Perfect Hash Families: The Generalization to Higher Indices

Ryan E. Dougherty and Charles J. Colbourn

Abstract Perfect hash families are often represented as combinatorial arrays encoding partitions of k items into v classes, so that every t or fewer of the items are completely separated by at least a specified number of the chosen partitions. This specified number is the index of the hash family. The case when each t-set must be separated at least once has been extensively researched; they arise in diverse applications, both directly and as fundamental ingredients in a column replacement strategy for a variety of combinatorial arrays. In this paper, construction techniques and algorithmic methods for constructing perfect hash families are surveyed, in order to explore extensions to the situation when each t-set must be separated by more than one partition.

1 Introduction

Suppose that there are k items, and each is assigned one of v values. Our objective is to ensure that each set of t items receives t different values; when this occurs, the t items are *separated*. Evidently if $v \geq k$, each item can be assigned a value that is different from all others assigned, so that every set of t items is separated. However, when $v < k$, some two items necessarily receive the same value; then any t-set containing these two cannot be separated. When this occurs, suppose that N assignments of values to items are chosen, rather than one. Then one can ask: how small can N be so

R.E. Dougherty
Computing, Informatics, and Decision Systems Engineering, Arizona State University
e-mail: ryan.dougherty@asu.edu

C.J. Colbourn
Computing, Informatics, and Decision Systems Engineering, Arizona State University
e-mail: charles.colbourn@asu.edu

that every t-set of items is separated in at least λ of the assignments? This easily stated combinatorial question is challenging, and many open problems remain despite substantial research effort. It is an important question as well, with applications described next.

Mehlhorn [49] originally examined this question to provide an efficient way to store and retrieve frequently used information; in that context, the assignment of values to the items is treated as a *hash function* [32], and hence the question is phrased as one about families of hash functions. Applications to derandomization [6], circuit complexity [51] , and cryptography [16, 38, 61] arose. Subsequently, Stinson, Trung, and Wei [62] established applications of such families (with $\lambda = 1$) to construct numerous other combinatorial objects, such as separating systems, key distribution patterns, cover-free families, and secure frameproof codes. A general strategy, column replacement, has extended their range of applications into testing and measurement [24] and compressive sensing [27], among others.

In many of these applications, error correction through redundancy in the separation is needed; a few examples are given in [1, 44, 57]. Despite this, there has been little examination of such hash families with $\lambda > 1$, with the notable exception of [3,4]. In this paper, we therefore survey a number of the main construction methods for such hash families, with an eye to extending them to treat cases with $\lambda > 1$ when possible. Our emphasis is on fixed values of $\lambda \geq 1$; we only treat cases when λ increases as a function of N in the concluding remarks. We focus on combinatorial aspects, discussing in particular constructive approaches to produce explicit examples for use in applications.

In order to develop and extend these ideas formally, we extend the presentation in [24], employing the very general language of t-restrictions [5]. Let N, k, $v_1, \cdots, v_N$, $x_1, \cdots, x_k$, t, and λ be positive integers. An *abstract simplical complex* (ASC), $\mathcal{A}$, is a family of non-empty finite subsets of a set Γ that is closed under non-empty subsets; the *dimension* of an ASC, $\mathsf{dim}(\mathcal{A})$, is the maximum of $|X| - 1$, for all $X \in \mathcal{A}$. Let $\mathcal{H}$ be an abstract simplical complex on k vertices such that the maximum cardinality of any set in $\mathcal{H}$ is t; label the vertices of $\mathcal{H}$ as $c_1, \cdots, c_k$. Let Σ_i be a v_i-ary alphabet not containing $\star$ for all $1 \leq i \leq N$, and let Δ_j be an x_j-ary alphabet also not containing $\star$ for all $1 \leq j \leq k$. Define an $N \times k$ array A in which the ith row of A contains symbols from $\Sigma_i \cup \{\star\}$, and the jth column of A contains symbols from $\Delta_j \cup \{\star\}$. If there exist i, j for which $\Sigma_i \cap \Delta_j = \emptyset$, the entry in this cell must be $\star$. Let $\Delta = \bigcup_{j=1}^{k} \Delta_j$. A *$t$-restriction* is a χ-tuple $\mathcal{T} = ((\mathcal{P}_1, T_1), \cdots, (\mathcal{P}_\chi, T_\chi))$, where $\mathcal{P}_i \subseteq \Delta^t$ and $T_i \in \{\exists, \forall\}$. Each set $\mathcal{P}_i$ is a *demand*. For each $\mathcal{P}_i$, if $T_i = \exists$, then at least λ rows of A contains some element of $\mathcal{P}_i$; if $T_i = \forall$, then for each element of $\mathcal{P}_i$, at least λ rows contain that element. Let $\partial^i(\mathcal{S})$ be the set of $\binom{t}{i}$ sets of $(t-i)$-tuples, obtained by deleting the i chosen columns from each $s \in \mathcal{S}$. An array $\mathsf{A} = (a_{ij})$ *λ-satisfies* a given $\mathcal{P}_i$ and T_i if and only if for all $0 \leq j \leq t$ and any set $S \in \mathcal{H}$,

1. if $T_i = \exists$, then for each $\mathcal{P} \in \partial^j(\mathcal{P}_i)$, there exist λ rows $1 \le r_1 < \cdots < r_\lambda \le N$ such that $(a_{r_\ell,c_{i_1}}, \cdots, a_{r_\ell,c_{i_{|S|}}}) \in \mathcal{P}$ for all $1 \le \ell \le \lambda$ and $S \in \mathcal{H}$ when $|S| = t - j$; or
2. if $T_i = \forall$, then for each $\mathcal{P} \in \partial^j(\mathcal{P}_i)$, and for all $(\sigma_1, \cdots, \sigma_{t-j}) \in \mathcal{P} \cap \prod_{m=1}^{t-j} \Delta_{c_m}$, there exist λ rows $1 \le r_1 < \cdots < r_\lambda \le N$ such that $(a_{r_\ell,c_{i_1}}, \cdots, a_{r_\ell,c_{i_{|S|}}}) = (\sigma_1, \cdots, \sigma_{|S|})$ for all $1 \le \ell \le \lambda$ and $S \in \mathcal{H}$ when $|S| = t - j$.

If the array λ-satisfies each of the given $\mathcal{P}_i$ and T_i, then the array λ-*satisfies* $\mathcal{T}$. If an array A on N rows and k columns (and corresponding symbol set cardinalities for rows and columns) λ-satisfies a t-restriction $\mathcal{T}$, denote it by $\mathsf{TRA}_\lambda(N, k, \mathcal{H}, (v_1, \cdots, v_N), (x_1, \cdots, x_k), \mathcal{T})$. If $v_1 = \cdots = v_N$, A is *uniform*; otherwise, it is *mixed*. If $x_1 = \cdots = x_k$, A is *homogeneous*; otherwise, it is *heterogeneous*. When $\lambda = 1$, we omit it from the notation. When $T_1 = \cdots = T_\chi$, $\mathcal{T}$ is a *monotone* t-restriction. Most literature has concentrated on monotone t-restrictions with $\mathcal{H}$ being the hypergraph containing all possible hyperedges of size at most t on k vertices, and $\lambda = 1$.

This framework is very general, and it encompasses a number of well-studied combinatorial arrays. We establish more restrictive notation for some of them next. Choose an integer t, and form the set $\mathcal{M}_t$ of multisets whose elements contain nonnegative integers, for which the sums of each element in a multiset sum to t. Let $\mathcal{W} \subseteq \mathcal{M}_t$. A $\mathcal{W}$-*separating hash family* meets the following condition: when $C = \{c_1, \cdots, c_t\} \subseteq \binom{[k]}{t}$ and $W_1, \cdots, W_s$ is a partition of C with $\{|W_1|, \cdots, |W_s|\} \in \mathcal{W}$, define $\mathcal{D} = \{(y_1, \ldots, y_t) \in \Delta_{c_1} \times \cdots \times \Delta_{c_t} : y_c = y_{c'}$ only if c, c' belong to the same class of $W\}$. Then the demand $(\mathcal{D}, \exists)$ is met. When each demand is the stricter requirement that $\mathcal{D} = \{(y_1, \ldots, y_t) \in \Delta_{c_1} \times \cdots \times \Delta_{c_t} : y_c = y_{c'}$ if and only if c, c' belong to the same class of $W\}$, the hash family is $\mathcal{W}$-*partitioning*. When $\mathcal{W}$ consists of all partitions in $\mathcal{M}_t$ containing s parts, a $\mathcal{W}$-separating hash family is (t, s)-*distributing*. In both cases, when $\mathcal{W}$ contains a single set $W = \{w_1, \ldots, w_s\}$, the family is separatng (or partitioning) of *type* $\{w_1, \ldots, w_s\}$.

Of primary concern here are the (t, s)-distributing hash families with $s = t$. Such a family is a *perfect hash family*. In order to refer to objects of this type, we employ standard notation. A perfect heterogeneous hash family is denoted as a $\mathsf{PHHF}_\lambda(N; k, (v_1, \cdots, v_N), t)$, and a homogeneous one is written as a $\mathsf{PHF}_\lambda(N; k, v, t)$. If a PHHF_λ A λ-satisfies a given $\mathcal{P}_i$, then it λ-*separates* these columns.

An example of a (homogeneous) $\mathsf{PHF}_1(6; 12, 3, 3)$ is given in Figure 1. It is a 6×12 array (6 rows, 12 columns) on the three symbols $\{0, 1, 2\}$, in which every 3-set of columns is 1-separated. For the 6×3 subarray involving columns $8, 9$, and 10, only the last row consists of distinct symbols. Also, 148 of the 3-sets are exactly 1-separated; 44 are exactly 2-separated; 19 are exactly 3-separated; 4 are exactly 4-separated; and none are 5 or 6-separated. There is no $\mathsf{PHF}(5; 12, 3, 3)$ [9], so this array has the fewest possible rows.

								↓	↓	↓		
	0	1	2	2	1	2	2	0	1	1	0	0
	0	2	1	0	2	2	2	1	0	1	2	1
	1	0	0	2	2	2	1	1	2	1	0	2
	2	0	1	1	2	0	2	0	1	1	2	1
	2	0	2	1	2	1	0	2	2	1	1	0
→	2	0	1	2	1	1	2	[2]	[0]	[1]	2	1

Fig. 1 A perfect hash family $\mathsf{PHF}_1(6; 12, 3, 3)$.

The notation $\mathsf{SHHF}(N; k, (v_1, \cdots, v_N), \{w_1, \cdots, w_s\})$ is used for a separating hash family. More simply $\mathsf{SHF}(N; k, v, \{w_1, \cdots, w_s\})$ is used when it is homogeneous. Figure 2 gives an example of a (homogeneous) $\mathsf{SHF}(3; 16, 4, \{1, 2\})$. It is a 3×16 array on the four symbols $\{1, 2, 3, 4\}$ that is not a perfect hash family, because columns 11, 15, and 16 are separated by none of the three rows. However, in the 3×3 subarray consisting of these three columns, each of the three $\{1, 2\}$-separations is accomplished by a row.

										↓			↓	↓
	1	1	1	1	2	2	2	2	3	3	3	4	4	4
	1	2	3	4	1	2	3	4	1	2	3	4	1	2
→	1	2	3	4	2	1	4	3	3	4	[1]	2	4	3

													⌜2⌝	[1]
	3	4	4	4	4								⌞ ⌟	

Fig. 2 A $\mathsf{SHF}(3; 16, 4, \{1, 2\})$.

A distributing hash family is denoted by $\mathsf{DHHF}(N; k, (v_1, \cdots, v_N), t, s)$; as before, a homogeneous DHHF is a $\mathsf{DHF}(N; k, v, t, s)$. Figure 3 gives a (heterogeneous) $\mathsf{DHHF}(10; 13, \mathbf{v}, 5, 2)$ with $\mathbf{v} = (9, 9, 9, 3, 3, 3, 3, 4, 5, 2)$. Often one uses an exponential notation that indicates the repetition in the exponent: $\mathbf{v} = (9^3 3^4 4^1 5^1 2^1)$.

				↓	↓			↓		↓	↓
	6	7	8	3	4	0	2	2	3	0	5	1	1
	3	1	1	7	2	6	8	4	3	0	2	0	5
	8	5	1	4	2	3	2	6	7	0	1	3	0
	0	2	0	2	2	0	0	1	1	1	1	2	0
	0	0	2	1	1	1	2	0	0	2	2	0	1
→	1	1	2	2	[2]	⌜0⌝	1	0	0	[2]	1	⌜0⌝	⌜0⌝
	1	0	1	2	0	0	2	0	0	1	2	2	1
	1	1	0	1	0	3	2	0	2	0	1	0	2
	0	0	3	0	1	0	0	2	4	0	0	1	0
	0	⋆	⋆	⋆	⋆	1	⋆	⋆	1	⋆	⋆	0	1

Fig. 3 A $\mathsf{DHHF}(10; 13, \mathbf{v}, 5, 2)$ with $\mathbf{v} = (9^3 3^4 4^1 5^1 2^1)$.

Hash families in general, and perfect hash families in particular, play a central role in the construction of arrays that satisfy various t-restrictions.

Indeed, they form the essential ingredients in a general technique known as *composition* or *column replacement*, which we describe next.

Construction 1 *Suppose there exist:*

1. A, *a* $\mathsf{PHF}_\chi(M;\ell,k,t)$; *and*
2. B, *a* $\mathsf{TRA}_\lambda(N;k,\mathcal{H},(v_1^{s_1}\cdots,v_\rho^{s_\rho}),x^k,\mathcal{T})$ *with* $N=\sum_{i=1}^{\rho}s_iv_i$ *and* $\mathcal{H}=\binom{[k]}{t}$.

Construct an $NM\times\ell$ *array,* C, *by replacing each symbol* γ *in* A *by the column indexed by* γ *in* B. *Then* C *is a*

$$\mathsf{TRA}_{\chi\lambda}(NM;\ell,\mathcal{H}',((M\cdot v_1)^{s_1}\cdots,(M\cdot v_\rho)^{s_\rho}),x^\ell,\mathcal{T})$$

with $\mathcal{H}'=\binom{[\ell]}{t}$.

Construction 1 provides strong motivation for the study of perfect hash families, as it underlies the easy generation of 'large' arrays meeting t-restrictions. We outline one example of this, introducing a well-studied t-restriction that employs universal quantification. When for every t-set $\{c_1,\ldots,c_t\}$ of columns, the demand $(\Delta_{c_1}\times\cdots\times\Delta_{c_t},\forall)$ is to be met, the array is a *mixed-level covering array*, denoted by $\mathsf{MCA}(N;t,(v_1,\cdots,v_k))$; when the array is homogeneous, it is a *covering array*, denoted by $\mathsf{CA}(N;t,k,v)$. In any $\mathsf{CA}(N;k,v,t)$, symbols can be permuted *in each column independently* so that the first row consists entirely of a single symbol. This yields a *constant row*, and when the CA has been modified in this way, it is *standardized*. Figure 4 gives an example of a standardized $\mathsf{CA}(13;3,10,2)$.

```
0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1
1 1 1 0 1 0 0 0 0 1 1
1 0 1 1 0 1 0 1 0 0 0
1 0 0 0 1 1 1 0 0 0 0
0 1 1 0 0 1 0 0 1 0 0
0 0 1 0 1 0 1 1 1 0 0
1 1 0 1 0 0 1 0 1 0 0
0 0 0 1 1 1 0 0 1 1 1
0 0 1 1 0 0 1 0 0 1 1
0 1 0 1 1 0 0 1 0 0 1
1 0 0 0 0 0 0 1 1 1 1
0 1 0 0 0 1 1 1 0 1 1
```

Fig. 4 A $\mathsf{CA}(13;3,10,2)$.

In [23], a restriction on DHHFs is applied to gain an additional improvement on Construction 1 when the TRA is a covering array. *Partitioning* hash families are distributing hash families except not only for any partition of t-columns into s parts (possibly empty) the entries in any two different parts

are pairwise disjoint, but the symbols in each part are all equal; denote it by a $\mathsf{PaHF}(N; k, v, t, s)$. It is shown there that a $\mathsf{DHF}(N; k, 2, t, 2)$ is a $\mathsf{PaHF}(N; k, 2, t, 2)$. Notably, partitioning hash families are appealing because if a $\mathsf{CA}(N + \rho; v, k, v)$ with ρ constant rows and a $\mathsf{PaHF}(M; \ell, k, t, v)$ exist, then a $\mathsf{CA}(NM + \rho; t, \ell, v)$ exists; this shows that the ingredient CA can be of a different strength than the PaHF. However, PaHFs appear difficult to construct. For recent work on probabilistic methods for PaHFs, see Cassels and Godbole [20].

For some specific t-restrictions, such as covering arrays, one may achieve a smaller number of rows while still satisfying the restriction (e.g., standardizing the CA). Introducing heterogeneity can in some cases provide even more improvements; we content ourselves for now with the homogeneous case. By observing the generality of the framework, much of this survey can be appropriately applied to other types of t-restrictions.

Because a PHF with M rows leads to a TRA with NM rows, one wants the PHF ingredient to have as few rows as possible. The *perfect hash family (row) number*, $\mathsf{PHFN}_\lambda(k, v, t)$, is the minimum N for which a $\mathsf{PHF}_\lambda(N; k, v, t)$ exists. This notation does not extend naturally to heterogeneous hash families, because the number of rows is to be determined. To circumvent this notational issue, we often instead consider maximizing the number of columns rather than minimizing the number of rows. More formally, the *perfect hash family (column) number* $\mathsf{PHHFK}_\lambda(N, \mathbf{v}, t)$ is defined to be the maximum k for which a $\mathsf{PHHF}_\lambda(N; k, \mathbf{v}, t)$ exists. For homogeneous hash families, the notation $\mathsf{PHFK}_\lambda(N, v, t)$ is used. For homogeneous families, one can easily change between row and column numbers:

$$\mathsf{PHFN}_\lambda(k, v, t) = \min(N : \mathsf{PHFK}_\lambda(N, v, t) \geq k)$$

$$\mathsf{PHFK}_\lambda(N, v, t) = \max(k : \mathsf{PHFN}_\lambda(k, v, t) \leq N)$$

A $\mathsf{PHF}_\lambda(N; k, v, t)$ is *optimal* if $N = \mathsf{PHFN}_\lambda(k, v, t)$. Much study has been devoted to determining perfect hash family numbers for as many parameters as possible, as well as what structure underlies optimal PHFs. Moreover, one would hope to provide an explicit representation of the PHF with those parameters, particularly for constructing other combinatorial objects and t-restrictions. If this is not possible, then knowing asymptotics on this quantity is important in helping determine asymptotics for other objects.

2 The Basics

First we state elementary relationships among perfect hash family numbers. In order to treat heterogeneous situations as well, we employ perfect hash family column numbers.

Additional rows cannot reduce the number of columns that can be achieved:

Fact 1 $\mathsf{PHHFK}_\lambda(N,(v_1,\ldots,v_N),t) \le \mathsf{PHHFK}_\lambda(N{+}1,(v_1,\ldots,v_{N+1}),t)$ *whenever* $v_{N+1} \ge 0$.

Reducing the size of column sets to be separated also cannot reduce the number of columns.

Fact 2 $\mathsf{PHHFK}_\lambda(N,(v_1,\ldots,v_N),t) \le \mathsf{PHHFK}_\lambda(N,(v_1,\ldots,v_N),t-1)$ *if* $t \ge 2$.

Reducing λ enables one to remove rows without reducing the number of columns.

Fact 3 $\mathsf{PHHFK}_\lambda(N,(v_1,\ldots,v_N),t) \le \mathsf{PHHFK}_{\lambda-1}(N{-}1,(v_1,\ldots,v_{i-1},v_{i+1},\ldots v_N),t)$ *if* $\lambda \ge 2$ *and* $1 \le i \le N$.

Increasing the number of symbols in a row cannot reduce the number of columns.

Fact 4 $\mathsf{PHHFK}_\lambda(N,(v_1,\ldots,v_N),t) \le \mathsf{PHHFK}_\lambda(N,(v_1,\ldots,v_{i-1},v_i{+}1,v_{i+1},\ldots v_N),t)$ *if* $1 \le i \le N$.

Changing the number of columns is also of interest. Removing a column is straightforward, but adding a column can leave $\binom{k}{t-1}$ t-sets of columns unseparated. Naively one could add λ rows for each to obtain

Fact 5 $\mathsf{PHFN}_\lambda(k,v,t) \le \mathsf{PHFN}_\lambda(k+1,v,t) \le \mathsf{PHFN}_\lambda(k,v,t) + \lambda\binom{k}{t-1}$.

Walker and Colbourn [65] show a better bound, later generalized by Martirosyan and van Trung [48]. [Ryan: maybe add unpublished work of mine as a short mention here?]

In order to avoid situations in which a row does not have enough symbols to separate any t-set of columns, we have:

Fact 6 $\mathsf{PHHFK}_\lambda(N,(v_1,\ldots,v_N),t) = \mathsf{PHHFK}_\lambda(N{-}1,(v_1,\ldots,v_{i-1},v_{i+1},\ldots v_N),t)$ *if* $v_i < t$.

One can also consider reducing the number of symbols in a row,:

Fact 7 $\mathsf{PHHFK}_\lambda(N,(v_1,\ldots,v_N),t) \le \left\lceil \frac{v_i-1}{v_i}\mathsf{PHHFK}_\lambda(N,(v_1,\ldots,v_{i-1},v_i-1,v_{i+1},\ldots v_N),t)\right\rceil$ *if* $1 \le i \le N$.

Iterating Fact 7 until Fact 6 applies, one obtains

Theorem 1. (Martirosyan and van Trung [48, Theorem 7.2])
$\mathsf{PHFN}(\lceil\frac{k(t-1)}{v}\rceil,v,t) \le \mathsf{PHFN}(k,v,t)-1$. *Equivalently,* $\mathsf{PHFK}(N-1,v,t) \ge \lceil\frac{t-1}{v}\mathsf{PHFK}(N,v,t)\rceil$.

Finally we describe a *row amalgamation* method for reducing the number of rows, which essentially comes from [15,59]. In a $\mathsf{PHHF}_\lambda(N; k, (v_1, \ldots, v_N), t)$, select two rows i and j with $1 \le i < j \le N$. From these two, form a single row whose entries are ordered pairs, with the first coordinate being the entry from row i and the second from row j. Delete rows i and j (with v_i and v_j symbols), and add the new row with $v_i v_j$ symbols. This method can reduce the number of times a t-set of columns is separated, but this number cannot be reduced to 0.

Fact 8 $\mathsf{PHHFK}_\lambda(N, (v_1, \ldots, v_N), t) \le \mathsf{PHHFK}_{\max(1,\lambda-1)}(N-1, (w_1, \ldots, w_{N-1}), t)$ *whenever* $1 \le i < j \le N$, $\{v_1, \ldots, v_N\} \setminus \{v_i, v_j\} = \{w_1, \ldots, w_{N-2}\}$, *and* $w_{N-1} = v_i v_j$.

Now let us dispense with some easier parameter sets. If $N < \lambda$, there are insufficient rows to λ-separate any t-set; so we assume that $N \ge \lambda$. Now $\mathsf{PHFN}_\lambda(k, v, 1) = \lambda$ for all k, $v \ge 1$, and $\lambda \ge 1$, because any row separates all 1-sets of columns. Henceforth we only consider cases with $t \ge 2$. Fact 3 underlies the following:

Fact 9 $\mathsf{PHHFK}_\lambda(\lambda, (v_1, \ldots, v_\lambda), t) = \min(v_i : 1 \le i \le \lambda)$.

Because of this, we concentrate on cases in which no λ rows are each permitted to contain k or more distinct symbols. It is natural to ask whether one can obtain larger values of k when the number of rows is allowed to exceed λ.

In general, *recursive* constructions combine ingredient PHFs to make 'larger' ones. Many of the facts given provide easy examples of recursive constructions. Of course, because being a perfect hash family of index λ is a t-restriction, so column replacement or composition (Construction 1) is a recursive construction. In Section 3, Theorem 3 also provides a recursive construction.

3 Few rows

No PHHF_λ with fewer than λ rows exists; when there are λ rows, Fact 9 applies. Suppose that $v_1 \ge \cdots \ge v_N \ge t$ and that a $\mathsf{PHHF}_\lambda(N; v_\lambda + 1, (v_1, \ldots, v_N), t)$ exists. Using Fact 3 we remove the first $\lambda - 1$ rows to obtain a $\mathsf{PHHF}_1(N - \lambda + 1; v_\lambda + 1, (v_\lambda, \ldots, v_N), t)$. Each row contains at least one pair of columns in which a symbol is repeated. Let $\{\gamma_i, \gamma_i'\}$ be such a pair of column indices with a repeated symbol in row i for $\lambda \le i \le N$. Then $\bigcup_{i=\lambda}^{N} \{\gamma_i, \gamma_i'\}$ is a set of at most $2(N - \lambda + 1)$ columns that is separated by no row of the PHHF_1. When $N \le \frac{t}{2} + \lambda - 1$, this is a contradiction. Hence we conclude

Fact 10 *When* $v_1 \ge \cdots \ge v_N \ge t$, $\mathsf{PHHFK}_\lambda(N, (v_1, \ldots, v_N), t) > v_\lambda$ *only if* $N \ge \lceil \frac{t+1}{2} \rceil + \lambda - 1$.

Although this condition is not sufficient whenever $v_1 \geq \cdots \geq v_N \geq t$, it does establish, for example, that $\mathsf{PHFK}_1(N, v, t) = v$ whenever $1 \leq N \leq \frac{t}{2}$. In order to increase the number of columns, one therefore requires further rows.

We describe a $\mathsf{PHF}_\lambda(s+\lambda; m(s+\lambda), m(s+\lambda-1)+1, 2s+1)$ whenever $m \geq 2$ and $s \geq 1$, generalizing a result of Walker and Colbourn [65] when $\lambda = 1$.

Construction 2 *Let $s \geq 1$, $m \geq 2$, and $\lambda \geq 1$. A $\mathsf{PHF}_\lambda(s+\lambda; m(s+\lambda), m(s+\lambda-1)+1, 2s+1)$ is constructed as follows. Form a set of $m(s+\lambda-1)$ elements X, and let ∞ be an element not in X. Then the desired PHF_λ contains exactly one occurrence of ∞ in each column, and contains each element of X exactly once in each row.*

Now Construction 2 yields more columns than symbols, and by Fact 10 it has the fewest rows for which this is possible with strength $t = 2s + 1$. As a function of v, k grows linearly. This linear relationship is not restricted to the minimum number of rows, Blackburn [13] explored this phenomenon when $\lambda = 1$, and explicit computations, again for $\lambda = 1$, are pursued in [26]. We apply Blackburn's techniques to treat all λ.

To begin, we suppose that $k > v_1 \geq \cdots \geq v_N$, for otherwise we can either reduce λ by Fact 3 or conclude that one row suffices when $\lambda = 1$. Then every row contains at least one element that is repeated. The key idea is to classify the entries in each row; an entry is a *singleton* for this row when it appears exactly once in the row, and a *replicate* otherwise. Now suppose that a $\mathsf{PHHF}_\lambda(N; k, (v_1, \ldots, v_N), N-(\lambda-2)+s)$ with $0 \leq s \leq \frac{t-1}{2}$ has a column γ with at most $s+\lambda-1$ singletons, and hence at least $N-s-\lambda+1 = t-2s-1$ replicates. Form a set C of $t-2s$ column indices by including γ, and a column index from each of the $t-2s-1$ rows that contains the same symbol as in column γ. Now choose any s further rows, and for each add a pair of column indices for columns containing the same symbol in this row to C. In total, C now contains at most $t-2s+2s = t$ column indices, and C is not separated in any of $t-2s-1+s = N-\lambda+1$ rows. But then C is not λ-separated, because at most $\lambda-1$ rows remain.

So $\lambda + s = t - N + 2(\lambda-1)$ is the minimum number of singletons in each column. In a row having v_i symbols, at most $v_i - 1$ can be singletons. However, there must be at least $k(t - N + 2(\lambda-1))$ singletons in total. It follows that

$$k(t - N + 2(\lambda-1)) \leq \sum_{i=1}^{N} (v_i - 1).$$

Hence we obtain

Lemma 1. *A $\mathsf{PHHF}_\lambda(N; k, (v_1, \ldots, v_N), t)$ with $N - (\lambda-2) \leq t \leq 2(N-(\lambda-2)) - 1$ satisfies*

$$k \leq \max\left(t, v_1, \ldots, v_N, \frac{\sum_{i=1}^{N}(v_i - 1)}{t - N + 2(\lambda - 1)}\right).$$

Lemma 1 ensures that for a $\mathsf{PHF}_\lambda(N; k, v, t)$ with $N - (\lambda - 2) \leq t \leq 2(N - (\lambda - 2)) - 1$, (or, equivalently, $\frac{t+1}{2} + \lambda - 2 \leq N \leq t + \lambda - 2$), k grows linearly as a function of v.

For $\lambda = 1$, Blackburn [13] establishes that when $N = t + \lambda - 1$, k grows superlinearly in v. We extend his construction to treat all values of λ next.

Construction 3 *Let $t \geq 2$, $\lambda \geq 1$, and $a \geq 2$. Then there exists a $\mathsf{PHF}_\lambda(t + \lambda - 1; a^{t+\lambda-1}, a^{t-\lambda-2}, t)$. The set of all vectors from $\{1, \ldots, a\}^{t+\lambda-1}$ index the columns. In each column, in the ith row place the vector from $\{1, \ldots, a\}^{t+\lambda-2}$ obtained by deleting the entry in the ith coordinate of the column index.*

The verification that this is a $\mathsf{PHF}_\lambda(t + \lambda - 1; a^{t+\lambda-1}, a^{t-\lambda-2}, t)$ comes essentially from [13]. Suppose to the contrary that there are t rows $\rho_1, \ldots, \rho_t$ in which t columns $\gamma_1, \ldots, \gamma_t$ are not separated. Form a graph G on vertex set $\{\gamma_1, \ldots, \gamma_t\}$; for each $\rho \in \{\rho_1, \ldots, \rho_t\}$, place an edge in G between some two vertices whose columns share a symbol in row ρ, and colour the edge with ρ. Now G has t vertices and t edges (of t different colours), and hence contains a cycle, say on vertices $\{v_0, \ldots, v_\ell\}$. Let $e_i = \{v_i, v_{i+1}\}$ for $0 \leq i < \ell$ have colour c_i, and let $e_\ell = \{v_\ell, v_0\}$ have colour c_ℓ. For $0 \leq i \leq \ell$, the two columns indexed by e_i agree in coordinate c_i *and in no other*. Because all edge colours in the cycle are distinct, the columns indexed by each of $\{e_0, \ldots, e_{\ell-1}\}$ agree in coordinate c_ℓ, but e_ℓ requires that they disagree, which yields the contradiction.

Fact 1 now guarantees that k grows superlinearly in v whenever $N \geq t + \lambda - 1$, in contrast with the requirement that $\mathsf{PHFK}_\lambda(t + \lambda - 2, v, t) \leq \max(v, \frac{1}{\lambda}(t + \lambda - 2)(v - 1))$ from Lemma 1.

Of course, practical interest is in obtaining a large number of columns, but understanding the situation with few rows has an important consequence.

Theorem 2. *Let $N = \alpha(t - 1) + \beta$ with $1 \leq \beta \leq t - 1$. Then $\mathsf{PHFK}_\lambda(N + \lambda - 1, v, t) \leq \mathsf{PHFK}_1(N, v, t) \leq v^\alpha(t - 1 + \beta(v - 1)) \leq (t - 1)v^{\lceil \frac{N}{t-1} \rceil}$.*

Proof. Consider a $\mathsf{PHF}_1(N; k, v, t)$. Repeatedly amalgamate rows (Fact 8) to form a $\mathsf{PHHF}_1(t - 1; k, ((v^{\alpha+1})^\beta (v^\alpha)^{t-1-\beta}), t)$. Apply Lemma 1 to conclude that $k \leq v^\alpha(t - 1 + \beta(v - 1))$.

Row amalgamation can reduce λ when it exceeds 1, and hence Theorem 2 employs amalgamation only when $\lambda = 1$. Consequently, it yields a useful upper bound when $\lambda = 1$, but we anticipate that the bound is weak when $\lambda > 1$.

In the 'linear' range when $\frac{t+1}{2} + \lambda - 2 \leq N \leq t + \lambda - 2$, Lemma 1 establishes that for some constant $c_{N-(\lambda-1),t-N-(\lambda-1),\lambda}$, the existence of a $\mathsf{PHF}_\lambda(N; k, v, t)$ requires that $k \leq c_{N-(\lambda-1),t-N-(\lambda-1),\lambda} v$. Blackburn [13] devises a linear programming formulation to explicitly determine the constant

$d_{N,t-N}$ so that $k = d_{N,t-N}v(1 + o(1))$ when $\lambda = 1$. In order to establish the lower bound asymptotically, he develops a construction technique using coverings. In [26], the method is extended to produce explicit constructions for small values of v, and to treat the generalization to distributing hash families (with $\lambda = 1$).

The key idea is to employ hash families with an additional property. A $\mathsf{DHHF}(t; k, (v_1, \ldots, v_t), t, p)$ is *fractal* if $t \leq 2$, or if, for each row j, deleting row j yields a fractal $\mathsf{DHHF}(t-1; k, (v_1, \ldots, v_{j-1}, v_{j+1}, \ldots, v_t), t-1, \min(p, t-1))$. Fractal PHHFs are fractal DHHFs with $p = t$. A $\mathsf{DHHF}(t; k, (v_1, \ldots, v_n), t, p)$ is *α-fractal* if it is fractal and at least α rows of the DHHF contain all distinct symbols. In addition to fractal DHHFs, we require a second ingredient. An *(n, m, d)-covering* of *type* $(\rho_0, \ldots, \rho_{m-1})$ is a collection of n subsets $\{P_0, \ldots, P_{n-1}\}$ of $\{0, \ldots, m-1\}$ satisfying:

1. $|\{P_r : 0 \leq r < n, P_r \ni c\}| = \rho_c$ for $0 \leq c < m$; and
2. For every $S \subseteq \{0, \ldots, m-1\}$ with $|S| = d$, S is a subset of some set in $\{P_1, ..., P_{n-1}\}$.

Extending the methods in [13] to use fractal and heterogeneous ingredients, and distributing hash families rather than just perfect ones, we have the following.

Theorem 3. [26] *Suppose that there exist*

- *an (n, m, d)-covering $\mathcal{P} = \{P_0, \ldots, P_{n-1}\}$ of type $(\rho_0, \ldots, \rho_{m-1})$, and*
- *for each $0 \leq c < m$, a ρ_c-fractal $\mathsf{DHHF}(n; k_c, (v_{0,c}, \ldots, v_{n-1,c}), n, p)$ in which, for $0 \leq r < n$, row r contains all distinct symbols when $c \in P_r$.*

Then there exists a $\mathsf{DHHF}(n; \sum_{c=0}^{m-1} k_c, (w_0, \ldots, w_{n-1}), n + d, p')$ where

$$w_r = \textstyle\sum_{c=0}^{m-1} v_{r,c} \text{ for } 0 \leq r < n, \text{ and}$$
$$p' = \begin{cases} p & \text{if } p < n - d \\ n + d & \text{if } p \geq n - d. \end{cases}$$

One easy result uses the trivial $(\binom{m}{d}, m, d)$-covering:

Corollary 1. *Let $m > d \geq 1$ be integers. Suppose that a fractal*

$$\mathsf{PHHF}(\tbinom{m-1}{d}; \kappa, (w_0, \ldots, w_{\binom{m-1}{d}-1}), \tbinom{m-1}{d})$$

exists. Let σ be the sum of the $m - d$ largest elements in $\{w_i : 0 \leq i \leq \binom{m-1}{d} - 1\}$. Then a $\mathsf{PHF}(\binom{m}{d}; m\kappa, d\kappa + \sigma, \binom{m}{d} + d)$ exists.

A sufficient condition for a PHHF to be fractal establishes that fractal PHHFs are common; nevertheless, not all PHHFs are fractal.

Lemma 2. [26] *If a $\mathsf{PHHF}(t; k, (v_1, \ldots, v_t), t)$ has at most one singleton in each row, it is fractal.*

It is not clear whether there is a useful generalization of the fractal definition to cases when the strength is not the same as the number of rows, although a variant of this idea is used in [30] in a recursive construction. Theorem 3 gives a powerful tool for the construction of DHHFs with few rows, but one needs many ingredients in order to apply it effectively. We now therefore turn to construction techniques.

4 The Connection with Codes

One direct method for constructing a variety of hash families relies on the existence of error-correcting codes. A *code* with parameters $(n, k, d)_q$ is a set of k distinct vectors (*codewords*) of length n over an alphabet of size q, so that every two distinct codewords are at Hamming distance at least d. Then $A_q(n, d)$ denotes the largest k for which there is a $(n, k, d)_q$ code, and $A(n, d, w)$ denotes the largest k for which there is a $(n, k, d)_2$ code in which each cod word has weight w (i.e., w 1's). See [19] for some bounds on $A(n, d, w)$. Determining exact values for $A_q(n, d)$ and $A(n, d, w)$ in general remains a major challenge.

Alon [2] shows a connection between codes and PHF_1s; see also Atici et al. [10]. We give the easy generalization for higher index:

Theorem 4. *If there is an $(n, k, d)_q$ code, then for any t, λ such that $\binom{t}{2} < \frac{n-\lambda+1}{n-d}$, there is a $\mathsf{PHF}_\lambda(n; k, q, t)$.*

Proof. Let C be an $(n, k, d)_q$ code. Construct an array A that has the codewords of C as its columns. Let $L = \{c_1, \cdots, c_t\}$ be a set of t columns of A. Two distinct columns of L can agree in at most $n - d$ rows, so the number of rows in which not all columns of L disagree is at most $(n - d)\binom{t}{2}$. Provided that $(n - d)\binom{t}{2} < n - \lambda + 1$, A has at least λ rows that separate L.

In general, Theorem 4 yields a PHF from a code, but not every PHF need arise in this way. However, when $t = 2$ the correspondence is exact (see Mehlhorn [49] and Atici, Magliveras, Stinson, and Wei [10] when $\lambda = 1$):

Theorem 5. *An $(n, k, \lambda)_q$ code is equivalent to a $\mathsf{PHF}_\lambda(n; k, q, 2)$.*

It follows that $\mathsf{PHFK}_1(N, v, 2) = v^N$ and hence $\mathsf{PHFN}_1(k, v, 2) = \lceil \frac{\log k}{\log v} \rceil$. By considering all codewords in $\{0, \ldots, v-1\}^N$ whose entries sum to 0 (mod v), one has $\mathsf{PHFK}_2(N, v, 2) = v^{N-1}$. When $\lambda \geq 3$, one wants $(n, k, d)_q$ codes with $d \geq 3$. Numerous constructions and bounds are known [47], but in general exact values are not.

There is a $\mathsf{PHHF}_1(N; \prod_{i=1}^{N} v_i, (v_1, \cdots, v_N), 2)$ for any $N \geq 1$ and $v_1, \cdots, v_N \geq 2$, obtained by taking all possible column vectors. In the heterogeneous case, one has a correspondence with codewords in which each coordinate has its own alphabet, but such codes have not been much studied.

Turning to cases with $t \geq 3$, Theorem 4 has been extensively employed, particularly to Reed-Solomon codes to make PHFs with $\lambda = 1$ [10]. We formulate a generalization using design-theoretic terminology. A *transversal design*, $\mathsf{TD}(s, k, n)$ is a triple $(V, \mathcal{G}, \mathcal{B})$ where V is a set of kn points, partitioned into k groups $\mathcal{G} = \{G_1, \cdots, G_k\}$, and $|G_i| = n$ for all i. Furthermore, $\mathcal{B}$ contains n^s blocks of size k with $|B_i \cap G_j| = 1$ for all i, j, and $|B_i \cap B_j| \leq s$ for all $i \neq j$. A standard construction of transversal designs over the finite field $\mathbb{F}_q$ follows.

Construction 4 *A* $\mathsf{TD}(s, k, q)$ *with* $k \leq q+1$ *exists. Let* $X = \{x_1, \ldots, x_k\} \subseteq \mathbb{F}_q \cup \{\infty\}$. *The elements of the* TD *are* $\mathbb{F}_q \times X$. *For each polynomial* $a_0 + a_1 y + \cdots + a_{s-1} y^{s-1}$ *of degree* $s-1$ *with coefficients from* $\mathbb{F}_q$, *form a block that contains element* (b, z) *whenever* $z \in X$ *and (1)* $b = a_0$ *when* $z = \infty$, *or (2)* $b = a_0 + a_1 z + \cdots + a_{s-1} z^{s-1}$ *otherwise (all arithmetic performed in* $\mathbb{F}_q$*).*

A transversal design constructed in this manner is *linear*. Treating the blocks of the $\mathsf{TD}(s, k, q)$ from Construction 4 as columns and the elements of X as rows, we obtain a $k \times q^s$ array C on q symbols. In fact, because the difference between two polynomials of degree at most $s-1$ is also a polynomial of degree at most $s-1$, and such a polynomial has at most $s-1$ roots, the columns of C form a $(k, q^s, k-s)_q$ code, so Theorem 4 applies. When constructed in this way, the PHF_λ is *linear* as well. However, because the code has a natural algebraic interpretation, much more can be said. Suppose that there is a set X for which every set t polynomials of degree $s-1$ disagree on some value of X. This can arise when $|X| \leq (s-1)\binom{t}{2}$. For example, Blackburn [14] shows that a $\mathsf{PHF}(3; r^3, r^2, 3)$ exists for all $r \geq 2$, and that a $\mathsf{PHF}(6; p^2, p, 4)$ exists for all primes $p \geq 17$ or $p = 11$. This phenomenon has been extensively examined when $\lambda = 1$. A PHF is *optimal linear* if it is linear and no linear PHF exists having fewer rows. Blackburn [14] provides explicit constructions of PHFs, some of which are optimal.

Blackburn and Wild [17] showed that if q is a sufficiently large prime power, there is an optimal linear $\mathsf{PHF}(s(t-1); q^s, q, t)$ for $s, t \geq 2$. For specific choices of s and t, characterizations of the number of rows that suffice for small prime powers q have been carried out by Barwick and Jackson [11,12], and Colbourn and Ling [29]. These provide numerous explicit examples of PHF_1s that are easily constructed. The extension of larger values of λ is straightforward.

It remains an open question whether an optimal PHF exists whenever an optimal linear PHF exists [14, 17]. The linear perfect hash families always consist of rows in which the numbers of occurrences of each symbol are as equal as possible. Of course, this equireplication cannot be required for all parameter sets; consider, for example, Construction 2. When $s = 1, m = 2$, and $\lambda = 1$, it is possible to prove that the corresponding PHF_2s have a single equivalence class, and so every row of any $\mathsf{PHF}_1(2; k, v, 3)$ arising from this construction must have this property. Nevertheless, it appears plausible that once the number of rows is large enough, every row can be required to be

nearly equireplicated. If true, this constraint could simplify the development of further constructions.

However, we do not expect that linear PHFs lead to the largest number of columns. Consider, for example, $\mathsf{PHFK}(3, v, 3)$. By [14], $\mathsf{PHFK}(3, v, 3) = \Omega(v^{1.5})$; however, Walker and Colbourn [65] found solutions for small v that suggest a larger growth rate, and posed the question of whether $\mathsf{PHFK}(3, v, 3) = \mathrm{o}(v^2)$. Fuji-Hara [39] constructs $\mathsf{PHF}(3; v^5, v^3, 3)$ and $\mathsf{PHF}(3; v^2(v+1), v^2, 3)$ for a prime power $v \geq 3$, to establish that $\mathsf{PHFK}(3, v, 3) = \Omega(v^{5/3})$. Shangguan and Ge [58] solved the question of Walker and Colbourn: For sufficiently large v and arbitrary $\varepsilon > 0$, that $q^{2-\varepsilon} < \mathsf{PHFK}(3, v, 3) = \mathrm{o}(v^2)$. A similar result for $\mathsf{PHFK}(4, v, 4)$ is also proved.

One does not need transversal designs constructed over the field $\mathbb{F}_q$ in order to produce a code. It is well known that a $\mathsf{TD}(2, k, v)$ is equivalent to $k-2$ mutually orthogonal latin squares of side v (see [25], for example). Via this connection, one can generalize a result of Stinson, Wei, and Zhu [63] to $\lambda \geq 1$, by also employing Theorem 4:

Theorem 6. [63] *If there are at least* $s = \binom{t}{2} + \lambda - 2$ *MOLS of order* n, *there exists a* $\mathsf{PHF}_\lambda(s + 2 + \lambda; n^2, n, t)$.

The same authors generalize this statement to mutually orthogonal $n \times m$ latin rectangles on $\max(m, n)$ symbols, obtaining a PHF_1 with mn columns. Dinitz, Ling, and Stinson [34] establish in some cases that the number of rows employed by Theorem 6 can be reduced by ensuring that the corresponding TD avoids certain forbidden configurations.

Another generalization of transversal designs is to block designs. Let X be a set of v *points*, and $\mathcal{B}$ be a set of b subsets of X, called *blocks* of X each with k points. Then $(X, \mathcal{B})$ is a *balanced incomplete block design* (BIBD) if every point occurs in r blocks, and every pair of points occurs in λ blocks. We denote this by $\mathsf{BIBD}(v, b, r, k, \lambda)$. By a simple counting argument, $vr = bk$ and $\lambda(v-1) = r(k-1)$. A BIBD is *resolvable* if $\mathcal{B}$ can be partitioned into r *parallel classes*, and each class contains $\frac{v}{k}$ disjoint blocks. We denote this by $\mathsf{RBIBD}(v, b, r, k, \lambda)$. Brickell [18] and Atici, Magliveras, Stinson, and Wei [10] proved that if there is an $\mathsf{RBIBD}(v, b, r, k, \lambda)$ and $r > \lambda\binom{w}{2}$, there is a $\mathsf{PHF}_1(r; v, \frac{v}{k}, w)$. For results on the existence and asymptotics of RBIBDs, see [25].

5 Asymptotic Bounds and Algorithms

The probabilistic method yields bounds on PHFN_λ. A basic bound is obtained as follows. Let A be a random $N \times k$ array on v symbols; by this, each entry is selected uniformly at random and independent of all other entries. Choose t distinct column indices $C = \{c_1, \cdots, c_t\}$. The probability that C is separated in a single row A is the probability that in this row, all columns of C contain

distinct entries. This probability is $\phi_{t,v} = \frac{\binom{v}{t}t!}{v^t}$, and the probability that C is not separated in this row is $1 - \phi_{t,v}$. Often we write ϕ and $1 - \phi$ when t and v are fixed in this context.

Because all rows are chosen independently, the number of rows in which C is not separated is equal to ρ with probability $\binom{N}{\rho}(1-\phi)^\rho\phi^{N-\rho}$. Define the probability $\psi_{\lambda,N,t,v}$ (or, more simply, $\psi_{\lambda,N}$) to be $\sum_{\rho=0}^{\lambda-1}\binom{N}{\rho}(1-\phi)^\rho\phi^{N-\rho}$. By linearity of expectation, the expected number of t-sets of columns that are not λ-separated in A is $\binom{k}{t}\psi_{\lambda,N}$. When this expected number is less than 1, a $\mathsf{PHF}_\lambda(N;k,v,t)$ surely exists. When $\lambda = 1$, $\psi_{1,N} = (1-\phi)^N$. Then taking logarithms of $\binom{k}{t}(1-\phi_{t,v})^N < 1$, we obtain an easily stated bound, first established by Mehlhorn:

Theorem 7. [49] $\mathsf{PHFN}_1(k,v,t) \le \left\lceil \frac{\log\binom{k}{t}}{f(v,t)} \right\rceil$, *where* $f(v,t) = \log(v^t) - \log(v^t - t!\binom{v}{t})$.

Stinson, van Trung, and Wei [62] improved this bound with an expurgation method. We generalize their technique (for $\lambda = 1$) using the technique of post-processing [64] or oversampling [28].

Theorem 8. $\mathsf{PHFN}_1(k,v,t) \le \min_{x\ge 0}\left(N : \binom{k+x}{t}(1-\phi_{t,v})^N < x+1\right)$.

Proof. Start with an array with $k+x$ columns, and let E be the expected number of unseparated t-sets of columns in this array if each entry is chosen uniformly and independently. If $E < x+1$, then by deleting one column from each of the (at most) x unseparated t-sets, we must have a $\mathsf{PHF}_1(N;k,v,t)$. Then $\binom{k+x}{t}(1-p)^N < x+1$; find the minimum value of N such that there exists an $x \ge 0$ for which this inequality holds.

In a related context [28], the choice $x = \frac{k}{t-1}$ was shown to be the best; the same value works for PHF_1s as well. Of course, this method extends in the natural way to higher values of λ.

Stein [60], Lovász [46], and Johnson [42] devise a greedy strategy for explicitly producing solutions to certain covering problems of the size guaranteed by the basic probabilistic method. Their method encompasses t-restrictions. In the context of perfect hash families with $\lambda = 1$, the method constructs the family one row at a time. As long as there remains at least one t-set to separate, the method chooses a next row to maximize the number of t-sets separated for the first time by this row. Let $\mathcal{U}_i$ denote the set of t-sets of columns that are not yet separated after i rows have been selected. If row $i+1$ is selected at random, linearity of expectations determines that the expected size of $\mathcal{U}_{i+1}$ is $(1-\phi)|\mathcal{U}_i|$. Hence the greedy selection guarantees that $|\mathcal{U}_{i+1}| \le \lfloor(1-\phi)|\mathcal{U}_i|\rfloor$. Our goal is to find the smallest N such that $|\mathcal{U}_N| < 1$; this ensures that we have found an array with N rows that separates all t-sets. Because $|\mathcal{U}_0| = \binom{k}{t}$, we have $|\mathcal{U}_N| \le \binom{k}{t}(1-\phi)^N$. Consequently this

greedy strategy leads to explicit constructions meeting the bounds of Theorem 7 and Theorem 8. In fact, it typically yields smaller values of N than are guaranteed by these results, because (1) expectations can be replaced by their integer floors, and (2) the greedy strategy may choose a row that separates many more t-sets than the expectation.

Unfortunately, even when t and v are fixed, a naive implementation of this greedy strategy is not efficient, because it asks for the examination of v^k rows in order to select a maximum. No efficient algorithm is known for computing a row that separates the maximum number of t-sets among a given set $\mathcal{U}$ of t-sets. One could, however, be content with a row that separates at least the expected number rather than the maximum, without affecting the analysis. A randomized approach to select row $i+1$ could first calculate $\phi|\mathcal{U}_i|$, the expected number separated, and generate random rows until one separates at least this expected number from $\mathcal{U}_i$. Such a method guarantees to produce a solution meeting the bound on N, but does not guarantee to run in polynomial time.

Using conditional expectations to derandomize this approach, Colbourn [22] devised the density method, a deterministic method for constructing perfect hash families that is efficient when t and v are fixed. We wish to produce the row $i+1$ efficiently to separate at least $\phi|\mathcal{U}_i|$ t-sets from $\mathcal{U}_i$. The key idea is to consider partial rows, in which some entries have been *fixed* to one of the v values, and some are *free* to take on any value. For such a partial row, one can efficiently calculate the expected number of t-sets in $\mathcal{U}_i$ that are separated when the free entries are chosen uniformly at random. Then, as long as free entries remain, we can choose one and tentatively fix it to each of the v possible values, computing the expectation for each. Then we fix the free entry permanently to a value that leads to a largest expectation. At each stage, the expected number of t-sets of $\mathcal{U}_i$ separated by this (partial) row cannot decrease. Hence, once all entries are fixed, a row has been efficiently found that separates at least the expected number for a random row.

In practice, the density method often completes with a number of rows that is significantly smaller than guaranteed by Theorem 7 (see [22]). We expect that improvements in a similar method for covering arrays [54,55] can be applied to hash families as well.

Unfortunately, the extension of these algorithmic methods to cases with $\lambda > 1$ is not immediate. One row cannot, by itself, λ-separate a t-set. Hence rather than simply tracking t-sets that are 0-separated, progress towards constructing a perfect hash family of index $\lambda > 1$ is measured by the number of ℓ-separated t-sets for each $0 \leq \ell < \lambda$. Nevertheless, we outline one method to follow the Stein-Lovász-Johnson paradigm. Suppose that i rows have been selected, and that the t-sets that are not yet separated at least λ times are in one of $\mathcal{U}_{i,0}, \ldots, \mathcal{U}_{i,\lambda-1}$ where $\mathcal{U}_{i,\ell}$ contains all t-sets that are ℓ-separated in the first i rows. Extending the earlier analysis, when t, v, and λ are fixed, one can efficiently calculate the expected number $\psi((\mathcal{U}_{i,0}, \ldots, \mathcal{U}_{i,\lambda-1}), N-i)$ of t-sets that are not separated at least λ times if $N-i$ further rows are

selected uniformly at random. Then one (greedily) chooses row $i+1$ so that

$$\psi((\mathcal{U}_{i+1,0},\ldots,\mathcal{U}_{i+1,\lambda-1}),N-i-1)\le\psi((\mathcal{U}_{i,0},\ldots,\mathcal{U}_{i,\lambda-1}),N-i).$$

Set $\mathcal{U}_{0,0}$ to contain all $\binom{k}{t}$ t-sets, and $\mathcal{U}_{0,\ell}=\emptyset$ for $1\le\ell<\lambda$. Choose N so that $\psi((\mathcal{U}_{0,0},\ldots,\mathcal{U}_{0,\lambda-1}),N)<1$. The key difference when $\lambda>1$ is that one presupposes the determination of N, the target number of rows, a step that is not needed when $\lambda=1$.

The bound of Theorem 7 can be improved in a different manner using the Lovász local lemma, a tool extensively used in combinatorics [7, 37]. We employ the symmetric version here:

Theorem 9. *Let $E_1,\cdots,E_n$ be events in a probability space, E_i is mutually dependent of at most d other events for all i, and $\Pr[E_i]\le p$ for all i. If $ep(d+1)\le 1$, then with nonzero probability all of the events simultaneously do not occur.*

In the setting of PHF_1s, an event E_i are that the ith t-set of columns is not separated, so that $\Pr[E_i]=(1-p)^N$. Events E_i and E_j are mutually dependent if and only if the corresponding t-sets have nonempty intersection, so $d=\binom{k}{t}-\binom{k-t}{t}-1$. When $e\,(1-p)^N\left(\binom{k}{t}-\binom{k-t}{t}\right)\le 1$, there is a PHF_1 with those parameters:

Theorem 10. [33] $\mathsf{PHFN}_1(k,v,t)\le\left\lceil\frac{\log\left(\binom{k}{t}-\binom{k-t}{t}\right)+1}{f(v,t)}\right\rceil$.

The same argument shows that when $e\psi_{\lambda,N,t,v}\left(\binom{k}{t}-\binom{k-t}{t}\right)\le 1$, there is a $\mathsf{PHF}_\lambda(N;k,v,t)$.

An improvement for many parameter sets was found by Procacci and Sanchis [53] using the algorithmic cluster expansion local lemma [56].

In landmark work, Moser and Tardos [50] develop a constructive method for objects whose sizes match the bound provided by the LLL, and their techniques have been shown to extend to the cluster expansion local lemma in [8, 52]. The method is randomized and runs in expected polynomial time when the number of rows is as dictated by the (original or cluster expansion) LLL bound. In the context of hash families, one first computes the value of N that the LLL bound asserts is sufficient, and generates a random $N\times k$ array on v symbols. Now order the $\binom{k}{t}$ t-sets of columns arbitrarily, and fix this ordering throughout. To check the array, consider each t-set in order until one is not λ-separated or all are checked. If all are checked, the current array is the desired PHF_λ. Otherwise, for the first t-set of columns that fails, randomly resample the entries in each row in each column of the t-set, and start checking again from the first t-set. Remarkably, this *column resampling* method terminates with a solution in expected polynomial time when t is fixed [8, 50, 52].

Because the value for N used is an upper bound, but typically not the exact value, one can apply the same column resampling approach for smaller values of N. When N is smaller, there is no guarantee of expected polynomial running time, and indeed no guarantee that the method will terminate at all. (This is discussed in a different context in [21].) Nevertheless, column resampling provides a practical algorithm for producing perfect hash families that are substantially smaller than the bounds. Of course, when resampling an $N \times t$ subarray, the number of t-sets that are separated fewer than λ times may increase. Moreover, the likelihood of this occurring increases as N decreases, as one would expect. Therefore one finds that when N is well below the bound, most column resamplings make the situation worse. To avoid wholesale changes in the array resulting in a significant increase in the number of unseparated t-sets, one could replace a single column within an unseparated t-set rather than all t. In general, one could consider partial resamplings of the type in [41].

Starting with a $\mathsf{PHF}_\lambda(N; k, v, t)$ and adding a randomly chosen column, one can choose always to replace the new column when there is an unseparated t-set. This *random extension* method [28] amounts to choosing possible random columns to adjoin until one is found that λ-separates all $\binom{k}{t-1}$ t-sets of columns containing the new one. In practical computations, often many additional columns can be adjoined. We do not report computational results for the methods discussed, instead referring the reader to [35].

Density-based and column resampling algorithms are conceptually simple, and while they can be useful for making perfect hash families with moderately large parameters, one should not expect either to produce optimal PHFs in general; indeed, the optimal linear perfect hash families have dramatically fewer rows than are guaranteed by the probabilistic methods. Unfortunately, known direct constructions address quite limited parameter sets. Nevertheless, we have seen that recursive constructions rely on finding suitable ingredients with small values of the parameters. For this reason, it is natural to consider *metaheuristics* that identify a set of changes allowed and a rule for determining when one applies the change to the array, and when one rejects the change and keeps the current array. Evidently the long-term objective of any such method is to reduce the number of t-sets that are unseparated, eventually to 0.

Arguably, the Moser-Tardos algorithm is such a method, in which the possible changes are the resamplings of all entries of an unseparated t-set of columns, whose rule accepts every change. Applying the method of [41] would choose from a smaller set of possible changes but still accept all. One might prefer changes that reduce, or at least do not increase, the number of unseparated t-sets. *Local optimization* or *hill-climbing* accepts changes only when the change does not increase the number of t-sets. As one might expect, it can happen that none of the offered changes are accepted, and the method is stuck at a local optimum with no escape. For this reason, more sophisticated metaheuristic methods (simulated annealing [43], tabu search [40], the

great deluge algorithm [36], constraint or answer-set programming [45], for example) can be applied. In general, these methods require either that the separation status of each t-set be stored or frequently recalculated; whichever is done, one small change can affect the status of many t-sets. As a result these methods are (at least at present) limited to 'small' values of the parameters. Many such perfect hash families that are the smallest known have been found using metaheuristic methods [35], although none guarantees an improvement on column resampling or the density-based methods in general.

A different strategy, *post-optimization*, starts with a $\mathsf{PHF}_\lambda(N; k, v, t)$. Through a series of changes alterations of individual entries without making any t-set unseparated, the method attempts to make an entire row contain only $\star$ entries. When this can be done, the row can be deleted to form a $\mathsf{PHF}_\lambda(N-1; k, v, t)$; again, some computational results are reported in [35]. This strategy is discussed for general t-restrictions in [31].

6 Concluding Remarks

In this selective survey of combinatorial aspects of perfect hash families, we have found that many methods generalize in a natural manner to treat existence for fixed $\lambda > 1$. Perfect hash families are central in the construction of a wide variety of other combinatorial objects, particularly because of Construction 1. The extension of existence results for $\lambda = 1$ to higher λ can therefore provide a valuable tool in the construction of arrays for a variety of practical applications. An example arises in the construction of locating arrays (a type of t-restriction) with separation greater than 1 [57]; in that context, the increase in separation entails a substantial increase in the number of rows, and hence also in computation time. We argue that a viable alternative is to instead construct a perfect hash family with index $\lambda > 1$, and employ column replacement to address the t-restriction. We anticipate that this framework can prove useful in similar applications in which the effects of a single row are prone to error or mismeasurement.

In addition to ensuring that every t-set of columns be separated at least λ times, one might address the more stringent requirement that every t-set be separated at least $\underline{\lambda}$ and at most $\overline{\lambda}$ times. When $\underline{\lambda} = \overline{\lambda}$, such a PHF is *perfectly balanced* [3]. Alon and Gutner [3] establish that a perfectly balanced $\mathsf{PHF}_\lambda(N; k, v, t)$ can exist only when $N = \Omega(k^{\lfloor t/2 \rfloor})$ for t fixed. Contrast this with the $\Theta(\log k)$ growth rate for PHF_1s to understand why perfectly balanced PHFs are not frequently used. On the other hand, a $\mathsf{PHF}(N; k, v, t)$ is δ-*balanced* for some $\delta \geq 1$ if there is a value $T > 0$ so that every t-set of columns is separated at least $\frac{T}{\delta}$ and at most δT times [4]. Alon and Gutner [4] show that for any fixed $\delta > 1$, there is a δ-balanced $\mathsf{PHF}(N; k, v, t)$ with N close to $2^{O(t \log \log t)} \log k$; so, for fixed t, the growth rate is the same as for PHF_1s. Their approach relies (in small part) on the binomial distribution

of the number of times a t-set is separated and the application of Chernoff bounds. Moreover, their techniques yield an explicit construction method in principle; its practical effectiveness for intermediate values of k has not been explored.

When δ-balanced PHFs are used in Construction 1 with different t-restrictions, the array constructed inherits from the balanced PHF a lower bound on the number of rows in which the t-restriction is met. However, the t-restriction may be met in a row arising from a row of the PHF despite failure of the PHF to separate in this row; hence balanced PHFs need not result in balanced t-restrictions through Construction 1. For these reasons, it is reasonable to focus on extending known methods, and finding new methods, for constructing perfect hash families of index $\lambda > 1$.

Acknowledgments

This work is supported in part by the U.S. National Science Foundation grants #1421058 and #1813729. Thanks to Randy Compton, Stephanie Forrest, Erin Lanus, Kaushik Sarkar, and Violet Syrotiuk for helpful discussions.

References

1. Akhtar, Y., Phoa, F.K.H.: A construction of cost-efficient designs with guaranteed repeated measurements on interaction effects. preprint (2019)
2. Alon, N.: Explicit construction of exponential sized families of k-independent sets. Discrete Mathematics **58**(2), 191–193 (1986)
3. Alon, N., Gutner, S.: Balanced hashing, color coding and approximate counting. In: Parameterized and Exact Computation, 4th International Workshop, IWPEC 2009, Copenhagen, Denmark, pp. 1–16 (2009)
4. Alon, N., Gutner, S.: Balanced families of perfect hash functions and their applications. ACM Trans. Algorithms **6**(3), 54:1–54:12 (2010)
5. Alon, N., Moshkovitz, D., Safra, S.: Algorithmic construction of sets for k-restrictions. ACM Trans. Algorithms **2**, 153–177 (2006)
6. Alon, N., Naor, M.: Derandomization, witnesses for Boolean matrix multiplication and construction of perfect hash functions. Algorithmica **16**(4-5), 434–449 (1996)
7. Alon, N., Spencer, J.H.: The probabilistic method. John Wiley & Sons (2004)
8. Alves, R.G., Procacci, A.: Witness trees in the Moser-Tardos algorithmic Lovász local lemma and Penrose trees in the hard-core lattice gas. Journal of Statistical Physics **156**(5), 877–895 (2014)
9. Atici, M.: Hash families: recursive constructions and applications to cryptography. PhD dissertation, University of Nebraska (1996)
10. Atici, M., Magliveras, S.S., Stinson, D.R., Wei, W.D.: Some recursive constructions for perfect hash families. Journal of Combinatorial Designs **4**(5), 353–363 (1996)
11. Barwick, S.G., Jackson, W.A.: A sequence approach to linear perfect hash families. Designs, Codes and Cryptography **45**(1), 95–121 (2007)
12. Barwick, S.G., Jackson, W.A.: Geometric constructions of optimal linear perfect hash families. Finite Fields and Their Applications **14**(1), 1–13 (2008)

13. Blackburn, S.R.: Perfect hash families with few functions. Unpublished manuscript
14. Blackburn, S.R.: Perfect hash families: Probabilistic methods and explicit constructions. Journal of Combinatorial Theory, Series A **92**(1), 54–60 (2000)
15. Blackburn, S.R.: Frameproof codes. SIAM Journal on Discrete Mathematics **16**(3), 499–510 (2003)
16. Blackburn, S.R., Burmester, M., Desmedt, Y., Wild, P.R.: Efficient multiplicative sharing schemes. In: Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, pp. 107–118 (1996)
17. Blackburn, S.R., Wild, P.R.: Optimal linear perfect hash families. Journal of Combinatorial Theory, Series A **83**(2), 233–250 (1998)
18. Brickell, E.F.: A problem in broadcast encryption. In: 5th Vermont Summer Workshop on Combinatorics and Graph Theory (1991)
19. Brouwer, A.E., Etzion, T.: Bounds for binary constant weight codes. IEEE Transactions on Information Theory **36**, 1334–1380 (1990)
20. Cassels, J., Godbole, A.: Covering arrays for equivalence classes of words. Journal of Combinatorial Designs **to appear** (2019)
21. Catarata, J.D., Corbett, S., Stern, H., Szegedy, M., Vyskocil, T., Zhang, Z.: The Moser-Tardos resample algorithm: Where is the limit? (an experimental inquiry). In: Proceedings of the Ninteenth Workshop on Algorithm Engineering and Experiments ALENEX, pp. 159–171 (2017)
22. Colbourn, C.J.: Constructing perfect hash families using a greedy algorithm. Coding and Cryptology pp. 109–118 (2008)
23. Colbourn, C.J.: Distributing hash families and covering arrays. Journal of Combinatorics, Information, and System Sciences **34**, 113–126 (2009)
24. Colbourn, C.J.: Covering arrays and hash families. NATO Science for Peace and Security Series, D: Information and Communication Security **29**(Information Security, Coding Theory and Related Combinatorics), 99–135 (2011)
25. Colbourn, C.J., Dinitz, J.H.: Handbook of Combinatorial Designs. CRC Press (2007)
26. Colbourn, C.J., Dougherty, R.E., Horsley, D.: Distributing hash families with few rows. Theoretical Computer Science (2018). (Accepted)
27. Colbourn, C.J., Horsley, D., Syrotiuk, V.R.: Strengthening hash families and compressive sensing. Journal of Discrete Algorithms **16**, 170–186 (2012)
28. Colbourn, C.J., Lanus, E., Sarkar, K.: Asymptotic and constructive methods for covering perfect hash families and covering arrays. Designs, Codes and Cryptography pp. 1–31 (2017)
29. Colbourn, C.J., Ling, A.C.H.: Linear hash families and forbidden configurations. Designs, Codes and Cryptography **52**(1), 25–55 (2009)
30. Colbourn, C.J., Ling, A.C.H.: A recursive construction for perfect hash families. Journal of Mathematical Cryptology **3**(4), 291–306 (2009)
31. Colbourn, C.J., Nayeri, P.: Randomized post-optimization for t-restrictions. In: Information theory, combinatorics, and search theory, *Lecture Notes in Computer Science*, vol. 7777, pp. 597–608. Springer, Heidelberg (2013)
32. Czech, Z.J., Havas, G., Majewski, B.S.: Perfect hashing. Theoretical Computer Science **182**, 1–143 (1997)
33. Deng, D., Stinson, D.R., Wei, R.: The Lovász local lemma and its applications to some combinatorial arrays. Designs, Codes and Cryptography **32**(1-3), 121–134 (2004)
34. Dinitz, J.H., Ling, A.C.H., Stinson, D.R.: Perfect hash families from transversal designs. The Australasian Journal of Combinatorics **37**, 233–242 (2007)
35. Dougherty, R.E.: Perfect hash family tables for t=3 to 11 (2017). URL http://www.public.asu.edu/∼redoughe/phf_pages/phf_tables.html
36. Dueck, G.: New optimization heuristics: The great deluge algorithm and the record-to-record travel. Journal of Computational Physics **104**(1), 86–92 (1993)
37. Erdős, P., Lovász, L.: Problems and results on 3-chromatic hypergraphs and some related questions. In: Infinite and finite sets, pp. 609–627. North-Holland, Amsterdam (1975)

38. Fiat, A., Naor, M.: Broadcast encryption. In: Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO '93, pp. 480–491 (1994)
39. Fuji-Hara, R.: Perfect hash families of strength three with three rows from varieties on finite projective geometries. Designs, Codes and Cryptography **77**(2-3), 351–356 (2015)
40. Glover, F., Laguna, M.: Tabu search. In: Handbook of combinatorial optimization, Vol. 3, pp. 621–757. Kluwer Acad. Publ., Boston, MA (1998)
41. Harris, D.G., Srinivasan, A.: The Moser-Tardos framework with partial resampling. In: 2013 IEEE 54th Annual Symposium on Foundations of Computer Science—FOCS 2013, pp. 469–478. IEEE Computer Soc., Los Alamitos, CA (2013)
42. Johnson, D.S.: Approximation algorithms for combinatorial problems. Journal of Computer and System Sciences **9**, 256–278 (1974)
43. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science **220**(4598), 671–680 (1983)
44. Knill, E., Bruno, W.J., Torney, D.C.: Non-adaptive group testing in the presence of errors. Discrete Applied Mathematics **88**(1), 261–290 (1998)
45. Lifschitz, V.: Answer set programming and plan generation. Artificial Intelligence **138**(1), 39–54 (2002)
46. Lovász, L.: On the ratio of optimal integral and fractional covers. Discrete Math. **13**(4), 383–390 (1975)
47. MacWilliams, F.J., Sloane, N.J.A.: The theory of error-correcting codes. North-Holland Publishing Co., Amsterdam-New York-Oxford (1977)
48. Martirosyan, S., Tran Van Trung: Explicit constructions for perfect hash families. Designs, Codes and Cryptography **46**(1), 97–112 (2008)
49. Mehlhorn, K.: On the program size of perfect and universal hash functions. In: 23rd Annual Symposium on Foundations of Computer Science (sfcs 1982), pp. 170–175 (1982)
50. Moser, R.A., Tardos, G.: A constructive proof of the general Lovász local lemma. Journal of the ACM (JACM) **57**(2), 11 (2010)
51. Newman, I., Wigderson, A.: Lower bounds on formula size of Boolean functions using hypergraph entropy. SIAM Journal on Discrete Mathematics **8**(4), 536–542 (1995)
52. Pegden, W.: An extension of the Moser-Tardos algorithmic local lemma. SIAM Journal on Discrete Mathematics **28**(2), 911–917 (2014)
53. Procacci, A., Sanchis, R.: Perfect and separating hash families: new bounds via the algorithmic cluster expansion local lemma. Annales de l'Institut Henri Poincaré D. Combinatorics, Physics and their Interactions **5**(2), 153–171 (2018)
54. Sarkar, K., Colbourn, C.J.: Upper bounds on the size of covering arrays. SIAM Journal on Discrete Mathematics **31**(2), 1277–1293 (2017)
55. Sarkar, K., Colbourn, C.J.: Two-stage algorithms for covering array construction. Journal of Combinatorial Designs **to appear** (2019)
56. Scott, A.D., Sokal, A.D.: The repulsive lattice gas, the independent-set polynomial, and the Lovász local lemma. J. Stat. Phys. **118**(5-6), 1151–1261 (2005)
57. Seidel, S.A., Sarkar, K., Colbourn, C.J., Syrotiuk, V.R.: Separating interaction effects using locating and detecting arrays. In: International Workshop on Combinatorial Algorithms, pp. 349–360 (2018)
58. Shangguan, C., Ge, G.: Separating hash families: A Johnson-type bound and new constructions. SIAM Journal on Discrete Mathematics **30**(4), 2243–2264 (2016)
59. Staddon, J.N., Stinson, D.R., Wei, R.: Combinatorial properties of frameproof and traceability codes. IEEE Transactions on Information Theory **47**, 1042–1049 (2001)
60. Stein, S.K.: Two combinatorial covering theorems. Journal of Combinatorial Theory. Series A **16**, 391–397 (1974)
61. Stinson, D.R.: On some methods for unconditionally secure key distribution and broadcast encryption. Designs, Codes and Cryptography **12**(3), 215–243 (1997)

62. Stinson, D.R., Tran Van Trung, Wei, R.: Secure frameproof codes, key distribution patterns, group testing algorithms and related structures. Journal of Statistical Planning and Inference **86**(2), 595–617 (2000)
63. Stinson, D.R., Wei, R., Zhu, L.: New constructions for perfect hash families and related structures using combinatorial designs and codes. Journal of Combinatorial Designs **8**(3), 189–200 (2000)
64. Van Den Berg, E., Candès, E., Chinn, G., Levin, C., Olcott, P.D., Sing-Long, C.: Single-photon sampling architecture for solid-state imaging sensors. Proceedings of the National Academy of Sciences **110**(30), E2752–E2761 (2013)
65. Walker II, R.A., Colbourn, C.J.: Perfect hash families: Constructions and existence. Journal of Mathematical Cryptology **1**(2) (2007)