$ with $\sigma = 154326$

	$\langle\sigma\rangle$	$216543\,\langle\sigma\rangle$	$354162\,\langle\sigma\rangle$	$461325\,\langle\sigma\rangle$	$532614\,\langle\sigma\rangle$	$645231\,\langle\sigma\rangle$
X	123456	216543	354162	461325	532614	645231
$X\sigma$	154326	516234	423165	361452	245613	632541

PSCA$(7, 3, 2)$ with $\sigma = 4261735$

	$\langle\sigma\rangle$	$1573426\,\langle\sigma\rangle$	$3275641\,\langle\sigma\rangle$	$3617524\,\langle\sigma\rangle$	$5243176\,\langle\sigma\rangle$	$5164327\,\langle\sigma\rangle$
X	1234567	1573426	3275641	3617524	5243176	5164327
$X\sigma$	4261735	4756123	6257314	6345721	7216453	7431625

3.3 Construction of PSCA(n, k, λ) as union of cosets

In this section, we identify an algebraic structure for a perfect sequence covering array that underlies our most powerful and fruitful search method.

3.3.1 Motivation

We have seen in Section 3.2.2 that certain perfect sequence covering arrays can be represented as a union of cosets of a subgroup G of S_n. The examples previously given have three features in common: the cosets are left cosets; the subgroup G has order 2; and the parameter λ has the value 2. We now demonstrate that there are interesting examples of perfect sequence covering arrays that do not require these features.

Recall firstly that Mathon and van Trung [MvT99] used both right and left coset structure in their examination of directed k-packings: they found a PSCA$(6, 4, 1)$ which is the union of three *right* cosets of a subgroup $G \cong D_8$ of S_6 (see Theorem 2.46 (2)); and they found a 304-subset $P \subset S_8$ for which $\mathrm{rv}_6(P)$ has each entry at most 1, where P is the union of 38 *left* cosets of a subgroup $G \cong C_2^3$ of S_8 (see comments following Theorem 2.46). We shall use both left and right cosets of a subgroup in our construction of perfect sequence covering arrays. However, we shall see that the use of left cosets leads to a richer existence pattern (involving a larger subgroup G) and a simplification in the search using conjugacy classes.

We next note that the subgroup G need not have order 2 and need not be cyclic. Indeed, the PSCA$(6, 4, 1)$ of Theorem 2.46 (1) is an entire order 24 subgroup of S_6, while the PSCA$(6, 4, 1)$ of Theorem 2.46 (2) is a union of three cosets of an order 8 subgroup. Likewise, the PSCA$(6, 3, 2)$ which we represented in Section 3.2.2 as six left cosets of an order 2 subgroup can instead be represented as the entire order 12 subgroup $\langle 154326, 216543\rangle \cong D_{12}$.

We remark that the parameter λ need not take the value 2. In Proposition 3.18 we shall give an example of a PSCA$(5, 3, 3)$ which is the union of 9 left cosets of an order 2 subgroup of S_5.

We further note that a single perfect sequence covering array can admit more than one representation as a union of cosets of a subgroup. For example, the PSCA$(7, 3, 2)$ which we considered in Section 3.2.2 has at least the following five representations.

(1) Six left cosets of the order 2 subgroup $\langle 4261735 \rangle$ (as given in Section 3.2.2):

$\langle\sigma\rangle$	$1573426\,\langle\sigma\rangle$	$3275641\,\langle\sigma\rangle$	$3617524\,\langle\sigma\rangle$	$5243176\,\langle\sigma\rangle$	$5164327\,\langle\sigma\rangle$
1234567	1573426	3275641	3617524	5243176	5164327
4261735	4756123	6257314	6345721	7216453	7431625

(2) Six left cosets of the order 2 subgroup $\langle 5243176 \rangle$:

$\langle\sigma\rangle$	$1573426\,\langle\sigma\rangle$	$3275641\,\langle\sigma\rangle$	$3617524\,\langle\sigma\rangle$	$6257314\,\langle\sigma\rangle$	$6345721\,\langle\sigma\rangle$
1234567	1573426	3275641	3617524	6257314	6345721
5243176	5164327	4261735	4756123	7216453	7431625

(3) Six left cosets of the order 2 subgroup $\langle 6257314 \rangle$:

$\langle\sigma\rangle$	$1573426\,\langle\sigma\rangle$	$3275641\,\langle\sigma\rangle$	$3617524\,\langle\sigma\rangle$	$4261735\,\langle\sigma\rangle$	$4756123\,\langle\sigma\rangle$
1234567	1573426	3275641	3617524	4261735	4756123
6257314	6345721	5243176	5164327	7216453	7431625

(4) Six right cosets of the order 2 subgroup $\langle 3617524 \rangle$:

$\langle\sigma\rangle$	$\langle\sigma\rangle\,1573426$	$\langle\sigma\rangle\,3275641$	$\langle\sigma\rangle\,4261735$	$\langle\sigma\rangle\,4756123$	$\langle\sigma\rangle\,5164327$
1234567	1573426	3275641	4261735	4756123	5164327
3617524	7216453	7431625	6345721	5243176	6257314

(5) Four left cosets of the order 3 subgroup $\langle 3275641 \rangle$:

$\langle\sigma\rangle$	$1573426\,\langle\sigma\rangle$	$4261735\,\langle\sigma\rangle$	$4756123\,\langle\sigma\rangle$
1234567	1573426	4261735	4756123
3275641	3617524	5243176	5164327
7216453	7431625	6257314	6345721

In view of the observations and examples given above, we shall focus on the construction a PSCA(n, k, λ) that is a union of one or more (left or right) cosets of a nontrivial subgroup G of S_n.

3.3.2 Left or right cosets

We next show that, in the context of a prescribed subgroup for a PSCA(n, k, λ), there is a fundamental distinction between left and right cosets.

Definition 3.4 (conjugacy of subgroups)**.** Subgroups G and H of S_n are *conjugate* in S_n if $H = xGx^{-1}$ for some $x \in S_n$. Conjugacy is an equivalence relation on the set of subgroups of S_n, and the equivalence class of G under conjugation is the *conjugacy class* $\mathrm{Cl}(G) := \{xGx^{-1} : x \in S_n\}$.

Theorem 3.5 (union of left cosets)**.** *Let G be a subgroup of S_n and let $H \in \mathrm{Cl}(G)$. Suppose there is a* PSCA(n, k, λ) *that is a union of left cosets of H. Then there is a* PSCA(n, k, λ) *which is a union of left cosets of G.*

Proof. Let P be a PSCA(n, k, λ) that can be written as

$$P = \bigcup_{\pi \in \mathcal{R}} \pi H \tag{3.2}$$

for a set $\mathcal{R}$ of left coset representatives for H. Since $H \in \mathrm{Cl}(G)$, for some $x \in S_n$ we can write

$$\begin{aligned} P &= \bigcup_{\pi \in \mathcal{R}} \pi(xGx^{-1}) \\ &= \Big(\bigcup_{\mu \in \mathcal{R}x} \mu G \Big) x^{-1}. \end{aligned}$$

Then $Px = \bigcup_{\mu \in \mathcal{R}x} \mu G$ is a union of left cosets of G, and is a PSCA(n, k, λ) by Result 2.42. ■

Remark 3.6.

1. Consider searching exhaustively over all nontrivial subgroups G of S_n for a perfect sequence covering array as a union $\bigcup_{\pi \in \mathcal{R}} \pi G$ of left cosets of G. We may not restrict attention to a single representative G from each *isomorphism* class of subgroups of S_n. For example, we have seen in Section 3.2.2 that there is a PSCA$(5, 3, 2)$ that is a union of left cosets of $\langle 13254 \rangle \cong C_2$, but by exhaustive search there is no PSCA$(5, 3, 2)$ that is a union of left cosets of $\langle 12354 \rangle \cong C_2$.

   However, by Theorem 3.5 it is sufficient to restrict attention to a single representative G from each *conjugacy* class of subgroups of S_n. This drastically reduces the required computation for an exhaustive search. For example, S_7 contains 11299 nontrivial subgroups, but only 95 nontrivial conjugacy classes of subgroups.

2. Now consider searching exhaustively over all nontrivial subgroups G of S_n for a perfect sequence covering array as a union $\bigcup_{\sigma\in\mathcal{R}} G\sigma$ of right cosets of G. Theorem 3.5 no longer holds when we replace left cosets by right cosets, because it relies on Result 2.42 which fails when the right action of σ on P is replaced by the left action. Therefore an exhaustive search must consider all nontrivial subgroups of S_n.

   However, there is now a worthwhile simplification in that we may assume the subgroup G itself is one of the right cosets contained in the perfect sequence covering array $\bigcup_{\sigma\in\mathcal{R}} G\sigma$: let $x \in \mathcal{R}$, and note from Result 2.42 that $\left(\bigcup_{\sigma\in\mathcal{R}} G\sigma\right)x^{-1} = \bigcup_{\sigma\in\mathcal{R}} G(\sigma x^{-1})$ is a perfect sequence covering array, and it contains the right coset $G(xx^{-1}) = G$.

3.3.3 Search algorithm when the subgroup is prescribed

In this section, we modify Algorithm 2 to Algorithm 3 to find all possible examples of a PSCA(n, k, λ) that is a union of one or more cosets of a nontrivial subgroup G of S_n. Since the number of n-sequences in a PSCA(n, k, λ) is $k!\lambda$, and each coset has order $|G|$, the number of cosets is $\frac{k!\lambda}{|G|}$ and so $|G|$ must divide $k!\lambda$. We obtain a key computational advantage by prescribing the subgroup G. The modified algorithm attempts to build a PSCA(n, k, λ) one coset of G at a time without covering any k-subsequence more than λ times. If it terminates without output then there is no PSCA(n, k, λ) that is a union of cosets of G.

Algorithm 3 follows the same principles as Algorithm 2, but operates on the following compressed version of the (n, k)-incidence matrix in which the $|G|$ repetition vectors corresponding to the sequences of a coset are replaced by their sum: the entire coset is either contained or not contained in the perfect sequence covering array.

Definition 3.7 ((n, k)-incidence matrix for a subgroup G)**.** Let G be a subgroup of S_n, and let $\mathcal{R}$ be a complete set of left (or right) coset representatives for G in S_n. The *left (or right)* (n, k)*-incidence matrix for* G is the $\frac{n!}{|G|} \times \binom{n}{k}k!$ array over $\{0, 1, \ldots, |G|\}$ whose rows are $\{\mathrm{rv}_k(\pi G) : \pi \in \mathcal{R}\}$ (or $\{\mathrm{rv}_k(G\sigma) : \sigma \in \mathcal{R}\}$).

When displaying the (n, k)-incidence matrix for a subgroup G, we shall arrange the rows in order of the lexicographically least element of each coset and then label the rows $1, 2, \ldots, \frac{n!}{|G|}$. However, the outcome of the modified algorithm is independent of the ordering of the rows.

Example 3.8. The left (4,3)-incidence matrix for the subgroup $\langle 2341\rangle$ of S_4 is given in Table 3.5 shown below. Each row is the sum of four rows of the (4,3)-incidence matrix (see Example 2.35) corresponding to a left coset of $\langle 2341\rangle$ in S_4. The lexicographically least elements of the six left cosets of $\langle 2341\rangle$ are $\{1234, 1243, 1324, 1342, 1423, 1432\}$.

We use Algorithm 3 together with Algorithm 4 (union of left cosets) or Algorithm 5 (union of right cosets). By using Algorithms 3 and 4, we find the following example of a

Table 3.5: The left (4,3)-incidence matrix for $\langle 2341 \rangle$

			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
			1	1	1	1	1	1	2	2	2	2	2	2	3	3	3	3	3	3	4	4	4	4	4	4
			2	2	3	3	4	4	1	1	3	3	4	4	1	1	2	2	4	4	1	1	2	2	3	3
			3	4	2	4	2	3	3	4	1	4	1	3	2	4	1	4	1	2	2	3	1	3	1	2
1	$\langle\sigma\rangle$	1234 2341 3412 4123	2	1	0	1	0	0	0	0	1	2	1	0	1	0	0	0	2	1	2	1	0	1	0	0
2	$1243\langle\sigma\rangle$	1243 2314 3421 4132	1	1	1	0	0	1	0	1	1	1	0	1	0	1	1	0	1	1	1	1	1	0	0	1
3	$1324\langle\sigma\rangle$	1324 2431 3142 4213	0	1	1	1	1	0	1	0	1	0	1	1	1	1	0	1	0	1	0	1	1	1	1	0
4	$1342\langle\sigma\rangle$	1342 3124 4231 2413	0	1	1	1	1	0	1	0	1	0	1	1	1	1	0	1	0	1	0	1	1	1	1	0
5	$1423\langle\sigma\rangle$	1423 3241 4312 2134	1	0	0	1	1	1	1	1	0	1	1	0	1	0	1	1	1	0	1	0	0	1	1	1
6	$1432\langle\sigma\rangle$	1432 2143 3214 4321	0	0	1	0	1	2	1	2	0	0	0	1	0	1	2	1	0	0	0	0	1	0	1	2

PSCA$(7,4,2)$ which is the union of left cosets of a cyclic order 6 subgroup. Since $g(7,4) > 1$ by Result 2.31 and Theorem 2.27, this gives the new g-value $g(7,4) = 2$ via Result 1.11.

Proposition 3.9. *The following* 48 *sequences form a* PSCA$(7,4,2)$ *as a union of 8 left cosets of the subgroup* $G = \langle 2745163 \rangle \cong C_6$*:*

G	$1254763\,G$	$1537264\,G$	$1634527\,G$	$1637254\,G$	$1647253\,G$	$1652743\,G$	$1745263\,G$
1234567	1254763	1537264	1634527	1637254	1647253	1652743	1745263
2745163	2715364	2143765	2645173	2643715	2653714	2617354	2351764
3412765	3472561	3715462	3612745	3615472	3625471	3674521	3527461
4527361	4537162	4321567	4627351	4621537	4671532	4635172	4173562
5173462	5143267	5472163	5673412	5672143	5632147	5641237	5234167
7351264	7321465	7254361	7651234	7654321	7614325	7623415	7412365

Therefore $g(7,4) = 2$*.*

3.3.4 Idea of algorithm

Let G be the prescribed nontrivial subgroup of S_n. Whereas Algorithm 2 keeps track of a shrinking set L of rows and a shrinking set J of columns of the (n,k)-incidence matrix, Algorithm 3 does the same but in relation to the (n,k)-incidence matrix for G. This requires some modifications to Algorithm 2 because the entries $a_{i,j}$ of the latter matrix are no longer restricted to $\{0,1\}$.

3.3.5 Pseudocode

Algorithm 3 Tree search for union-of-cosets PSCA(n, k, λ) by backtracking

Input: PSCA parameters (n, k, λ) and $A = (a_{i,j})$ = left or right (n, k)-incidence matrix for a subgroup G of S_n, where $|G|$ divides $k!\lambda$.

1: **procedure** SEARCH(I, J, R, L, M)
2: % I = indices of cosets in partial PSCA(n, k, λ) P
3: % write $(r_j) = (\sum_{i \in I} a_{i,j})$ = sum of rows of A indexed by I
4: % $J = \{j : r_j < \lambda\}$ = indices of k-subsequences not yet covered λ times by P
5: % $R = (r_j : j \in J))$
6: % $L = \{\ell \notin I : a_{\ell,j} + r_j \leq \lambda \text{ for all } j\}$ = candidates for enlarging I
7: % write $(m_j) = (\sum_{\ell \in L} a_{\ell,j})$ = sum of rows of A indexed by L
8: % $M = (m_j : j \in J)$
9: **if** $|I| = \frac{k!\lambda}{|G|}$ **then**
10: record I as the index set of cosets whose union is a PSCA(n, k, λ).
11: **return**
12: **end if**
13: **if** $r_j + m_j < \lambda$ for some $j \in J$ **then**
14: **return** % terminate branch early
15: **end if**
16: let J' be the set of indices for which r_j attains its maximum value over $j \in J$.
17: choose an arbitrary $j = j^*$ for which m_j attains its minimum value over $j \in J'$.
18: $L(j^*) \coloneqq \{\ell \in L : a_{\ell,j^*} \neq 0\}$
19: **for each** $i \in L(j^*)$ **do**
20: $I_{\text{new}} \coloneqq I \cup \{i\}$
21: $J_{\text{new}} \coloneqq J \setminus \{j : r_j + a_{i,j} = \lambda\}$
22: $R_{\text{new}} \coloneqq (r_j + a_{i,j} : j \in J_{\text{new}})$
23: $B \coloneqq \{i\} \cup \{\ell \in L : a_{l,j} + r_j + a_{i,j} > \lambda \text{ for some } j \in J\}$
24: $L_{\text{new}} \coloneqq L \setminus B$
25: $M_{\text{new}} \coloneqq (m_j - \sum_{\ell \in B} a_{\ell,j} : j \in J_{\text{new}})$
26: SEARCH$(I_{\text{new}}, J_{\text{new}}, R_{\text{new}}, L_{\text{new}}, M_{\text{new}})$
27: **end for**
28: **end procedure**

The following lines of Algorithm 2 are modified in Algorithm 3 to search for a perfect sequence covering array as a union of cosets of a prescribed nontrivial subgroup G. Algorithm 3 reduces to Algorithm 2 if we take G to be the trivial subgroup.

Line 9. The target size of $|I|$ is the number $\frac{k!\lambda}{|G|}$ of cosets, rather than the number $k!\lambda$ of n-sequences.

Line 18. The entries of the (n, k)-incidence matrix over G lie in $\{0, 1, \ldots, |G|\}$ rather than $\{0, 1\}$, so we replace the test $a_{\ell,j^*} = 1$ by $a_{\ell,j^*} \neq 0$.

Lines 6. The set L of candidates for enlarging I is more constrained, because inclusion of a row containing an entry $a_{\ell,j} > 1$ could cause the sum $a_{\ell,j} + r_j$ to exceed λ. We

therefore impose that $a_{\ell,j} + r_j \leq \lambda$ over all values of j (not just those lying outside J). We can rewrite L as specified in Line 6 as

$$\{\ell \notin I : a_{\ell,j} = 0 \text{ for all } j \notin J\} \cap \{\ell \notin I : a_{\ell,j} + r_j \leq \lambda \text{ for all } j \in J\}, \tag{3.3}$$

to see that the set L in Algorithm 3 is a subset of the set L in Algorithm 2.

Line 23. The set B used to update L at Line 24 follows from the definition of L at Line 6 and the updated value of R at Line 22.

We call the function SEARCH of Algorithm 3 from Algorithm 4 (union of left cosets) or Algorithm 5 (union of right cosets). In both cases, we first test the necessary condition that $|G|$ divides $k!\lambda$. We then exclude from the initial candidate set L the index of each row of the (n,k)-incidence matrix for G that contains one or more entries greater than λ. For left cosets (Algorithm 4) we must check each row individually, but for right cosets it is sufficient by Lemma 2.43 to check a single row (and return without output if it contains an entry greater than λ).

Algorithm 4 Search for PSCA(n,k,λ) as union of left cosets of subgroup G of S_n

Input: PSCA parameters (n,k,λ) and $|G|$ and $A = (a_{i,j})$ = left (n,k)-incidence matrix for G

1: **procedure** FINDPSCALEFT
2: **if** $|G|$ divides $k!\lambda$ **then**
3: $L_{\text{initial}} = [n!] \setminus \{i : a_{i,j} > \lambda \text{ for some } j\}$
4: $M_{\text{initial}} = (\sum_{\ell \in L_{\text{initial}}} a_{l,j})$
5: SEARCH($\emptyset$, $[\binom{n}{k}k!]$, $(0)^{\binom{n}{k}k!}$, L_{initial}, M_{initial})
6: **end if**
7: **end procedure**

Algorithm 5 Search for PSCA(n,k,λ) as union of right cosets of subgroup G of S_n

Input: PSCA parameters (n,k,λ) and $|G|$ and $A = (a_{i,j})$ = right (n,k)-incidence matrix for G

1: **procedure** FINDPSCARIGHT
2: **if** $|G|$ divides $k!\lambda$ **and** $a_{1,j} \leq \lambda$ for all j **then**
3: SEARCH$\Big(\emptyset$, $[\binom{n}{k}k!]$, $(0)^{\binom{n}{k}k!}$, $[n!]$, $(\frac{n!}{k!})^{\binom{n}{k}k!}$ $\Big)$
4: **end if**
5: **end procedure**

3.3.6 Example: partial search for PSCA(5,3,2)

We illustrate Algorithm 3 by presenting part of a search for all possible examples of a PSCA$(5,3,2)$ that is a union of 6 left cosets of the prescribed order 2 subgroup $\langle\sigma\rangle$

of S_5, where $\sigma = 13254$. We use the same 3-step visualization as in Section 3.1.3: early termination condition; finding j^* and $L(j^*)$; and updating I, J, R, L, M.

We begin with the left (5,3)-incidence matrix A for $\langle\sigma\rangle$, having $\frac{5!}{2} = 60$ rows and $\binom{5}{3}3! = 60$ columns. Suppose that the first three iterations added indices $1, 51, 28$ corresponding to left cosets $\langle\sigma\rangle$, $43215\langle\sigma\rangle$, $24315\langle\sigma\rangle$ to I in that order, and that we are now entering SEARCH at Line 1 with $|L| = 6$ and $|J| = 38$, as shown in Table 3.6. As previously, M is the sum of all rows of the remaining matrix, and R is the sum of the three red rows that correspond to the indices in I.

Iteration 4. Table 3.7 shows Steps 1 and 2. The green positions are those at which R attains its maximum value of 1. The light blue positions are those at which M attains its minimum value of 1 over the green positions. Suppose we choose $j^* = 40$ from the light blue positions. The dark blue row indices $L(j^*) = \{47\}$ are those containing a nonzero entry in column j^*, and we recurse over i in this set.

Table 3.8 illustrates Step 3. The pink columns, to be deleted from J, are those at which the sum of R and row 47 equals $\lambda = 2$. In Algorithm 2, we determined the orange row indices B to be deleted from L as the row index of the recursion together with the indices of all rows containing an entry 1 in at least one of the pink columns. In contrast, we now determine the orange row indices B as the row index 47 of the recursion together with the indices of all rows whose sum with R + row 47 contains an entry greater than $\lambda = 2$: this test is not restricted to the pink column positions. In this case, we colour rows 55 and 56 orange because their sum with R + row 47 is greater than 2 at the column positions shown in yellow. We then update M by subtracting (in column positions that are not pink) the sum Σ of the orange rows.

This leaves the 3×34 matrix shown in Table 3.9 for *Iteration 5.*

Table 3.6: The submatrix of the left (5,3)-incidence matrix for $\langle 13254 \rangle$ relative to L and J

		J	1	4	7	8	9	10	11	12	13	19	20	22	23	24	25	31	32	33	34	35	37	38	40	41	42	44	46	47	48	49	50	53	55	56	57	58	59	60
L			1 2 3	1 3 2	1 4 2	1 4 3	1 4 5	1 5 2	1 5 3	1 5 4	2 1 3	2 4 1	2 4 3	2 5 1	2 5 3	2 5 4	3 1 2	3 4 1	3 4 2	3 4 5	3 5 1	3 5 2	4 1 2	4 1 3	4 2 1	4 2 3	4 2 5	4 3 2	4 5 1	5 1 2	5 1 3	5 2 3	5 3 1	5 3 2	5 3 4	5 4 1	5 4 2	5 4 3
1	$\langle\sigma\rangle$	12345 13254	1	1			1			1						1				1																				
51	$43215\langle\sigma\rangle$	43215 52314																							1		1	1						1	1		1			
28	$24315\langle\sigma\rangle$	24315 35214										1	1								1	1																		
12	$14523\langle\sigma\rangle$	14523 15432	1	1	2	2	1	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	0	1	0	1	0	0	1	1
13	$14532\langle\sigma\rangle$	14532 15423	1	1	2	2	1	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	0	1	0	1	0	0	1	1
35	$25413\langle\sigma\rangle$	25413 34512	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1	0	1	1	0	0	0	0	1	0	1
47	$42513\langle\sigma\rangle$	42513 53412	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	1	1	1	1	0	1	0	1	1	1	0	1	1	1	1	1	0
55	$45123\langle\sigma\rangle$	45123 54132	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0	1	0	1	1	1	1	2	2	1	0	1	0	1	1	1
56	$45132\langle\sigma\rangle$	45132 54123	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0	1	0	1	1	1	1	2	2	1	0	1	0	1	1	1
R			1	1	0	0	1	0	0	1	0	1	1	0	0	1	0	0	0	1	1	1	0	0	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0
M			4	4	4	4	2	4	4	2	2	1	1	2	2	1	2	2	2	1	1	1	6	6	1	5	1	4	4	5	5	6	6	4	1	5	1	4	5	5

Table 3.7: Algorithms 3 and 4, Iteration 4, Steps 1 and 2

L \ J	1	4	7	8	9	10	11	12	13	19	20	22	23	24	25	31	32	33	34	35	37	38	40	41	42	44	46	47	48	49	50	53	55	56	57	58	59	60
1 51 28																																						
12	1	1	2	2	1	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	0	1	0	1	0	0	1	1
13	1	1	2	2	1	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	0	1	0	1	0	0	1	1
35	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1	0	1	1	0	0	0	0	1	0	1
47	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	1	1	1	1	0	1	0	1	1	1	0	1	1	1	1	1	0
55	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0	1	0	1	1	1	1	2	2	1	0	1	0	1	1	1
56	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0	1	0	1	1	1	1	2	2	1	0	1	0	1	1	1
R	1	1	0	0	1	0	0	1	0	1	1	0	0	1	0	0	0	1	1	1	0	0	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0
M	4	4	4	4	2	4	4	2	2	1	1	2	2	1	2	2	2	1	1	1	6	6	1	5	1	4	4	5	5	6	6	4	1	5	1	4	5	5

Table 3.8: Algorithms 3 and 4, Iteration 4, Step 3

L \ J	1	4	7	8	9	10	11	12	13	19	20	22	23	24	25	31	32	33	34	35	37	38	40	41	42	44	46	47	48	49	50	53	55	56	57	58	59	60
1																																							
51																																						
28																																						
12	1	1	2	2	1	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	0	1	0	1	0	0	1	1
13	1	1	2	2	1	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	0	1	0	1	0	0	1	1
35	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1	0	1	1	0	0	0	0	1	0	1
47	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	1	1	1	1	1	0	1	0	1	1	1	0	1	1	1	1	1	0
55	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0	1	0	1	1	1	1	2	2	1	0	1	0	1	1	1
56	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0	1	0	1	1	1	1	2	2	1	0	1	0	1	1	1
R	1	1	0	0	1	0	0	1	0	1	1	0	0	1	0	0	0	1	1	1	0	0	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0
R + row 47	1	1	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	2	1	1	0	1	1	1	1	2	1	2	1	1	0
M	4	4	4	4	2	4	4	2	2	1	1	2	2	1	2	2	2	1	1	1	6	6	1	5	1	4	4	5	5	6	6	4	1	5	1	4	5	5
Σ	2	2	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	5	5		3		2	3	2	3	5	5	2		3		3	3	2
M_{new}	2	2	4	4	2	4	4	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1		2		2	1	3	2	1	1	2		2		1	2	3

Table 3.9: Algorithms 3 and 4, Iteration 5, Step 1

L \ J	1	4	7	8	9	10	11	12	13	19	20	22	23	24	25	31	32	33	34	35	37	38	41	44	46	47	48	49	50	53	56	58	59	60
1																																		
51																																		
28																																		
47																																		
12	1	1	2	2	1	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	1	1	0	1	1
13	1	1	2	2	1	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	1	1	0	1	1
35	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	1	1	0	0	1	0	1
R	1	1	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	0
M	2	2	4	4	2	4	4	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2	1	3	2	1	1	2	2	1	2	3

3.3.7 Comments

1. Algorithm 3 combines ideas due to Mathon and van Trung contained in Algorithm 1, extensions to their method for $\lambda > 1$ contained in Algorithm 2, our reinterpretation of the Yuster construction using left cosets of an order 2 subgroup, and generalization to larger order subgroups and to right cosets.

2. The critical computational advantage of Algorithm 3 over Algorithm 2 is that the maximum search depth is reduced by a factor of $|G|$ (as specified in Line 9). This gives a dramatic speed improvement, even for $|G| = 2$.

3. We expect the overhead for precalculating the (n, k)-incidence matrix for the prescribed subgroups G to be small in relation to the running time of Algorithm 3. However, we chose to precalculate these incidence matrices using GAP as an interpreted language, and to carry out the recursive procedure SEARCH in C. In consequence, we found that the running time of the precalculation can sometimes be longer than that of the recursion. However, this applies only to searches whose total running time is relatively small, and we therefore did not attempt to compile the GAP code.

4. Algorithm 3 searches for a PSCA(n, k, λ) as a union of left or right cosets of a prescribed nontrivial subgroup of S_n. As noted in Section 2.8, the right action of a permutation on a set of n-sequences relabels the symbols of the sequences, whereas the left action permutes the positions of the symbols. The Kramer-Mesner construction method for designs with a prescribed automorphism group [KM76],[KÖ06, Chap. 9.2] searches for symmetries of the design under symbol relabelling. This method is therefore related to Algorithm 3 for the case of right cosets but not left cosets; our most powerful search results using Algorithm 3 arise for left cosets (see Section 3.4).

3.4 Search results

In this section, we present exhaustive search results for a PSCA(n, k, λ) formed as a union of cosets of a prescribed nontrivial subgroup G of S_n. We use Algorithms 3 and 4 for left cosets, and Algorithms 3 and 5 for right cosets. As noted in Section 3.3.2, for left cosets it is sufficient to search over a single representative of each conjugacy class of nontrivial subgroups of S_n; whereas for right cosets we must search over all nontrivial subgroups of S_n, although we may assume the presence of the identity coset.

Our objectives in searching for examples with this structure are to form a deeper understanding of the existence pattern for perfect sequence covering arrays, and to determine new g-values. We find that $g(7, 4) = 2$ (Proposition 3.9) and $g(8, 3) \in \{2, 3\}$ (Theorem 3.10).

For a nontrivial subgroup G of S_n, abbreviate the condition "there exists a PSCA(n, k, λ) formed as a union of (left or right) cosets of G" as "G is a (left or right) subgroup for

(n, k, λ)". If G is a subgroup for (n, k, λ), then every nontrivial subgroup H of G is also a subgroup for (n, k, λ). For each parameter set (n, k, λ), we therefore wish to determine the maximal left and right subgroups (according to inclusion). For simplicity, we report only the maximal isomorphism classes of these maximal subgroups.

For example, recall that Mathon and van Trung [MvT99] determined that (up to equivalence) a PSCA$(6, 4, 1)$ is either a subgroup $G_1 \cong S_4$ of S_6, or else is a union of three right cosets of a subgroup $G_2 \cong D_8$ of S_6 (see Theorem 2.46). Therefore G_1 and G_2 are each maximal right subgroups for $(6, 4, 1)$. Even though G_2 is not a subgroup of G_1, there is a subgroup $G_3 \cong D_8$ of G_1 that is a (non-maximal) right subgroup for $(6, 4, 1)$. The isomorphism classes of the maximal right subgroups are S_4 and D_8, and we report only the single maximal isomorphism class S_4.

For further illustration, let $n > 4$. Then $G \cong S_4$ is a maximal left subgroup for (n, k, λ) if no subgroup of S_n that properly contains G is a left subgroup for (n, k, λ); and S_4 is then a maximal isomorphism class of left subgroups for (n, k, λ) if no subgroup of S_n containing a subgroup isomorphic to S_4 is a left subgroup for (n, k, λ). In order to determine whether there are maximal isomorphism classes other than S_4, we then need to check whether there are subgroups of S_n that are not subgroups of S_4 (for example D_{12}, C_5, C_6) that are also left subgroups for (n, k, λ).

It follows from Section 3.3.7 (2) that the larger $|G|$ is, the more dramatic the speed improvement of Algorithm 3 over Algorithm 2 is. In order to determine whether there exists some nontrivial subgroup G that is a left or right subgroup for (n, k, λ) (and thereby establish the existence of a PSCA(n, k, λ)), it is therefore most efficient to examine the required subgroups G in decreasing order of $|G|$. We do not take G to be the trivial group: in that case, the method reduces to Algorithm 2 and, even if a perfect sequence covering array is found, no additional structure is identified. In order to determine the maximal isomorphism classes of maximal left or right subgroups for (n, k, λ), we also examine subgroups in decreasing order of $|G|$; if, for example, we find an isomorphism class for some even $|G| > 2$ then it is not necessary to examine subgroups G of order 2.

3.4.1 Left cosets

We present the maximal isomorphism classes of maximal left subgroups for various parameter sets (n, k, λ) in Table 3.10, and include a source for an example perfect sequence covering array with the specified structure.

We already know from the Levenshtein construction of Theorem 2.18 that $g(n, n-1) = 1$ for each $n \geq 2$. The results for $\lambda = 1$ and $4 \leq n \leq 7$ are in agreement with this result, but display additional structure that is not apparent from that construction.

Table 3.10: Maximal isomorphism classes of nontrivial maximal left subgroups

(n,k)	λ	isomorphism class	example
$(4,3)$	1	C_2	Proposition 3.11
$(5,4)$	1	C_4, $C_2 \times C_2$	Proposition 3.12
$(6,4)$	1	S_4	Theorem 2.46
$(6,5)$	1	D_{10}, S_4	Proposition 3.13
$(7,6)$	1	S_4	Proposition 3.14
$(5,3)$	2	C_4, $C_2 \times C_2$, S_3	Proposition 3.15
$(6,3)$	2	C_4, D_{12}, A_4	Proposition 3.16
$(7,3)$	2	C_6, S_3	Proposition 3.17
$(7,4)$	2	C_6	Proposition 3.9
$(8,3)$	2	none	
$(8,4)$	2	none except possibly C_2	
$(9,3)$	2	none except possibly C_2	
$(5,3)$	3	C_2	Proposition 3.18
$(6,3)$	3	C_6, S_3	Proposition 3.19
$(7,3)$	3	C_2	Proposition 3.20
$(8,3)$	3	C_2	Proposition 3.21

We see from Table 3.10 that $g(8,3) \leq 3$. Since $g(5,3) > 1$ by Theorem 2.25, we also have $g(8,3) > 1$ by Result 1.11. Combining these results gives the following conclusion.

Theorem 3.10. *We have* $2 \leq g(8,3) \leq 3$.

We were not able to determine whether $g(8,3) = 2$: an exhaustive search for a PSCA$(8,3,2)$ using Algorithm 2 did not complete. However, we see from Table 3.10 that if a PSCA$(8,3,2)$ exists then it does not occur as a union of left cosets of a nontrivial subgroup of S_8. Likewise, if a PSCA$(8,4,2)$ or PSCA$(9,3,2)$ exists then it does not occur as a union of left cosets of a subgroup of S_8 of order greater than 2.

If there exists a PSCA(n,k,λ) and a disjoint PSCA(n,k,μ), then their union is a PSCA$(n,k,\lambda+\mu)$. However, for $n \in \{5,6,7\}$ we cannot construct a PSCA$(n,3,3)$ as the union of a PSCA$(n,3,2)$ and a disjoint PSCA$(n,3,1)$: for these values of n we know from Propositions 2.41 and 3.1 that $g(n,3) = 2$ and so there is no PSCA$(n,3,1)$. Therefore, the examples of a PSCA$(n,3,3)$ for $n \in \{5,6,7\}$ shown in Table 3.10 are interesting and nontrivial.

Table 3.10 indicates the current limits of computation using Algorithms 3 and 4: for $(8,4,2)$ and for $(9,3,2)$, we were able to complete the search for all conjugacy classes of left subgroups G when $|G| > 2$, but not when $|G| = 2$.

Proposition 3.11. *The following* 6 *sequences form a* PSCA$(4,3,1)$ *as a union of* 3 *left cosets of the subgroup* $G = \langle 3412 \rangle \cong C_2$ *of* S_4*:*

G	$1432\,G$	$2413\,G$
1234	1432	2413
3412	3214	4231

Proposition 3.12. *Each of the following sets of* 24 *sequences forms a* PSCA$(5,4,1)$ *as a union of left cosets of the subgroup* G *of* S_5*.*

(1) 2 *left cosets of* $G = \langle 24135 \rangle \cong C_4$*:*

G	$13254\,G$	$14352\,G$	$15243\,G$	$15342\,G$	$51423\,G$
12345	13254	14352	15243	15342	51423
24135	21453	23154	25431	25134	52341
31425	34152	32451	35124	35421	53214
43215	42351	41253	45312	45213	54132

(2) 2 *left cosets of* $G = \langle 34125, 43215 \rangle \cong C_2 \times C_2$*:*

G	$13254\,G$	$14253\,G$	$15243\,G$	$15342\,G$	$51432\,G$
12345	13254	14253	15243	15342	51432
21435	24153	23154	25134	25431	52341
34125	31452	32451	35421	35124	53214
43215	42351	41352	45312	45213	54123

Proposition 3.13. *Each of the following sets of* 120 *sequences (listed explicitly in Appendix A) forms a* PSCA$(6,5,1)$ *a union of left cosets of the subgroup* G *of* S_6*.*

(1) 12 *left cosets of* $G = \langle 245316, 532416 \rangle \cong D_{10}$*:*

$123564\,G$	$124536\,G$	$124635\,G$	$126534\,G$	$132645\,G$	$134526\,G$
$135462\,G$	$136254\,G$	$162345\,G$	$163524\,G$	$612543\,G$	$613425\,G$

(2) 5 *left cosets of* $G = \langle 125634, 346521 \rangle \cong S_4$*:*

G	$132546\,G$	$132645\,G$	$135642\,G$	$136524\,G$

Proposition 3.14. *The following* 720 *sequences (listed explicitly in Appendix A) form a* PSCA$(7, 6, 1)$ *as a union of* 30 *left cosets of the subgroup* $G = \langle 1756432, 7235461\rangle \cong S_4$ *of* S_7*:*

$1234657\,G$	$1235476\,G$	$1237456\,G$	$1273564\,G$	$1324576\,G$
$1325467\,G$	$1326475\,G$	$1326574\,G$	$1342756\,G$	$1345267\,G$
$1352764\,G$	$1356427\,G$	$1357246\,G$	$3124756\,G$	$3125746\,G$
$3126745\,G$	$3127654\,G$	$3145627\,G$	$3154267\,G$	$3154762\,G$
$3156742\,G$	$3412657\,G$	$3415276\,G$	$3421567\,G$	$3425176\,G$
$3426715\,G$	$3427651\,G$	$3456172\,G$	$3457126\,G$	$3457621\,G$

Proposition 3.15. *Each of the following sets of* 12 *sequences forms a* PSCA$(5, 3, 2)$ *as a union of left cosets of the subgroup* G *of* S_5*.*

(1) 3 *left cosets of* $G = \langle 23541\rangle \cong C_4$*:*

G	$15342\,G$	$41325\,G$
12345	15342	41325
23541	21543	42531
35142	32145	43152
51243	53241	45213

(2) 3 *left cosets of* $G = \langle 21543, 35142\rangle \cong C_2 \times C_2$*:*

G	$13245\,G$	$41523\,G$
12345	13245	41523
21543	25143	42315
35142	31542	43251
53241	52341	45132

(3) 2 *left cosets of* $G = \langle 15324, 35142\rangle \cong S_3$*:*

G	$15342\,G$
12345	24153
14352	25341
15324	42351
32154	45123
34125	52143
35142	54321

Proposition 3.16. *Each of the following sets of* 12 *sequences forms a* PSCA$(6, 3, 2)$ *as a union of left cosets of the subgroup* G *of* S_6.

(1) 3 *left cosets of* $G = \langle 235614 \rangle \cong C_4$*:*

$125436\,G$	$145263\,G$	$416325\,G$
125436	145263	416325
231654	261345	436152
352416	342561	624531
513624	563142	654213

(2) the identity coset of $G = \langle 154326, 216543 \rangle \cong D_{12}$*:*

G		
123456	154326	216543
245613	354162	361452
423165	461325	516234
532614	632541	645231

(3) the identity coset of $G = \langle 153624, 245163 \rangle \cong A_4$*:*

G		
123456	153624	245163
265341	321654	351426
416235	436512	542361
562143	614532	634215

Proposition 3.17. *Each of the following sets of* 12 *sequences forms a* PSCA$(7, 3, 2)$ *as a union of left cosets of the subgroup* G *of* S_7.

(1) 2 *left cosets of* $G = \langle 4735621 \rangle \cong C_6$*:*

$1274635\,G$	$1357642\,G$
1274635	1357642
2567431	2316475
4715236	4361257
5146732	5324761
6452137	6375124
7621534	7342516

(2) 2 left cosets of $G = \langle 4731652, 5432176 \rangle \cong S_3$*:*

$1245637\,G$	$1365427\,G$
1245637	1365427
2657134	2317564
4716532	4356172
5421736	5371246
6174235	6324715
7562431	7342651

Proposition 3.18. *The following* 18 *sequences form a* PSCA$(5, 3, 3)$ *as a union of* 9 *left cosets of the subgroup* $G = \langle 35142 \rangle \cong C_2$ *of* S_5*:*

G	$12354\,G$	$14325\,G$	$15342\,G$	$21543\,G$	$24315\,G$	$25143\,G$	$41523\,G$	$42513\,G$
12345	12354	14325	15342	21543	24315	25143	41523	42513
35142	35124	34152	32145	53241	54132	52341	43251	45231

Proposition 3.19. *Each of the following sets of* 18 *sequences forms a* PSCA$(6, 3, 3)$ *as a union of left cosets of the subgroup* G *of* S_7*.*

(1) 3 left cosets of $G = \langle 562341 \rangle \cong C_6$*:*

G	$132645\,G$	$145326\,G$
123456	132645	145326
245163	254316	216543
351624	315462	362154
416235	461523	423615
562341	526134	534261
634512	643251	651432

(2) 3 left cosets of $G = \langle 254613, 645231 \rangle \cong S_3$*:*

G	$142536\,G$	$154623\,G$
123456	142536	154623
254613	265143	216354
361542	356412	345261
432165	413625	461532
516324	531264	523416
645231	624351	632145

Proposition 3.20. *The following* 18 *sequences form a* PSCA$(7, 3, 3)$ *as a union of* 9 *left cosets of the subgroup* $G = \langle 5327164 \rangle \cong C_2$ *of* S_4*:*

G	$1254367\,G$	$1643572\,G$	$1735462\,G$	$2745631\,G$
1234567	1254367	1643572	1735462	2745631
5327164	5317264	5672143	5421763	3471625

$4326175\,G$	$4721653\,G$	$6231745\,G$	$6451273\,G$
4326175	4721653	6231745	6451273
7236541	7435612	6325471	6715342

Proposition 3.21. *The following* 18 *sequences form a* PSCA$(8, 3, 3)$ *as a union of* 9 *left cosets of the subgroup* $G = \langle 85672341 \rangle \cong C_2$ *of* S_8*:*

G	$15468237\,G$	$17624385\,G$	$27561843\,G$	$28461573\,G$
12345678	15468237	17624385	27561843	28461573
85672341	82731564	84357612	54238176	51738246

$31864275\,G$	$32654187\,G$	$37461528\,G$	$47218653\,G$
31864275	32654187	37461528	47218653
68137542	65327814	64738251	74581326

3.4.2 Right cosets

We summarize the maximal isomorphism classes of maximal right subgroups for various parameter sets (n, k, λ) in Table 3.11. As previously noted, the searches for right cosets are significantly larger than those for left cosets. Furthermore, although we obtain some positive results, the structure uncovered is less rich than for left cosets (compare with Table 3.10). We therefore did not attempt to carry out some of the larger searches.

In particular, in each case we found that if C is an isomorphism class of maximal right subgroups for (n, k, λ), then C is also an isomorphism class of maximal left subgroups for (n, k, λ). This does not appear to have a trivial explanation. For example, the PSCA$(7, 3, 2)$ in Section 3.3.1 can be represented as six right cosets of the order 2 subgroup $\langle 3617524 \rangle$ of S_7. However, it cannot be represented as six left cosets of the same subgroup, even though it can be written as six left cosets of (several) other order 2 subgroups of S_7.

Three of the entries of Table 3.11 can be predicted from previous results. By Theorem 2.46, there is a PSCA$(6, 4, 1)$ that is a single subgroup $G \cong S_4$ of S_6, so S_4 is a maximal isomorphism class of right as well as left subgroups for $(6, 4, 1)$. Similarly, by Proposition 3.16 we have that D_{12} and A_4 are each a maximal isomorphism class of right as

well as left subgroups for $(6,3,2)$. There are only two other entries of Table 3.11 for which the isomorphism class is larger than C_2, and we provide examples for these in Propositions 3.22 and 3.23.

Table 3.11: Maximal isomorphism classes of nontrivial maximal right subgroups

(n,k)	λ	isomorphism class
$(4,3)$	1	none
$(5,4)$	1	C_2
$(6,4)$	1	S_4
$(6,5)$	1	C_2
$(7,6)$	1	C_2
$(5,3)$	2	C_2
$(6,3)$	2	C_4, D_{12}, A_4
$(7,3)$	2	C_2
$(7,4)$	2	C_2
$(8,3)$	2	none
$(5,3)$	3	none
$(6,3)$	3	C_3, C_2
$(7,3)$	3	none

Proposition 3.22. *The following* 12 *sequences form a* PSCA$(6,3,2)$ *as a union of* 3 *right cosets of the subgroup* $G = \langle 352614 \rangle \cong C_4$ *of* S_6*:*

G	$G\,164352$	$G\,263541$
123456	164352	263541
215436	456213	346125
352614	541263	432165
531624	615342	624531

Proposition 3.23. *The following* 18 *sequences form a* PSCA$(6,3,3)$ *as a union of* 6 *right cosets of the subgroup* $G = \langle 345612 \rangle \cong C_3$ *of* S_6*:*

G	$G\,154263$	$G\,163254$	$G\,214365$	$G\,253164$	$G\,264153$
123456	154263	163254	214365	253164	264153
345612	426315	325416	436521	316425	415326
561234	631542	541632	652143	642531	532641

3.4.3 Updated table of g-values

Table 3.12 is an updated table of g-values for small n and k, with our new results shown in red:

- $g(6,3) = g(7,3) = 2$ (Proposition 3.1, using Algorithm 2)
- $g(7,4) = 2$ (Proposition 3.9, using Algorithms 3 and 4),
- $2 \leq g(8,3) \leq 3$ (Theorem 3.10, using Algorithms 3 and 4).

Table 3.12: Updated g-values

n \ k	2	3	4	5	6	7
2	1					
3	1	1				
4	1	1	1			
5	1	2	1	1		
6	1	2	1	1	1	
7	1	2	2	>1	1	1
8	1	2 or 3			>1	1

3.4.4 Search times

Table 3.13 shows the CPU time required to search for all possible PSCA(n,k,λ) under the specified conditions, using a C implementation on an Intel Xeon E5-2680 (1 core) and excluding the time required to precalculate incidence matrices in GAP for Algorithm 3. The comparison times for the parameter sets $(7,5,1)$ and $(8,6,1)$ taken from [MvT99] refer to searches on an Ultra SPARCstation 5 in 1999. Algorithm 2 assumes the perfect sequence covering array contains the sequence $12\cdots n$. Algorithms 3 and 4 search each conjugacy class of subgroups of S_n of order g using left cosets. Algorithms 3 and 5 search each subgroup of S_n of order g using right cosets, and assume the perfect sequence covering array contains the identity coset. The times shown for Algorithms 3 and 4, and for Algorithms 3 and 5, are for all required subgroups of order g.

Table 3.13: CPU time to search for all possible PSCA(n,k,λ) under the specified conditions

method	(n,k,λ)	g	CPU time	notes
Algorithm 2	$(7,5,1)$		0.1 seconds	5 minutes in 1999 [MvT99]
Algorithm 2	$(8,6,1)$		100 minutes	100 hours in 1999 [MvT99]
Algorithm 2	$(6,3,2)$		3 seconds	
Algorithm 2	$(7,3,2)$		40 minutes	
Algorithms 3 and 4	$(7,3,2)$	6	0.04 seconds	
Algorithms 3 and 4	$(7,4,2)$	6	3 seconds	
Algorithms 3 and 5	$(7,3,2)$	2	8 seconds	

Chapter 4

Review and Open Problems

4.1 Review

The central objective in the study of perfect sequence covering arrays is to determine $g(n, k)$ (the smallest value of λ for which a PSCA(n, k, λ) exists) for all n and k. Yuster [Yus20] determined in 2020 the first known exact g-value greater than 1, namely $g(5, 3) = 2$, but concluded:

> "Proving additional exact values of $g(n, k)$ which are not of unit multiplicity in addition to $g(5, 3)$ also seems challenging".

Despite this, we have determined that $g(6, 3) = g(7, 3) = g(7, 4) = 2$ and $g(8, 3) \in \{2, 3\}$, by modifying the search method of Algorithm 1 due to Mathon and van Trung [MvT99], and by identifying and prescribing an algebraic structure for perfect sequence covering arrays. These insights bring into reach various searches that would otherwise be intractable, and provide a deeper understanding of the existence pattern for perfect sequence covering arrays.

Algorithm 2 improves on Algorithm 1 in two ways: it extends the exhaustive recursive search for a PSCA$(n, k, 1)$ to a search for a PSCA(n, k, λ) for arbitrary $\lambda \geq 1$, and reduces the required computation by passing as a recursion parameter the sum of certain rows of the remaining incidence matrix.

Algorithm 3 is motivated by the observation that the Yuster construction [Yus20] of a PSCA$(5, 3, 2)$ can be reinterpreted using left cosets of an order 2 subgroup of S_n. Restricting attention to perfect sequence covering arrays that are formed as a union of left or right cosets of a nontrivial subgroup G of S_n is a key insight that allows us to combine combinatorial and algebraic properties. Algorithm 3 modifies Algorithm 2 by assuming that a perfect sequence covering array has this algebraic structure, building the array one coset of G at a time rather than one sequence at a time. This reduces the search depth and so dramatically reduces the search time. For example, as shown in Table 3.13, the time required to search exhaustively using Algorithm 2 for a PSCA$(7, 3, 2)$ is 40 minutes; whereas the time required to search

exhaustively using Algorithms 3 and 4 over all conjugacy classes of order 6 subgroups G for a PSCA$(7, 3, 2)$ that is a union of left cosets of G is only 0.04 seconds.

Table 3.10 shows a rich existence pattern for the maximal isomorphism classes of nontrivial maximal left subgroups for various parameter sets (n, k, λ). This strongly suggests that the algebraic structure proposed here for a perfect sequence covering array is a valuable concept. The corresponding data in Table 3.11 for right subgroups are less rich, and the searches require more time. We note that the k-(n, n, λ) directed designs corresponding to the PSCA(n, k, λ) examples given in Chapter 3 are not restricted to directed Steiner systems (having $\lambda = 1$: see Definition 1.13).

4.2 Open problems

We conclude by proposing several open problems arising from our findings.

1. Determine further exact values of $g(n, k)$ (see Table 3.12). In particular, determine which of $g(8, 3) = 2$ and $g(8, 3) = 3$ is correct.

2. The recursive search methods presented here appear to encounter memory constraints when attempting to settle the smallest open case of Conjecture 2.24, namely whether $g(9, 7) > 1$. Are there theoretical techniques or improved search methods for handling this case?

3. Find a convincing explanation for why the existence pattern in Table 3.10 for left subgroups is richer than that in Table 3.11 for right subgroups.

4. Can the data summarized in Section 3.4 be used to infer the construction of a new infinite family of perfect sequence covering arrays?

5. Find more combinatorial nonexistence results, for example by developing arguments similar to those described in Section 2.4.

6. Is it true that $g(n - 1, k) \leq g(n, k)$? This holds under the convention that a perfect sequence covering array can be a multiset (see Remark 1.12), but the argument used to establish this does not hold under our convention that it must be a set.

7. Is it true that if a PSCA(n, k, λ) exists, then there exists a PSCA(n, k, λ) that is a union of left cosets of some nontrivial subgroup of S_n? The data in Table 3.10 are consistent with the answer "yes" (noting that we do not currently know whether a PSCA$(8, 3, 2)$ exists). By exhaustive search, we find that the following PSCA$(5, 3, 2)$ is not a union of cosets of a nontrivial subgroup of S_5, and so the statement is not true if we replace "then there exists a PSCA(n, k, λ) that is" by "then every PSCA(n, k, λ) is":

   {12345, 13254, 14532, 15423, 24513, 25314, 34512, 35214, 42315, 43215, 52413, 53412}.

8. Determine further examples of a PSCA(n, k, λ) for which $\lambda > g(n, k)$ and which cannot be decomposed into two smaller disjoint (n, k)-perfect sequence covering arrays (see Section 3.4.1 for the cases $(n, k, \lambda) = (5, 3, 3)$, $(6, 3, 3)$, $(7, 3, 3)$).

Bibliography

[ARA09] B. Apilli, L. Richardson, and C. Alexander. Fault-based combinatorial testing of web services. In *Proceedings of the 24th ACM SIGPLAN Conference Companion on Object Oriented Programming Systems Languages and Applications*, pages 731–732, 2009.

[BM07] F. E. Bennett and A. Mahmoodi. Directed designs. In C. J. Colbourn and J. H. Dinitz, editors, *Handbook of Combinatorial Designs*, pages 441–444. Chapman and Hall/CRC, Boca Raton, 2nd edition, 2007.

[CCHZ13] Y. M. Chee, C. J. Colbourn, D. Horsley, and J. Zhou. Sequence covering arrays. *SIAM Journal on Discrete Mathematics*, 27(4):1844–1861, 2013.

[Col11] C. J. Colbourn. Covering arrays and hash families. In *Information Security, Coding Theory and Related Combinatorics*, NATO Science for Peace and Security Series-D: Information and Communication Security, pages 99–135, Amsterdam, 2011. IOS Press.

[DSS84] J. E. Dawson, J. Seberry, and D. B. Skillicorn. The directed packing numbers $\mathrm{DD}(t, v, v), t \geq 4$. *Combinatorica (Budapest. 1981)*, 4(2–3):121–130, 1984.

[Für96] Z. Füredi. Scrambling permutations and entropy of hypergraphs. *Random Structures & Algorithms*, 8(2):97–104, 1996.

[GAP20] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.11.0*, 2020.

[HCM10] S. Huang, M. B. Cohen, and A. M. Memon. Repairing GUI test suites using a genetic algorithm. In *2010 Third International Conference on Software Testing, Verification and Validation (ICST)*, pages 245–254, 2010.

[Ish96] Y. Ishigami. An extremal problem of d permutations containing every permutation of every t elements. *Discrete Mathematics*, 159(1–3):279–283, 1996.

[KHL+12] D. R. Kuhn, J. M. Higdon, J. Lawrence, R. Kacker, and Y. Lei. Combinatorial methods for event sequence testing. In *2012 IEEE Fifth International Conference on Software Testing, Verification and Validation*, pages 601–609, 2012.

[Kle04] A. Klein. On perfect deletion-correcting codes. *Journal of Combinatorial Designs*, 12(1):72–77, 2004.

[KM76] E. S. Kramer and D. M. Mesner. t-designs on hypergraphs. *Discrete Mathematics*, 15:263–296, 1976.

[KÖ06] P. Kaski and P. R. J. Östergård. *Classification Algorithms for Codes and Designs*. Springer, Berlin, 2006.

[Lev90] V. I. Levenshtein. Perfect deletion-correcting codes as combinatorial designs. *II International Workshop Algebraic and Combinatorial Coding Theory*, pages 137–140, 1990.

[Lev91] V. I. Levenshtein. Perfect codes in the metric of deletions and insertions. *Diskretnaya Matematika*, 3(1):3–20, 1991. English translation in: *Discrete Mathematics and Applications*, 2(3):241–258, 1992.

[Mat97] R. Mathon. Searching for spreads and packings. In J. W. P. Hirschfeld, S. S. Magliveras, and M. J. de Resmini, editors, *Geometry, Combinatorial Designs and Related Structures*, Proceedings of the First Pythagorean Conference, pages 161–176. Cambridge University Press, 1997.

[MvT99] R. Mathon and T. van Trung. Directed t-packings and directed t-Steiner systems. *Designs, Codes and Cryptography*, 18(1):187–198, 1999.

[Rad03] J. Radhakrishnan. A note on scrambling permutations. *Random Structures & Algorithms*, 22(4):435–439, 2003.

[Spe72] J. Spencer. Minimal scrambling sets of simple orders. *Acta Mathematica Hungarica*, 22(3–4):349–353, 1972.

[Wil90] R. M. Wilson. A diagonal form for the incidence matrices of t-subsets vs. k-subsets. *European Journal of Combinatorics*, 11(6):609–615, 1990.

[WLS+09] W. Wang, Y. Lei, S. Sampath, R. Kacker, R. Kuhn, and J. Lawrence. A combinatorial approach to building navigation graphs for dynamic web applications. In *2009 IEEE International Conference on Software Maintenance*, page 211–220, 2009.

[WSLK08] W. Wang, S. Sampath, Y. Lei, and R. Kacker. An interaction-based test sequence generation approach for testing web applications. In *2008 11th IEEE High Assurance Systems Engineering Symposium*, page 209–218, 2008.

[YCM11] X. Yuan, M. B. Cohen, and A. M. Memon. GUI interaction testing: Incorporating event context. *IEEE Transactions on Software Engineering*, 37(4):559–574, 2011.

[YM10] X. Yuan and A. M. Memon. Generating event sequence-based test cases using GUI runtime state feedback. *IEEE Transactions on Software Engineering*, 36(1):81–95, 2010.

[Yus20] R. Yuster. Perfect sequence covering arrays. *Designs, Codes and Cryptography*, 88(3):585–593, 2020.

Appendix A

Large left coset examples

The following 120 sequences form a PSCA$(6, 5, 1)$ as a union of 12 left cosets of the subgroup $G = \langle 245316, 532416 \rangle \cong D_{10}$ of S_6:

$123564\,G$	123564	154263	213465	245163	341562
	352461	425361	431265	514362	532164
$124536\,G$	124536	153246	215436	243156	342516
	351426	421356	435216	512346	534126
$124635\,G$	124635	153642	215634	243651	342615
	351624	421653	435612	512643	534621
$126534\,G$	126534	156243	216435	246153	346512
	356421	426351	436215	516342	536124
$132645\,G$	132645	145632	231654	254631	314625
	325614	413652	452613	523641	541623
$134526\,G$	134526	143256	235416	253146	312546
	321456	415236	451326	524136	542316
$135462\,G$	135462	142365	234561	251364	315264
	324165	412563	453162	521463	543261
$136254\,G$	136254	146523	236145	256413	316452
	326541	416325	456231	526314	546132
$162345\,G$	162345	165432	261354	264531	364125
	365214	462513	463152	561423	563241
$163524\,G$	163524	164253	263415	265143	361542
	362451	461235	465321	562134	564312
$612543\,G$	612543	615234	621453	624135	634521
	635412	642315	643251	651324	653142
$613425\,G$	613425	614352	623514	625341	631245
	632154	641532	645123	652431	654213

The following 120 sequences form a PSCA$(6, 5, 1)$ as a union of 5 left cosets of the subgroup $G = \langle 125634, 346521 \rangle \cong S_4$ of S_6:

G	123456	124365	125634	126543	213465	214356
	215643	216534	341256	342165	345612	346521
	431265	432156	435621	436512	561234	562143
	563412	564321	651243	652134	653421	654312
$132546\,G$	132546	142635	152364	162453	231645	241536
	251463	261354	314526	324615	354162	364251
	413625	423516	453261	463152	516324	526413
	536142	546231	615423	625314	635241	645132
$132645\,G$	132645	142536	152463	162354	231546	241635
	251364	261453	314625	324516	354261	364152
	413526	423615	453162	463251	516423	526314
	536241	546132	615324	625413	635142	645231
$135642\,G$	135642	146532	153462	164352	236541	245631
	254361	263451	315624	326514	351264	362154
	416523	425613	452163	461253	513426	524316
	531246	542136	614325	623415	632145	641235
$136524\,G$	136524	145623	154326	163425	235614	246513
	253416	264315	316542	325641	352146	361245
	415632	426531	451236	462135	514362	523461
	532164	541263	613452	624351	631254	642153

The following 720 sequences form a PSCA$(7, 6, 1)$ as a union of 30 left cosets of the subgroup $G = \langle 1756432, 7235461 \rangle \cong S_4$ of S_7:

$1234657\,G$	1234657	1243567	1256437	1265347	1734562	1743652	1756342	1765432
	2136457	2145367	2154637	2163547	2736541	2745631	2754361	2763451
	7135462	7146352	7153642	7164532	7235641	7246531	7253461	7264351
$1235476\,G$	1235476	1246375	1253674	1264573	1736425	1745326	1754623	1763524
	2135674	2146573	2153476	2164375	2734615	2743516	2756413	2765314
	7136524	7145623	7154326	7163425	7234516	7243615	7256314	7265413
$1237456\,G$	1237456	1247365	1257634	1267543	1732465	1742356	1752643	1762534
	2137654	2147563	2157436	2167345	2731645	2741536	2751463	2761354
	7132564	7142653	7152346	7162435	7231546	7241635	7251364	7261453
$1273564\,G$	1273564	1274653	1275346	1276435	1723654	1724563	1725436	1726345
	2173546	2174635	2175364	2176453	2713456	2714365	2715634	2716543
	7123645	7124536	7125463	7126354	7213465	7214356	7215643	7216534
$1324576\,G$	1324576	1374625	1423675	1473526	1526374	1576423	1625473	1675324
	2316574	2376415	2415673	2475316	2514376	2574613	2613475	2673514
	7315624	7325416	7416523	7426315	7513426	7523614	7614325	7624513
$1325467\,G$	1325467	1376452	1426357	1475362	1523647	1574632	1624537	1673542
	2315647	2374651	2416537	2473561	2513467	2576431	2614357	2675341
	7316542	7324561	7415632	7423651	7514362	7526341	7613452	7625431

	1326475	1375426	1425376	1476325	1524673	1573624	1623574	1674523
$1326475\,G$	2314675	2375614	2413576	2476513	2516473	2573416	2615374	2674315
	7314526	7326514	7413625	7425613	7516324	7524316	7615423	7623415
	1326574	1375624	1425673	1476523	1524376	1573426	1623475	1674325
$1326574\,G$	2314576	2375416	2413675	2476315	2516374	2573614	2615473	2674513
	7314625	7326415	7413526	7425316	7516423	7524613	7615324	7623514
	1342756	1347265	1432765	1437256	1562734	1567243	1652743	1657234
$1342756\,G$	2361754	2367145	2451763	2457136	2541736	2547163	2631745	2637154
	7351264	7352146	7461253	7462135	7531246	7532164	7641235	7642153
	1345267	1346752	1435762	1436257	1563247	1564732	1653742	1654237
$1345267\,G$	2364751	2365147	2453761	2456137	2543167	2546731	2634157	2635741
	7354261	7356142	7463251	7465132	7534162	7536241	7643152	7645231
	1352764	1367254	1457263	1462753	1532746	1547236	1637245	1642735
$1352764\,G$	2347156	2351746	2437165	2461735	2531764	2567134	2641753	2657143
	7342165	7361245	7432156	7451236	7541263	7562143	7631254	7652134
	1356427	1365472	1456372	1465327	1534627	1543672	1634572	1643527
$1356427\,G$	2345671	2354617	2436571	2463517	2536417	2563471	2645317	2654371
	7346521	7364512	7435621	7453612	7546312	7564321	7635412	7653421
	1357246	1362745	1452736	1467235	1537264	1542763	1632754	1647253
$1357246\,G$	2341765	2357164	2431756	2467153	2537146	2561743	2647135	2651734
	7341256	7362154	7431265	7452163	7542136	7561234	7632145	7651243
	3124756	3174265	3216754	3276145	3715264	3725146	4123765	4173256
$3124756\,G$	4215763	4275136	4716253	4726135	5126734	5176243	5214736	5274163
	5713246	5723164	6125743	6175234	6213745	6273154	6714235	6724153
	3125746	3176245	3215764	3274165	3716254	3724156	4126735	4175236
$3125746\,G$	4216753	4273156	4715263	4723165	5123764	5174263	5213746	5276143
	5714236	5726134	6124753	6173254	6214735	6275134	6713245	6725143
	3126745	3175246	3214765	3275164	3714256	3726154	4125736	4176235
$3126745\,G$	4213756	4276153	4713265	4725163	5124763	5173264	5216743	5273146
	5716234	5724136	6123754	6174253	6215734	6274135	6715243	6723145
	3127654	3172564	3217456	3271546	3712465	3721645	4127563	4172653
$3127654\,G$	4217365	4271635	4712356	4721536	5127436	5172346	5217634	5271364
	5712643	5721463	6127345	6172435	6217543	6271453	6712534	6721354
	3145627	3146572	3264571	3265417	3754621	3756412	4135672	4136527
$3145627\,G$	4253671	4256317	4763521	4765312	5163427	5164372	5243617	5246371
	5734612	5736421	6153472	6154327	6234517	6235471	6743512	6745321
	3154267	3164752	3246751	3256147	3745261	3765142	4153762	4163257
$3154267\,G$	4235761	4265137	4736251	4756132	5136247	5146732	5234167	5264731
	5743162	5763241	6135742	6145237	6243157	6253741	6734152	6754231
	3154762	3164257	3246157	3256741	3745162	3765241	4153267	4163752
$3154762\,G$	4235167	4265731	4736152	4756231	5136742	5146237	5234761	5264137
	5743261	5763142	6135247	6145732	6243751	6253147	6734251	6754132
	3156742	3165247	3245167	3254761	3746152	3764251	4156237	4165732
$3156742\,G$	4236157	4263751	4735162	4753261	5134762	5143267	5236741	5263147
	5746231	5764132	6134257	6143752	6245731	6254137	6735241	6753142
	3412657	3417562	3571462	3572641	3621457	3627541	4312567	4317652
$3412657\,G$	4521367	4527631	4671352	4672531	5371642	5372461	5421637	5427361
	5612437	5617342	6321547	6327451	6471532	6472351	6512347	6517432
	3415276	3416725	3574216	3576124	3624715	3625174	4315726	4316275
$3415276\,G$	4523716	4526173	4673215	4675123	5374126	5376214	5423176	5426713
	5613274	5614723	6324175	6325714	6473125	6475213	6513724	6514273

	3421567	3471652	3517642	3527461	3612547	3672451	4321657	4371562
$3421567\,G$	4512637	4572361	4617532	4627351	5317462	5327641	5412367	5472631
	5621347	5671432	6312457	6372541	6417352	6427531	6521437	6571342
	3425176	3476125	3516724	3524716	3615274	3674215	4326175	4375126
$3425176\,G$	4516273	4573216	4615723	4623715	5314726	5326714	5413276	5476213
	5623174	5674123	6314275	6375214	6413725	6425713	6524173	6573124
	3426715	3475216	3514276	3526174	3614725	3675124	4325716	4376215
$3426715\,G$	4513726	4576123	4613275	4625173	5316274	5324176	5416723	5473126
	5624713	5673214	6315724	6374125	6415273	6423175	6523714	6574213
	3427651	3472561	3512467	3521647	3617452	3671542	4327561	4372651
$3427651\,G$	4517362	4571632	4612357	4621537	5312647	5321467	5417632	5471362
	5627431	5672341	6317542	6371452	6412537	6421357	6527341	6572431
	3456172	3465127	3546712	3564721	3645217	3654271	4356127	4365172
$3456172\,G$	4536217	4563271	4635712	4653721	5346721	5364712	5436271	5463217
	5634172	5643127	6345271	6354217	6435721	6453712	6534127	6543172
	3457126	3462175	3541726	3562714	3641275	3657214	4352176	4367125
$3457126\,G$	4531276	4567213	4631725	4652713	5342716	5361724	5437216	5461273
	5637124	5642173	6347215	6351274	6432715	6451723	6532174	6547123
	3457621	3462571	3541627	3562417	3641572	3657412	4352671	4367521
$3457621\,G$	4531672	4567312	4631527	4652317	5342617	5361427	5437612	5461372
	5637421	5642371	6347512	6351472	6432517	6451327	6532471	6547321