Quality Control in Manufacturing Oligo Arrays: A Combinatorial Design Approach *

Rimli Sengupta and Martin Tompa

Technical Report #2000-08-03

September 7, 2000

Department of Computer Science and Engineering
Box 352350
University of Washington
Seattle, WA 98195-2350
{rimli,tompa}@cs.washington.edu

* This material is based upon work supported in part by a Sloan/DOE Fellowship in Computational Molecular Biology, by the National Science Foundation and DARPA under grant DBI-9601046, and by the National Science Foundation under grant DBI-9974498.

Abstract

The advent of the DNA microarray technology has brought with it the exciting possibility of simultaneously observing the expression levels of all genes in an organism. One such microarray technology, called "oligo arrays", manufactures short single strands of DNA (called *probes*) onto a glass surface using photolithography. An altered or missed step in such a manufacturing protocol can adversely affect all probes using this failed step, and is in general impossible to disentangle from experimental variation when using such a defective array. The idea of designing special quality control probes to detect a failed step was first formulated by Hubbell and Pevzner. We consider an alternative formulation of this problem and use a combinatorial design approach to solve it. Our results improve over prior work in guaranteeing coverage of all protocol steps and in being able to tolerate a greater number of unreliable probe intensities.

Keywords: DNA microarray, oligo array, photolithography, quality control, combinatorial design, error-correcting code, 2-design.

1. Introduction

Recent advances in DNA microarray technology have allowed biologists to obtain expression profiles of the genes in an organism in a quantitative and high throughput fashion. This has catalyzed a major paradigm shift in how biological knowledge is pursued. Computational analysis of such DNA microarray data has led to interesting biological hypotheses of unprecedented scope. For example, analysis of the expression profiles of all 6200 genes in *S. cerevisiae* during sporulation [2] revealed the possible participation of nearly 1000 genes that were previously not known to be involved in sporulation. There has been a recent explosion of similar experiments and analyses using DNA microarrays.

An important class of DNA microarray technology, called "oligo arrays", manufactures short single strands of DNA (called *probes*) onto a glass surface using photolithography [11]. The glass surface (or *array*) has a well-defined set of addresses (or *spots*) where the probes are grown. The manufacturing *protocol* is a sequence of steps $N_1 N_2 \ldots N_n$, each with an associated nucleotide $N_i \in \{\text{A, C, G, T}\}$. Conceptually, at the i^{th} step of the protocol a *mask* is placed on the glass array and the array is exposed to a solution containing the nucleotide N_i. This causes the probes at the positions on the array that are not masked to be extended by one base, N_i. The rest of the probes do not change during this step. The process is repeated with a new mask at each step, to build a diverse assortment of probes.

When completed, the array is employed as follows. A mixture of single-stranded DNA molecules (called *targets*) are each fluorescently tagged, and the mixture is applied to the array for hybridization to the array's probes. (See Appendix A for a brief explanation of DNA complementarity and hybridization.) After washing away any unbound targets, the fluorescence intensities of all array spots are measured. Since the array's probe sequences are known, this procedure measures the abundance of the bound complementary target sequences.

An altered or missed step in the array's manufacturing protocol can adversely affect all probes using the failed step, and thus their hybridization behavior with targets. The error ensuing from a faulty manufacturing step may well be impossible to disentangle from experimental variation when using the defective array. The problem of developing a quality control mechanism that detects during the manufacturing process if a step has failed is therefore of clear practical importance.

One approach to the quality control problem, formulated first by Hubbell and Pevzner [8], is to design a small set of special quality control probes, which they called "fidelity probes." Their ingenious idea was to manufacture the same probe sequence at a number of different spots, each spot using a different schedule of steps of the protocol. A protocol step i therefore has an associated set P_i of quality control spots that use this manufacturing step. These quality control probes are then hybridized with a complementary fluorescent target. The intensities within the set P_i provide a "signature" for the quality of step i. If many of the intensities within P_i are significantly lower than the remaining intensities, this is a good indication of step i being flawed. This is because all the spots have the same sequence and should therefore have similar hybridization behavior (hence similar intensities) if they are correctly manufactured. The focus of the work of Hubbell and Pevzner is to generate sets P_i that are sufficiently large and sufficiently unique that a failed step can be identified even in the presence of some unreliable spot intensities. This method is then used repeatedly for each probe in a supplied set $\mathcal{S}$ of probes. However, there may be steps in a protocol that cannot be used in manufacturing any of the probes in a given set $\mathcal{S}$. Assuming that $\mathcal{S}$ is supplied implies that the failure of such a step cannot be detected. Moreover, since there is no coordination among the solutions generated for distinct probes (the algorithm being used separately on each probe), Hubbell and Pevzner do not exploit the ability of the probes to collectively make the set of spots using a protocol step as large and as unique as possible.

We consider an alternative formulation of this problem that does not assume that the quality control probe sequences are supplied. We take the choice of the probe sequences into our own hands in order to guarantee that every protocol step is well covered by the quality control mechanism. Our design ensures that the number of distinct probes is small and that they hybridize poorly with themselves and with each other. This is a necessary constraint because if probes hybridize well with themselves or each other, then their corresponding complementary targets will too, rendering them unavailable to hybridize to the probes [15]. Our design further ensures that each probe hybridizes well only with the target that is complementary to it, and hybridizes poorly with the targets meant for the other probes. This property allows us to use multiple quality control targets (up to 4 in our current designs) simultaneously, thereby relaxing the requirement of Hubbell and Pevzner [8] that all probes are complementary to substrings of a single target.

The fact that we want balanced and sufficiently unique signatures for all steps in the protocol suggests a connection to the elegant theory of combinatorial design. For our purposes, a combinatorial design is just a 0-1 matrix with appropriate balance and uniqueness properties. The chief contribution of this work is to solve the quality control problem by developing a framework that builds on techniques from combinatorial design. For a preview, see Figure 5.

The rest of the paper is organized as follows. In Section 2 we state our formulation of the quality control problem along with the assumptions we make, and characterize the criterion that allows us to identify a failed step. Section 3 describes the combinatorial design approach we take in solving this formulation of the quality control problem. Section 4 presents the specific combinatorial designs that solve the quality control problem for the protocol ACGT ACGT ... and for a wide range of values of the number of protocol steps, the number of spots, and tolerance for the number of spots that may show unreliable intensities. These results are generalized in Section 5 to all periodic protocols with period 4. Section 6 poses some open questions.

2. The Quality Control Problem

A quality control scheme for a protocol with n steps using m spots can be viewed as an $m \times n$ 0-1 matrix $\mathcal{Q}$, with each column representing a protocol step and each row representing a spot. Each column of $\mathcal{Q}$ is labelled with the nucleotide used in that step. The entry $\mathcal{Q}_{ij}$ is 1 if and only if step j was used in manufacturing the probe at spot i. We will refer to such a matrix $\mathcal{Q}$ as a *Quality Control* (QC) matrix. The sequence of the oligonucleotide at spot i can be read out by concatenating the labels of the columns at which row i has a 1:

Definition 2.1: Let $\mathcal{P}$ be a protocol $N_1 N_2 \cdots N_n$, where N_j is the nucleotide used at the j^{th} step, $1 \leq j \leq n$. Given an $m \times n$ QC matrix $\mathcal{Q}$ for protocol $\mathcal{P}$, the *probe* p_i *at row* i of $\mathcal{Q}$ is defined to be $p_i = q_{i1} q_{i2} \ldots q_{in}$, where

$$q_{ij} \quad = \quad \begin{cases} N_j, & \text{if } \mathcal{Q}_{ij} = 1 \\ \text{the empty string} & \text{if } \mathcal{Q}_{ij} = 0 \end{cases} .$$

The probes manufactured at the m quality control spots are not all different. There will in general be c distinct probes, with several spots containing the same probe but manufactured using different schedules of steps of the protocol. Typical values of m, c, and n based on previous work [8] are: $m = 128$, $c = 4$, and $60 \leq n \leq 100$.

To actually perform quality control of a protocol, the quality control probes defined by $\mathcal{Q}$ are manufactured using the protocol onto m reserved spots on each chip of a wafer [7]. The manufacturer takes one chip from the wafer and tests it as follows: the chip is hybridized with fluorescent targets complementary to the c probes, scanned, and the resulting vector of m intensity values is used to determine which step, if any, failed. The remaining chips of the wafer are thus unaffected by the quality control process and their quality can be assessed under the assumption that a step failure affects every chip on a wafer.

The quality control problem for oligo arrays is essentially the problem of designing a QC matrix $\mathcal{Q}$ with the following property: each step in the protocol is used in a set of spots that is sufficiently large as well as sufficiently different from the set for any other step, so that any single failed step induces a unique signature on the intensity vector. This should be true even when not all intensities are reliable. The problem we would like to solve is the following:

Definition 2.2: (QC Problem) Given a protocol $\mathcal{P}$ with n steps up to 1 of which may fail, and a budget of m quality control spots up to d of which may be unreliable, construct an $m \times n$ QC matrix $\mathcal{Q}$ such that an intensity vector $\mathcal{I}$ of the m spots manufactured using $\mathcal{Q}$ allows unique identification of the failed step, if any.

One reason why there is interest in *identifying* the failed step, rather than simply detecting *whether* some step failed, is so that the manufacturer can correct any errors in the failed step's mask before remanufacturing the chip.

At least two natural optimization versions of the QC problem are immediate: construct $\mathcal{Q}$ as above that (i) for given n and m maximizes the spot fault tolerance d; (ii) for given n and d minimizes the spot usage m. [1]

The problem we solve in this work is not quite as general as the one stated in Definition 2.2. We cannot hope to take arbitrary parameter values n, m, and d as input and produce a QC matrix $\mathcal{Q}$ that meets the specifications. We explain in Section 2.4 why solving this general version would entail solving long-standing open questions in combinatorial design. However we are able to produce QC matrices for a wide range of values of n, m, and d that covers the desired settings in practice. We also do not solve this for arbitrary protocols $\mathcal{P}$, but rather a specific set of 24 periodic protocols, namely, $[\pi(\mathrm{ACGT})]^{n/4}$, where π is any permutation and n is a multiple of 4 in the range $60 \leq n \leq 132$. Again, this covers the typical protocols in practice.

2.1. Assumptions

We state the assumptions we make in formulating the QC problem, and contrast them when possible with the assumptions of Hubbell and Pevzner [8].

1. The manufacturing protocol is $[\pi(\mathrm{ACGT})]^{n/4}$, where π is any permutation and n is a multiple of 4. Up to one step may fail and the impact of this failure on the chip is spatially uniform. (Hubbell and Pevzner allowed an arbitrary protocol.)

2. Spot failure: up to d spots may show arbitrarily unreliable intensities due to experimental variations in hybridization, or due to chip faults. (The parameter d is implicit in both the *MinSize* and *MinDiff* parameters of Hubbell and Pevzner [8].)

3. Step failure model: when a step fails, a spot will show a low intensity if and only if the failed step was used in manufacturing the probe at that spot, with up to d exceptions. When no step fails, each spot will show a high intensity, with up to d exceptions. (The step failure model was not made explicit by Hubbell and Pevzner [8].)

[1] We note that this version subsumes nonadaptive combinatorial group testing [6], with columns of the QC matrix corresponding to the elements in the universe and the rows corresponding to the query sets, in two ways. First, answers to d of the queries could be lies. This version of the nonadaptive group testing problem is open (Yuan Ma, personal communication). Second, in group testing the order of universe elements is immaterial, whereas the column order of a QC matrix is critical to the probe sequences and their properties.

4. Spots containing different probes in general may have different hybridization behaviors. (See Appendix A.) Hence we will not compare intensity values of two different probe sequences. We will also not make the assumption that, within the set of spots sharing the same probe, we can distinguish between all intensities high and all low. Formally, defining the real number $\mathcal{I}_i$ to be the intensity value at spot i, we assume that for every probe k there is a nonnegative constant ϵ_k such that two reliable spots i and i' for this probe that show both high or both low intensities must satisfy $|\mathcal{I}_i - \mathcal{I}_{i'}| \leq \epsilon_k$, and reliable spots i low and i' high must satisfy $\mathcal{I}_{i'} - \mathcal{I}_i > \epsilon_k$. (The focus of the work of Hubbell and Pevzner [8] is the fidelity probe generation problem. The problem of identifying the failed step is not explicitly addressed, hence no explicit statements are made about intensity values.)

5. We are allowed multiple quality control targets that are designed so as to hybridize poorly to themselves and to each other. Each probe is designed to hybridize poorly to all but one of these targets. (In the concrete test case cited by Hubbell and Pevzner [8], they assume they are supplied a single 20-mer quality control target and the probes are the four possible 17-mers that hybridize perfectly to the length 17 substrings of this target.) Unlike Hubbell and Pevzner, our designs use up to 4 different quality control targets. Based on commercial availability of inexpensive oligo synthesis techniques, manufacturing several quality control targets poses no problem.

The following definition crystallizes our assumption about what constitutes poor hybridization.

Definition 2.3: We say that two single-stranded nucleotide sequences *hybridize poorly* if and only if, when they are arranged in antiparallel fashion, shifted an arbitrary offset with respect to each other, at least two out of every four consecutive pairs of aligned bases are not complementary; see Figure 1. (See Appendix A for an explanation of DNA complementarity and hybridization.) A set S of such sequences is said to *hybridize poorly* if and only if every sequence $s \in S$ hybridizes poorly to itself, to every other sequence in S, and to the complement of every sequence in S that is not a rotation of s.

For example, the sequence CACG CACG is a rotation of the sequence ACGC ACGC. The reason for the last condition in Definition 2.3 is to allow, for example, the use of both CACG CACG and ACGC ACGC as probes, with (a sufficient quantity of) the single complementary target GCGT GCGT G.

2.2. Identifying the Failed Step

In this section we define a property of a QC matrix $\mathcal{Q}$, called "separation," and establish that high separation is sufficient to identify any one failed step when up to d spots may show unreliable intensities.

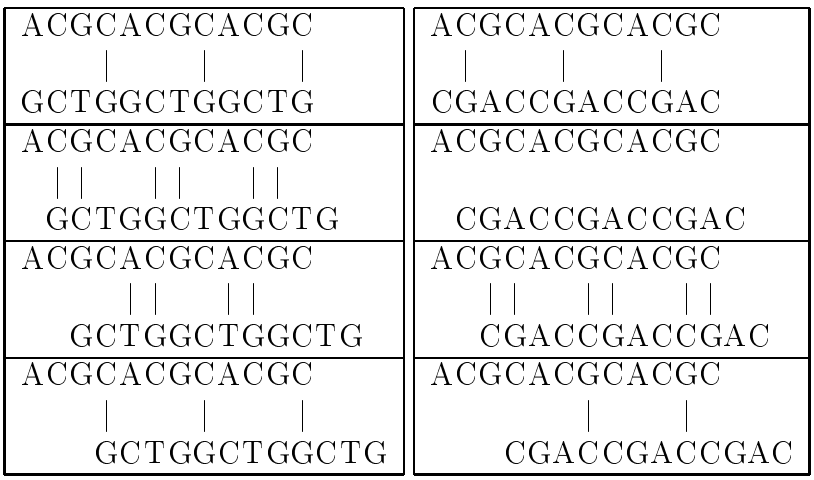

Figure 1: ACGC ACGC ACGC hybridizes poorly to GTCG GTCG GTCG, and also to its reverse complement CAGC CAGC CAGC.

2.2.1. Separation

Definition 2.4: Let $\mathcal{Q}$ be an $m \times n$ QC matrix with c distinct probes $\{q_k \mid 1 \leq k \leq c\}$. Let p_i be the probe at row i, $1 \leq i \leq m$. By convention, define $\mathcal{Q}_{i0} = 0$ for all $1 \leq i \leq m$. For any k with $1 \leq k \leq c$, and any pair $j \neq j'$ with $0 \leq j, j' \leq n$, let

$$\begin{aligned} D_{\mathcal{Q},k}(j,j') &= \#\{i \mid p_i = q_k \text{ and } \mathcal{Q}_{ij} \neq \mathcal{Q}_{ij'}\}, \\ L_{\mathcal{Q},k}(j,j') &= \#\{i \mid p_i = q_k \text{ and } (\mathcal{Q}_{ij} \neq 1 \text{ or } \mathcal{Q}_{ij'} \neq 0)\}, \\ R_{\mathcal{Q},k}(j,j') &= \#\{i \mid p_i = q_k \text{ and } (\mathcal{Q}_{ij} \neq 0 \text{ or } \mathcal{Q}_{ij'} \neq 1)\}. \end{aligned}$$

The subscript $\mathcal{Q}$ will be omitted when it is obvious from the context.

The *separation* of $\mathcal{Q}$ is defined to be:

$$sep(\mathcal{Q}) = \min_{\substack{0 \leq j,j' \leq n \\ j \neq j'}} \sum_{k=1}^{c} \min(D_k(j,j'), L_k(j,j'), R_k(j,j')). \tag{1}$$

Note that the convention concerning $\mathcal{Q}_{i0}$ is just a convenience for the purpose of defining separation. The QC matrix does not actually contain such a 0^{th} column.

The D_k portion of Definition 2.4 has an intuitive explanation based on the *Hamming distance* between two vectors, which is the number of corresponding positions at which the two vectors have unequal values. A large Hamming distance between columns j and j' of $\mathcal{Q}$ is necessary in order to be able to detect the difference between step j failing and step j' failing. Similarly, a large Hamming distance between column j of $\mathcal{Q}$ and the conventional column

0 (i.e., a large number of ones in column j) is necessary in order to detect the difference between step j failing and no step failing.

Note the similarity of the D_k portion of Definition 2.4 to error-correcting codes where, to correct up to d errors, it is sufficient that every pair of codewords (analogous to columns in the QC matrix) be separated by a distance of at least $2d+1$. The problem of identifying a failed step is like error correction rather than error detection, because we are interested in the identity of the failed step rather than simply whether any step failed. (In the latter case, the separation requirement of Theorem 2.6 below would be reduced from $2d+1$ to $d+1$.)

The L_k and R_k portions of Definition 2.4, which have no analog in error-correcting codes, capture the part of Assumption 4 from Section 2.1 that one may not be able to differentiate between all probe intensities high and all low, which is why the D_k portion alone is not sufficient. For example, suppose step j were used in *every* spot i. Even if no spot failed, if step j were to fail all spots would show equal (low) intensities. One might well not be able to distinguish this case from no step failing, in which all spots would also show equal (high) intensities. Definition 2.4 and Theorem 2.6 below guarantee that we will be able to distinguish these cases. Using a similar explanation to one given above, this portion implies that each column of $\mathcal{Q}$ has a large number of zeros.

2.2.2. Interpreting the Intensity Readings

The intensity vector $\mathcal{I}$ is a vector of m real numbers, giving an intensity reading for each of the m spots. We wish to interpret these real numbers as high ("0"), low ("1"), or unreadable ("?"). This interpretation is subject to reasonable constraints (given in Assumption 4 of Section 2.1, and Definition 2.5 below) that two similar intensities of the same probe are not interpreted as one high and one low, and two distant intensities of the same probe are not interpreted as both high or both low.

Let $\Phi(\mathcal{I}) \in \{0,1,?\}^m$ be such an interpretation of intensity vector $\mathcal{I} \in \Re^m$, where $\Re$ is the set of real numbers. The reason why high intensity corresponds to "0" and low to "1" is because the object is to use this interpretation vector to identify which column of the QC matrix it resembles most. When step j fails and none of the spots are faulty, the intensity vector interpretation $\Phi(\mathcal{I})$ one expects to see is exactly the 0-1 vector forming the j^{th} column of the QC matrix. In general up to d spots may be unreliable, so if step j fails, $\Phi(\mathcal{I})$ will equal the j^{th} column of the QC matrix with at most d exceptional positions. Note that not all the d unreliable spots need be interpreted as "?": some may be erroneously interpreted as high or low. The next definition formalizes this notion of interpretation.

Definition 2.5: Let $\mathcal{Q}$ be an $m \times n$ QC matrix with c distinct probes $\{q_k \mid 1 \le k \le c\}$. Let p_i be the probe at row i, $1 \le i \le m$. An *interpretation* $\Phi : \Re^m \to \{0,1,?\}^m$ of an intensity vector $\mathcal{I}$ satisfies the following for $1 \le i, i' \le m$ and $1 \le k \le c$.

- If $p_i = p_{i'} = q_k$ and $\Phi(\mathcal{I})_i = \Phi(\mathcal{I})_{i'} \in \{0,1\}$, then $|\mathcal{I}_i - \mathcal{I}_{i'}| \le \epsilon_k$. (In words, if spots i and i' have the same probe and are interpreted as both high or both low, then their intensities $\mathcal{I}_i$ and $\mathcal{I}_{i'}$ are similar.)

- If $p_i = p_{i'} = q_k$ and $\Phi(\mathcal{I})_i = 1$ and $\Phi(\mathcal{I})_{i'} = 0$, then $\mathcal{I}_{i'} - \mathcal{I}_i > \epsilon_k$. (In words, if spot i is interpreted as low and spot i' as high, then their intensities $\mathcal{I}_i$ and $\mathcal{I}_{i'}$ are not too similar.)

As an example, suppose there were twelve spots with probe q_k, their intensity readings were 2, 2, 3, 5, 6, 6, 6, 7, 7, 8, 8, 9, and $\epsilon_k = 2$. Then one possible interpretation (the one that minimizes the number of unreadable intensities) would interpret intensites 2–3 as low ("1"), 6–8 as high ("0"), and 5 and 9 as unreadable ("?").

2.2.3. Characterizing the Identity of the Failed Step

Theorem 2.6: Suppose $sep(\mathcal{Q}) \geq 2d + 1$ and $\mathcal{I}$ is the intensity vector of the m spots. Then, for $1 \leq j \leq n$, step j fails if and only if there is an interpretation Φ of $\mathcal{I}$ such that $\delta(\mathcal{Q}_{*j}, \Phi(\mathcal{I})) \leq d$, where δ is the Hamming distance and $\mathcal{Q}_{*j}$ is the j^{th} column of $\mathcal{Q}$. No step fails if and only if there is an interpretation Φ of $\mathcal{I}$ such that $\delta(0^m, \Phi(\mathcal{I})) \leq d$.

Proof: "Only if" clauses: Suppose step j fails. Then $\mathcal{I}_i$ is low if and only if $\mathcal{Q}_{ij} = 1$, with exceptions only for at most d spots that fail. (See Assumption 3 in Section 2.1.) For each probe q_k, choose the two intensities l_k and h_k, with $h_k - l_k > \epsilon_k$, to maximize the number of spots i such that

$$p_i = q_k \text{ and } ((\mathcal{Q}_{ij} = 1 \text{ and } \mathcal{I}_i \in [l_k - \epsilon_k, l_k]) \text{ or } (\mathcal{Q}_{ij} = 0 \text{ and } \mathcal{I}_i \in [h_k, h_k + \epsilon_k])).$$

Assign

$$\Phi(\mathcal{I})_i = \begin{cases} 1, & \text{if } p_i = q_k \text{ and } \mathcal{I}_i \in [l_k - \epsilon_k, l_k] \\ 0, & \text{if } p_i = q_k \text{ and } \mathcal{I}_i \in [h_k, h_k + \epsilon_k] \\ ?, & \text{if } p_i = q_k \text{ and } \mathcal{I}_i \notin [l_k - \epsilon_k, l_k] \cup [h_k, h_k + \epsilon_k] \end{cases}.$$

Because there are at most d exceptions to the condition that $\mathcal{I}_i$ is low if and only if $\mathcal{Q}_{ij} = 1$, $\delta(\mathcal{Q}_{*j}, \Phi(\mathcal{I})) \leq d$.

Suppose no step fails. Then $\mathcal{I}_i$ is high, with exceptions only for at most d spots that fail. (See Assumption 3 in Section 2.1.) For each probe q_k, choose the intensity h_k that maximizes the number of spots i such that

$$p_i = q_k \text{ and } \mathcal{I}_i \in [h_k, h_k + \epsilon_k].$$

Assign

$$\Phi(\mathcal{I})_i = \begin{cases} 0, & \text{if } p_i = q_k \text{ and } \mathcal{I}_i \in [h_k, h_k + \epsilon_k] \\ ?, & \text{if } p_i = q_k \text{ and } \mathcal{I}_i \notin [h_k, h_k + \epsilon_k] \end{cases}.$$

Because there are at most d exceptions to the condition that $\mathcal{I}_i$ is high, $\delta(0^m, \Phi(\mathcal{I})) \leq d$.

"If" clauses: Suppose j does not fail, meaning some $j' \neq j$ fails. (The case of no step failing is handled by the convention $j' = 0$ together with the convention from Definition 2.4

$\Phi(\mathcal{I})$		Q_{*j}		$Q_{*j'}$		$\Phi'(\mathcal{I})$
0		0	$\neq$	1	$\neq$	?
0	$\neq$	1	$\neq$	0		0
0		0		0		0
1		1	$\neq$	0	$\neq$	1
1		1		1		?
	e_k		$D_k(j,j')$		d_k	

Figure 2: Five sample spots with probe q_k, illustrating why every $Q_{ij} \neq Q_{ij'}$ contributes to $d_k + e_k$.

that $Q_{*0} = 0^m$.) By the "only if" clause above, there exists an interpretation Φ' such that $\delta(\mathcal{Q}_{*j'}, \Phi'(\mathcal{I})) \leq d$. Let Φ be any interpretation, and $e = \delta(\mathcal{Q}_{*j}, \Phi(\mathcal{I}))$. We will finish the proof by showing that $e > d$.

For each probe q_k, let

$$\begin{aligned} d_k &= \#\{i \mid p_i = q_k \text{ and } \Phi'(\mathcal{I})_i \neq \mathcal{Q}_{ij'}\}, \text{ and} \\ e_k &= \#\{i \mid p_i = q_k \text{ and } \Phi(\mathcal{I})_i \neq \mathcal{Q}_{ij}\}, \end{aligned}$$

so that

$$\begin{aligned} \sum_{k=1}^{c} d_k &\leq d \text{ and} \\ \sum_{k=1}^{c} e_k &= e. \end{aligned}$$

Consider the cases for how the two interpretations of the same vector $\mathcal{I}$ "line up" within some probe q_k:

Case 1: There exists an i such that $p_i = q_k$ and $\Phi(\mathcal{I})_i = \Phi'(\mathcal{I})_i \in \{0, 1\}$. Then for all i' such that $p_{i'} = q_k$,

$$\Phi(\mathcal{I})_{i'} = \Phi'(\mathcal{I})_{i'} \text{ or } \Phi(\mathcal{I})_{i'} = ? \text{ or } \Phi'(\mathcal{I})_{i'} = ?,$$

since $\Phi(\mathcal{I})_{i'} = 0$ and $\Phi'(\mathcal{I})_{i'} = 1$ would mean $|\mathcal{I}_i - \mathcal{I}_{i'}| \leq \epsilon_k$ and $(\mathcal{I}_{i'} - \mathcal{I}_i) > \epsilon_k$. But then every i for which $p_i = q_k$ and $\mathcal{Q}_{ij} \neq \mathcal{Q}_{ij'}$ contributes at least 1 to $d_k + e_k$, so that $d_k + e_k \geq D_k(j, j')$. The reason each such i contributes at least 1 is illustrated in Figure 2 and explained as follows. If $\Phi(\mathcal{I})_i = \Phi'(\mathcal{I})_i$ and $\mathcal{Q}_{ij} \neq \mathcal{Q}_{ij'}$, then either $\Phi'(\mathcal{I})_i \neq \mathcal{Q}_{ij'}$ or $\Phi(\mathcal{I})_i \neq \mathcal{Q}_{ij}$. Otherwise, one of the interpretations is ?, and that certainly differs from the corresponding $\mathcal{Q}_{i\cdot}$ entry.

Case 2: There exists an i such that $p_i = q_k$ and $\Phi(\mathcal{I})_i = 1$ and $\Phi'(\mathcal{I})_i = 0$. Then for all i' such that $p_{i'} = q_k$,

$$(\Phi(\mathcal{I})_{i'} = 1 \text{ and } \Phi'(\mathcal{I})_{i'} = 0) \text{ or } \Phi(\mathcal{I})_{i'} = ? \text{ or } \Phi'(\mathcal{I})_{i'} = ?.$$

But then every i for which $p_i = q_k$ and ($\mathcal{Q}_{ij} \neq 1$ or $\mathcal{Q}_{ij'} \neq 0$) contributes at least 1 to $d_k + e_k$, so that $d_k + e_k \geq L_k(j, j')$.

Case 3: There exists an i such that $p_i = q_k$ and $\Phi(\mathcal{I})_i = 0$ and $\Phi'(\mathcal{I})_i = 1$. Then $d_k + e_k \geq R_k(j, j')$, analogous to case 2.

Case 4: For all i, $p_i = q_k$ implies $\Phi(\mathcal{I})_i =?$ or $\Phi'(\mathcal{I})_i =?$. Then

$$d_k + e_k \geq \#\{i \mid p_i = q_k\} \geq D_k(j, j').$$

Combining the conclusions of these four cases,

$$d + e \geq \sum_{k=1}^{c}(d_k + e_k) \geq \sum_{k=1}^{c} \min(D_k(j, j'), L_k(j, j'), R_k(j, j')) \geq sep(\mathcal{Q}) \geq 2d + 1.$$

Hence $e > d$. □

2.2.4. An Algorithm that Identifies the Failed Step

Given spot failure tolerance d, an $m \times n$ QC matrix $\mathcal{Q}$ with $sep(\mathcal{Q}) \geq 2d + 1$, an intensity vector $\mathcal{I} \in \Re^m$, and an intensity window ϵ_k for each probe, $1 \leq k \leq c$, Theorem 2.6 can be applied to identify which protocol step, if any, has failed. An algorithm solving this problem must check if, for any j, $0 \leq j \leq n$, there exists an interpretation Φ such that $\delta(\mathcal{Q}_{*j}, \Phi(\mathcal{I})) \leq d$. If so, it returns the value j as the step that has failed. (As in Definition 2.4, Q_{*0} by convention is the vector 0^m, and a returned value of $j = 0$ corresponds to no step having failed.) If no such interpretation exists, the algorithm reports that more than one step must have failed, again by Theorem 2.6. Figure 3 describes an $O(mn + m \log m)$ time algorithm for performing this task.

A few definitions are prerequisites to the algorithm in Figure 3. Let vector $\hat{\mathcal{I}}$ be obtained by sorting the intensity vector $\mathcal{I}$, each set of rows corresponding to the same probe sorted separately in nondecreasing order. Let $\hat{\mathcal{Q}}$ be the result of performing the same permutation on the rows of $\mathcal{Q}$. Let $\mathcal{R}_k$ be the set of rows of $\hat{\mathcal{I}}$ corresponding to probe k, $1 \leq k \leq c$. Define $m_k = |\mathcal{R}_k|$, so that $\sum_{k=1}^{c} m_k = m$.

Given a column $\hat{\mathcal{Q}}_{*j}$ and the vector $\hat{\mathcal{I}}$, we define two lists, $\mathcal{L}_k$ and $\mathcal{H}_k$, for each probe k. The lists $\mathcal{L}_k$ and $\mathcal{H}_k$ are generated by partitioning the intensities in $\mathcal{R}_k$ according to whether the corresponding row of $\hat{\mathcal{Q}}_{*j}$ has a one or zero, respectively. Note that each $\mathcal{L}_k$ and $\mathcal{H}_k$, $1 \leq k \leq c$, is a sorted list of real-valued intensities. Letting $\mathcal{L}_k[i]$ be the i^{th} item in $\mathcal{L}_k$, define $\#\mathcal{L}_k[i]$ to be the number of indices $i' \leq i$ such that $\mathcal{L}_k[i] - \epsilon_k \leq \mathcal{L}_k[i'] \leq \mathcal{L}_k[i]$. Define $\#\mathcal{H}_k[i]$ similarly as the number of indices $i' \geq i$ such that $\mathcal{H}_k[i] \leq \mathcal{H}_k[i'] \leq \mathcal{H}_k[i] + \epsilon_k$.

The central idea behind the subroutine **FindLH** in Figure 3 is to find a "buddy" $l \in \mathcal{L}$ for each item $h \in \mathcal{H}$, such that $h - l > \epsilon_k$ and $\#l$ is maximized. The motivation for finding the buddy l of h is that, if $[h, h + \epsilon_k]$ were chosen as the range of high intensities, then $[l - \epsilon_k, l]$ would be the best choice for the range of low intensities, in the sense that it captures the greatest possible number of observed spot intensities. The list $\mathcal{B}$ defined within this subroutine maintains the index in $\mathcal{L}$ of the buddy for each item in $\mathcal{H}$, so that $\mathcal{H}[j]$ and

$\mathcal{L}[\mathcal{B}[j]]$ are buddies, $1 \leq j \leq |\mathcal{H}|$. Since both $\mathcal{L}$ and $\mathcal{H}$ are sorted lists, it must be the case that either $\mathcal{B}[j+1] = \mathcal{B}[j]$, or $\mathcal{B}[j+1] > i$, where i is any index satisfying $\mathcal{H}[j] - \mathcal{L}[i] > \epsilon_k$. Hence there is no need to backtrack in the list $\mathcal{L}$ as j increases through the list $\mathcal{H}$.

The list $\mathcal{B}$ is thus constructed via a single sweep through $\mathcal{L}$. The contribution from the "while" loop (line 4.1) over the execution of the entire "for" loop (line 4) is therefore $O(|\mathcal{L}| + |\mathcal{H}|)$. Since line 4.2 contributes $O(|\mathcal{H}|)$ to the cost of this "for" loop, the total cost of line 4 is $O(|\mathcal{L}| + |\mathcal{H}|)$. This dominates the $O(|\mathcal{H}|)$ contribution from line 5. By similarly using two monotonically advancing pointers, line 1 can be implemented in time $O(|\mathcal{L}| + |\mathcal{H}|)$. The total running time of **FindLH** is therefore $O(|\mathcal{L}| + |\mathcal{H}|)$.

This implies that the cost of line 2.1.2 in **DetectFaultyStep** is $O(|\mathcal{L}_k| + |\mathcal{H}_k|) = O(m_k)$. Since $\mathcal{L}_k$ and $\mathcal{H}_k$ can be constructed through a single sweep of $\mathcal{R}_k$, the cost of line 2.1.1 is also $O(m_k)$. The cost of the "for" loop in line 2.1 is then $\sum_{k=1}^{c} O(m_k) = O(m)$. The total running time of the "for" loop in line 2 is therefore $O(mn)$. Since each set of rows in $\mathcal{R}_k$ is sorted separately, the cost of sorting in line 1 is $\sum_{k=1}^{c} O(m_k \log m_k) = O(m \log m)$. Thus, **DetectFaultyStep** is an $O(mn + m \log m)$ time algorithm. Note that when a protocol step (say j) has failed, the "for" loop in line 2 will terminate early, so the actual cost of the algorithm would be $O(mj + m \log m)$.

2.3. Combining QC Matrices

The following theorem provides one simple way to combine QC matrices, and illustrates a tradeoff between the goals of maximizing separation and minimizing the number of spots.

Theorem 2.7: Suppose that $\mathcal{Q}_1$ is an $m_1 \times n$ QC matrix, and $\mathcal{Q}_2$ is an $m_2 \times n$ QC matrix. Then the union $\mathcal{Q}_1 + \mathcal{Q}_2$ of their rows has n steps, $m_1 + m_2$ spots, and

$$sep(\mathcal{Q}_1 + \mathcal{Q}_2) \geq sep(\mathcal{Q}_1) + sep(\mathcal{Q}_2).$$

Proof: For any $j \neq j'$ with $0 \leq j, j' \leq n$, let s be the separation of columns j and j' in $\mathcal{Q}_1 + \mathcal{Q}_2$, that is,

$$\begin{aligned} s &= \sum_{k=1}^{c} \min\left(D_{\mathcal{Q}_1+\mathcal{Q}_2,k}(j,j'), L_{\mathcal{Q}_1+\mathcal{Q}_2,k}(j,j'), R_{\mathcal{Q}_1+\mathcal{Q}_2,k}(j,j')\right) \\ &= \sum_{k=1}^{c} \min\Big(\begin{array}{l} D_{\mathcal{Q}_1,k}(j,j') + D_{\mathcal{Q}_2,k}(j,j'), \\ L_{\mathcal{Q}_1,k}(j,j') + L_{\mathcal{Q}_2,k}(j,j'), \\ R_{\mathcal{Q}_1,k}(j,j') + R_{\mathcal{Q}_2,k}(j,j')). \end{array} \end{aligned}$$

Using the inequality $\min(a+x, b+y, c+z) \geq \min(a,b,c) + \min(x,y,z)$,

$$\begin{aligned} s &\geq \sum_{k=1}^{c} \min(D_{\mathcal{Q}_1,k}(j,j'), L_{\mathcal{Q}_1,k}(j,j'), R_{\mathcal{Q}_1,k}(j,j')) \\ &\quad + \min(D_{\mathcal{Q}_2,k}(j,j'), L_{\mathcal{Q}_2,k}(j,j'), R_{\mathcal{Q}_2,k}(j,j')) \\ &\geq sep(\mathcal{Q}_1) + sep(\mathcal{Q}_2). \end{aligned}$$

DetectFaultyStep($\mathcal{Q}, \mathcal{I}, d, \epsilon_1, \ldots, \epsilon_c$)

begin

1. Sort $\mathcal{I}$, each set of positions corresponding to the same probe sorted separately in nondecreasing order. Induce the same row permutation on matrix $\mathcal{Q}$.
2. for j from 0 to n do
   /* for each j check if there is a Φ with $\delta(\Phi(\mathcal{I}), \mathcal{Q}_{*j}) \leq d$ */
   - 2.1 for k from 1 to c do
     - 2.1.1 Compute $\mathcal{L}_k$ and $\mathcal{H}_k$.
     - 2.1.2 $max_k = \textbf{FindLH}(\mathcal{L}_k, \mathcal{H}_k, \epsilon_k)$

     end for
   - 2.2 if $\sum_{k=1}^{c} max_k \geq m - d$ then return j
     /* $\sum_{k=1}^{c} max_k$ is the number of matches between $\Phi(\mathcal{I})$ and $\mathcal{Q}_{*j}$ */

   end for
3. return "Multiple steps failed"

end

FindLH($\mathcal{L}$, $\mathcal{H}$, ϵ)

/* Given sorted lists $\mathcal{L}$ and $\mathcal{H}$ find $l \in \mathcal{L}$ and $h \in \mathcal{H}$ such that $h - l > \epsilon$ and $\#h + \#l$ is maximized. Return this maximum value. */

begin

1. For each item $\mathcal{L}[i]$ in list $\mathcal{L}$, compute $\#\mathcal{L}[i]$. Do the same for $\mathcal{H}$.
2. $cur = 1$ /* initialize scan of list $\mathcal{L}$ */
3. /* $\mathcal{H}[1]$ may not have a buddy. */
   - 3.1 $\mathcal{B}[1] = null$
   - 3.2 $\#\mathcal{L}[\mathcal{B}[1]] = 0$
4. for j from 1 to $|\mathcal{H}|$ do
   /* for each element in $\mathcal{H}$, continue scan of $\mathcal{L}$ to find its "buddy" */
   - 4.1 while $\mathcal{H}[j] - \mathcal{L}[cur] > \epsilon$ and $cur \leq |\mathcal{L}|$ do
     - 4.1.1 if $\#\mathcal{L}[cur] > \#\mathcal{L}[\mathcal{B}[j]]$ then $\mathcal{B}[j] = cur$
     - 4.1.2 $cur = cur + 1$

     end while
   - 4.2 $\mathcal{B}[j+1] = \mathcal{B}[j]$
     /* carry over current buddy to initiate scan for the next item in $\mathcal{H}$ */

   end for
5. Compute max, the maximum of $\#\mathcal{H}[j] + \#\mathcal{L}[\mathcal{B}[j]]$ for all j, $1 \leq j \leq |\mathcal{H}|$.
6. **return** max

end

Figure 3: An $O(mn + m \log m)$ time algorithm for detecting a failed step.

Since any pair of distinct columns of $\mathcal{Q}_1 + \mathcal{Q}_2$ has separation at least $sep(\mathcal{Q}_1) + sep(\mathcal{Q}_2)$, the theorem follows. □

2.4. Precise Problem Formulation

We are now in a position to state the precise design problem we solve. The array manufacturer specifies as inputs the number n of steps, the protocol, and the length k of each probe. The QC design problem is to construct an $m \times n$ QC matrix $\mathcal{Q}$ with k ones per row such that the number m of spots is small and $sep(\mathcal{Q})$ is large. Furthermore, the set of c distinct probes hybridizes poorly, according to Definition 2.3. In our designs, we never use more than $c = 8$ distinct probes.

As discussed at the beginning of Section 2, the manufacturer uses the QC matrix $\mathcal{Q}$ by manufacturing the probes $p_1, p_2, \ldots, p_m$ onto m reserved spots, and hybridizing with the complementary fluorescent targets. The resulting intensity vector $\mathcal{I}$ is then used along with $\mathcal{Q}$ to identify the failed step, if any, using the algorithm of Section 2.2.4.

One cannot expect to optimize both the objective functions m and $sep(\mathcal{Q})$ in a single QC matrix. For instance, Theorem 2.7 says that duplicating the spots of $\mathcal{Q}$ simultaneously doubles m and $sep(\mathcal{Q})$. Instead, in Section 4 we will construct a variety of QC matrices $\mathcal{Q}$ that offer the manufacturer a spectrum of choices for m and $sep(\mathcal{Q})$.

The following theorem demonstrates a lower bound on the number of spots, in terms of the other parameters.

Theorem 2.8: For any $m \times n$ QC matrix $\mathcal{Q}$ with k ones per row,

$$m \geq \frac{sep(\mathcal{Q})}{k} \cdot n.$$

Proof: The number of ones in $\mathcal{Q}$ is mk. By Definition 2.4 (the D_k portion with $j = 0$), the number of ones per column is at least $sep(\mathcal{Q})$. □

One should also not expect to find an algorithm that, given arbitrary values n and m, computes an $m \times n$ QC matrix $\mathcal{Q}$ that maximizes $sep(\mathcal{Q})$. This is likely to be infeasible at the present time, because even the existence of certain combinatorial designs (such as a Hadamard matrix of order $4t$, which is equivalent to a $(4t-1) \times (4t-1)$ QC matrix $\mathcal{Q}$ with $sep(\mathcal{Q}) = 2t - 1$) is a long-standing open problem [5]. To see the equivalence, suppose there were such a matrix $\mathcal{Q}$. Since $sep(\mathcal{Q}) = 2t - 1$, the number of ones per column is between $2t - 1$ and $2t$. Suppose the number of ones per row (and hence column) is exactly $2t - 1$. (If $2t$, consider the complement of $\mathcal{Q}$ instead.) Since $sep(\mathcal{Q}) = 2t - 1$, the Hamming distance between any two columns is at least $2t - 1$, so any two columns both have ones in *at most* $t - 1$ rows. But the total number of unordered pairs of ones in the same row, summed over all rows, is $(4t-1)(2t-1)(t-1)$. Hence, the *average* pair of columns also both have ones in $t - 1$ rows, so that *every* pair of columns both have ones in exactly $t - 1$ rows. That makes $\mathcal{Q}$ the incidence matrix of a 2-design (see Definition 3.2) with parameters

$(4t-1, 4t-1, 2t-1, 2t-1, t-1)$, which is equivalent to a Hadamard matrix of order $4t$ [5, Construction 24.7].

3. A Combinatorial Design Approach

We will assume for the moment that the protocol is $(\mathrm{ACGT})^{n/4}$, generalizing to other protocols in Section 5.

3.1. Relationship to Error-Correcting Codes

A good QC matrix $\mathcal{Q}$ has many of the properties of a good error-correcting code, which is a type of combinatorial design: if one thinks of the columns of $\mathcal{Q}$ as binary codewords, then one part of Definition 2.4 (the constraint on D_k) guarantees that the Hamming distance between any pair of codewords is at least $sep(\mathcal{Q})$. However, good QC matrices have many more constraints that make their design more complicated than that of error-correcting codes:

1. The order of the columns, which would not matter in a code, is critical in a QC matrix. In particular, the ones in the rows must "spell out" a small number c of probes according to Definition 2.1. Furthermore, these probes must hybridize poorly according to Definition 2.3.

2. Each row must contain the same number k of ones, which has no analogy in codes. This is to enforce the constraint that each probe has the same length k.

3. Another consequence of Definition 2.4 (when $j = 0$) is that each column must contain between $sep(\mathcal{Q})$ and $m - sep(\mathcal{Q})$ ones.

4. The constraints of Definition 2.4 on L_k and R_k have no analogy in codes.

We solve this design problem using a hierarchical approach. In Section 3.2 we introduce a new type of combinatorial design. This is a "balanced" version of binary codes that takes care of items 2 and 3 from the list above, and we show how to construct such a balanced code C. We then substitute a small appropriate matrix, called a "QC block", for each 1 in C, and an equal size matrix of zeros for each 0 in C. The QC block is designed so that the resulting cross product satisfies all four properties of the list above. An example of an 8×8 QC block is shown in Figure 4. The definition of QC blocks is given in Section 3.3.

3.2. Balanced Codes

Definition 3.1: A *balanced binary code* with parameters $(v, b, r_{\min}, r_{\max}, k, d_{\min})$ is a $b \times v$ 0-1 matrix with the following properties:

1. Every row contains exactly k ones.

2. The minimum number of ones in any column is $r_{\min}$, and the maximum is $r_{\max}$.

3. The minimum Hamming distance between any pair of columns is $d_{\min}$.

A subset of the codewords from certain types of error-correcting codes, such as Hadamard codes and quadratic residue codes [16], form balanced codes. (To achieve k ones per row, choose $k/2$ pairs of complementary codewords.) However, our major source of balanced code constructions comes from 2-designs:

Definition 3.2 (Colbourn and Dinitz [4]): A *2-design* with parameters (v, b, r, k, λ) is a $b \times v$ 0-1 matrix D with the following properties:

1. Every row contains exactly k ones.

2. Every column contains exactly r ones.

3. For every pair j, j' of distinct columns, there are exactly λ rows i such that $D_{i,j} = D_{i,j'} = 1$.

Proposition 3.3: Any 2-design with parameters (v, b, r, k, λ) is a balanced code with parameters $(v, b, r, r, k, 2(r - \lambda))$.

Proof: Since there are λ rows in which columns j and j' each contain 1, there are $r - \lambda$ rows in which column j contains 1 and column j' contains 0, and another $r - \lambda$ rows in which column j contains 0 and column j' contains 1. □

While most of our balanced codes come from 2-designs, the latter are more stringent than we need: every column contains *exactly* r ones, and the Hamming distance between any pair of columns is *exactly* $2(r - \lambda)$. Constructing less stringent balanced codes would lead to a richer collection of QC matrices.

Another source of balanced codes comes from the following product construction.

Theorem 3.4: Let C' be a balanced code with parameters $(v', b', r'_{\min}, r'_{\max}, k', d'_{\min})$ and C be a balanced code with parameters $(v, b, r_{\min}, r_{\max}, k, d_{\min})$. Then there is a balanced code $C' \times C$ with parameters

$$(v'v, b'b, r'_{\min}r_{\min}, r'_{\max}r_{\max}, k'k, \min(d'_{\min}r_{\min}, d_{\min}r'_{\min})).$$

Proof: Replace every one in C' by a copy of C, and every zero in C' by a $b \times v$ matrix of zeros. (See the leftmost twelve columns of Figure 10(d) for an example.) If two columns j and j' of this product both lie in the same column of C', their Hamming distance is at least $d_{\min}r'_{\min}$. If they lie in different columns of C', their Hamming distance is at least $d'_{\min}r_{\min}$. □

A	C	G			C		
			T	A		G	T
	C			A	C	G	
A		G	T				T
A	C			A			T
		G	T		C	G	
A			T	A	C		
	C	G				G	T

Figure 4: An 8×8 QC block with parameters (8,8,4,4,4,4). For ease of visualization, the figure shows blanks instead of zeros, and the appropriate nucleotide from the protocol instead of ones.

3.3. QC Blocks

Balanced codes do not capture the notion of poor hybridization. A "QC block" is just a balanced code with an additional hybridization constraint:

Definition 3.5: A *QC block* for a protocol $\mathcal{P}$ is a $b \times v$ balanced code in which the b probes $p_1, p_2, \ldots, p_b$ (using the length v prefix of $\mathcal{P}$ as the protocol in Definition 2.1) are all distinct and, for every integer s, the set $\{p_1^s, p_2^s, \ldots, p_b^s\}$ hybridizes poorly (see Definition 2.3).

An example of an 8×8 QC block with parameters $(8, 8, 4, 4, 4, 4)$ is given in Figure 4. Its eight poorly hybridizing probes are $(\text{ACGC})^s$, $(\text{TAGT})^s$, $(\text{CACG})^s$, $(\text{AGTT})^s$, $(\text{ACAT})^s$, $(\text{GTCG})^s$, $(\text{ATAC})^s$, and $(\text{CGGT})^s$. Its four complementary targets are GCGT ... GCGT G, AACT ... AACT A, ATGT ... ATGT AT, and CGAC ... CGAC CG.

3.4. Product Construction of QC Matrices

The method we will use to construct good QC matrices is to apply the product construction of Theorem 3.4, with C' a balanced code and C a QC block. Figure 5 shows an example, where C' consists of ten codewords from the 8-Hadamard code [16], and C is the QC block of Figure 4.

If the parameters of C' are $(v', b', r'_{\min}, r'_{\max}, k', d'_{\min})$ and the parameters of C are $(v, b, r_{\min}, r_{\max}, k, d_{\min})$, then the QC matrix $C' \times C$ will have $v'v$ steps, $b'b$ spots, and b distinct probes, each of length $k'k$ and each occurring at b' distinct spots. More specifically, if $p_1, p_2, \ldots, p_b$ are the distinct probes of C, then $p_1^{k'}, p_2^{k'}, \ldots, p_b^{k'}$ are the distinct probes of $C' \times C$. By Definition 3.5, this set of distinct probes hybridizes poorly.

What remains is to determine $sep(C' \times C)$, in order to be able to apply Theorem 2.6.

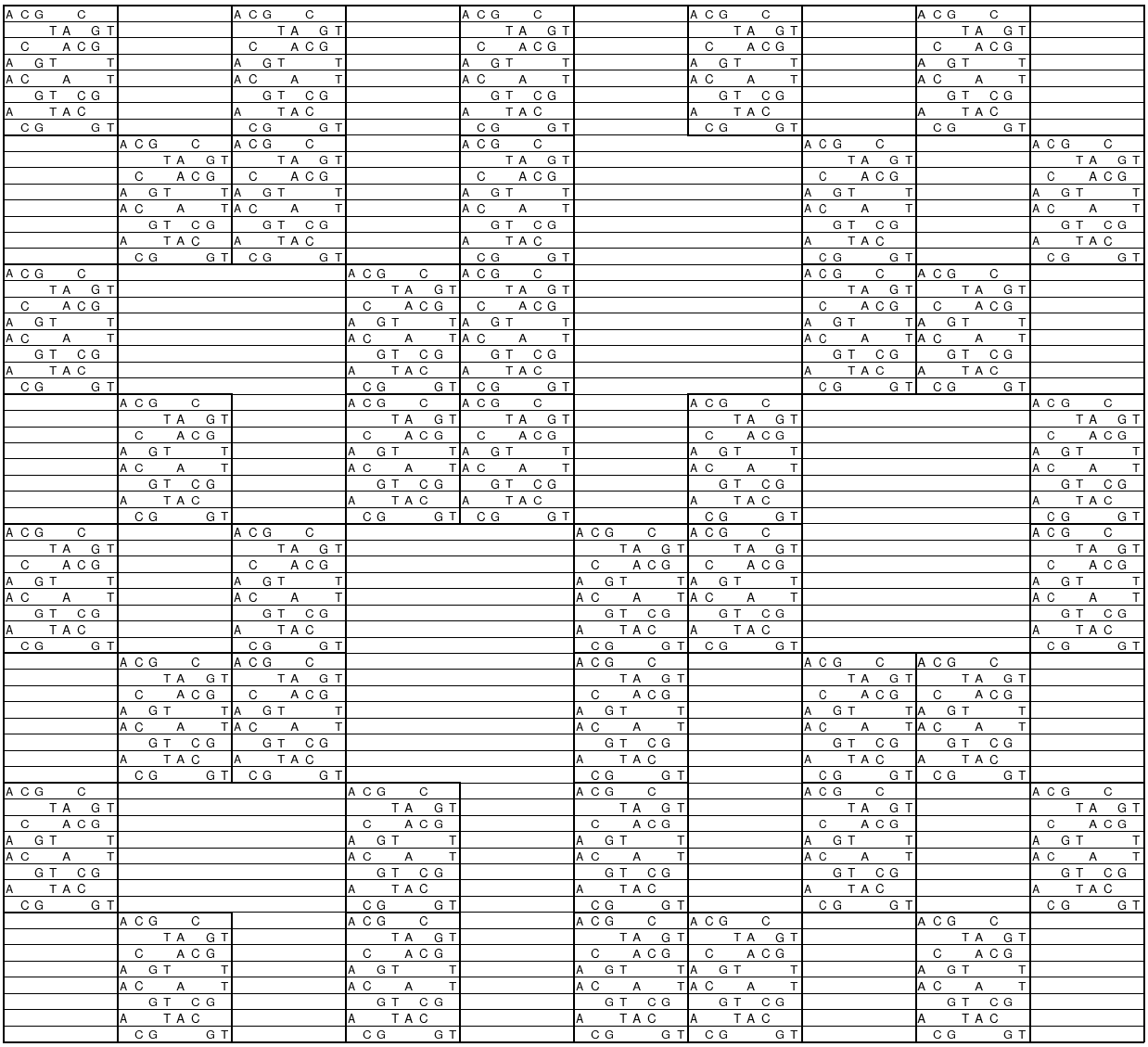

Figure 5: The product of 10 codewords from the 8-Hadamard code and the 8×8 QC block of Figure 4, resulting in a 64×80 QC matrix $\mathcal{Q}$ with minimum separation $sep(\mathcal{Q}) = 16$.

Theorem 3.6: If C' is a balanced code with parameters $(v', b', r'_{\min}, r'_{\max}, k', d'_{\min})$ and C is a QC block with parameters $(v, b, r_{\min}, r_{\max}, k, d_{\min})$, then

$$\begin{aligned} sep(C' \times C) = \min(\ & d'_{\min} r_{\min}, \\ & r'_{\min} \min(r_{\min}, d_{\min}), \\ & (b' - r'_{\max}) \min(r_{\min}, d_{\min})\ ). \end{aligned}$$

As an example, if C is the 8×8 QC block of Figure 4, then

$$sep(C' \times C) = 4 \min(d'_{\min}, r'_{\min}, b' - r'_{\max}).$$

Proof: We will identify any column j of the product $C' \times C$, $1 \le j \le v'v$, by the pair (g, h), where $1 \le g \le v'$, $1 \le h \le v$, and column j is the result of replacing every one in column g of C' by column h of C, and replacing every zero by the vector 0^b. The proof proceeds by examining the four possible cases for the pair j, j' of columns of $C' \times C$ in Definition 2.4, and computing the separation of these columns. (The reader may find it helpful to identify an example of each case in Figure 5 while reading the proof for that case.) Note for all four cases that the $b'b$ spots of $C' \times C$ are partitioned into b distinct probes, corresponding to the b rows of C, as described above.

Case 1: $j = (g, h)$ and $j' = 0$. (Recall the convention from Definition 2.4 that column 0 is the vector $0^{b'b}$.) Suppose column g of C' has r' ones, where $r'_{\min} \le r' \le r'_{\max}$, and column h of C has r ones, where $r_{\min} \le r$. For any probe q_k corresponding to a zero in column h of C, $D_k(j, j') = 0$, so q_k contributes 0 to the separation of j and j' in Equation (1) of Definition 2.4. For the r distinct probes q_k corresponding to the ones in column h of C, $D_k(j, j') = r'$, $L_k(j, j') = b' - r'$, and $R_k(j, j') = b'$. Thus,

$$\sum_{k=1}^{b} \min(D_k(j, j'), L_k(j, j'), R_k(j, j')) = r \min(r', b' - r') \ge r_{\min} \cdot \min(r'_{\min}, b' - r'_{\max}),$$

with equality when $r = r_{\min}$, and either $r' = r'_{\min}$ or $b' - r' = b' - r'_{\max}$, whichever is less.

Case 2: $j = (g, h)$ and $j' = (g, h')$, with $h \ne h'$. Suppose column g of C' has r' ones, where $r'_{\min} \le r' \le r'_{\max}$, and columns h and h' of C differ in d positions, where $d_{\min} \le d$. For any probe q_k corresponding to a row of C in which columns h and h' do not differ, $D_k(j, j') = 0$, so q_k contributes 0 to the separation of j and j' in Equation (1). For the d distinct probes q_k corresponding to the rows of C in which columns h and h' do differ, $D_k(j, j') = r'$, one of $L_k(j, j')$ and $R_k(j, j')$ is $b' - r'$, and the other is b'. Thus,

$$\sum_{k=1}^{b} \min(D_k(j, j'), L_k(j, j'), R_k(j, j')) = d \min(r', b' - r') \ge d_{\min} \cdot \min(r'_{\min}, b' - r'_{\max}),$$

with equality when $d = d_{\min}$, and either $r' = r'_{\min}$ or $b' - r' = b' - r'_{\max}$, whichever is less.

Case 3: $j = (g, h)$ and $j' = (g', h)$, with $g \ne g'$. Suppose column h of C has r ones, where $r_{\min} \le r$, and suppose there are e' rows i in C' for which $C'_{ig} = 1$ and $C'_{ig'} = 0$, and f' rows i in C' for which $C'_{ig} = 0$ and $C'_{ig'} = 1$, where $d'_{\min} \le e' + f'$ and $e', f' \le r'_{\max}$. For any probe

q_k corresponding to a row of C in which column h has a zero, $D_k(j, j') = 0$, so q_k contributes 0 to the separation of j and j' in Equation (1). For the r distinct probes q_k corresponding to the ones in column h, $D_k(j, j') = e' + f'$, $L_k(j, j') = b' - e'$, and $R_k(j, j') = b' - f'$. Thus,

$$\sum_{k=1}^{b} \min(D_k(j, j'), L_k(j, j'), R_k(j, j')) = r \min(e' + f', b' - e', b' - f') \geq r_{\min} \cdot \min(d'_{\min}, b' - r'_{\max}),$$

with equality possible when $r = r_{\min}$ and $e' + f' = d'_{\min}$.

Case 4: $j = (g, h)$ and $j' = (g', h')$, with $g \neq g'$ and $h \neq h'$. Suppose columns h and h' of C differ in d positions, where $d_{\min} \leq d$. For a probe q_k corresponding to a row i of C in which columns h and h' differ, assume without loss of generality that $C_{ih} = 1$ and $C_{ih'} = 0$. Suppose that column g of C' has r' ones, where $r'_{\min} \leq r' \leq r'_{\max}$. Then $D_k(j, j') = r'$, $L_k(j, j') = b' - r'$, and $R_k(j, j') = b'$. Thus,

$$\sum_{k=1}^{b} \min(D_k(j, j'), L_k(j, j'), R_k(j, j')) \geq d \min(r', b' - r') \geq d_{\min} \cdot \min(r'_{\min}, b' - r'_{\max}).$$

□

4. Results: Achieved QC Matrices

Table 1 shows some of the QC matrices achievable by using the product construction of Section 3.4. Each row of the table describes a QC matrix that is the product of the balanced code specified in the last column and the QC block specified in the penultimate column. For example, the QC matrix shown in Figure 5 corresponds to the row of the table with 80 steps and 64 spots.

Table 1 focuses on ranges of parameters comparable to those of Hubbell and Pevzner [8], namely between 60 and 132 protocol steps, probe lengths between 16 and 20, and fewer than 400 spots. (These parameters are given in columns 1–3 of the table.) The separations in column 4 of the table are calculated using Theorem 3.6. For each fixed number of steps (column 1), the table offers a small spectrum of designs to suit the manufacturer's spot budget and spot failure tolerance (columns 3–4). Arbitrary linear combinations of these designs can be formed according to Theorem 2.7, to provide a broader spectrum of choices.

The manufacturer uses Table 1 to look up the QC matrix $\mathcal{Q}$ for the appropriate choice of parameters in the first four columns of the table, where the "sep" parameter is chosen to be greater than twice the number of faulty spots the manufacturer is willing to tolerate. The QC matrix $\mathcal{Q}$ is used to manufacture the quality control probes onto reserved spots, which are hybridized with complementary fluorescent targets. The resulting intensity vector $\mathcal{I}$ is then used along with $\mathcal{Q}$ to identify the failed step, if any, using the algorithm of Section 2.2.4.

The 8×8 QC block has already been presented in Figure 4. The 6×12, 6×8, and 4×4 QC blocks are given in Figures 6, 7, and 8, respectively. They each use a subset of the probes used by the 8×8 QC block.

Table 1: Some basic QC matrices achievable by the product construction of Section 3.4. The second column shows the probe length. The last two columns show the QC block and balanced code whose product yields the QC matrix. In the last column, a list of 5 parameters indicates a 2-design (Definition 3.2), "×" indicates a product code (Theorem 3.4), "$+i$" indicates the addition of i extra columns that maintain the balanced code properties (see Appendix C), and GF(q) refers to balanced codes derived from polynomials over finite fields (see Appendix D). The 2-designs referenced in the last column can be found in the compendium of Mathon and Rosa [13], and the error-correcting codes in the survey of Tonchev [16]. For the latter, the balanced code employs codewords in complementary pairs, and the number of codewords used is n/v, where n is the number of steps and v the number of columns in the QC block.

steps	leng	spots	sep	spots/sep	steps/leng	block	balanced code
60	16	60	14	4.29	3.75	4x4	(15,15,8,8,4)
60	18	140	28	5.00	3.33	4x4	(15,35,21,9,12)
60	20	168	28	6.00	3.00	4x4	(15,42,28,10,18)
64	16	42	6	7.00	4.00	6x8	7-Hadamard code
64	16	44	10	4.40	4.00	4x4	11-Hadamard code
64	16	48	12	4.00	4.00	4x4	12-Hadamard code
64	16	64	16	4.00	4.00	8x8	8-Hadamard code
64	20	64	12	5.33	3.20	4x4	(16,16,10,10,6)
64	16	120	30	4.00	4.00	4x4	(16,30,15,8,7)
64	18	320	70	4.57	3.56	4x4	(16,80,45,9,24)
68	16	136	32	4.25	4.25	4x4	(17,34,16,8,7)
72	18	44	10	4.40	4.00	4x4	11-Hadamard code
72	18	48	12	4.00	4.00	4x4	12-Hadamard code
72	16	54	8	6.75	4.50	6x8	(3,3,2,2,1) x (3,3,2,2,1)
72	16	72	16	4.50	4.50	8x8	(3,3,2,2,1) x (3,3,2,2,1)
72	20	84	12	7.00	3.60	6x8	(7,14,8,4,4)+2
72	20	108	16	6.75	3.60	6x8	(9,18,10,5,5)
72	20	112	24	4.67	3.60	8x8	(7,14,8,4,4)+2
72	18	136	34	4.00	4.00	4x4	(18,34,17,9,8)
72	20	144	32	4.50	3.60	8x8	(9,18,10,5,5)
76	18	76	18	4.22	4.22	4x4	(19,19,9,9,4)
76	20	76	18	4.22	3.80	4x4	(19,19,10,10,5)
80	20	44	10	4.40	4.00	4x4	11-Hadamard code
80	20	48	12	4.00	4.00	4x4	12-Hadamard code
80	20	64	16	4.00	4.00	8x8	8-Hadamard code
80	16	90	12	7.50	5.00	6x8	(10,15,6,4,2)
80	16	120	24	5.00	5.00	8x8	(10,15,6,4,2)
80	20	152	38	4.00	4.00	4x4	(20,38,19,10,9)
80	18	160	24	6.67	4.44	4x4	(4,4,3,3,2) x (5,10,6,3,3)
80	16	380	76	5.00	5.00	4x4	(20,95,38,8,14)
84	16	42	6	7.00	5.25	6x12	(7,7,4,4,2)
84	20	126	12	10.50	4.20	6x12	(7,21,15,5,10)
84	18	140	30	4.67	4.67	4x4	(21,35,15,9,6)
84	20	168	40	4.20	4.20	4x4	(21,42,20,10,9)
88	20	66	10	6.60	4.40	6x8	(11,11,5,5,2)
88	20	84	12	7.00	4.40	6x8	(7,14,6,3,2)+4
88	20	88	20	4.40	4.40	8x8	(11,11,5,5,2)
88	20	112	24	4.67	4.40	8x8	(7,14,6,3,2)+4
88	20	144	32	4.50	4.40	8x8	(9,18,8,4,3)+2
88	16	264	48	5.50	5.50	4x4	(22,66,24,8,8)
88	20	308	70	4.40	4.40	4x4	(22,77,35,10,15)
96	16	42	6	7.00	6.00	6x12	7-Hadamard code
96	16	48	8	6.00	6.00	6x12	8-Hadamard code
96	18	48	4	12.00	5.33	4x4	(4,4,3,3,2) x 3-Hadamard
96	18	64	8	8.00	5.33	4x4	(4,4,3,3,2) x 4-Hadamard
96	16	84	14	6.00	6.00	6x12	(8,14,7,4,3)
96	20	90	12	7.50	4.80	6x8	(10,15,6,4,2)+2
96	20	120	24	5.00	4.80	8x8	(10,15,6,4,2)+2
96	18	160	20	8.00	5.33	4x4	(4,4,3,3,2) x (6,10,5,3,2)
96	16	276	46	6.00	6.00	4x4	(24,69,23,8,7)
100	18	100	18	5.56	5.56	4x4	(25,25,9,9,3)
100	20	160	32	5.00	5.00	4x4	(25,40,16,10,6)
100	16	300	48	6.25	6.25	4x4	(25,75,24,8,7)
104	16	78	8	9.75	6.50	6x8	(13,13,4,4,1)
104	16	104	16	6.50	6.50	8x8	(13,13,4,4,1)
104	20	234	30	7.80	5.20	6x8	(13,39,15,5,5)
104	20	260	50	5.20	5.20	4x4	(26,65,25,10,9)
104	20	312	60	5.20	5.20	8x8	(13,39,15,5,5)
108	18	36	4	9.00	6.00	4x4	degree 2 over GF(3)
108	16	54	8	6.75	6.75	6x12	(3,3,2,2,1) x (3,3,2,2,1)
108	20	84	12	7.00	5.40	6x12	(7,14,8,4,4)+2
108	20	108	16	6.75	5.40	6x12	(9,18,10,5,5)
108	18	156	26	6.00	6.00	4x4	(27,39,13,9,4)
108	20	216	40	5.40	5.40	4x4	(3,3,2,2,1) x (9,18,10,5,5)
112	20	108	12	9.00	5.60	6x8	(3,3,2,2,1) x (4,6,3,2,1) + 2
112	18	112	12	9.33	6.22	4x4	(4,4,3,3,2) x (7,7,3,3,1)
112	20	144	24	6.00	5.60	8x8	(3,3,2,2,1) x (4,6,3,2,1) + 2
112	20	168	30	5.60	5.60	4x4	(28,42,15,10,5)
112	18	336	54	6.22	6.22	4x4	(28,84,27,9,8)
116	16	232	32	7.25	7.25	4x4	(29,58,16,8,4)
120	20	42	6	7.00	6.00	6x12	7-Hadamard code
120	20	48	8	6.00	6.00	6x12	8-Hadamard code
120	20	66	10	6.60	6.00	6x12	11-quadratic residue code
120	16	90	12	7.50	7.50	6x12	(10,15,6,4,2)
120	20	108	18	6.00	6.00	6x12	(10,18,9,5,4)
120	20	168	28	6.00	6.00	6x12	(8,28,14,4,6)+2
120	16	240	32	7.50	7.50	8x8	(3,3,2,2,1) x (5,10,4,2,1)
120	20	348	58	6.00	6.00	4x4	(30,87,29,10,9)
124	20	124	20	6.20	6.20	4x4	(31,31,10,10,3)
128	16	96	8	12.00	8.00	6x8	degree 1 over GF(4)
128	16	120	10	12.00	8.00	6x8	(16,20,5,4,1)
128	16	128	16	8.00	8.00	8x8	degree 1 over GF(4)
128	16	160	20	8.00	8.00	8x8	(16,20,5,4,1)
128	16	192	24	8.00	8.00	8x8	(4,6,3,2,1) x 4-Hadamard
128	20	288	30	9.60	6.40	6x8	(16,48,15,5,4)
128	20	384	60	6.40	6.40	8x8	(16,48,15,5,4)
132	20	66	10	6.60	6.60	6x12	(11,11,5,5,2)
132	20	84	12	7.00	6.60	6x12	(7,14,6,3,2)+4
132	20	108	16	6.75	6.60	6x12	(9,18,8,4,3)+2
132	18	176	24	7.33	7.33	4x4	(33,44,12,9,3)
132	16	330	40	8.25	8.25	6x12	(11,55,20,4,6)

A	C	G							C		
			T					A		G	T
	C			A	C					G	
A						G	T				T
		G	T		C	G					
				A			T	A	C		

Figure 6: A 6×12 QC block with parameters (12,6,2,2,4,2).

A	C	G			C		
			T	A		G	T
A	C			A			T
		G	T		C	G	
A			T	A	C		
	C	G				G	T

Figure 7: A 6×8 QC block with parameters (8,6,3,3,4,2).

QC matrices involving 4×4 QC blocks are constructed in a slightly different manner from the others. These are actually a pair of mated blocks, as shown in Figure 8. When forming the product with a balanced code C', these two mates are alternately substituted for the ones in any given row of C'. An example of this product construction is shown in Figure 9, in which C' is a 2-design with parameters (19,19,9,9,4) [13, Table 1.26]. Because the proof of Theorem 3.6 relies on the substitution of a single QC block for all the ones in C', that theorem is not general enough to provide the separation values for QC matrices constructed from the mated 4×4 QC blocks. The same result does in fact hold for such QC matrices, and the proof is given in Appendix B.

QC matrices using the 4×4 QC blocks require only two targets, ATGT ... ATGT AT and CGAC ... CGAC CG. There are two entries in Table 1 that appear as though they could be derived by combining earlier entries via Theorem 2.7: (1) the 276×96 QC matrix has the same probe length, number of spots, and separation as four copies of the 48×96 plus one copy of the 84×96, and (2) the 348×120 the same as five copies of the 48×120 plus one copy of the 108×120. However, in both cases the single larger QC matrix, constructed via the 4×4 QC block, requires only two targets, whereas the combined equivalent QC matrix requires four targets.

By Theorem 2.8, column 5 of Table 1 is no less than column 6. When they are equal, the QC matrix is optimal, in the sense that it uses the fewest possible number of spots for its separation.

An open problem in combinatorial design theory [13] that has bearing on our practical range of parameters is the existence of a 2-design with parameters $(22, 33, 12, 8, 4)$ which, together with the 4×4 QC block, would yield a QC matrix with 88 steps, probe length 16,

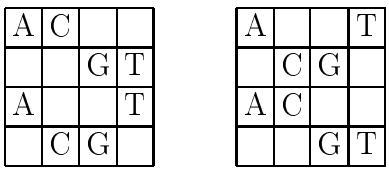

Figure 8: A pair of 4×4 QC blocks each with parameters (4,4,2,2,2,2).

132 spots, and separation 24.

5. Covering Protocols Other Than ACGT ACGT ...

In this section we show that all the achievable parameter settings for the protocol $\mathcal{P} = (\text{ACGT})^{n/4}$ obtained in Section 4 can also be achieved for any protocol $\mathcal{P}' = [\pi(\text{ACGT})]^{n/4}$, where π is any permutation. To obtain a QC matrix for any protocol $\mathcal{P}'$ we can start with a QC matrix $\mathcal{Q}$ with its columns labelled according to the steps in protocol $\mathcal{P}$, and relabel the columns according to the steps in $\mathcal{P}'$. The resulting QC matrix $\mathcal{Q}'$ certainly has the same parameter values as $\mathcal{Q}$. The only possible impediment to this being a valid QC matrix for $\mathcal{P}'$ is that the probe set associated with $\mathcal{Q}'$ may no longer hybridize poorly according to Definition 2.3. We overcome this impediment and exhibit transformations on valid QC matrices that preserve the validity of the resulting probe set for all 24 protocols $[\pi(\text{ACGT})]^{n/4}$. Of these, 16 permutations are obtained via general transformations that operate on the total QC design, and would apply to any valid QC matrix with periodic probes. The remaining 8 are specific to the QC blocks used in the product designs described in Section 4.

5.1. Rotations

All our probes are periodic, with period 4. Given such a periodic probe set $\mathcal{S}$, the probe set obtained by rotating some of the probes some number of positions has the same hybridization behavior as $\mathcal{S}$ with respect to Definition 2.3. Thus, a poorly hybridizing probe set remains so under such rotations.

Given a QC matrix $\mathcal{Q}$ for $(\text{ACGT})^{n/4}$, we can rotate the columns of $\mathcal{Q}$ right one position (the n^{th} column becomes the 1^{st}, and the i^{th} column becomes the $(i+1)^{st}$, $1 \leq i \leq n-1$) to obtain a QC matrix of identical parameters as $\mathcal{Q}$, for the protocol $(\text{TACG})^{n/4}$. Similarly, QC matrices of identical parameters for the protocols $(\text{GTAC})^{n/4}$ and $(\text{CGTA})^{n/4}$ can be obtained from $\mathcal{Q}$ by rotating the columns right two and three positions, respectively.

Note that the resulting QC matrices are not necessarily product designs as constructed in Section 3.4. Note also that similar rotations could be performed on valid QC matrices for any protocol, not just $(\text{ACGT})^{n/4}$. In the following sections, we will do exactly that.

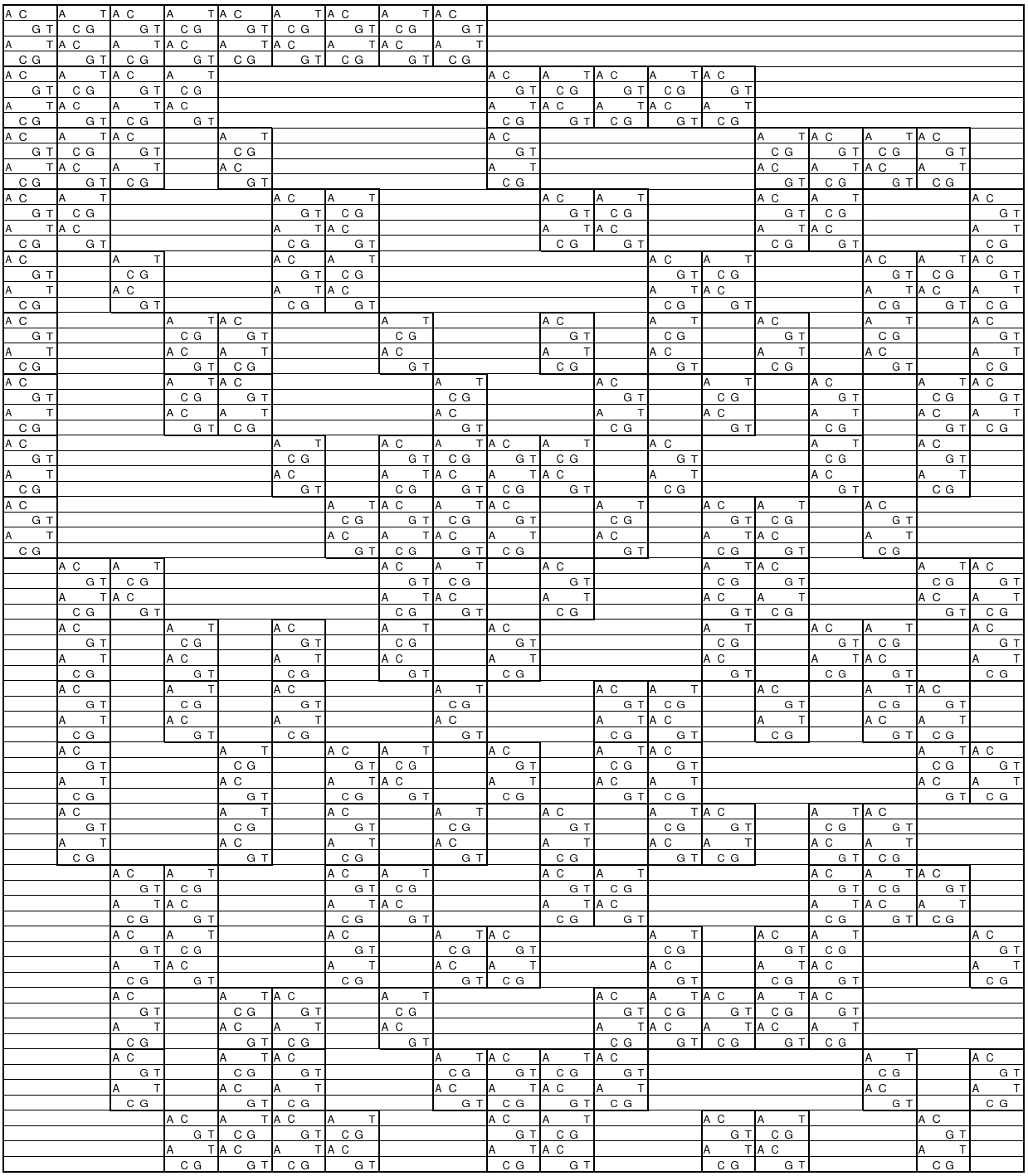

Figure 9: The product of a (19,19,9,9,4) 2-design and the pair of 4×4 QC blocks of Figure 8, resulting in a 76×76 QC matrix $\mathcal{Q}$ with minimum separation $sep(\mathcal{Q}) = 18$.

5.2. Substitutions Within Complementary Bases

Given a probe set, substituting each A with T (and vice versa), or each C with G (and vice versa) in every probe, does not change its hybridization behavior. This is because the substitutions are between complementary bases, so the hybridization efficiency of any two given probes remains unchanged. Hence a poorly hybridizing probe set remains so under this transformation.

Given a QC matrix $\mathcal{Q}$ for $(\mathrm{ACGT})^{n/4}$, to enforce the type of transformation on the probe set mentioned above we replace the label of every A column with T and every T column with A, and/or every G column with C and every C column with G. This leads to valid QC matrices of identical parameters as that of $\mathcal{Q}$, for the protocols $(\mathrm{TCGA})^{n/4}$, $(\mathrm{TGCA})^{n/4}$, and $(\mathrm{AGCT})^{n/4}$. The resulting designs can then be rotated, as in Section 5.1, to get valid QC matrices for $(\mathrm{ATCG})^{n/4}$, $(\mathrm{GATC})^{n/4}$, $(\mathrm{CGAT})^{n/4}$, $(\mathrm{ATGC})^{n/4}$, etc.

5.3. The Remaining Permutations

The two transformations above cover 16 of the 24 permutations. To get the remaining 8, we cover the two permutations $\pi_1 : \mathrm{ACGT} \rightarrow \mathrm{CAGT}$ and $\pi_2 : \mathrm{ACGT} \rightarrow \mathrm{ACTG}$, and get the rest via rotations as in Section 5.1. The permutations π_1 and π_2 are covered by checking that the probe set of the 8×8 QC block of Figure 4 remains poorly hybridizing under the transformation that substitutes A with C and vice versa (for π_1) or G with T and vice versa (for π_2), in each probe. Since the probe sets of the 4×4, 6×8, and 6×12 QC blocks are all subsets of that of the 8×8 QC block, these probe sets remain valid as well.

To obtain valid QC matrices for $(\mathrm{CAGT})^{n/4}$ or $(\mathrm{ACTG})^{n/4}$ from a valid QC matrix $\mathcal{Q}$ for $(\mathrm{ACGT})^{n/4}$, we relabel the columns in $\mathcal{Q}$ by replacing each A with C and vice versa (for π_1) or each G with T and vice versa (for π_2). The resulting design can then be rotated, as in Section 5.1, to get valid QC matrices for $(\mathrm{TCAG})^{n/4}$, $(\mathrm{GACT})^{n/4}$, $(\mathrm{GTCA})^{n/4}$, $(\mathrm{TGAC})^{n/4}$, $(\mathrm{AGTC})^{n/4}$, and $(\mathrm{CTGA})^{n/4}$.

6. Open Problems

The work reported here can be extended in various directions and raises several interesting open questions. We list a few here in no particular order.

1. Handle more than one step failure. Binary superimposed codes [9] appear to be a promising way to extend our hierarchical design approach to handle multiple step failures.

2. Relax the step fault model. When a step fails, not every spot using that step will have the same low intensity. The change in intensity more realistically will be a function of how far from the center of the probe the failed step is (Lipschutz *et al.* [11]).

3. Relax the assumption that the intensity window ϵ_k is supplied for each probe.

4. Handle a wider variety of protocols, i.e., with period greater than four.

5. Develop a general technique for designing balanced codes. These designs appear not to have been studied prior to this, even in the combinatorial design literature [3]. Alon and Tompa [1] have developed one such technique, resulting in many new balanced codes and QC matrices.

6. Improve the results of Table 1 by filling in more entries, or optimizing those entries for which $m/sep(\mathcal{Q}) > n/k$ (see Theorem 2.8). For instance, we have no optimal QC matrices with 60 steps. On this particular question, it is interesting to note that the product construction with our current QC blocks cannot produce QC matrices $\mathcal{Q}$ with $m/sep(\mathcal{Q}) < 4$: From Theorem 3.6,

$$sep(C' \times C) \leq d_{\min} \cdot \min(r'_{\min}, b' - r'_{\max}) \leq d_{\min} \cdot b'/2 = d_{\min} \cdot m/(2b),$$

so that $m/sep(C' \times C) \geq 2b/d_{\min}$. For the 8×8 and 4×4 QC blocks, $2b/d_{\min} = 4$; for the 6×12 and 6×8 QC blocks, $2b/d_{\min} = 6$.

Acknowledgments

We thank Noga Alon, Charlie Colbourn, Earl Hubbell, Yuan Ma, and David Smith for sharing their expertise with us.

Appendices

A. Basics of DNA Hybridization

Single-stranded DNA is a molecule composed by concatenating building blocks called *nucleotides*. Nucleotides come in four types called A, C, G, and T, named after the nucleotide's base component, so that a DNA molecule can be abstracted as a string over the alphabet $\{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}$.

The nucleotides occur in the complementary *base pairs* $\{\mathrm{A}, \mathrm{T}\}$ and $\{\mathrm{C}, \mathrm{G}\}$: these pairs *bind* or *hybridize* well to each other via hydrogen bonds. The complement of a DNA molecule is the result of replacing each of its nucleotides by the complementary nucleotide.

Two entire DNA molecules can only hybridize if they are arranged in an antiparallel alignment, meaning that they are aligned with one of them reversed. Thus, for example, the two DNA molecules ACGC ACGC ACGC and GCGT GCGT GCGT would hybridize extremely well to each other, because when they are aligned in antiparallel fashion

```
A C G C A C G C A C G C
| | | | | | | | | | | |
T G C G T G C G T G C G
```

all the aligned base pairs are complementary. This is a desirable situation if one of these molecules is a DNA array probe and the other is a target. However, it is undesirable if both of these molecules are targets, because then the target molecules bind to each other and are unavailable for binding to their complementary probes on the array.

A DNA molecule need not be exactly the reverse complement of another DNA molecule in order for the two to hybridize. Near complementarity suffices for reasonably good hybridization. However, if they are far from complementary, then they will not hybridize well at all.

All pairs of complementary DNA molecules do not hybridize with equal affinity or *binding energy*. There are many complicated reasons for this, but the simplest is that C and G hybridize with three hydrogen bonds whereas A and T form only two. This means that complementary molecules with a high G-C content tend to hybridize better than complementary molecules with a low G-C content. This observation underlies Assumption 4 in Section 2.1.

For more information on hybridization and binding energy, see any textbook on molecular biology, for instance Lewin [10].

B. Theorem 3.6 for Mated QC Blocks

In order to extend Theorem 3.6 to handle mated QC blocks such as the 4×4 blocks of Figure 8, we need to impose some conditions on the mates. Note, though, that the following theorem is general enough to apply to all of the QC blocks in this paper, since a single QC block such as the 8×8 can be considered as being mated with itself.

Theorem B.1: Let C' be a balanced code with parameters $(v', b', r'_{\min}, r'_{\max}, k', d'_{\min})$. Let C and D be a pair of mated QC block each with parameters $(v, b, r_{\min}, r_{\max}, k, d_{\min})$, and satisfying the following additional conditions:

1. $d_{\min} \leq r_{\min}$.
2. For any j, the columns C_{*j} and D_{*j} are either identical or complementary. That is, either $C_{ij} = D_{ij}$ for all rows i, or $C_{ij} = 1 - D_{ij}$ for all rows i.
3. Any column of C and any column of D are either identical, or have Hamming distance at least $d_{\min}$.

Let $\mathcal{Q}$ be the QC matrix that results from alternately substituting C and D for the ones in each row of C', and substituting a $b \times v$ matrix of zeros for each zero of C'. Then

$$sep(\mathcal{Q}) \geq \min(d'_{\min} r_{\min}, r'_{\min} d_{\min}, (b' - r'_{\max}) d_{\min}).$$

As an example, if C and D are the 4×4 QC blocks of Figure 8, then

$$sep(\mathcal{Q}) = 2\min(d'_{\min}, r'_{\min}, b' - r'_{\max}).$$

Proof: As in the proof of Theorem 3.6, we will identify any column j of the product $\mathcal{Q}$, $1 \leq j \leq v'v$, by the pair (g, h), where $1 \leq g \leq v'$, $1 \leq h \leq v$, and $j = (g-1)v + h$. What complicates this proof is that each such column j generally contains a mixture of column h from C and column h from D. The proof proceeds by examining four possible cases for the pair j, j' of columns of $\mathcal{Q}$ in Definition 2.4, and computing the separation of these columns. (The reader may find it helpful to identify an example of each case in Figure 9 while reading the proof for that case.)

Recall for all four cases that the $b'b$ spots of $\mathcal{Q}$ are partitioned into b distinct probes, corresponding to the b rows of C and D. Without loss of generality, assume that probe q_k corresponds to row k of C and D, for $1 \leq k \leq b$. For any k, $1 \leq k \leq b$, let e_k be the number of rows of $\mathcal{Q}$ with probe q_k that contain a one in column j and a zero in column j', and f_k be the number of rows of $\mathcal{Q}$ with probe q_k that contain a zero in column j and a one in column j'. Thus, $0 \leq e_k \leq r'_{\max}$, $0 \leq f_k \leq r'_{\max}$, $D_k(j, j') = e_k + f_k$, $L_k(j, j') = b' - e_k$, and $R_k(j, j') = b' - f_k$. Let the separation between columns j and j' be

$$s = \sum_{k=1}^{b} \min(D_k(j, j'), L_k(j, j'), R_k(j, j')) = \sum_{k=1}^{b} \min(e_k + f_k, b' - e_k, b' - f_k).$$

Case 1: $j = (g, h)$ and $j' = 0$. (Recall the convention from Definition 2.4 that column 0 is the vector $0^{b'b}$.) Then $\sum_{k=1}^{b} e_k \geq r_{\min} \cdot r'_{\min}$ and $f_k = 0$.

Case 1.1: For every k, $e_k \leq b' - e_k$. Then

$$s = \sum_{k=1}^{b} e_k \geq r_{\min} \cdot r'_{\min}.$$

Case 1.2: There exists k such that $b' - e_k < e_k$. Suppose that $C_{kh} = 1$, the case where $D_{kh} = 1$ being dual. By property 2 of the theorem, each of the values of k' with $C_{k'h} = 1$, of which there are at least $r_{\min}$, satisfies $e_{k'} = e_k$, since the column D_{*h} is either identical or complementary to C_{*h}. Thus,

$$s \geq r_{\min}(b' - e_k) \geq r_{\min}(b' - r'_{\max}).$$

Case 2: $j = (g, h)$ and $j' = (g, h')$, with $h \neq h'$. Then $\sum_{k=1}^{b}(e_k + f_k) \geq d_{\min} \cdot r'_{\min}$.

Case 2.1: For every k, $e_k + f_k \leq b' - \max(e_k, f_k)$. Then

$$s = \sum_{k=1}^{b}(e_k + f_k) \geq d_{\min} \cdot r'_{\min}.$$

Case 2.2: There exists k such that $b' - \max(e_k, f_k) < e_k + f_k$. Without loss of generality, suppose $e_k \geq f_k$, and that $C_{kh} = 1$ and $C_{kh'} = 0$, the case where this occurs in D being dual.

By property 2 of the theorem, each of the values of k' with $C_{k'h} \neq C_{k'h'}$, of which there are at least $d_{\min}$, satisfies $\{e_{k'}, f_{k'}\} = \{e_k, f_k\}$, since the columns D_{*h} and $D_{*h'}$ are either identical or complementary to C_{*h} and $C_{*h'}$, respectively. Thus,

$$s \geq d_{\min}(b' - e_k) \geq d_{\min}(b' - r'_{\max}).$$

Case 3: $j = (g, h)$ and $j' = (g', h')$, with $g \neq g'$, and columns h of C and h' of D are either identical or complementary. (The possibility $h = h'$ is included in this case.) By property 2 of the theorem, this means that column h of C is either identical or complementary to each of column h' of C, column h of D, and column h' of D. Since $g \neq g'$, $\sum_{k=1}^{b}(e_k + f_k) \geq r_{\min} \cdot d'_{\min}$.

Case 3.1: For every k, $e_k + f_k \leq b' - \max(e_k, f_k)$. Then

$$s = \sum_{k=1}^{b}(e_k + f_k) \geq r_{\min} \cdot d'_{\min}.$$

Case 3.2: There exists k such that $b' - \max(e_k, f_k) < e_k + f_k$. Without loss of generality, suppose $e_k \geq f_k$, and that $C_{kh} = 1$, the case where $D_{kh} = 1$ being dual. Each of the values of k' with $C_{k'h} = 1$, of which there are at least $r_{\min}$, satisfies $e_{k'} = e_k$ and $f_{k'} = f_k$, since $C_{kh} = C_{k'h}$ implies $D_{kh} = D_{k'h}$, $C_{kh'} = C_{k'h'}$, and $D_{kh'} = D_{k'h'}$, all these columns being either identical or complementary. Thus,

$$s \geq r_{\min}(b' - e_k) \geq r_{\min}(b' - r'_{\max}).$$

Case 4: $j = (g, h)$ and $j' = (g', h')$, with $g \neq g'$, and columns h of C and h' of D are neither identical nor complementary. By property 2 of the theorem this means that, if choosing column h either from C or from D, and choosing column h' either from C or from D, the two chosen columns cannot be identical. Hence, by property 3, the two chosen columns must have Hamming distance at least $d_{\min}$.

Let r'_1 and r'_2 be the number of ones in columns g and g' of C', respectively, and d' be the Hamming distance between these two columns of C'. Then the number of rows of C' in which both of these columns contain a one is $\frac{1}{2}(r'_1 + r'_2 - d')$, so that

$$\sum_{k=1}^{b}(e_k + f_k) \geq r_{\min} \cdot d' + \frac{1}{2} d_{\min}(r'_1 + r'_2 - d').$$

Suppose that there are t values of k for which $e_k + f_k > b' - \max(e_k, f_k)$.

Case 4.1: $t > d_{\min}$. Then

$$s \geq t(b' - r'_{\max}) > d_{\min}(b' - r'_{\max}).$$

Case 4.2: $0 \leq t \leq d_{\min}$. Each of the t values of k can reduce s by at most $\frac{1}{2}(r'_1 + r'_2 + d')$ from $\sum_{k=1}^{b}(e_k + f_k)$, namely the d' rows of C' where columns g and g' differ, plus the

$\frac{1}{2}(r'_1 + r'_2 - d')$ where both columns contain a one. Each of these t values also increases s by at least $b' - r'_{\max}$. Thus,

$$\begin{aligned} s &\geq \sum_{k=1}^{b}(e_k + f_k) - \frac{1}{2}t(r'_1 + r'_2 + d') + t(b' - r'_{\max}) \\ &\geq r_{\min} \cdot d' + \frac{1}{2}d_{\min}(r'_1 + r'_2 - d') - \frac{1}{2}t(r'_1 + r'_2 + d') + t(b' - r'_{\max}). \end{aligned} \tag{2}$$

Since expression (2) is a linear function of t, it achieves its minimum value at one of its endpoints $t = 0$ or $t = d_{\min}$.

Case 4.2.1: $t = 0$. Since $d' \leq r'_1 + r'_2$, Inequality (2) yields

$$s \geq r_{\min} \cdot d' \geq r_{\min} \cdot d'_{\min}.$$

Case 4.2.2: $t = d_{\min}$. Then

$$s \geq r_{\min} \cdot d' - d_{\min} \cdot d' + d_{\min}(b' - r'_{\max}) \geq d_{\min}(b' - r'_{\max}),$$

the last inequality following from property 1 of the theorem.

Combining the results of all of the cases,

$$\begin{aligned} sep(\mathcal{Q}) &\geq \min(r_{\min}d'_{\min}, \min(r_{\min}, d_{\min})r'_{\min}, \min(r_{\min}, d_{\min})(b' - r'_{\max})) \\ &= \min(r_{\min}d'_{\min}, d_{\min}r'_{\min}, d_{\min}(b' - r'_{\max})). \end{aligned}$$

□

C. Balanced Codes with Added Columns

This appendix provides constructions for those balanced codes in Table 1 labeled "$+i$", meaning that i extra columns have been added to some other balanced code. A few of these augmented balanced codes come from the following simple construction.

Proposition C.1: If there is a 2-design D with parameters (v, b, r, k, λ), then there is a balanced code with parameters

$$(v + 2, 2b, \min(b, 2r), \max(b, 2r), k + 1, \min(b, 4(r - \lambda))).$$

Proof: Duplicate the rows of D to obtain a 2-design D' with parameters $(v, 2b, 2r, k, 2\lambda)$. By Proposition 3.3, D' is a balanced code with parameters $(v, 2b, 2r, 2r, k, 4(r - \lambda))$. Add two additional columns to D' with entries $(1, 0)$ in those rows corresponding to one copy of D, and $(0, 1)$ in those rows corresponding to the second copy of D. (See the leftmost nine columns of Figure 10(a) for an illustration.) These two new columns each have exactly b ones, Hamming distance exactly b from each of the columns of D', and Hamming distance exactly $2b$ from each other. □

Proposition C.1 yields two of the balanced codes of Table 1:

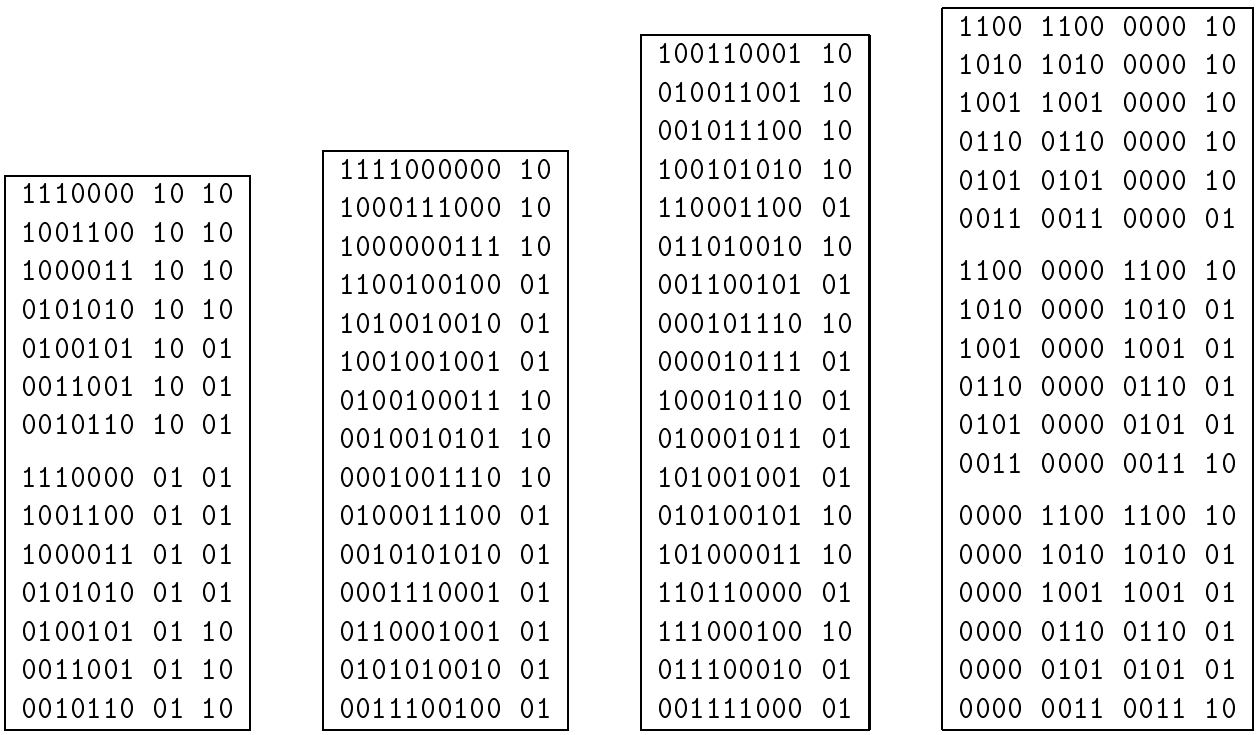

Figure 10: These balanced codes have the following parameters, and the following labels in Table 1: (a) Balanced code $(11, 14, 6, 7, 5, 6)$, labeled $(7, 14, 6, 3, 2) + 4$. (b) Balanced code $(12, 15, 6, 9, 5, 6)$, labeled $(10, 15, 6, 4, 2) + 2$. (c) Balanced code $(11, 18, 8, 9, 5, 9)$, labeled $(9, 18, 8, 4, 3) + 2$. (d) Balanced code $(14, 18, 6, 9, 5, 6)$, labeled $(3, 3, 2, 2, 1) \times (4, 6, 3, 2, 1) + 2$.

1. the entry labeled "$(7, 14, 8, 4, 4)$+2", which is derived from the 2-design with parameters $(7, 7, 4, 4, 2)$, and is a balanced code with parameters $(9, 14, 7, 8, 5, 7)$, and

2. the entry labeled "$(8, 28, 14, 4, 6) + 2$", which is derived from the 2-design with parameters $(8, 14, 7, 4, 3)$, and is a balanced code with parameters $(10, 28, 14, 14, 5, 14)$.

The 2-designs from which these are derived can be found in the compendium of Mathon and Rosa [13].

The remaining "$+i$" balanced codes were augmented from known 2-designs [13] by a simple exhaustive program, and are given in Figure 10.

D. Balanced Codes from Polynomials Over Finite Fields

This section describes the construction of those balanced codes labeled "degree d over GF(q)" in Table 1. These codes have appeared numerous times in the literature. For example, they were described by Nisan and Wigderson [14, Lemma 2.5], who called them simply "(d, q)-designs", as part of a construction for pseudorandom number generators.

Theorem D.1: Let q be any integral power of a prime number, and d be any nonnegative integer. Then there is a balanced code C with parameters $(q^{d+1}, q^2, q, q, q^d, 2(q-d))$.

Proof: Let F be the finite field with q elements, and $F[x]$ the integral domain of polynomials in the indeterminate x with coefficients in F. (See Lipson [12] for an introduction to the algebra of finite fields and polynomials.) C has a row for each of the q^2 pairs $(x_0, y_0) \in F \times F$, and a column for each of the q^{d+1} polynomials $p(x) \in F[x]$ of degree at most d. The entry in C for row (x_0, y_0) and column $p(x)$ is 1 if $p(x_0) = y_0$, and 0 otherwise. Thus there are exactly q ones per column (since any x_0 and $p(x)$ uniquely determine $y_0 = p(x_0)$) and q^d ones per row (since any x_0, y_0, and the d high degree coefficients of $p(x)$ uniquely determine the lowest degree coefficient). Finally, the Interpolation Theorem [12, Section IV.3.3, Theorem 5] states that any $d+1$ points (x_0, y_0) uniquely determine a polynomial $p(x)$ of degree at most d that passes through these points, so that any two columns of D can have at most d rows in which both columns contain a one. Thus, the Hamming distance between any two columns is at least $2(q-d)$. □

References

[1] Noga Alon and Martin Tompa. Balanced codes from near difference sets. In preparation, 2000.

[2] S. Chu, J. DeRisi, M. Eisen, J. Mulholland, D. Botstein, P. O. Brown, and I. Herskowitz. The transcriptional program of sporulation in budding yeast. *Science*, 282:699–705, 23 October 1998.

[3] Charles Colbourn, 2000. Personal communication.

[4] Charles J. Colbourn and Jeffrey H. Dinitz, editors. *The CRC Handbook of Combinatorial Designs*. CRC Press, 1996.

[5] R. Craigen. Hadamard matrices and designs. In Colbourn and Dinitz [4], pages 370–377.

[6] Ding-Zhu Du and Frank H. Hwang. *Combinatorial Group Testing and Its Applications*. World Scientific, 1993.

[7] Earl Hubbell, 1999. Personal communication.

[8] Earl Hubbell and Pavel A. Pevzner. Fidelity probes for DNA arrays. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 113–117, Heidelberg, Germany, August 1999. AAAI Press.

[9] W. H. Kautz and R. C. Singleton. Non-random binary superimposed codes. *IEEE Transactions on Information Theory*, 10:363–377, 1964.

[10] Benjamin Lewin. *Genes VI.* Oxford University Press, 1997.

[11] Robert J. Lipshutz, Stephen P. A. Fodor, Thomas R. Gingeras, and David J. Lockhart. High density synthetic oligonucleotide arrays. *Nature Genetics Supplement*, 21:20–24, 1999.

[12] John D. Lipson. *Elements of Algebra and Algebraic Computing.* Addison-Wesley, Reading, MA, 1981.

[13] Rudolf Mathon and Alexander Rosa. $2-(v,k,\lambda)$ designs of small order. In Colbourn and Dinitz [4], pages 3–40.

[14] Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994.

[15] David Smith. Affymetrix, 1999. Personal communication.

[16] Vladimir D. Tonchev. Codes. In Colbourn and Dinitz [4], pages 517–542.