ELSEVIER
Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science 321 (2004) 125–148
Theoretical Computer Science
www.elsevier.com/locate/tcs

Packing arrays

Brett Stevens[a,*,1], Eric Mendelsohn[b,2]

[a] School of Mathematics and Statistics, Carleton University, 1125 Colonel By Dr., Ottawa, Ont, Canada K1S 5B6

[b] Department of Mathematics, University of Toronto, Toronto, Ont., Canada M5S 3G3

Received 31 July 2002; received in revised form 4 February 2003; accepted 6 July 2003

Abstract

A packing array is a $b \times k$ array of values from a g-ary alphabet such that given any two columns, i and j, and for all ordered pairs of elements from the g-ary alphabet, (g_1, g_2), there is at most one row, r, such that $a_{r,i} = g_1$ and $a_{r,j} = g_2$. A central question is to determine, for given g and k, the maximum possible b. We develop general direct and recursive constructions and upper bounds on the sizes of packing arrays. We introduce the consideration of a set of disjoint rows in a packing array which allows these constructions and additionally gives a new upper bound on the size of all packing arrays. We also show the equivalence of the problem to a matching problem on graphs and a class of resolvable pairwise balanced designs. We provide tables of the best known upper and lower bounds.

Keywords: Orthogonal arrays; Transversal designs; Packing arrays; MDS codes; Matchings; Partial latin squares; Partial match queries

1. Introduction

A $g^2 \times k$ array filled with elements from a g-ary alphabet such that each ordered pair from the alphabet occurs exactly once in each pair of columns is called an *orthogonal array*. By having the first column index the rows of a $g \times g$ square, the second column index its columns, and the remaining $k-2$ columns give the entries of $k-2$ different squares, orthogonal arrays are transformed into a set of $k-2$ *mutually orthogonal*

* Corresponding author.
E-mail addresses: brett@math.carleton.ca (B. Stevens), mendelso@math.toronto.edu (E. Mendelsohn).
URLs: http://mathstat.carleton.ca/~brett/, http://www.math.toronto.edu/mendelso/
[1] Research supported by NSF Graduate Fellowship GER9452870, SFU, PIMS, MITACS, and IBM Watson Research.
[2] Research supported by NSERC of Canada Grant No. OGP007681.

0304-3975/$ - see front matter
doi:10.1016/j.tcs.2003.06.004

latin squares (MOLS). Both are known never to exist for $k > g+1$ [3]. It is natural to ask for structures that have similarly useful properties as orthogonal arrays for larger k. One generalization is to require that all pairwise interactions be covered at least once. These objects are known as covering arrays or transversal covers and have been extensively studied, see [8–11,13] and their references. The other natural generalization of orthogonal arrays is the packing array. We define it here with the addition of a substructure which will turn out to be very useful.

Definition 1. A *packing array*, $\mathrm{PA}(k, g:n)$, is a $b \times k$ array with values from a g-ary alphabet such that given any two columns, i and j, and for all ordered pairs of elements from the g-ary alphabet, (g_1, g_2), there is at most one row, r, such that $a_{r,i} = g_1$ and $a_{r,j} = g_2$. Further, there is a set of at least n rows that pairwise differ in each column: they are called disjoint. The largest number of rows possible in a $\mathrm{PA}(k, g:n)$ is denoted by $\mathrm{pa}(k, g:n)$.

One might question the usefulness of requiring a set of disjoint rows, but we carry this parameter n throughout the paper because this substructure is necessary to permit the constructions discussed, most notably Theorems 12 and 13. If this set does not occur inside the ingredient arrays of these recursive constructions then the arrays formed will have many repeated pairs which is forbidden in a packing array. The same constructions motivate the inclusion of the study of sets of disjoint rows in covering arrays [10,11]. In that case these sets allowed the optimization of the construction, here they are absolutely necessary.

It is important to note that if disjoint rows are not required, which may be the case for most applications, one can simply focus on $n = 1$, and not disregard this article as uninteresting. On the other hand, in the quest for optimal packing arrays with $n = 1$ we utilize packing arrays with larger n. For example, Wilson's construction requires their existence and can be used to produce the $\mathrm{PA}(6, 6:1)$ of size 26 and the $\mathrm{PA}(7, 6:1)$ of size 16. These are packing arrays with $n = 1$ (suitable for applications). They are the best currently known with these parameters, are not constructible without consideration of $n > 1$. Additionally it should be emphasized that consideration of disjoint rows directly gives a new upper bound on the size of packing arrays which, in particular, directly gives results on the utility of one of the applications of packing arrays. This bound is given in Theorem 6 and its corollaries in Section 2.2. Finally, in [12], considering sets of disjoint rows leads to very exciting connections between packing arrays and the existence of finite projective planes.

Row and column permutations, as well as permuting symbols within each column, leave the packing conditions intact.

Example 2. A $\mathrm{PA}(5, 3:1)$ with six rows:

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 2 & 1 \\ 1 & 0 & 1 & 2 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 2 & 1 & 0 & 1 \\ 1 & 2 & 2 & 1 & 0 \end{matrix}$$

We often and without loss of generality use $\mathbb{Z}_g$ as the symbol set on each column and let $r_{i,j}$ equal the number of times symbol $i \in \mathbb{Z}_g$ appears in column j.

Packing arrays are also called *transversal packings* [9] and *mutually orthogonal partial latin squares* [1,2]. Observing that any two rows of the packing array must have Hamming distance at least $k-1$ we see that packing arrays are also error correcting codes, specifically, maximal distance separating (MDS) codes or partial MDS codes.

Abdel-Ghaffar and Abbadi [2] use MDS codes to allocate large database files to multiple hard disk systems so that the retrieval time is optimal. For storage on multiple disks, each attribute space D_i, $1 \leqslant i \leqslant k$ of a data base file with n records and k attributes is broken up into g parts, D_{ij}, $1 \leqslant i \leqslant k$ and $1 \leqslant j \leqslant g$. The database file can be seen as a subset of $D_1 \times D_2 \times \cdots \times D_k$. The set of all records with entries in $D_{1j_1} \times D_{2j_2} \times \cdots \times D_{kj_k}$ for $1 \leqslant j_i \leqslant g$ and $1 \leqslant i \leqslant k$ is called a *bucket*, which can be made equivalent to an element of $\mathbb{Z}_g^k$. Abdel-Ghaffar and Abbadi begin by first dividing up the database into all its buckets and partitioning the set of all buckets into m pieces, each of which will be stored on a single disk in an array of m disks. A *partial match query* is a request for all records in the database file that match a choice of from zero to k attribute values. A search will begin by retrieving from disk all the buckets that could contain records matching the request. After retrieval, this set of buckets will be searched for the matching records therein. Abdel-Ghaffar and Abbadi concern themselves with minimizing the time required to retrieve the buckets from the disk array. If a request matches b_i buckets from disk i, $1 \leqslant i \leqslant m$ then they define the retrieval time to be $\max\{b_1, b_2, \ldots, b_m\}$ and ask how they can optimize this time. A response time is *strictly optimal* if

$$\max\{b_1, b_2, \ldots, b_m\} = \left\lceil \frac{\sum_{i=1}^{m} b_i}{m} \right\rceil .$$

The conditions necessary for strict optimality are very tight and cannot be achieved in a large number of reasonable situations. Abdel-Ghaffar and Abbadi discuss best possible response times when strict optimality is not possible, and show that if the set of buckets on each disk is an MDS error correcting code then the response time, although not strictly optimal, can be shown to be better than all other allocations schemes across m disks.

They establish bounds on the size of packing arrays, the smallest form of MDS codes, to yield extreme bounds on the efficiency of their system. The MDS or partial MDS codes that correspond to packing arrays would apply to optimal disk allocation over a large number of disks, specifically at least g^{k-2}. If packing arrays were used for optimal disk allocation in their model, any search with at least two specified attributes would yield search time of one. Their method guarantees that any partial match query with at least two specified attributes matches at most one bucket on each disk. This is the best possible response time to retrieve the buckets from the disk array.

If we define

$$f(b,g) = \max\{k \text{ such that there exists a PA}(k,g) \text{ with } b \text{ rows}\}.$$

Then a simple adaptation of proof of Theorem 3.4 from [2] gives new results on the efficacy of this application:

Theorem 3. *There is no partial MDS code of length k and size b over an alphabet of g letters if*

$$2 \leqslant \lceil \log_g b \rceil \leqslant k - f(b,g) + 1.$$

In particular the Corollaries from Section 2.2 give that no partial MDS code exists when either

$$2 \leqslant \lceil \log_g b \rceil \leqslant k - \frac{g(g-1)}{2} + \frac{r_{\max}(r_{\max}-1)}{2}$$

or

$$2 \leqslant \lceil \log_g b \rceil \leqslant k - \frac{g(g-1)}{2} + \frac{\lceil b/g \rceil(\lceil b/g \rceil - 1)}{2},$$

where $r_{\max}$ is the most frequently occurring letter in any single coordinate position. In a similar manner, the bound obtained from Theorem 4 can give results about the range of possible application.

In addition to refining the application range, it is also necessary to construct partial MDS codes if one is ever to implement the application. This is another task of this article. Decomposing $\mathbb{Z}_g^k$ into partial of full MDS codes is also required for any implementation. Abdel-Ghaffar and Abbadi [2] discuss this aspect but do not fully solve it. In this paper we address the existence of partial MDS codes and leave the decomposition problem associated with them to further research. We believe that constructing examples is a worthwhile endeavor and is not mitigated by not completing the decomposition problem which is a separate research area in its own right.

In our terminology, Abdel-Ghaffar and Abbadi [2] show that

$$\mathrm{pa}(k,g:1) \geqslant g+1 \quad \Rightarrow \quad k \leqslant \frac{g^2+g}{2},$$

which together with $\mathrm{pa}(k,g:1) \geqslant g$ implies

$$\mathrm{pa}((g^2+g+2)/2, g:1) = g \quad \text{and} \quad \mathrm{pa}((g^2+g)/2, g:1) \geqslant g.$$

Although not in these precise terms, Abdel-Ghaffar modifies the Plotkin bound for packing arrays with $n=1$ and shows that this bound is met when $k \geqslant 2g-1$ [1]. We extend this result to include all n. He also completely solves the case where $n=1$ and $g=3,4$ and shows that a packing array with g^2-1 rows can be completed to one with g^2 rows.

In Section 2, two sets of upper bounds on packing arrays are derived. The first set of bounds consists of our modification of the Plotkin bound to account for sets of disjoint rows and the implications of non-integrality. The second set comes from an observation on the constraints that sets of disjoint rows place on these structures. In Section 3, we discuss constructions which yield lower bounds. We review some constructive techniques motivated by design theory that are also useful to construct transversal covers [11]. We also present a direct construction using matchings in graphs. Finally, we present a set of recursive constructions based on this matching problem. Both these constructions extend Abdel-Ghaffar's result to packing arrays with sets of disjoint rows.

2. Packing arrays: upper bounds

2.1. Modification of the Plotkin bound

The rows of a $\mathrm{PA}(k, g : n)$ form a code on a g-ary alphabet with word length k and Hamming distance at least $k - 1$. The set of n disjoint rows form a set of n codewords with mutual Hamming distance k. If the $\mathrm{PA}(k, g : n)$ has $\mathrm{pa}(k, g : n)$ rows then the number of codewords in the code is maximum among the codes containing such a specified set of n codewords.

One of the most potent coding theory bounds is the Plotkin bound. Abdel-Ghaffar derives a generalization of the Plotkin bound for packing arrays. We adapt this bound further to include consideration of sets of disjoint rows and to extend its utility to parameter sets where the number of rows is not a multiple of g. We establish upper bounds not only to discuss the range of application of packing arrays but also to determine when we are using optimal objects in our constructions.

Theorem 4 (Plotkin bound modified for packing arrays). *A $b \times k$ packing array must satisfy the following bound for all $\beta \leqslant b$, $\beta = ug + v$ where $0 \leqslant v < g$:*

$$k((g - v)u^2 + v(u + 1)^2) \leqslant \beta^2 - \beta - n^2 + n + k\beta.$$

Proof. Since deletion of rows does not affect the packing conditions, if a $b \times k$ packing array exists then a $\beta \times k$ packing array must exist for all $\beta \leqslant b$. Any bounds on the parameters of packing arrays must therefore hold for all $\beta \leqslant b$ as well.

Let N be the number of ordered pairs of different symbols in the columns of a $\mathrm{PA}(k, g : n)$. Since pairs of rows from the set of n disjoint rows never intersect (i.e. never have the same symbol in the same position) and any other pair of rows can share a common symbol in the same position in at most one place, we get

$$N \geqslant \beta(\beta - 1)(k - 1) + n(n - 1).$$

On the other hand, we can determine this value, N, in another way. For each symbol, i, there are $b - r_{i,j}$ rows where a different symbol occurs in column j. So, with $\sum_{i=1}^{g} r_{ij} = \beta$, we have

$$\sum_{i=1}^{g} r_{i,j}(\beta - r_{i,j}) = \beta^2 - \sum_{i=1}^{g} r_{i,j}^2 \leqslant \beta^2 - (g - v)u^2 - v(u + 1)^2 \tag{1}$$

with equality if exactly v symbols have replication $u + 1$ and $g - v$ symbols have replication u. These two bounds on N give

$$k(\beta^2 - (g - v)u^2 - v(u + 1)^2) \geqslant N \geqslant \beta(\beta - 1)(k - 1) + n(n - 1),$$

which reduces to the desired bound, with equality if the replication numbers are as stated above and every pair of rows meets in exactly one position except those in the set of disjoint rows. □

In the event that we can prove that there must be more than two replication numbers this bound can be tightened in Eq. (1). On the other hand, when equality is reached in Theorem 4, strong implications on the structure of the code arise: the bound must be an integer (although the standard Plotkin bound might not be integral in many cases); any two code words, except those in the set of disjoint rows must intersect; and each symbol must appear nearly equally often in each column (either u or $u+1$ times). Let us examine these consequences for packing arrays.

We dualize the packing array into a block design.

Definition 5. A *pairwise balanced block design* (PBD(v, K, λ)) is a base set V of v elements, and a collection $\mathscr{B}$ of subsets $B \subseteq V$, and $|B| \in K$, called *blocks*, such that every pair of points from V occur in exactly λ blocks. Furthermore if the collection of blocks can be partitioned so that each part contains every element of V exactly once, we say that the PBD is *resolvable*. A generalization of PBDs admits holes. A subset, $H \subseteq V$ is called a *hole* if every pair of elements of H occurs in no blocks.

The base set of the design of interest is the rows of the array. The blocks of this design are any maximal collection of rows that have the same symbol in some column. Each column, therefore, defines a spanning subset of blocks, or resolution class. This dual structure is a resolvable PBD($v, \{u, u+1\}, 1$) with a hole of order n. Each resolution class must have each block size occurring the same number of times as every other resolution class. If b is a multiple of g then there is only one block size in the resolvable PBD. In fact, for $n=1$, these structures are a particular class of design: a *class-uniformly resolvable design* (CURD). This recently studied class of designs provides insight into resolvable structures and have important applications [4,7,14].

If $b \leqslant 2g$, consideration of Theorem 4 implies that the dual of this structure is a packing of pairwise edge disjoint $(b-g)$-matchings into $K_b - K_n$. We discuss this specific case in Section 3.2.

2.2. Disjoint row bound

If we have a symbol appearing $m > n$ times in one column of a PA($k, g:n$) then deleting this column yields a PA($k-1, g:m$). Therefore, bounds on the sizes of packing arrays can be translated into bounds on the admissible replication numbers for packing arrays with one more column. To this end, we calculate the maximum number of columns possible in a packing array with n disjoint rows and at least $g+1$ rows.

Theorem 6. *The maximum number of columns in a packing array with at least $g+1$ rows is*

$$\frac{g(g+1)}{2} - \frac{n(n-1)}{2}.$$

Proof. Since the number of columns is non-increasing as the number of rows increases we need only establish the result for exactly $g+1$ rows. Given an arbitrary packing array with $g+1$ rows, we manipulate it row by row. By permuting rows and symbols within columns, we can assume that the first n rows of the packing array are the row of all 0's, the row of all 1's and so on up to the row of all $(n-1)$'s. Since, within each column we can permute the symbols, when we consider each next row, i, (except the last) the symbols in this row can be all the symbols that we have used in previous rows and at most one new symbol. We can also assume that underneath a previously occurring symbol in row i that the symbol, i, new to row $i+1$ does not occur. If it did we could permute the symbols in this column to swap symbol i and $i-1$.

Define $l_{i,n}$ to be the largest number of previously used symbols that can appear in row i in a packing array with a set of n disjoint rows. Clearly $l_{n+1,n}=n$; after the first n disjoint rows, each symbol in them appears at most once in every following row. These numbers satisfy the recursion $l_{i+1,n}=l_{i,n}+i$. We can assume, by performing permutations of the columns, that each symbol new to a row appears in a contiguous rightmost set of indices or columns. To the left of these indices we can assume are a set of entries that only contain previously used symbols appearing directly below symbols that were new to row $i-1$. Finally we can assume that the remaining leftmost set of indices are previously used entries in both row i and $i-1$. Then, by assumption, at most the first $l_{i,n}$ symbols of row i will be previously appearing symbols and the remaining $k-l_{i,n}$ will be the symbol $i-1$, new to row i. Thus the right-hand side of the packing array under manipulation will look like a large set of disjoint rows; rows will only share the same symbols in a column on the left hand side. When considering the row $i+1$, the new symbol is i; at most i of the old symbols in this, row $i+1$, are underneath the right hand part of row i where symbol $i-1$ appears. This is the middle set of indices described above. This portion, along with the first $l_{i,n}$ positions in row $i+1$, is the largest possible set of positions in which old symbols can appear.

Iterating, we see that $l_{i,n}=(i(i-1)-n(n-1))/2$ and $l_{g,n}=(g(g-1)-n(n-1))/2$. If a $g+1$st row exists, it can only contain previously used symbols, and so we must have a set of at most g right-most columns, restricted to which, the array so far manipulated, consists of disjoint rows. So $k-l_{g,n}\leqslant g$ or

$$k \leqslant \frac{g(g+1)}{2} - \frac{n(n-1)}{2}.$$

To see that there is an array achieving this bound we use the following algorithm for arranging the old symbols to each new row. In row i, the first $l_{i-1,n}$ columns will have the old symbol, $i-2$ (new in the previous row, $i-1$). The next $i-1$ columns will have each of the old symbols, $0,1,\ldots,i-2$, once. The remaining $(g(g+1)-n(n-1))/2-l_{i,n}$ columns of this row will have the new symbol, $i-1$. The last, $g+1$st row, follows the same pattern but will have no columns remaining for new symbols (which is good seeing as there are not any). Each row intersects every other in at most one point and thus satisfies the packing conditions. It is straightforward to check that the first g rows of a matrix using $l_{i,n}$ previously used symbols in row i can only be those constructed above, up to permutation of rows, columns and symbol sets within each column. □

Corollary 7. *If, in a* $\mathrm{PA}(k,g:n)$ *with more than* g *rows, we define* $r_{\max} = \max\{r_{ij} : i \in \mathbb{Z}_g,\ 1 \leqslant j \leqslant k\}$, *then*

$$k-1 \leqslant \frac{g(g+1)}{2} - \frac{r_{\max}(r_{\max}-1)}{2}.$$

Proof. Remove any column which has a point achieving $r_{\max}$. This yields a $\mathrm{PA}(k-1, g:\max(n, r_{\max}))$. □

Observing that if a $\mathrm{PA}(k,g:n)$ has more than mg rows, it must have a point with replication number at least $m+1$, we have the following corollary.

Corollary 8. *The maximum number of columns in a packing array with more than* mg *rows is*

$$\frac{g(g+1)}{2} - \frac{m(m-1)}{2} + 1,$$

or alternatively, noting that $\mathrm{pa}(k,g:n)$ *is non-increasing in* k, *we have*

$$\mathrm{pa}\left(\frac{g(g+1)}{2} - \frac{m(m-1)}{2} + 2, g:n\right) \leqslant mg.$$

One notable consequence of this is the fact that

$$\mathrm{pa}(g+2, g:n) \leqslant g^2 - g.$$

Finally, as discussed in the introduction, these corollaries directly give new results on the range of useful application for disk allocation.

3. Packing arrays: constructions and lower bounds

3.1. Constructions from design theory

Definition 9. An *incomplete orthogonal array* is the same as an orthogonal array except there are s mutually disjoint subsets of the alphabet, H_i, called *holes* with cardinalities b_i, such that every ordered pair, both from H_i, $1 \leqslant i \leqslant s$, never occurs in any pair of columns.

They are also called incomplete transversal designs ($\mathrm{ITD}(k,g:b_1,b_2,\ldots,b_s)$) and are used to construct transversal covers [11]. This construction can be similarly formulated for packing arrays to yield.

Theorem 10. *If there exists an* $\mathrm{ITD}(k,g;b_1,b_2,\ldots,b_s)$ *then*

$$\mathrm{pa}(k,g:i) \geqslant \max_{\substack{i_i+i_2+\cdots+i_s=i \\ i_j \leqslant b_j}} \left(g^2 - \sum_{j=1}^{s} (b_j^2 - \mathrm{pa}(k,b_j:i_j))\right).$$

Proof. Fill the holes of the ITD with the PA$(k, b_j : i_j)$. These holes are disjoint, thus the union of the sets of disjoint rows from the PA$(k, b_j : i_j)$ will also be a set of disjoint rows. $\square$

Example 11. The existence of ITD$(4,6;2)$ and ITD$(6,10;2)$, yield pa$(4,6) = 34$, as also stated by Abdel-Ghaffar [1] and pa$(6,10) = 98$ or 100. PA$(4,6)$ is explicitly

$$\left(\begin{array}{cccccccccccccccccccccccccccccccccc} 0&0&0&0&0&0&1&1&1&1&1&1&2&2&2&2&2&2&3&3&3&3&3&3&4&4&4&4&5&5&5&5&4&5\\ 0&1&2&3&4&5&0&1&2&3&4&5&0&1&2&3&4&5&0&1&2&3&4&5&0&1&2&3&0&1&2&3&5&4\\ 4&5&2&3&0&1&1&0&5&4&2&3&5&4&0&1&3&2&3&2&4&5&1&0&0&3&1&2&2&1&3&0&4&5\\ 0&1&4&5&2&3&5&4&0&1&3&2&3&2&5&4&0&1&4&5&3&2&1&0&1&3&2&0&2&0&1&3&4&5 \end{array}\right)^{\mathrm{T}},$$

where the last two rows are the filled hole. We observe that this is actually a PA$(4,6:5)$: rows 1, 8, 18, 27 and 34 are a set of disjoint rows.

3.1.1. Generalizing Wilson's construction to packings

Theorem 12. *Let C be a* PA$(k+l,t)$ *with columns $G_1, G_2, \dots, G_k$ and $H_1, H_2, \dots, H_l$. Let $\mathscr{S}$ be any set of choices of a subset of each symbol set on $H_1, \dots, H_l$, let h_i be the number of symbols chosen from column H_i, $\sum_{i=1}^{l} h_i = u$, and m be any nonnegative integer. For any row A of C let $u_A = |\mathscr{S} \cap A|$, the number of times the symbol in the intersection of column H_i with row A is in the set of symbols chosen for H_i summed from H_1 to H_l. Then*

$$\mathrm{pa}(k, mt+u) \geqslant \sum_{A} (\mathrm{pa}(k, m+u_A : u_A) - u_A) + \sum_{i=1}^{l} \mathrm{pa}(k, h_i).$$

Proof. The proof is a straightforward generalization of Wilson's construction for transversal designs [15]. $\square$

If $l = 0$, we get the obvious generalization of MacNeish's theorem.

$$\mathrm{pa}(k, g:n) \geqslant \max_{\substack{2 \leqslant i \leqslant \lfloor g/2 \rfloor \\ \max(1, n/\lfloor g/i \rfloor) \leqslant j \leqslant \min(n,i)}} ((\mathrm{pa}(k, i:j)\,\mathrm{pa}(k, \lfloor g/i \rfloor : \lceil n/j \rceil)). \tag{2}$$

When $l = 1$, using some probabilistic analysis as in [11], we can achieve:

$$\begin{aligned} \mathrm{pa}(k, mt+u:n) \geqslant \max_{(i,j,\ell) \in A_{n,u,t,m}} & (\mathrm{pa}(k, u:\ell) + (t+u)\,\mathrm{pa}(k+1, t:i)\,\mathrm{pa}(k, m:j)/t \\ & + u\,\mathrm{pa}(k+1, t:i)\,(\mathrm{pa}(k, m+1 : j+1) - 1)/t), \end{aligned} \tag{3}$$

where

$$A_{n,u,t,m} = \{(i,j,\ell) \in \mathbb{N}^3 : ij + \ell \geqslant n,\ 1 \leqslant \ell \leqslant u,\ 1 \leqslant i \leqslant t,\ 1 \leqslant j \leqslant m\}.$$

For $l > 1$ the potential of this construction is large but there are difficulties. For transversal designs, the tight balance conditions often permit counting the number of blocks that intersect S. We do not have such strong structural information about packing arrays and this makes the case $l > 1$ difficult to state as a recursion on the packing numbers alone. It requires analyzing the particular master array. One case where this has be done is $m=0$.

Theorem 13. *Given a* $(v,\{2,3,\ldots,g-1\},1)$*-design, and for each point* x, *a chosen block to represent* x, B_x, *with* $x \in B_x$, *we can construct a* $\mathrm{PA}(k,g)$. *For each block*, B, *of the design, we define* u_B *to be the number of points on this block not represented by it. Then*

$$\mathrm{pa}(k,g) \geqslant \sum_B \mathrm{pa}(k,|B|:u_B) - u_B.$$

For a full explanations of the constructions of this section see [11].

3.2. Direct construction for $b \leqslant 2g$: matching packings

The equivalence, when $b \leqslant 2g$, between optimal packing arrays and packings of matchings into $K_b - K_n$ also yields a number of constructive results. We prove a number of results from a purely graph theoretical point of view before applying the results to packing arrays.

3.2.1. Graph theoretical results

We will need a result on graph edge colourings before we proceed. the proof can be found in [5].

Lemma 14. $K_v - K_n$ is $(v-1)$ *edge colourable if and only if* v *is even or* $n^2-n \geqslant v-1$.

The next two theorems are similar to the interesting work of Rodger and Hoffman on the number of edge-disjoint perfect matching in graphs [6]. We are concerned with the number of edge-disjoint *partial* matchings.

Theorem 15. *The maximum number*, k, *of edge disjoint copies of an* m*-matching that can be packed into a* $K_v - K_n$ *is subject to the following conditions*:

$$2km \leqslant v(v-1) - n(n-1),$$
$$k(2m - v + n) \leqslant n(v-n)$$

and a packing achieving the largest integer k *subject to these conditions always exists.*

Proof. First note that since the m-matching must fit in the graph, then $v \geqslant 2m$ and since every one of the edges in a matching must have at least one endpoint not in the hole, then $v-n \geqslant m$. Since each of the k m-matchings uses m edges and there are only $(v(v-1)-n(n-1))/2$ edges and we obtain

$$2km \leqslant v(v-1) - n(n-1).$$

Let k_1 equal the greatest integer that satisfies this bound.

There are $n(v-n)$ edges between the n vertices in the hole and the remaining $v-n$ vertices. These are divided into k classes so some matching uses at most $\lfloor (n(v-n))/k \rfloor$ of these edges. Each of the k m-matching spans $2m$ vertices and misses $v-2m$ vertices so if

$$v - 2m < n - \left\lfloor \frac{n(v-n)}{k} \right\rfloor .$$

We cannot complete the edges from $K_{n,v-n}$ into a m-matching and so if $2m-v+n>0$, we derive the second necessary condition:

$$k(2m - v + n) \leqslant n(v - n).$$

Let k_2 equal the greatest integer that satisfies this bound. Both k_1 and k_2 are upper bounds on the number of m-matchings that can be packed into $K_v - K_n$.

When

$$\frac{v(v-1) - n(n-1)}{2m} \geqslant v - 1$$

then k_1 is the lesser upper bound and as long as either $k_1 > v-1$ or the conditions of Lemma 14 are met then an equitable k_1 colouring of $K_v - K_n$ with any

$$e = \frac{v(v-1) - n(n-1)}{2} - mk_1$$

edges removed give the desired packing and allow the leave graph to be any subgraph of $K_v - K_n$ with e edges. The only case not covered is when v is odd, $n^2 - n < v - 1$ and $k_1 = v - 1$. These conditions imply that $v = 2m + 1$. It is also clear that when $n_1 < n_2$, then $K_v - K_{n_2} \subseteq K_v - K_{n_1}$ and all we need consider is the largest n satisfying $n^2 - n < v - 1$.

Equitably $(v-1)$ edge colour K_{v-n}. Label the edges in each colour class from the set $\{n, \dots, m\}$ such that no two edges in the same colour class get the same label. Fill in a $v \times (v-1)$ matrix in the following manner. Label the rows consecutively $0, 1, \dots, v-1$. All cells in the first n rows get the same symbol as their row. To fill in the remaining rows assign one colour class from the $(v-1)$ edge colouring of K_{v-n} to each column of the matrix and assign one vertex of K_{v-n} to each of the remaining rows of the matrix. If edge $\{i, j\}$ is in colour class k with label l then put symbol l in cells (i, k) and (j, k).

After this is done, each row (not from the first n) will have n unfilled cells. There will be $(v - 1 + n^2 - n)/2$ columns with $(n-1)$ unfilled cells and $(v - 1 - n^2 + n)/2$ columns with $(n+1)$ unfilled cells. Each of these last columns will be missing exactly one symbol from $\{n, \dots, m\}$. In each such column arbitrarily place this symbol in exactly one of the unfilled cells. Now each row and column has at most n unfilled cells. Form the natural bipartite graph representing these holes: The two parts of this graph will be the set of the bottom $(v-n)$ rows and the set of columns. An edge will be placed from vertex i to vertex j if the cell (i, j) is empty. This graph has $\Delta = n$ and is therefore n edge colourable with colours $\{0, \dots, n-1\}$ using this colouring fill in the remaining unfilled cells. Every symbol from $\{0, \dots, m\}$ appears at least once in each column, no symbol more than twice and exactly one symbol once in each column. The pairs of rows that have identical symbols in the ith column are the sets of edges in the ith m-matching packed into $K_v - K_n$.

The remaining case is when

$$\frac{v(v-1)-n(n-1)}{2m} < v-1$$

and $k_2 \leqslant k_1$. Remove all but $k_2(v-n-m)$ edges from K_{v-n} This is possible because $(v-n)(v-n-1)/2 \geqslant (n(v-n)(v-n-m))/(2m+n-v)$. Since the maximum degree of K_{v-n} was $v-n-1$ before the edges were removed, this is still the largest possible maximum degree. Now k_2 edge colour the remaining graph. This is possible since

$$\Delta + 1 \leqslant v-n \leqslant \left\lfloor \frac{n(v-n)}{(2m-v+n)} \right\rfloor = k_2.$$

As above fill in the cells of a $v \times k_2$ array. There are $(2m-v+n)$ empty cells in each column and at most

$$\frac{k_2(2m-v+n)}{(v-n)}$$

in each of the last $(v-n)$ rows (there are no empty cells in the first n rows). Now in the bipartite graph defined on the set

$$\{\text{last } v-n \text{ rows}\} \cup \{k_2 \text{ columns}\}.$$

Edge $\{i,j\}$, $i \in \{\text{last } v-n \text{ rows}\}$, $j \in \{k_2 \text{ columns}\}$ is in the graph if and only if cell (i,j) is empty. The maximum degree of this bipartite graph is

$$\Delta = \max\left(2m-v+n, \left\lfloor \frac{k(2m-v+n)}{(v-n)} \right\rfloor\right) \leqslant n.$$

So this graph is n edge colourable. Use this colouring to fill in the empty cells of the matrix and define the partial matchings as above. It is easy to check that each matching has at least m edges, so we can remove any excess and obtain the desired packing of m-matchings in $K_v - K_n$. $\square$

Theorem 16. *The maximum number*, k, *of edge disjoint copies of an m-matching that can be packed into a $K_{v_1,v_2} - K_{n_1,n_2}$ (assume without loss of generality that $v_1 \geqslant v_2$) is subject to the following constraints*:

$$k(m-v_1+n_1) \leqslant n_1(v_2-n_2),$$
$$k(m-v_2+n_2) \leqslant n_2(v_1-n_1),$$
$$km \leqslant v_1v_2 - n_1n_2$$

and the largest integer k satisfying all these is always achievable.

Proof. First we establish the necessary conditions. Since the m-matching must fit in the graph, we have $v_2 \geqslant m$ and since every one of the edges in a matching must have at least one endpoint not in the hole, then $v_1+v_2-n_1-n_2 \geqslant m$. Since each m-matching uses m edges and there are only $v_1v_2-n_1n_2$ edges and we obtain

$$k \leqslant \left\lfloor \frac{v_1v_2-n_1n_2}{m} \right\rfloor := k_1.$$

The union of the k m-matchings form a subgraph of $K_{v_1,v_2}-K_{n_1,n_2}$ that is, by construction, k colourable. Therefore this subgraph must have degree at most k. Therefore it must be possible to remove $v_1v_2-n_1n_2-mk$ edges so as to reduce the degree of all vertices to at most k. There are v_1-n_1 vertices of degree v_2 and so we get

$$v_1v_2-n_1n_2-mk \geqslant (v_1-n_1)(v_2-k).$$

If $m-v_1+n_1>0$ this reduces to

$$k \leqslant \left\lfloor \frac{n_1(v_2-n_2)}{m-v_1+n_1} \right\rfloor := k_2.$$

There are v_2-n_2 vertices of degree v_1 and so we get

$$v_1v_2-n_1n_2-mk \geqslant (v_2-n_2)(v_1-k).$$

If $m-v_2+n_2>0$ this reduces to

$$k \leqslant \left\lfloor \frac{n_2(v_1-n_1)}{m-v_2+n_2} \right\rfloor := k_3.$$

It is also true that there are n_1 vertices of degree v_2-n_2 and n_2 vertices of degree v_1-n_1 but the bounds on k derived from these two facts are never stronger than k_1.

To establish the sufficiency of these conditions we shall examine three cases:

Case 1: *k_1 is the least.*

In particular, if $m-v_2+n_2>0$, so k_3 is defined, then $k_1 \leqslant k_3$:

$$\begin{aligned}
&\frac{v_1v_2-n_1n_2}{m} \leqslant \frac{n_2(v_1-n_1)}{m-v_2+n_2},\\
&mv_1(v_2-n_2) \leqslant v_1v_2(v_2-n_2)-n_1n_2(v_2-n_2),\\
&v_1 \leqslant \frac{v_1v_2-n_1n_2}{m},\\
&v_1 \leqslant k_1.
\end{aligned}$$

On the other hand if $m-v_2+n_2 \leqslant 0$ then $m \leqslant v_2-n_2$ and we get

$$\begin{aligned}
&n_1 \leqslant v_1,\\
&v_1v_2-v_1n_2 \leqslant v_1v_2-n_1n_2,\\
&v_1 \leqslant \frac{v_1v_2-n_1n_2}{m},\\
&v_1 \leqslant k_1.
\end{aligned}$$

Since v_1 is the maximum degree of the bipartite graph $K_{v_1,v_2}-K_{n_1,n_2}$, we can remove all but k_1m edges from the graph and equitably k_1 edge colour it and the colour classes will be the desired number of m-matchings.

Case 2: *k_2 is the least.*

Since k_2 is defined then we know that $m-v_1+n_1>0$. If $m-v_2+n_2 \leqslant 0$, equivalently, k_3 is not defined, then

$$\begin{aligned}
&n_1 \leqslant v_1,\\
&v_1v_2 - v_1n_2 \leqslant v_1v_2 - n_1n_2,\\
&v_1m \leqslant v_1v_2 - n_1n_2,\\
&v_2m \leqslant v_1v_2 - n_1n_2,\\
&v_2m(v_1-n_1) \leqslant (v_1v_2-n_1n_2)(v_1-n_1),\\
&mv_1v_2 - mn_1n_2 - (v_1v_2-n_1n_2)(v_1-n_1) \leqslant mv_2n_1 - mn_1n_2,\\
&(m-v_1+n_1)(v_1v_2-n_1n_2) \leqslant mn_1(v_2-n_2),\\
&k_1 \leqslant k_2
\end{aligned}$$

and so k_1 is also the least and the results of the first case apply.

So we can assume that both $m-v_1+n_1>0$ and $m-v_2+n_2>0$. In what follows we will remove x edges from the K_{n_1,v_2-n_2} portion of the graph, y edges from the $K_{v_1-n_1,v_2-n_2}$ portion of the graph and z edges from the $K_{v_1-n_1,n_2}$ portion of the graph. We must assign values to x, y and z so that all the degrees of the remaining graph are no more than k_2 and there are k_2m edges remaining. At this point simply k_2 edge colouring will produce the desired set of m-matchings. The conditions that must be satisfied are:

$$\begin{aligned}
&x \geqslant (v_2-n_2-k_2)(n_1),\\
&x+y \geqslant (v_1-k_2)(v_2-n_2),\\
&y+z \geqslant (v_2-k_2)(v_1-n_1),\\
&z \geqslant (v_1-n_1-k_2)n_2,\\
&x+y+z = v_1v_2-n_1n_2-k_2m.
\end{aligned}$$

Since $m \leqslant v$, this implies that $m-v_1+n_1 \leqslant n_1$. Together with the fact that v_2-n_2 must be integral means that $v_2-n_2 \leqslant \lfloor (n_1(v_2-n_2))/(m-v_1+n_1) \rfloor = k_2$. Thus we can set $x=0$, $y=(v_1-k_2)(v_2-n_2)$ and $z=v_1v_2-n_1n_2-k_2m-y=v_1v_2-n_1n_2-k_2m-(v_1-k_2)(v_2-n_2)$. It is easy to check that all the constraints are satisfied. The edges can easily be removed so as to reduce the degrees as evenly as possible in each component.

Case 3: *k_3 is the least.*

If k_3 is defined then we know that $m-v_2+n_2>0$. If $m-v_1+n_1 \leqslant 0$, so k_2 is not defined then $m \leqslant v_1-n_1$. Now if $k_3 \leqslant v_2$ then consider K_{m,k_3} as a subgraph of the $K_{v_1-n_1,v_2}$ that is a subgraph of $K_{v_1,v_2}-K_{n_1,n_2}$. We know that $m \leqslant v_1-n_1 \leqslant k_3$ so we can k_3 edge colour this graph to produce the needed number of m-matchings. So we may now assume that $k_3>v_2$. Like above we need to choose x, y and z to satisfy

$$\begin{aligned}
&x \geqslant (v_2-n_2-k_3)(n_1),\\
&x+y \geqslant (v_1-k_3)(v_2-n_2),
\end{aligned}$$

$$y+z \geqslant (v_2-k_3)(v_1-n_1),$$
$$z \geqslant (v_1-n_1-k_3)n_2,$$
$$x+y+z = v_1v_2-n_1n_2-k_3m.$$

But knowing that $k_3>v_2$ and $k_3\geqslant v_1-n_1$ means that the only one of these inequalities not automatically satisfied is

$$x+y \geqslant (v_1-k_3)(v_2-n_2).$$

So the evenly distributed removal of any $v_1v_2-n_1n_2-k_3m$ edges between the v_2-n_2 vertices of degree v_1 will reduce all degrees to no more k_3 and leave exactly k_3m edges. So equitably k_3 edge colouring the remaining graph produces the needed number of m-matchings.

In the only remaining case, both $m-v_1+n_1>0$ and $m-v_2+n_2>0$. Since $v_1-n_1\leqslant\lfloor(n_2(v_1-n_1))/(m-v_2+n_2)\rfloor=k_3$ we can set $z=0$. Then set $y=(v_2-k_3)(v_1-n_1)$ and $x=v_1v_2-n_1n_2-k_3m-y=v_1v_2-n_1n_2-k_3m-(v_2-k_3)(v_1-n_1)$. It is easy to check that all the constraints are satisfied. The edges can easily be removed so that they reduce the degrees as evenly as possible in each component. Equitably k_3 edge colouring once again produces the desired set of m-matchings. □

3.2.2. Packing array constructions

A direct application of Theorem 15 to packing arrays gives

Corollary 17.

$$\mathrm{pa}(k,g:n) \geqslant \min\left(2g, \frac{2kg-kn-n^2}{k-n}, \frac{2k+1-\sqrt{(1+2k)^2+4(n^2-n-2kg)}}{2}\right).$$

If $k\leqslant n\leqslant g$ then a PA$(k,g:n)$ with $2g$ rows is easily constructed using Theorem 15 and so we can ignore this bound if the denominator is zero or negative. Similarly if the discriminant in the third bound is negative then the corresponding bound from Theorem 15 is always satisfied, so it may also be ignored in that case. In particular, when $k\geqslant 2g-1$, Corollary 17 gives a packing array that meets the bound in Theorem 4 and is optimal.

Abdel-Ghaffar and Abbadi [2] ask and answer the question: when does a packing array have more than g rows? We extend this result to include sets of disjoint rows in Theorem 6. Theorems 15 and 4 also allow us to derive several exact packing arrays numbers:

Corollary 18.

$$\mathrm{pa}\left(\frac{g^2+3g+2-n^2+n}{4}, g:n\right) = g+2. \quad (4)$$

$$\mathrm{pa}\left(\frac{g^2+3g+2-n^2+n}{4}+1, g : n\right) = g+1. \tag{5}$$

When $k \geqslant 2g-1$ *and* $n>1$, *then*

$$\mathrm{pa}(k,g:n) = \left\lfloor \frac{2k+1-\sqrt{4k^2+4k+1-8kg+4n^2-4n}}{2} \right\rfloor.$$

And finally

$$\mathrm{pa}(2g-2, g:2) = 2g.$$

We also have recursive constructions.

Theorem 19. *Within a* $\mathrm{PA}(k,g:n)$ *fix* n' *rows which contain no members of the set of* n *disjoint rows. Let* $q_{i,j}$ *be the number of times that symbol* j *appears in* i*th column of a row in the union of this set of rows and the original* n *disjoint rows. Let* m_i, *and* $h_{i,j}$, $1 \leqslant i \leqslant k$, $1 \leqslant j \leqslant g$ *be a positive integers such that* $\mathrm{pa}(m_i, h_{i,j} : q_{i,j}) \geqslant r_{i,j}$. *Additionally, let* $h = \max_i \{\sum_{j=1}^{g} h_{i,j}\}$. *Then*

$$\mathrm{pa}\left(\left(\sum_{i=1}^{k} m_i\right), h : n+n'\right) \geqslant \mathrm{pa}(k,g:n).$$

Proof. This is the dual of Theorem 2.7 from [4]. □

We can use Theorem 16 to facilitate a recursive construction for packing arrays.

Theorem 20. *Suppose that there exists a* $\mathrm{PA}(k,g_1:n_1)$ *with* b_1 *rows and a* $\mathrm{PA}(k,g_2:n_2)$ *with* b_2 *rows and assume* $b_1 \geqslant b_2$. *If* $b_1 \leqslant g_1+g_2$ *then there exists a* $\mathrm{PA}(k+k', g_1+g_2:n_1+n_2)$ *with* b_1+b_2 *rows where* k' *is the largest integer satisfying*

$$\begin{aligned} k'(b_1+b_2-g_1-g_2) &\leqslant b_1b_2-n_1n_2 \\ k'(b_2-g_1-g_2+n_1) &\leqslant n_1(b_2-n_2) \\ k'(b_1-g_1-g_2+n_2) &\leqslant n_2(b_1-n_1) \end{aligned}$$

and there exists a $\mathrm{PA}(k+k'', g_1+g_2:\max(n_1,n_2))$ with b_1+b_2 rows where

$$k'' = \left\lfloor \frac{b_1b_2}{b_1+b_2-g_1-g_2} \right\rfloor.$$

Proof. Consider each PA as its dual, resolvable packings of $K_{b_1}-K_{n_1}$ with g_1 blocks in each class and $K_{b_2}-K_{n_2}$ with g_2 blocks in each class. We then take the disjoint union of both graphs and associate one resolution class from the first PA with a unique class from the second PA (this is equivalent to concatenating the two packing arrays vertically with distinct alphabets sets). Then using Theorem 16, with either the hole in the bipartite graph, or not, we add the maximum number of matchings, each one contributes another resolution class with g_1+g_2 blocks and these can all be viewed as new columns. It is worth noting that if $g_1+g_2 \geqslant b_1+(n_1n_2)/b_1$ then

$$k' = \left\lfloor \frac{b_1b_2-n_1n_2}{b_1+b_2-g_1-g_2} \right\rfloor. \qquad \square$$

Example 21. An example of Theorem 20 is given here. We start with two copies of a PA$(4,3:2)$, one on symbol set $\{0,1,2\}$ and the other on symbol set $\{3,4,5\}$, vertically concatenated.

$$\begin{array}{cccc}
0 & 0 & 0 & 0\\
1 & 1 & 1 & 1\\
0 & 1 & 2 & 2\\
2 & 2 & 0 & 1\\
2 & 0 & 1 & 2\\
1 & 2 & 2 & 0\\
3 & 3 & 3 & 3\\
4 & 4 & 4 & 4\\
3 & 4 & 5 & 5\\
5 & 5 & 3 & 4\\
5 & 3 & 4 & 5\\
4 & 5 & 5 & 3
\end{array}$$

Each one represents a resolvable packing in K_6-K_2. We find the maximum number of 6-matchings in the bipartite graph $K_{6,6}$ of edges between the two K_6. Each additional matching allows us to add one more column to the packing array:

$$\begin{array}{cccccccccc}
0 & 0 & 0 & 0 & \quad 0 & 0 & 0 & 0 & 0 & 0\\
1 & 1 & 1 & 1 & \quad 1 & 1 & 1 & 1 & 1 & 1\\
0 & 1 & 2 & 2 & \quad 2 & 2 & 2 & 2 & 2 & 2\\
2 & 2 & 0 & 1 & \quad 3 & 3 & 3 & 3 & 3 & 3\\
2 & 0 & 1 & 2 & \quad 4 & 4 & 4 & 4 & 4 & 4\\
1 & 2 & 2 & 0 & \quad 5 & 5 & 5 & 5 & 5 & 5\\
3 & 3 & 3 & 3 & \quad 0 & 5 & 4 & 3 & 2 & 1\\
4 & 4 & 4 & 4 & \quad 1 & 0 & 5 & 4 & 3 & 2\\
3 & 4 & 5 & 5 & \quad 2 & 1 & 0 & 5 & 4 & 3\\
5 & 5 & 3 & 4 & \quad 3 & 2 & 1 & 0 & 5 & 4\\
5 & 3 & 4 & 5 & \quad 4 & 3 & 2 & 1 & 0 & 5\\
4 & 5 & 5 & 3 & \quad 5 & 4 & 3 & 2 & 1 & 0
\end{array}$$

Thus we get that $\mathrm{pa}(10,6:2) \geqslant 12$ which is the best possible by the upper bound from Theorem 4.

3.3. Utility of the constructions

We implemented on computer the calculation of upper and lower bounds using the methods discussed in this article. We only give $3 \leqslant g \leqslant 9$, although the PBD bound

Table 1
Abbreviation list of methods constructing packing arrays

Abbreviation	Construction method
a	Only g or $g+1$ rows possible, Eq. (5) and Theorem 6
b	At least $g+2$ rows exists, by Eq. (4).
c	Generalization of MacNeish's theorem, Inequality 2
d	Monotonicity in g: $\mathrm{pa}(k,g:n) > \mathrm{pa}(k,g-1:n)$
e	Orthogonal arrays exist
f	Wilson's construction, Inequality 3
g	The PBD construction, Theorem 13
h	Dual of a RBIBD, Subsection 2.1
i	Monotonicity in k: $\mathrm{pa}(k,g:n) \geqslant \mathrm{pa}(k+1,g:n)$
j	Monotonicity in n: $\mathrm{pa}(k,g:n) \geqslant \mathrm{pa}(k,g:n+1)$
k	Removal of a column to increase n, see proof of Theorem 7
l	Direct Graph Construction, Theorem 15
m	First Recursive Construction from Theorem 20
n	Second Recursive Construction from Theorem 20
o	Incomplete transversal design method, Theorem 10
p	Construction from [12]
q	Dual of a CURD, Section 2.1
r	Construction of Abdel-Ghaffar [1]

Table 2
Abbreviations list of methods for packing array upper bounds

Abbreviation	Upper bound
α	Only g or $g+1$ rows possible, Eq. (5) and Theorem 6
β	Disjoint row bound, Theorem 7
γ	Theorem 4
δ	Monotonicity in k: $\mathrm{pa}(k,g:n) \leqslant \mathrm{pa}(k-1,g:n)$
ε	Monotonicity in n: $\mathrm{pa}(k,g:n) \leqslant \mathrm{pa}(k,g:n-1)$
ζ	A packing array with g^2-1 rows can be completed to one with g^2 rows [1]
η	Result found by hand
θ	Global Upper bound of g^2
ι	Upper bound from [12]

Table 3
Bounds on the sizes of ternary packing arrays

k	4	5	6
$\mathrm{pa}(k,3:1)$	***9*** eh $-$ $\alpha\beta\gamma\delta\varepsilon\theta\iota$	***6*** l $-$ $\beta\gamma$	***4*** l $-$ $\alpha\gamma$
$\mathrm{pa}(k,3:2)$	***6*** klp $-$ η	***4*** l $-$ $\alpha\gamma\iota$	***3*** l $-$ $\alpha\beta\gamma\iota$
$\mathrm{pa}(k,3:3)$	***3*** a $-$ $\alpha\beta$	***3*** l $-$ $\alpha\beta\gamma\delta$	***3*** l $-$ $\alpha\beta\gamma\delta\varepsilon$

Table 4
Bounds on the sizes of quatary packing arrays

k	5	6	7	8
pa(k, 4 : 1)	***16*** eh $-$ $\alpha\beta\gamma\delta\varepsilon\theta\iota$	***9*** qr $-$ γ	***8*** l $-$ γ	***5*** l $-$ $\alpha\gamma$
pa(k, 4 : 2)	9 jk $-$ 14γ	***8*** klm $-$ γ	***6*** l $-$ γ	***5*** l $-$ $\alpha\gamma\varepsilon$
pa(k, 4 : 3)	9 k $-$ 13$\gamma\iota$	7 l $-$ ι	***5*** l $-$ $\alpha\gamma\iota$	***4*** l $-$ $\alpha\beta\gamma\iota$
pa(k, 4 : 4)	***4*** a $-$ $\alpha\beta$	***4*** a $-$ $\alpha\beta\gamma\delta$	***4*** l $-$ $\alpha\beta\gamma\delta$	***4*** l $-$ $\alpha\beta\gamma\delta\varepsilon$

Table 5
Bounds on the sizes of pentary packing arrays

k	6	7	8	9	10
pa(k, 5 : 1)	***25*** eh $-$ $\alpha\beta\gamma\delta\varepsilon\theta\iota$	***15*** h $-$ γ	***10*** ijkl $-$ γ	***10*** l $-$ $\gamma\delta$	7 l $-$ γ
pa(k, 5 : 2)	15 jk $-$ 23γ	11 j $-$ 13γ	***10*** kl $-$ $\gamma\varepsilon$	***8*** l $-$ γ	7 l $-$ $\gamma\varepsilon$
pa(k, 5 : 3)	15 k $-$ 22γ	11 j $-$ 12γ	9 l $-$ 10$\gamma\varepsilon$	7 l $-$ γ	***6*** l $-$ $\alpha\gamma$
pa(k, 5 : 4)	11 i $-$ 19ι	***11*** p $-$ $\gamma\iota$	***6*** al $-$ $\alpha\gamma\iota$	***6*** l $-$ $\alpha\gamma\delta\iota$	***5*** l $-$ $\alpha\beta\gamma\iota$
pa(k, 5 : 5)	***5*** a $-$ $\alpha\beta$	***5*** a $-$ $\alpha\beta\delta$	***5*** a $-$ $\alpha\beta\gamma\delta$	***5*** l $-$ $\alpha\beta\gamma\delta$	***5*** l $-$ $\alpha\beta\gamma\delta\varepsilon$

was only implemented for $g \leqslant 7$ since the required enumeration of all PBDs has only been completed to seven points. Appendix A contains tables of these computations. Except for the optimum existence results (Corollary 18, Theorem 6, and the existence of MOLS), the relative utility of these constructions is not obvious. Corollary 18 and Theorem 6 obviously dominate the tables; but these existence results are for a restricted class of packing arrays. The other constructions are more interesting. Theorem 15 produces a large number of the best currently known lower bounds. The most useful constructions for $b > 2g$ seem to be Wilson's constructions.

4. Conclusion

A number of upper bounds and constructions for packing arrays were presented. The two upper bounds were the generalization of the Plotkin bound (Theorem 4) and the bound derived from the consideration of sets of disjoint rows (Theorem 6), a serendipitous bonus of considering disjoint rows which are required in the constructions. Abdel-Ghaffar completely determines pa($k, g : n$) when $n = 1$ and $k \geqslant 2g - 1$; by determining the packing number for matching into $K_b - K_n$, we extend his solution for all n. This restricts the investigation of packing arrays from k larger than the largest transversal design known to $k \leqslant 2g - 2$.

Two additional interesting combinatorial problems arise from the study of packing arrays. The first, the dual structure of a packing array which meets the generalized Plotkin bound, Theorem 4. The dual is a resolvable PBD on b points with a hole of size n where each resolution class has the same number, g, of blocks in it and the

Table 6
Bounds on the sizes of hexary packing arrays

k	3	4	5	6	7	8	9	10	11	12
pa(k, 6 : 1)	***36*** e − $\alpha\beta\gamma\delta\varepsilon\theta\iota$	***34*** j − ζ	26 dfgijk − 34δ	26 dgj − 34δ	16 dgj − 34δ	12 ijkl − 19γ	12 ijkl − 14γ	***12*** ijkl − γ	***12*** l − $\gamma\delta$	***9*** l − γ
pa(k, 6 : 2)	***36*** e − $\alpha\beta\gamma\delta\varepsilon\theta\iota$	***34*** j − ζ	26 dfgijk − 34$\delta\varepsilon$	26 dg − 34$\delta\varepsilon$	16 dg − 34$\gamma\delta\varepsilon$	12 ijklm − 19$\gamma\varepsilon$	12 ijklm − 14$\gamma\varepsilon$	***12*** klm − $\gamma\varepsilon$	***10*** l − γ	***8*** l − γ
pa(k, 6 : 3)	***36*** e − $\alpha\beta\gamma\delta\varepsilon\theta\iota$	***34*** j − ζ	26 djk − 34$\delta\varepsilon$	16 djk − 34$\delta\varepsilon$	12 dijl − 33γ	12 ijl − 19$\gamma\varepsilon$	12 l − 13γ	11 l − 12$\gamma\varepsilon$	***9*** l − γ	***8*** l − $\gamma\varepsilon$
pa(k, 6 : 4)	***36*** e − $\alpha\beta\gamma\delta\varepsilon\theta\iota$	***34*** j − ζ	26 djk − 34$\delta\varepsilon$	16 d − 34$\delta\varepsilon$	12 dijlm − 32γ	12 lm − 18γ	11 l − 13$\gamma\varepsilon$	9 lm − 12$\gamma\varepsilon$	***8*** l − γ	*7* l − $\alpha\gamma$
pa(k, 6 : 5)	***36*** e − $\alpha\beta\gamma\delta\varepsilon\theta\iota$	***34*** o − ζ	26 dgjk − 34$\delta\varepsilon$	12 dfijl − 34$\delta\varepsilon$	12 dl − 26ι	10 l − 13ι	***8*** blp − $\gamma\iota$	*7* a − $\alpha\gamma\iota$	*7* l − $\alpha\gamma\delta\iota$	***6*** l − $\alpha\beta\gamma\iota$
pa(k, 6 : 6)	***36*** e − $\alpha\beta\gamma\delta\varepsilon\theta\iota$	26 dfgi − 34ε	26 dg − 34$\delta\varepsilon$	12 l − 34$\delta\varepsilon$	***6*** a − $\alpha\beta$	***6*** a − $\alpha\beta\gamma\delta$	***6*** a − $\alpha\beta\gamma\delta$	***6*** a − $\alpha\beta\gamma\delta$	***6*** l − $\alpha\beta\gamma\delta$	***6*** l − $\alpha\beta\gamma\delta\varepsilon$

Table 7
Bounds on the sizes of septary packing arrays

k	8	9	10	11	12	13	14
pa(k, 7 : 1)	***49*** eh − $\alpha\beta\gamma\delta\varepsilon\theta\iota$	***28*** h − γ	***21*** h − γ	14 ijkl − 15γ	***14*** ijkl − γ	***14*** l − $\gamma\delta$	***10*** l − γ
pa(k, 7 : 2)	28 jk − 47γ	21 jk − 26γ	14 ijkl − 19γ	14 ijkl − 15$\gamma\varepsilon$	***14*** kl − $\gamma\varepsilon$	***12*** l − γ	***10*** l − $\gamma\varepsilon$
pa(k, 7 : 3)	28 jk − 46γ	21 k − 25γ	14 ijl − 18γ	14 l − 15$\gamma\varepsilon$	13 jl − 14$\gamma\varepsilon$	***11*** l − γ	***10*** l − $\gamma\varepsilon$
pa(k, 7 : 4)	28 k − 45γ	15 j − 24γ	14 l − 17γ	13 il − 15$\gamma\varepsilon$	13 l − 14$\gamma\varepsilon$	***10*** l − γ	***9*** l − γ
pa(k, 7 : 5)	15 ij − 44γ	15 j − 23γ	13 j − 16γ	12 l − 14γ	***9*** bilm − γ	***9*** l − $\gamma\delta$	***8*** l − $\alpha\gamma$
pa(k, 7 : 6)	15 i − 34ι	15 p − 18ι	***11*** l − ι	***8*** a − $\alpha\gamma\iota$	***8*** a − $\alpha\gamma\delta\iota$	***8*** l − $\alpha\gamma\delta\iota$	*7* l − $\alpha\beta\gamma\iota$
pa(k, 7 : 7)	*7* a − $\alpha\beta$	*7* a − $\alpha\beta\delta$	*7* a − $\alpha\beta\delta$	*7* a − $\alpha\beta\gamma\delta$	*7* a − $\alpha\beta\gamma\delta$	*7* l − $\alpha\beta\gamma\delta$	*7* l − $\alpha\beta\gamma\delta\varepsilon$

Table 8
Bounds on the sizes of octary packing arrays

k	9	10	11	12	13	14	15	16
pa(k, 8 : 1)	***64*** eh − $\alpha\beta\gamma\delta\varepsilon\theta\iota$	22 dj − 34γ	16 ijkl − 25γ	16 ijkl − 19γ	16 ijkl − 17γ	***16*** ijkl − γ	***16*** l − $\gamma\delta$	***12*** l − γ
pa(k, 8 : 2)	29 d − 62γ	22 d − 34$\gamma\varepsilon$	16 ijklm − 25$\gamma\varepsilon$	16 ijklm − 19$\gamma\varepsilon$	16 ijklm − 17$\gamma\varepsilon$	***16*** klm − $\gamma\varepsilon$	***14*** l − γ	***12*** l − $\gamma\varepsilon$
pa(k, 8 : 3)	22 djk − 61γ	16 ijl − 33γ	16 ijl − 25$\gamma\varepsilon$	16 ijl − 19$\gamma\varepsilon$	16 l − 17$\gamma\varepsilon$	15 jl − 16$\gamma\varepsilon$	***13*** l − γ	***11*** l − γ
pa(k, 8 : 4)	22 d − 60γ	16 ijlm − 33$\gamma\varepsilon$	16 ijlm − 24γ	16 lm − 18γ	15 il − 17$\gamma\varepsilon$	15 l − 16$\gamma\varepsilon$	***12*** l − γ	***11*** l − $\gamma\varepsilon$
pa(k, 8 : 5)	16 dijl − 59γ	16 ijl − 32γ	16 l − 24$\gamma\varepsilon$	15 lm − 17γ	14 l − 16γ	***12*** lm − γ	***11*** l − γ	***10*** l − γ
pa(k, 8 : 6)	16 dijl − 58γ	16 l − 32$\gamma\varepsilon$	14 ilm − 19γ	14 lm − 17$\gamma\varepsilon$	***11*** l − γ	***10*** bilm − γ	***10*** l − $\gamma\delta$	***9*** l − $\alpha\gamma$
pa(k, 8 : 7)	16 dl − 43ι	13 l − 20ι	12 l − 13ι	***10*** blmp − $\gamma\iota$	***9*** a − $\alpha\gamma\iota$	***9*** a − $\alpha\gamma\delta\iota$	***9*** l − $\alpha\gamma\delta\iota$	***8*** l − $\alpha\beta\gamma\iota$
pa(k, 8 : 8)	***8*** a − $\alpha\beta$	***8*** a − $\alpha\beta\gamma\delta$	***8*** a − $\alpha\beta\gamma\delta$	***8*** a − $\alpha\beta\gamma\delta$	***8*** a − $\alpha\beta\gamma\delta$	***8*** a − $\alpha\beta\gamma\delta$	***8*** l − $\alpha\beta\gamma\delta$	***8*** l − $\alpha\beta\gamma\delta\varepsilon$

Table 9
Bounds on the sizes of nonary packing arrays

k	10	11	12	13	14	15	16	17	18
pa$(k,9:1)$	***81*** eh $-\alpha\beta\gamma\delta\varepsilon\theta\iota$	27 ijk $-$ 45γ	27 ijk $-$ 30γ	***27*** h $-\gamma$	18 ijkl $-$ 21γ	18 ijkl $-$ 19γ	***18*** ijkl $-\gamma$	***18*** l $-\gamma\delta$	***14*** l $-\gamma$
pa$(k,9:2)$	27 ijk $-$ 79γ	27 ijk $-$ 43γ	27 jk $-$ 30$\gamma\varepsilon$	18 ijkl $-$ 25γ	18 ijkl $-$ 20γ	18 ijkl $-$ 19$\gamma\varepsilon$	***18*** kl $-\gamma\varepsilon$	***16*** l $-\gamma$	***14*** l $-\gamma\varepsilon$
pa$(k,9:3)$	27 ik $-$ 78γ	27 ik $-$ 42γ	27 k $-$ 30$\gamma\varepsilon$	18 ijl $-$ 24γ	18 ijl $-$ 20$\gamma\varepsilon$	18 l $-$ 19$\gamma\varepsilon$	17 jl $-$ 18$\gamma\varepsilon$	***15*** l $-\gamma$	***13*** l $-\gamma$
pa$(k,9:4)$	18 ijl $-$ 77γ	18 ijl $-$ 41γ	18 ijl $-$ 29γ	18 ijl $-$ 23γ	18 l $-$ 20$\gamma\varepsilon$	17 ijl $-$ 19$\gamma\varepsilon$	17 l $-$ 18$\gamma\varepsilon$	***14*** l $-\gamma$	***13*** l $-\gamma\varepsilon$
pa$(k,9:5)$	18 ijl $-$ 76γ	18 ijl $-$ 40γ	18 ijl $-$ 29$\gamma\varepsilon$	18 l $-$ 22γ	17 il $-$ 19γ	17 l $-$ 18γ	***14*** l $-\gamma$	***13*** l $-\gamma$	***12*** l $-\gamma$
pa$(k,9:6)$	18 ijl $-$ 75γ	18 ijl $-$ 39γ	18 l $-$ 28γ	17 l $-$ 21γ	16 il $-$ 19$\gamma\varepsilon$	16 l $-$ 18$\gamma\varepsilon$	***12*** ilm $-\gamma$	***12*** l $-\gamma\delta$	***11*** l $-\gamma$
pa$(k,9:7)$	18 ijl $-$ 74γ	18 l $-$ 38γ	16 l $-$ 27γ	15 il $-$ 20γ	15 l $-$ 18γ	***12*** lm $-\gamma$	***11*** bilm $-\gamma$	***11*** l $-\gamma\delta$	***10*** l $-\alpha\gamma$
pa$(k,9:8)$	18 lp $-$ 53ι	15 l $-$ 25ι	14 l $-$ 17ι	***11*** blmp $-\gamma\iota$	***10*** a $-\alpha\gamma\iota$	***10*** a $-\alpha\gamma\delta\iota$	***10*** a $-\alpha\gamma\delta\iota$	***10*** l $-\alpha\gamma\delta\iota$	***9*** l $-\alpha\beta\gamma\iota$
pa$(k,9:9)$	***9*** a $-\alpha\beta$	***9*** a $-\alpha\beta\delta$	***9*** a $-\alpha\beta\gamma\delta$	***9*** a $-\alpha\beta\delta$	***9*** a $-\alpha\beta\gamma\delta$	***9*** a $-\alpha\beta\gamma\delta$	***9*** a $-\alpha\beta\gamma\delta$	***9*** l $-\alpha\beta\gamma\delta$	***9*** l $-\alpha\beta\gamma\delta\varepsilon$

structure has k resolution classes: a subclass of restricted resolvable designs. As noted in [4,7] these designs have important applications. The second problem that has arisen from the study of packing arrays is the question of the maximum number of edge disjoint m-matchings that can be packed into a graph G.

Future directions for research include packing matchings into general graphs. To search for packing arrays with $b \leqslant 2g$ we should consider packing graphs with subgraphs that include not only points and disjoint edges, but triangles as well. Lastly, for the application to databases, the investigation of decomposing $\mathbb{Z}_g^k$ into packing arrays is the next important step.

Appendix A. Tables of upper and lower bounds

We include here tables showing the values for the upper and lower bounds on packing arrays. We have calculated upper and lower bounds for $3 \leqslant g \leqslant 9$ and the interesting range of $k \leqslant 2g$, since $k \geqslant 2g - 1$ is completely known. In all cases we encode the methods used to obtain these values. Values known to be optimal are emphasized in bold (Tables 1–9).

References

[1] K.A.S. Abdel-Ghaffar, On the number of mutually orthogonal partial latin squares, Ars Combin. 42 (1996) 259–286.
[2] K.A.S. Abdel-Ghaffar, A. El Abbadi, Optimal disk allocation for partial match queries, ACM Trans. Database Systems 18 (1) (1993) 132–156.
[3] C.J. Colbourn, J.H. Dinitz (Eds.), The CRC Handbook of Combinatorial Designs, CRC Press, Boca Raton, 1996.
[4] P. Danziger, B. Stevens, Class-uniformly resolvable designs, J. Combin. Des. 9 (2001) 79–99.
[5] D.G. Hoffman, C.A. Rodger, The chromatic index of complete multipartite graphs, J Graph Theory 16 (2) (1992) 159–163.
[6] D.G. Hoffman, C.A. Rodger, On the number of edge-disjoint one factors and the existence of k-factors in complete multipartite graphs, Discrete Math. 160 (1–3) (1996) 177–187.
[7] E. Lamken, R. Rees, S. Vanstone, Class-uniformly resolvable pairwise balanced designs with block sizes two and three, Discrete Math. 92 (1991) 197–209.
[8] N.J.A. Sloane, Covering arrays and intersecting codes, J. Combin. Des. 1 (1993) 51–63.
[9] B. Stevens, Transversal covers and packings, Ph.D. Thesis, University of Toronto, Toronto, 1998.
[10] B. Stevens, A. Ling, E. Mendelsohn, A direct construction of transversal covers using group divisible designs, Ars Combin. 63 (2002) 145–159.
[11] B. Stevens, E. Mendelsohn, New recursive methods for transversal covers, J. Combin. Des. 7 (3) (1999) 185–203.
[12] B. Stevens, E. Mendelsohn, Packing arrays and packing designs, Des. Codes Cryptogr. 27 (1/2) (2002) 165–176.
[13] B. Stevens, L. Moura, E. Mendelsohn, Lower bounds for transversal covers, Des. Codes Cryptogr. 15 (3) (1998) 279–299.
[14] D. Wevrick, S.A. Vanstone, Class-uniformly resolvable designs with block sizes 2 and 3, J. Combin. Des. 4 (1996) 177–202.
[15] R.M. Wilson, Concerning the number of mutually orthogonal Latin squares, Discrete Math. 9 (1974) 181–198.