Available online at www.sciencedirect.com
ScienceDirect
Discrete Applied Mathematics 155 (2007) 840–856
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Integer linear programming approaches for non-unique probe selection

Gunnar W. Klau[a, b, *], Sven Rahmann[c], Alexander Schliep[d], Martin Vingron[d], Knut Reinert[e]

[a] *Mathematics in Life Sciences, Free University Berlin, Arnimallee 3, D-14195 Berlin, Germany*
[b] *DFG Research Center MATHEON "Mathematics for Key Technologies", Berlin, Germany*
[c] *Algorithms and Statistics for Systems Biology, Genome Informatics, Technische Fakultät, Bielefeld University, D-33594 Bielefeld, Germany*
[d] *Computational Molecular Biology, Max Planck Institute for Molecular Genetics, Ihnestr. 73, D-14195 Berlin, Germany*
[e] *Algorithmic Bioinformatics, Free University Berlin, Takustr. 9, D-14195 Berlin, Germany*

Received 17 July 2004; received in revised form 17 January 2005; accepted 24 September 2005
Available online 31 October 2006

Abstract

In addition to their prevalent use for analyzing gene expression, DNA microarrays are an efficient tool for biological, medical, and industrial applications because of their ability to assess the presence or absence of biological agents, the *targets*, in a sample. Given a collection of genetic sequences of targets one faces the challenge of finding short oligonucleotides, the *probes*, which allow detection of targets in a sample by hybridization experiments. The experiments are conducted using either *unique* or *non-unique* probes, and the problem at hand is to compute a *minimal design*, i.e., a minimal set of probes that allows to infer the targets in the sample from the hybridization results. If we allow to test for more than one target in the sample, the design of the probe set becomes difficult in the case of non-unique probes.

Building upon previous work on group testing for microarrays we describe the first approach to select a minimal probe set for the case of non-unique probes in the presence of a small number of multiple targets in the sample. The approach is based on an integer linear programming formulation and a branch-and-cut algorithm. Our implementation significantly reduces the number of probes needed while preserving the decoding capabilities of existing approaches.

Keywords: Integer linear programming; Microarray; Probe; Oligonucleotide; Design; Group testing

1. Introduction

Microarrays are a widely used tool because they provide a cost-efficient way to determine levels of specified RNA or DNA molecules in a biological sample. Typically, one measures the amount of gene expression in a cell by observing hybridization of mRNA to different probes on a microarray, each probe targeting a specific gene. A distinct and likewise important application, arising, for example, in medicine, environmental sciences, industrial quality control, or biothreat

* Corresponding author. Mathematics in Life Sciences, Free University Berlin, Arnimallee 3, D-14195 Berlin, Germany.
E-mail address: gunnar@mi.fu-berlin.de (G.W. Klau).
URL: http://www.inf.fu-berlin.de/inst/ag-bio.

0166-218X/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2005.09.021

Table 1
Target-probe incidence matrix H

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9
t_1	1	1	1	0	1	1	0	0	0
t_2	1	0	1	1	0	0	1	1	0
t_3	0	1	1	1	0	1	1	0	1
t_4	0	1	0	0	1	0	1	1	1

reduction, is the identification of biological agents in a sample. This wide range of applications leads to the same methodological problem: to determine the presence or absence of *targets*—such as viruses or bacteria—in a biological sample.

Our work focuses on *oligonucleotide arrays*. To illustrate the general approach let us assume we would like to identify virus subtypes in a sample. If we test whether a number of *probes*, i.e., short oligonucleotides of size 8–32, hybridize to the genome of the virus, we can infer presence or absence of the virus if the hybridization pattern is unique among all viruses possibly contained in the sample. This problem is readily extended to the case of several viruses simultaneously present and the objective of identifying those which are indeed contained in thesample.

In the case of unique probes, neglecting errors, this extension is trivial, since we will exactly observe the union of those probes that hybridize to the viruses in the set and there is one unique set of targets for every observable hybridization pattern. However, finding unique probes is often difficult, for example in the case of closely related virus subtypes. One way around this fundamental problem is a method which can make use of *non-unique* probes: probes which hybridize to more than one target.

Assume we are given the target-probe incidence matrix $H = (H_{ij})$ as shown in Table 1: we set $H_{ij} = 1$ if and only if target t_i hybridizes to probe p_j. In a hybridization experiment with a sample that contains several targets, we will observe hybridization signals at all probes incident to *any* target present in the sample, i.e., we observe the logical OR of the row vectors corresponding to targets in the sample. If the probe set is not carefully chosen, this can easily lead to situations where we cannot resolve the hybridization pattern observed in the experiment.

We illustrate the problem using the example in Table 1. We have four targets $t_1, \ldots, t_4$ and nine probes $p_1, \ldots, p_9$. Assume first that we know that the sample contains at most one target. The goal is to choose a suitable *design matrix D*, i.e., to select a minimal set of probes that allows us to infer the presence of a single target. In our example it is sufficient to use probes p_1, p_2, and p_3 for detecting the presence of a single target (e.g., for target t_2 probes p_1 and p_3 hybridize, while p_2 does not). Minimizing the number of probes is a reasonable objective function, since it is proportional to the cost of the experiment. In the example, $\{p_1, p_2, p_3\}$ is an optimal choice, since we need at least three targets to distinguish between the five events "no target present", "t_1 present", ..., "t_4 present": The smaller choice $\{p_4, p_8\}$ cannot distinguish between the empty target set and the set $\{t_1\}$.

Now assume that target t_2 and target t_3 are *simultaneously* present in the sample (but none of the remaining targets is). In this case all three probes p_1, p_2, and p_3 hybridize. This case cannot be distinguished from the case that only t_1 is present. As a remedy, we could take all the probes $p_1, \ldots, p_9$. It can easily be checked that for each subset of two targets the hybridization pattern is different from every other subset of cardinality one or two. Taking all probes is, however, not necessary. With p_1, p_4, p_5, and p_9, all target sets of cardinality $\leqslant 2$ can be distinguished, with the exception of $\{t_1, t_3\}$ versus $\{t_2, t_4\}$. Adding probe p_8 to the design allows to make this last distinction, too.

Generally it is clear that taking all probes results in the best possible separation between all subsets. However, for a small number of targets in the sample, say up to three or four, we can often achieve the same quality with a (substantially) smaller number of probes.

In addition to the difficulty illustrated above, the problem is aggravated by the presence of errors. Usually the *false positive* error rate f_1 (i.e., the experiment reports a hybridization although there should be none) and the *false negative* rate f_0 (i.e., the experiment should report a hybridization but does not) are up to 5%. As a remedy it is customary to build some redundancy into the design; e.g., we demand that two targets are separated by more than one probe and that each target hybridizes to more than one probe.

It is not trivial to compute the target-probe incidence matrix H in the beginning. Among the potentially large set of possible non-unique probes, only a fraction satisfies the typical restrictions used for oligonucleotide probe selection.

For instance, all probes should exhibit the same hybridization affinity, expressed as the Gibbs free energy ΔG of the probe-target duplex, at a given temperature and salt concentration. The probes should neither be self complementary, nor should they cross-hybridize. Other constraints are possible (see for example [14]).

The three steps of (1) computing the target-probe incidence matrix, (2) computing a suitable design matrix D, and (3) decoding the result were recently addressed by [13]. In this work we address the second step, the computation of the design matrix D given the target-probe incidence matrix H, and use for the first and third step the methods proposed in the above mentioned paper, adopting its notation to a large extent. We emphasize, however, that our results do not depend on our choice of method for computing the target-probe incidence matrix H [7] nor on any of the typical constraints on oligonucleotide probes such as their length. While we strive to minimize the number of probes, we do not want to lose the ability to decode experimental results, i.e., the ability to infer the correct target set, even in the presence of errors. We show that the minimal designs we obtain have good decoding capabilities.

A similar problem has been considered by [11], where the authors have also used integer linear programming to determine a minimum cardinality set of substrings that distinguishes between all singleton target sets. In contrast to their work, we focus on a preselection of probes that satisfy the physical and chemical constraints discussed above, which are impossible to formulate by linear inequalities. In addition, we are able to distinguish not only between single targets but also the whole groups of targets. This is a much more complex problem: if we knew in advance that exactly (at most) one target is present, there are only m ($m + 1$) different outcomes to consider, m being the number of targets. Without this knowledge, there are 2^m possible outcomes. Furthermore, we present a more elaborate probabilistic decoding framework, explicitly modeling false positive and false negative hybridization errors.

This paper is the full version of the extended abstract presented at ISMB'04 [4]. It contains proofs of theorems, an additional, more elegant ILP formulation, and new computational experiments that settle some questions raised in the extended abstract.

1.1. Problem definition

We denote the m target sequences by t_i ($i \in \{1, \ldots, m\}$) and the n candidate probes by p_j ($j \in \{1, \ldots, n\}$), and define the target-probe incidence matrix $H = (H_{ij})$ by $H_{ij} := 1$ if target t_i hybridizes to probe candidate p_j, and $H_{ij} := 0$ otherwise. The design matrix D is the sub-matrix of H that contains those columns corresponding to probes included in the final design; it will be characterized by a binary vector $x = (x_1, \ldots, x_n)$, where $x_j = 1$ if and only if p_j is selected for the final design.

The set of probes hybridizing to target t_i, i.e., the index set of nonzero entries in row i of the incidence matrix H, is denoted by $P(i)$. Similarly, $T(j)$ denotes the set of target sequences probe p_j hybridizes to, or equivalently, the index set of nonzero entries in column j of H. If S is a *set* of target sequences, we say that a probe *p hybridizes to the set S* when p hybridizes to at least one target in S. By $P(S)$ we denote the set of all probes hybridizing to S, i.e., $P(S) := \bigcup_{t_i \in S} P(i)$.

Definition 1 (*d-separability*). Let S and T be two different target sets. Probe p *separates* S and T if $p \in P(S) \Delta P(T)$, i.e., if p hybridizes to either S or T, but not to both (Δ denotes symmetric set difference). Target sets S and T are *d-separable* if at least d probes separate them, i.e., if $|P(S) \Delta P(T)| \geqslant d$.

Consider the example in Table 1: the sets $S = \{t_1, t_2\}$ and $T = \{t_3, t_4\}$ are 2-separable (the separating probes are p_1 and p_9).

Formally, the probe selection problem is stated as follows.

Problem 2 (*Probe selection problem*). *Given an incidence matrix H and two parameters $d \in \mathbb{N}$ and $c \in \mathbb{N}$, select a minimal set of probes, such that all targets are covered by at least d probes and such that all different subsets of targets S and T up to cardinality c are d-separable using only selected probes.*

Note that asking for a target coverage of d explicitly is in fact unnecessary, because one of the sets S, T can be the empty set. Then the general separability constraint includes the coverage constraint. In practice, one might prefer to use different values $d_{\text{cov}} > d_{\text{sep}}$ for coverage and separability, respectively. Our methods allow this change easily.

Lemma 3. *The probe selection problem is NP-hard.*

Proof. This is a straightforward reduction from the set cover problem that appears as a special case ($d = 1$ and $c = 0$). □

Schliep et al. [13] describe a fast heuristic that allows the computation of a good design, and we review it shortly: the following procedure greedily computes a design that guarantees d-separability for *all pairs* of targets if possible and thus produces a feasible solution for the probe selection problem with cardinality parameter $c = 1$. Additionally, it randomly generates a fixed number of target subsets of larger cardinality and d-separates them. Note that due to the greedy nature of this algorithm, the chosen design is not guaranteed to be minimal.

(1) Add probes until every target is covered by at least d probes, i.e., every singleton target set $\{t_i\}$ is d-separated from the empty set, by calling SEPARATE($\{t_i\}, \{\}, d$) for all $i = 1, \ldots, m$.
(2) Ensure that all pairs of single targets are separated by at least d probes by calling SEPARATE($\{t_i\}, \{t_{i'}\}, d$) for all $1 \leqslant i < i' \leqslant m$.
(3) Randomly pick a number N of additional pairs of target sets S and T and d-separate each pair by calling SEPARATE(S, T, d). The parameter N can be chosen according to the time available to refine the design.

The procedure SEPARATE(S, T, d) ensures d-separation of S and T, or produces a warning if the candidate set allows only d'-separation for some $d' < d$ (see Algorithm 1).

Algorithm 1: Greedy design computation

SEPARATE(S, T, d);
Add probes to the current partial design D to d-separate S and T;
Let $C := P(S) \Delta P(T)$;
Partition C into $C = C_D \cup C'$, where $C_D := C \cap D$, and C' contains the separating probes not yet included in D;
if $|C_D| \geqslant d$ **then**
  | **return**; (*nothing to do*)
end
if $|C'| < (d - |C_D|)$ **then**
  | **warn** "Can only $(|C_D| + |C'|)$-separate S and T";
end
Add the $d - |C_D|$ highest-quality probes from C' to D;

This approach is simple and very practical. However, since a particular microarray is designed only once, the time spent to compute the design is less important than the size and quality of the design, that is, the number of probes it contains (the fewer the better), and its decoding capabilities in the presence of errors.

In order to reduce the number of probes in the design, we propose an approach based on Integer Linear Programming (ILP) that guarantees d-separability for each pair of targets as well as each pair of small target groups using the *minimal* number of probes.

2. Integer linear programming formulations

This section contains ILP formulations that characterize feasible solutions for the probe selection problem. Our first formulation, presented in Section 2.1, makes use of adding virtual probes to overcome problems with unseparable target sets and is also the one we use in our computational experiments. In Section 2.2 we present a more elegant formulation that directly solves the probe selection problem.

2.1. Virtual probe formulation

We start by formulating the problem as a variation of a set cover ILP. Let x_j $(1 \leqslant j \leqslant n)$ be a set of binary variables with $x_j = 1$ if probe p_j is chosen and $x_j = 0$ otherwise. Then the probe selection problem for $c = 1$ can be formulated

as the following integer linear program which we refer to as the *master ILP*:

$$\begin{aligned}
\min \quad & \sum_{j=1}^{n} x_j \quad \text{(master ILP)} \\
\text{s.t.} \quad & \sum_{j=1}^{n} H_{ij} x_j \geqslant d, \quad 1 \leqslant i \leqslant m, \\
& \sum_{j=1}^{n} |H_{ij} - H_{kj}| \cdot x_j \geqslant d, \quad 1 \leqslant i < k \leqslant m, \\
& x_j \in \{0, 1\}, \quad 1 \leqslant j \leqslant n.
\end{aligned}$$

Note that it can be easily checked whether it is possible to d-separate all pairs of targets. If not, then the set of feasible solutions of the above ILP is empty. As a remedy, we consider a variation of the problem: we add a sufficiently large number $l := m \cdot d$ of unique virtual probes that are only chosen if it is not possible to do the separation with the original set of candidate probes. We ensure this by setting the objective function coefficients of the virtual probes to a large number M, i.e., we change the objective in the master ILP to

$$\min \sum_{j=1}^{n} x_j + M \sum_{k=n+1}^{n+l} x_j, \tag{1}$$

and replace n by $n + l$ in the constraints of the above ILP. Having added the virtual probes we can now deal with input matrices H that do not allow d-separability.[1]

The master ILP guarantees the pairwise separation of all single targets, similar to the greedy heuristic. Solving the ILP, however, leads to the *minimal* number of probes necessary to do this. In the experimental section we show that the difference in the number of probes can be substantial.

We do not only want to guarantee d-separability between pairs of targets but between pairs of small target *groups*. Given a set of targets S (group), we denote by ω^S the vector that results from applying the logical OR to the rows in S. We call this vector the *signature* of S. It represents the probe set $P(S)$ as a binary vector and is formally defined by

$$\omega_j^S = \max_{i \in S} H_{ij} \quad (j = 1, \ldots, n).$$

Note that S and T are d-separable if and only if the Hamming distance between ω^S and ω^T is at least d. Our goal is to guarantee a sufficiently large Hamming distance for the signatures of all pairs $S \neq T$ with $|S|, |T| \leqslant c$. We call these additional requirements the *group constraints*.

Enumerating all pairs of small subsets and adding the corresponding group constraints to the ILP is not feasible, as already noted by Schliep et al. [13]. Hence we propose a *cutting plane approach*: whenever we have a feasible solution to the master ILP, we check for violated group inequalities and add them only if needed. This leads to a branch-and-cut algorithm (see, e.g., [15]), a linear programming-based branch and bound technique for solving mixed integer linear programs by dynamically adding violated inequalities (cuts).

2.1.1. Finding violated group inequalities

The main idea of our approach is to iteratively construct a *most violated group constraint* by looking at our current selection of probes. More precisely, let x^* be a solution vector of the master ILP and let $X = \{j \mid x_j^* = 1\}$, i.e., the index set of the currently chosen probes. Further, for a target set S, let $\omega^S|_X$ denote the restriction of ω^S to the columns in X. For brevity, we set $M := \{1, \ldots, m\}$ (target indices) and $N := \{1, \ldots, n\}$ (probe indices).

[1] Note that, instead of modifying the objective function by making the virtual probes expensive, we could also attack this problem in a two-step process: first, minimize the number of virtual probes, and then add the corresponding variables to the master ILP and solve it using the original objective function.

$$\max \sum_{j\in X}(\sigma_j^0+\sigma_j^1) \qquad \text{(slave ILP)}$$

$$\text{s.t. } \sigma_j^0 \leqslant 1-s_i \qquad \forall\,(i,j)\in M\times X \text{ with } H_{ij}=1 \qquad (2)$$

$$\sigma_j^0 \leqslant 1-t_i \qquad \forall\,(i,j)\in M\times X \text{ with } H_{ij}=1 \qquad (3)$$

$$\sigma_j^1 \leqslant \sum_{i\in M} H_{ij}s_i \qquad \forall\, j\in X \qquad (4)$$

$$\sigma_j^1 \leqslant \sum_{i\in M} H_{ij}t_i \qquad \forall\, j\in X \qquad (5)$$

$$\sum_{i\in M} s_i \geqslant 1 \qquad (6)$$

$$\sum_{i\in M} t_i \geqslant 1 \qquad (7)$$

$$s_i+t_i \leqslant 1 \qquad \forall\, i\in M \qquad (8)$$

$$\sum_{i\in M} s_i \leqslant c \qquad (9)$$

$$\sum_{i\in M} t_i \leqslant c \qquad (10)$$

$$0\leqslant \sigma_j^0 \leqslant 1 \qquad \forall\, j\in X \qquad (11)$$

$$0\leqslant \sigma_j^1 \leqslant 1 \qquad \forall\, j\in X \qquad (12)$$

$$s_i\in\{0,1\} \qquad \forall\, i\in M \qquad (13)$$

$$t_i\in\{0,1\} \qquad \forall\, i\in M \qquad (14)$$

Fig. 1. The slave ILP.

We solve another ILP, the *slave ILP* in Fig. 1, in order to find target groups S and T for which the Hamming distance of $\omega^S|_X$ and $\omega^T|_X$ is below the threshold d.

The aim of the slave ILP is to select (via the variable vectors s and t) two sets of targets (S and T) that yield a maximally violated group inequality. In other words, the ILP tries to create two groups that resemble each other as much as possible in terms of their signatures.

Variables σ_j^0 and σ_j^1 model the *similarity* of S and T for probe j, i.e., $\sigma_j^0=1$ if and only if both ω_j^S and ω_j^T are equal to zero and $\sigma_j^1=1$ if and only if both values are equal to one. Besides the trivial constraints (11)–(14) and inequality (8), which keep S and T disjoint,[2] we have three main classes of inequalities: The first class, given by inequalities (2) and (3), models that $\sigma_j^0=0$ if S or T contain a target that hybridizes to probe j. Similarly, (4) and (5) express that, if $\sigma_j^1=1$, at least one target in both S and T must hybridize to j. Finally, (6) and (7) avoid $S=\emptyset$ and $T=\emptyset$, and (9) and (10) control the size of S and T.

Lemma 4. *A feasible solution (s,t,σ^0,σ^1) of the slave ILP for a partial design with virtual probes characterized by X corresponds to two disjoint groups of targets, S and T. Furthermore, the value*

$$h := |X| - \sum_{j\in X}(\sigma_j^0+\sigma_j^1)$$

is equal to the Hamming distance of S and T with respect to X.

[2] In fact, this restricts the set of feasible solutions too much. The problem formulation only requires that $S\neq T$, not that $S\cap T=\emptyset$. In the next section we show how to formulate the $S\neq T$ requirement by introducing additional variables. For the sake of clarity, we stick to $S\cap T=\emptyset$ for the moment.

Proof. The proof of this lemma is similar to the proof of Lemma 6 in the next section where we give a more elegant ILP formulation that does not require virtual probes. □

If h is smaller than the minimum required Hamming distance d, we have found a violated group inequality, namely

$$\sum_{j=1}^{n+l} |\omega_j^S - \omega_j^T| \cdot x_j \geqslant d.$$

We add this inequality to the master ILP, solve it again, and iterate the process. If we do not find further violated inequalities, we have solved the group separation problem and know that our selection of probes is well-suited to additionally distinguish between target groups of small cardinality.

There are several drawbacks of this first ILP approach. First, and most importantly, the following property of virtual probes may prevent the first ILP approach from finding an optimal solution to Problem 2: suppose that two targets s and t are only d'-separable, with $d' < d$. Then, we have to use virtual probes, say for target s, to d-separate s and t. Unfortunately, this has consequences not only for the separation of the target sets $\{s\}$ and $\{t\}$, but also for all S and T with $s \in S$ and $s \notin T$: the virtual probe, which has to be included in any case, also serves as a separating probe for all such S and T, even though some of these pairs may be d-separable without virtual probes. Second, the slave ILP needs the integer solution of the current master ILP before a violated group inequality can be found; it cannot even be formulated with an intermediate solution of a LP relaxation. Finally, it would be preferable to have a purely combinatorial polynomial-time algorithm to solve the separation problem. In the next section we present a better formulation that does not suffer from the first two drawbacks. The last point remains an issue for further research.

2.2. Formulation without virtual probes

In this section we present a more elegant ILP formulation for the probe selection problem (Problem 2) that does not require virtual probes. We propose the following solution:

For two groups of targets S and T we define their *maximal possible separation*

$$h(S, T) = \sum_{j \in N} |\omega_j^S - \omega_j^T|,$$

and give a direct formulation for the probe selection problem, replacing the master and slave ILPs from the previous section.

$$\min \quad \sum_{j \in N} x_j \tag{15}$$

$$\text{s.t.} \quad x_j \in \{0, 1\} \quad \forall j \in N, \tag{16}$$

$$\sum_{j \in N} |\omega_j^S - \omega_j^T| \cdot x_j \geqslant \min\{d, h(S, T)\} \quad \forall S, T \subseteq M, \ |S| \leqslant c, \ |T| \leqslant c, \ S \neq T. \tag{17}$$

Again, the coverage constraints are satisfied by inequalities (17) with $S = \emptyset$ and $T = \{t_i\}$ for $i \in M$.

Clearly the number of constraints (17) is exponential. We therefore consider relaxing the formulation by these constraints and solving the corresponding separation problem at each node of the branch-and-bound tree:

Problem 5 (*Separation problem for group inequalities*). *Given an optimal solution x^* of the relaxation of the ILP* (15)–(16) *with only some of the constraints* (17) *satisfied, find $S, T \subseteq M$, $|S| \leqslant c$, $|T| \leqslant c$, $S \neq T$ with*

$$\sum_{j \in N} |\omega_j^S - \omega_j^T| \cdot x_j^* < \min\{d, h(S, T)\}, \tag{18}$$

or prove that no such sets exist.

Rewriting (18) as

$$\sum_{j\in N} |\omega_j^S - \omega_j^T| \cdot x_j^* < d \tag{19}$$

and

$$\sum_{j\in N} |\omega_j^S - \omega_j^T| \cdot x_j^* < h(S,T) \tag{20}$$

enables us to give the following ILP with variables $\omega^S \in [0,1]^N$, $\omega^T \in [0,1]^N$, $\delta \in [0,1]^N$, $z \in \mathbb{R}$, $s \in \{0,1\}^M$, and $t \in \{0,1\}^M$ to formulate the separation problem:

$$\min \quad \sum_{j\in N} \delta_j x_j^* - z \tag{21}$$

$$\text{s.t.} \quad \omega_j^S \geqslant s_i \quad \forall (i,j) \in M \times N \text{ with } H_{ij} = 1, \tag{22}$$

$$\omega_j^T \geqslant t_i \quad \forall (i,j) \in M \times N \text{ with } H_{ij} = 1, \tag{23}$$

$$\omega_j^S \leqslant \sum_{i\in M} s_i H_{ij}, \quad \omega_j^T \leqslant \sum_{i\in M} t_i H_{ij} \ \ \forall j \in N, \tag{24}$$

$$\delta_j \geqslant \omega_j^S - \omega_j^T, \quad \delta_j \geqslant \omega_j^T - \omega_j^S \ \ \forall j \in N, \tag{25}$$

$$\delta_j \leqslant \omega_j^S + \omega_j^T, \quad \delta_j \leqslant 2 - \omega_j^S - \omega_j^T \ \ \forall j \in N, \tag{26}$$

$$s_i + t_i \leqslant 1 \quad \forall i \in M, \tag{27}$$

$$\sum_{i\in M} s_i + t_i \geqslant 1, \tag{28}$$

$$\sum_{i\in M} s_i \leqslant c, \quad \sum_{i\in M} t_i \leqslant c, \tag{29}$$

$$z \leqslant \sum_{j\in N} \delta_j \tag{30}$$

$$z \leqslant d \tag{31}$$

$$s_i \in \{0,1\}, \quad t_i \in \{0,1\} \ \ \forall i \in M. \tag{32}$$

Similar to the slave ILP from the previous section, this formulation builds the sets S and T via the variable vectors s and t. The variables δ_j model the difference of the signatures of the two chosen groups at position j, that is, we have $\delta_j = 1$ if and only if $\omega_j^S \neq \omega_j^T$ and $\delta_j = 0$ if the two groups have the same signature at position j. Clearly, we want to find two groups of minimum difference or—as in the slave ILP—of maximum similarity.

The construction of the signature vector ω via inequalities is similar to the computation of vector σ in the slave ILP. Note, however, that we compute the signatures over the whole range of candidate probes. Inequalities (25) and (26) force variables $\delta_j = 1$ if and only if the signatures differ for probe j, and $\delta_j = 0$ otherwise. Again, constraints (27), (28), and (29) ensure $S \cap T = \emptyset$, $S \cup T \neq \emptyset$, and $|S| \leqslant c$, $|T| \leqslant c$, respectively.

The most important difference to the slave ILP in the previous section are constraints (30) and (31): they make sure that inequalities (19) and (20) hold. Variable z will be set to the minimum of the separation parameter d and the maximum possible separation of the chosen groups, and inequality (18) is violated if and only if the objective function yields a value lower than zero.

Note that restricting s and t to integers automatically guarantees that the components of δ, ω^S and ω^T are integers as well, so these constraints need not be enforced explicitly.

Lemma 6. *A feasible solution* $(s, t, \omega^S, \omega^T, \delta)$ *of ILP* (21)–(32) *corresponds to two target groups S and T with* $S \cap T = \emptyset$ *for which*

$$|\omega_j^S - \omega_j^T| = \delta_j \quad \forall j \in N \tag{33}$$

holds. Its objective function value is equal to

$$\sum_{j \in N} |\omega_j^S - \omega_j^T| \cdot x_j^* - \min\{d, h(S, T)\},$$

and if it is smaller than zero, we have found a violated group inequality.

Proof. Let $(s, t, \omega^S, \omega^T, \delta)$ be a feasible solution of the integer linear program. We construct two sets S and T based on the characteristic vectors s and t. Clearly, we have $S \cap T = \emptyset$ due to inequality (27). Inequalities (22)–(24) ensure that $\omega_j^S = 1$ if there is at least one $i \in S$ with $H_{ij} = 1$, and that $\omega_j^S = 0$ if no such i exists in S (likewise for T). Given the signatures at position j, inequalities (25)–(26) take care of the correct computation of δ_j. If ω_j^S and ω_j^T are equal, the third and fourth inequality force δ_j to zero. Otherwise, the first and second inequality force δ_j to one. Thus, $\delta_j = |\omega_j^S - \omega_j^T|$ for all $j \in N$. With this equation, constraint (30) changes to

$$z \leqslant |\omega_j^S - \omega_j^T| = h(S, T).$$

Together with (31) this yields

$$z = \min\{d, h(S, T)\},$$

which concludes the proof. □

As mentioned in the previous section, the disjointness constraint $S \cap T = \emptyset$ (27) can be changed into a set of constraints that ensure only $S \neq T$ as follows. We introduce a new variable vector, δ^{ST}, and replace (27) by

$$\delta_i^{ST} \geqslant s_i - t_i, \quad \delta_i^{ST} \geqslant t_i - s_i \ \ \forall i \in M, \tag{34}$$

$$\delta_i^{ST} \leqslant s_i + t_i, \quad \delta_i^{ST} \leqslant 2 - s_i - t_i \ \ \forall i \in M, \tag{35}$$

$$\sum_{i \in M} \delta_i^{ST} \geqslant 1. \tag{36}$$

The construction is similar to the one that measures the distinctness of the signatures. Whenever s and t are different at some position i, the corresponding value of δ_i^{ST} is set to one by inequalities (34) and (35) and to zero otherwise. Constraint (36) requires the vectors to be different at at least one position.

Corollary 7. *The integer linear program* (21)–(32) *with* (34)–(36) *instead of* (27) *solves the separation problem for group inequalities.*

3. Experimental validation

Schliep et al. [13] tested their group testing method employing a greedy heuristic on a set of 1230 28S rDNA sequences from different organisms present in the Meiobenthos [6]. The set contains redundancies and many close homologs, so the sequences were clustered at 99% sequence identity over at least 99% of the sequence length, and a representative for each cluster was picked arbitrarily. This procedure results in a test set of 679 target sequences. We have access to this data set and report on the results. Additionally, in order to evaluate the benefits of our new method more systematically, we also generate artificial data sets and compare the results of our method against the result of the greedy heuristic.

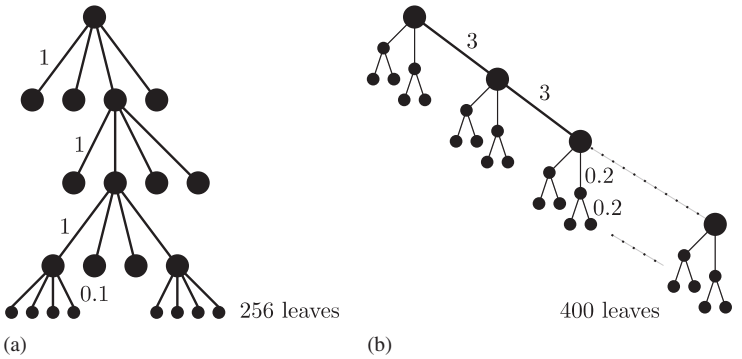

Fig. 2. The two evolutionary tree models used for sequence family generation. Branch lengths are indicated next to sample branches. The 256 (respectively, 400) leaf sequences were taken as family members. In (a) not all children of the nodes are shown.

3.1. Generating artificial data

3.1.1. Generating sequence families

To generate artificial data that closely models homologous sequence families, we use the REFORM (Random Evolutionary FORests Model) software developed by Rahmann [10] that allows to define arbitrary sets of evolutionary trees ("evolutionary forests") with either random or pre-defined root sequences. The sequences are evolved from the root through internal nodes to the leaves along the branches of the tree for a time proportional to the branch lengths, and may consist of several segments. For each segment it is possible to specify a separate evolutionary model.

The nucleotide substitution model is given as an evolutionary Markov process (EMP); for example as the simple model by Jukes and Cantor [3] that assigns equal probabilities to all mutation types. In general, the EMP is specified by any valid rate matrix $Q = (Q_{ij})$ with $Q_{ij} > 0$ for $i \neq j$ and $Q_{ii} = -\sum_{j \neq i} Q_{ij} < 0$. For $i \neq j$, Q_{ij} is the mutation rate $i \to j$, where i and j are different nucleotides. $|Q_{ii}|$ then measures the overall mutation rate away from i. Evolutionary time and branch lengths are measured in percentage of expected mutations. Therefore, the entries of Q are normalized by a calibration condition, such as $\sum_{i \neq j} \pi_i Q_{ij} = 0.01$, where π_i denotes the stationary or average frequency of nucleotide i. The condition states that on average, within one time unit, we expect $1/100$ mutation events per site.

An indel model is placed on top of the substitution process by specifying a deletion rate, an insertion rate, an indel length distribution, and a nucleotide distribution of inserted residues. During sequence evolution along a branch, at each position of the parent sequence, the probability of deleting one or several characters is given by the product of the branch length, the relative speed for the current segment, and the deletion rate. The length of the gap is then drawn from the specified gap length distribution. A similar rule is applied to inserts. Substitutions are only computed for non-deleted positions, but inserts can follow immediately after deletions.

For our experiments, we use two different forest models (see also Fig. 2). From each model, five independent test sets are generated.

(a) The first model produces a family of 256 sequences of average length 1000 nt. The root sequence consists of a random sequence of length 1000 nt with uniform nucleotide distribution. It is split into 5 segments of equal size with relative evolutionary speeds of 0.9, 0.95, 1, 1.05, and 1.1. Substitutions are generated according to the Jukes–Cantor model. The global delete and insert rates are set to 0.005, and the distribution of the gap lengths is given by the probability vector proportional to $(8, 1, 4, 2, 1, 0.5, 0.25, 0.125, \ldots)$. Inserted residues are drawn from the uniform distribution. The tree has three levels of internal nodes below the root for a total of $4 + 16 + 64$ internal nodes. Starting with the root, each internal node has 4 children. The distance between adjacent nodes corresponds to $t = 1$ percent of expected mutations. Each internal node on the third level has four leaf children at a distance of $t = 0.1$ for a total of 256 leaves with different distances to each other: For each given leaf, there are three leaves at distance 0.2, 12 leaves at distance 2.2, 48 leaves at distance 4.2, and 192 leaves at distance 6.2. The leaf sequences are subsequently used for probe candidate selection.

(b) In the second model, all global parameters are as in the first model, and the sequences consist of a single segment of average length 1000 nt. The topology differs considerably from the first model: the tree consists of a linear chain of 100 internal nodes (including the root) 3 time units apart; two "cherries" with branch lengths of 0.2 are attached to each internal node (Fig. 2b) for a total of 400 leaves.

These two particular model topologies were chosen because they produce difficult sets of very similar target sequences that cannot be easily separated with unique probes. Model (a) is strictly hierarchical, while model (b) has an overall linear structure.

3.1.2. Generating probe candidates

To generate probe candidates for each of the 10 families (5 instances of each model), we use the PROMIDE software [8,9]. Probe candidates are selected to be between 19 and 21 nt long and have a stability (Gibbs energy) of -20 to $-19.5\,\mathrm{kcal/mol}^{-1}$ at 40 °C and 0.075 M $[Na^+]$ according to the nearest neighbor model with parameters from SantaLucia [12].

We keep probes that occur as exact substrings in up to 50 family members. If, however, a probe candidate p has a long common substring (at least $|p| - 3$ nt) with another family member sequence t^*, but does not occur exactly in it, we discard this candidate because we cannot make a reasonably certain binary decision: cross-hybridization may or may not cause p to show a signal when t^* is present in a sample. The decision to keep only candidates where a clear decision is possible was made to keep the false positive and false negative error levels reasonably low.

We found that good probe candidates frequently occur in clusters in the target sequences; probes in the same cluster tend to have the same properties. If this happens, only one candidate from each cluster is selected.

3.2. Evaluating the selection

Minimizing the number of probes is an obvious objective function in the selection of the design. In the presence of errors, however, a reduction in number of probes is futile if the resulting set of probes performs worse. A natural performance measure is the ability to *decode* a microarray experiment, that is, to infer which biological agents were present in the sample. We will describe the statistical model for decoding and the Monte Carlo approach for the estimation of performance proposed by Schliep et al. [13], which is based on a group testing approach for screening clonal libraries [5].

From now on, we assume that a design has been fixed and denote the number of probes in the design by n (up to now, n was the number of probe candidates).

3.2.1. Statistical decoding

In an experiment, we observe whether each probe has hybridized or not. The experimental result can therefore be described as a binary vector $r = (r_1, \ldots, r_n) \in \{0, 1\}^n$. We are interested in the posterior probability that a subset T of targets is exactly the set of targets present in the sample,

$$\mathbb{P}[T|r] = \frac{\mathbb{P}[r|T] \cdot \mathbb{P}[T]}{\mathbb{P}[r]} \propto \mathbb{P}[r|T] \cdot \mathbb{P}[T].$$

In this Bayesian formulation we are left to choose the prior probabilities $\mathbb{P}[T]$ for all $T \subset M$ and devise a model to compute the likelihood $\mathbb{P}[T|r]$ of the observation r, given the knowledge of T.

To model the prior, we assume that targets are present or absent independently and that there are individual prevalences f_i of targets in the possible samples. Also, we assume that there is a prior on the number of targets *simultaneously* present, denoted c_k. Together this yields the combined prior for a set T of the form

$$\mathbb{P}[T] \propto c_{|T|} \cdot \prod_{t_i \in T} f_i \cdot \prod_{t_i \notin T} (1 - f_i).$$

To model the likelihood, we assume that false negative and false positive perturbations of the true result r associated to T act independently on each probe. Let us write $T(j)$ the set of targets that probe p_j hybridizes to. Then we should

get a hybridization signal for p_j if and only if $T \cap T(j) \neq \emptyset$. Because of the possibility of false positive and negative errors with rates f_1 and f_0, respectively,[3] we have the following probability distribution for r_j:

$$\mathbb{P}(r_j = 1|T) = \begin{cases} f_1 & \text{if } T \cap T(j) = \emptyset, \\ 1 - f_0 & \text{if } T \cap T(j) \neq \emptyset. \end{cases}$$

Because of the assumed conditional independence of probes, we can now compute the likelihood $\mathbb{P}[r|T]$.

Finding the set T maximizing the posterior $\mathbb{P}[T|r]$ is far from trivial. Also, there will be typically a large number of distinct sets with about equal posterior, which makes settling on one optimal T dubious. For that reason we compute the posterior probabilities $\mathbb{P}[t_i \in T|r]$ using Markov chain Monte Carlo with a Gibbs sampler, see [5] for details. The computation uses a Markov chain (MC) with $\mathbb{P}[T|r]$ as its equilibrium distribution and estimates $\mathbb{P}[t_i \in T|r]$ as the relative frequency of the event $t_i \in T$ over a large number of sets T sampled using the MC. The result of the decoding is a list of targets ranked according to their posterior.

3.2.2. Assessing performance

The performance of a design is measured by its ability to decode experiments even if multiple targets are present in a sample and the error rates in the hybridization experiments are high. Ideally, if we have a true target set T of size k, then the decoding described in the previous section should assign high probabilities to targets $t_i \in T$, and zero to all others. We would like to quantify in how far this is true for all possible T. This is clearly infeasible, so we resort to a Monte Carlo approach repeating the following in silico experiment.

We randomly choose a set of k targets, the *true* positives. That is, we assume that our artificial sample to be analyzed contains each of the k targets but no others. Neglecting errors at first, the design we are testing yields the set of probes which all should hybridize to our sample (recall that the design specifies the incidence of targets and probes). This gives us a set of true positive probes, the ones hybridizing to a chosen target, and true negative probes, the ones which do not.

Errors are introduced by independently changing result values for true positive probes to negative with probability f_0 and for true negative probes to positive with probability f_1. This noisy result is used as input to the MCMC-based decoding procedure described above.

The result of the decoding is a sorted list of the most probably present targets. To estimate the performance of a design, we repeat the process for a large number of random target sets for different choices of (small) k and count the fraction of true positive targets appearing at rank 1, 2, 3, ... of the result list. A design exhibits maximal performance if the proportion of true positive targets among the top k found by the decoding procedure is unity when choosing k-sized samples. For example, when evaluating the heuristic design, cf. Table 4, the value 0.93 for two positives among "top 2" implies that 93 of the 100 true positives in the 50 repetitions of the Monte-Carlo experiment were ranked among the top two by the decoding. Similarly, 98 of the 100 true positives were found among those ranked first to fourth.

Clearly, the performance degrades as k grows. Even for small k maximal performance cannot be expected for two reasons. First, in the presence of realistically large error rates, say 5%, the number of true positive probes is vastly outnumbered by the number of false positive ones. Second, the decoding procedure is stochastic and hence not guaranteed to give an exact result. We propose to use the proportion of true positives among the top $k + 1$ targets for k-cardinality samples. The choice of the maximal value of k to consider depends on the application and can be guided by the cardinality prior c_k. For example one could pick K as the minimum value for which $\sum_{k=1}^{K} c_k \geqslant 0.95$ holds.

3.3. Results

In [4] we reported on our results using the Markmann [6] data set (679 28S rDNA targets from different organisms present in the Meiobenthos) denoted (M), as well as the ten artificial data sets described in Section 3.1, denoted (a)1 to (a)5, and (b)1 to (b)5.

In our experiments, we used different parameters for coverage and separation of targets (these correspond to the unique parameter d used in the theoretical description of the approaches). We generated designs with minimum coverage

[3] The error rates f_0 and f_1 should not be confounded with the prevalences f_i despite the similar notation.

Table 2
For each artificial data set (a)1 to (b)5 and for the Markmann [6] meiobenthic data (M), the table shows the number m of targets, the number #cand of probe candidates, and the number of probes n chosen by the greedy design heuristic and the ILP approach, using pairwise separation only

Set	m	#cand	Greedy n		ILP n		n ratio	t ratio
(a) 1	256	2786	1163	(42%)	503	(18%)	2.31	0.23
(a) 2	256	2821	1137	(40%)	519	(18%)	2.19	0.21
(a) 3	256	2871	1175	(41%)	516	(18%)	2.28	0.25
(a) 4	256	2954	1169	(40%)	540	(18%)	2.17	0.17
(a) 5	256	2968	1175	(40%)	504	(17%)	2.33	0.24
(b) 1	400	6292	1908	(30%)	879	(14%)	2.17	0.02
(b) 2	400	6283	1885	(30%)	938	(15%)	2.01	0.02
(b) 3	400	6311	1895	(30%)	891	(14%)	2.13	0.06
(b) 4	400	6223	1888	(30%)	915	(15%)	2.06	0.02
(b) 5	400	6285	1876	(30%)	946	(15%)	1.98	0.07
(M)	679	15,139	3851	(25%)	3158	(21%)	1.22	0.08

Percentages represent the number of selected probes in relation to the number of probe candidates. The probe ratio $n_{\text{Greedy}}/n_{\text{ILP}}$ and the ratio $t_{\text{Greedy}}/t_{\text{ILP}}$ of the required design times are also shown.

Table 3
Comparison of design size between greedy and ILP solution

Data set	Targets	Candidates	Heuristic1	Heuristic2	ILP1	ILP2
M	679	15139	3879	3879	3158	3158
LSU	1165	11047	4901	4901	3887	3887
(a)1	256	2786	1164	1165	503	515
(cl)1	256	2530	1083	1083	542	566

The column *targets* contains the number of sequences in each data set and the column *candidates* the number of probe candidates in the incidence matrix H. The next four columns show the number of probes in the design matrix computed by the heuristic algorithm without and with random separation (500,000 random groups) and by the ILP-based algorithm without (ILP1) and with (ILP2) group separation (the maximal cardinality of groups was set to $c = 5$).

10 and minimum Hamming distance 5 with the greedy heuristic (a part of the PROMIDE package) and with the ILP approach described in Section 2, utilizing version 8.1 of the commercial CPLEX software with standard settings [2].

The results are shown in Table 2. Naturally, the heuristic runs faster, but it also generates a design that is often more than twice as large than the optimal design found with the ILP approach.

This is a general trend observed in the real and the artificial data sets. Our approach significantly reduces the amount of probes needed in the design at the cost of an increased running time. The absolute running times are in the range of 50–1700 s and hence quite practical.

In [4] we did not compute the group separations for all cases and stated that we expect only a moderate increase in the design size. In this work, we implemented the group separation for disjoint groups and investigate the effect on the data sets (M), (a)1 and two new data sets. The first new data set consists of 1165 rRNA Sequences that were obtained from the Ribosomal Database Project [1]. This set contains sequences of the Large Ribosomal Subunit from different organisms and of varying size. We also created another artificial data set with REFORM. The evolutionary model was the same as used for (a)1. However, the branch lengths of the first three levels were changed from 1.0 to 0.5. This change leads to rather homologous sequences. In the following we call these data sets (LSU) and (cl)1, respectively.

The results are shown in Table 3. Apparently the group separation had no influence on the designs of the two data sets (M) and (LSU) containing rRNA sequences. The design remains exactly the same and no new probes are included. This implies that the pairwise separation was already optimal for guaranteeing the group separation.

Furthermore, the additional random separation of the greedy algorithm has also almost no effect for all data sets, even though we generated 500,000 random pairs of groups. For the artificially generated data sets the group separation

Table 4
Decoding results for the greedy heuristic design and the ILP design on the Markmann [6] data set (M)

Heuristic design for (M)					
$k=$	1	2	3	4	5
Top 1	**0.92**	–	–	–	–
Top 2	**0.98**	**0.93**	–	–	–
Top 3	**0.98**	0.96	**0.94**	–	–
Top 4	1.00	**0.98**	0.95	0.87	–
Top 5	1.00	**0.98**	**1.00**	0.90	**0.92**
Top 10	1.00	0.98	1.00	0.94	**0.98**
ILP design for (M)					
$k=$	1	2	3	4	5
Top 1	0.86	–	–	–	–
Top 2	0.90	0.92	–	–	–
Top 3	0.96	0.96	0.91	–	–
Top 4	1.00	0.97	**0.98**	**0.88**	–
Top 5	1.00	0.97	0.99	**0.95**	0.83
Top 10	1.00	**0.99**	1.00	**1.00**	0.92

Reading example: the value 0.93 for $k = 2$ targets among "top 2" implies that 93 of the 100 true positives in the 50 repetitions of the Monte Carlo experiment were ranked among the top two by the decoding. Similarly, 98 of the 100 true positives were found among those ranked first to fourth.

Table 5
Decoding results for artificial data set (a)1

Heuristic design for (a)1					
$k=$	1	2	3	4	5
Top 1	0.98	–	–	–	–
Top 2	0.98	**1.00**	–	–	–
Top 3	0.98	**1.00**	**0.99**	–	–
Top 4	0.98	**1.00**	**1.00**	**0.93**	–
Top 5	0.98	**1.00**	**1.00**	**0.95**	**0.82**
Top 10	0.98	**1.00**	**1.00**	**0.97**	**0.91**
ILP design for (a)1					
$k=$	1	2	3	4	5
Top 1	**1.00**	–	–	–	–
Top 2	**1.00**	0.99	–	–	–
Top 3	**1.00**	0.99	0.95	–	–
Top 4	**1.00**	0.99	0.97	0.92	–
Top 5	**1.00**	0.99	0.97	0.93	0.6
Top 10	**1.00**	0.99	0.97	0.97	0.75

See Table 4 for further explanations.

leads only to a small increase in the design size. It should be noted that the separation routine takes considerable time. For all data sets the algorithms needed 5 h on average to find a solution.

The solution of both ILPs, with and without group separation, reduces the number of probes considerably. The question remains whether this reduction has any impact on the ability to decode the experiments. We do not expect to do better than the heuristic (except for random fluctuations in the Monte Carlo algorithm), but we expect to be almost as good with the minimal probe set as with the much larger heuristic probe set.

Let us first consider the designs computed by the pairwise separation only (ILP1). We chose a false positive and false negative rate of 5% and ran the decoding for each number of positives 50 times with different randomly chosen targets in the sample. Table 4 shows the result of the decoding procedure for data set (M), Table 5 for a representative of the first artificial data set (a), and Table 6 for a representative of the second artificial data set (b). The tables show for k true positives (first row) the percentage of true positive targets found at the first position (top 1), among the first two positions (top 2), etc. For ease of reading the best values are in boldface.

Table 6
Decoding results for artificial data set (b)3

Heuristic design for (b)3					
$k=$	1	2	3	4	5
Top 1	0.98	–	–	–	–
Top 2	1.00	0.99	–	–	–
Top 3	1.00	0.99	0.96	–	–
Top 4	1.00	0.99	0.96	0.96	–
Top 5	1.00	0.99	0.96	**0.98**	**0.78**
Top 10	1.00	0.99	0.96	0.98	**0.90**
ILP design for (b)3					
$k=$	1	2	3	4	5
Top 1	0.98	–	–	–	–
Top 2	1.00	**1.00**	–	–	–
Top 3	1.00	**1.00**	**0.97**	–	–
Top 4	1.00	**1.00**	**0.98**	0.96	–
Top 5	1.00	**1.00**	**0.99**	0.97	0.70
Top 10	1.00	**1.00**	**0.99**	0.98	0.84

See Table 4 for further explanations.

It can be clearly seen that the ILP solution—remember that it contains often less than half of the probes of the heuristic solution—does still have excellent decoding capabilities, indeed it is sometimes slightly better than the heuristic. Also it can be seen that for five true positives the decoding capability of our solution is indeed worse than that of the heuristic. Obviously the pairwise separation is not sufficient for larger k. The ILP solution has then more problems than the heuristic solution (which has many more probes).

We next investigate whether the decoding capabilities become indeed better when we add more probes to guarantee the group separation. The corresponding tables for the designs with group separation (ILP2) are given as Tables 7 and 8.

The results show that the decoding performance increases as expected if the group separation version of the ILP algorithm is used. Especially for four and five true positives the improved ILP solution is able to separate the targets much better.

The question is now why the group separation performs differently on the used data sets. Further investigation showed that the results correlate with the percentage of unique probe candidates: The data set a(1), which had the largest improvement in decoding capabilities, was the one with the lowest fraction of unique probe candidates.

Another question that arises is whether the use of virtual probes to guarantee feasibility of the solution has an impact on the decoding capabilities of the design. We adapt two strategies: in the first one we globally reduce the coverage constraint until we can solve the ILP without the use of virtual probes; in the second variant, we specifically relax some group inequalities that make the ILP infeasible. We could not find that the use of virtual probes significantly alters the decoding capabilities (data not shown).

4. Conclusions

We have presented an exact approach to the problem of selecting non-unique probes. We have given two different integer linear programming formulations for the problem, one based on adding virtual probes to overcome problems with unseparable target sets and a more elegant one without virtual probes. Based on the formulations, we have developed a branch-and-cut formulation for solving the probe selection problem in the general case. Further research should investigate the complexity status of the separation problem.

Our implementation is able to separate all pairs of targets optimally in reasonable computation time and achieves a considerable reduction of the numbers of probes needed compared to a previous greedy algorithm. The size reduction has only a mild effect on the decoding capabilities of the design, as verified with Monte Carlo simulations.

These results reinforce the findings of [13], namely that a group testing approach using non-unique probes is capable of accurately assessing the presence of small target sets, even when minimizing the cardinality of the probe set. Our

Table 7
Comparison of decoding capabilities for data set (a)1 using the heuristic design, the ILP design with group separation and the ILP design without group separation

k	Top 1	Top 2	Top 3	Top 4	Top 5	Top 10	Top 20
(a)1, Heuristic design, 5% error rate							
1	0.990	1.000	1.000	1.000	1.000	1.000	1.000
2	–	1.000	1.000	1.000	1.000	1.000	1.000
3	–	–	0.994	1.000	1.000	1.000	1.000
4	–	–	–	0.950	0.962	0.970	0.970
5	–	–	–	–	0.830	0.912	0.912
(a)1, ILP design (group sep.), 5% error rate							
1	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2	–	0.985	0.995	1.000	1.000	1.000	1.000
3	–	–	0.983	0.993	0.993	0.997	1.000
4	–	–	–	0.910	0.930	0.968	0.990
5	–	–	–	–	0.646	0.826	0.902
(a)1, ILP design (no group sep.), 5% error rate							
1	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2	–	0.990	0.990	0.990	0.990	0.990	0.990
3	–	–	0.947	0.967	0.967	0.973	0.973
4	–	–	–	0.915	0.930	0.965	0.965
5	–	–	–	–	0.600	0.752	0.752

Reading example: averaged over all experiments with $k = 5$ true targets, we found 91.2% of the true targets among the top 10 predicted ones in the heuristic design (1165 probes; cf. Table 3). The success rate decreases to 82.6% for the ILP design with group separation (515 probes) and to 75.2% without group separation (503 probes).

Table 8
Comparison of decoding capabilities for data set (cl)1, similar to Table 7

k	Top 1	Top 2	Top 3	Top 4	Top 5	Top 10	Top 20
(cl)1, Heuristic design, 5% error rate							
1	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2	–	1.000	1.000	1.000	1.000	1.000	1.000
3	–	–	1.000	1.000	1.000	1.000	1.000
4	–	–	–	0.985	0.988	1.000	1.000
5	–	–	–	–	0.868	0.976	0.996
(cl)1, ILP design (group sep.), 5% error rate							
1	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2	–	1.000	1.000	1.000	1.000	1.000	1.000
3	–	–	0.983	0.987	0.990	1.000	1.000
4	–	–	–	0.927	0.948	0.980	0.995
5	–	–	–	–	0.770	0.934	0.990
(cl)1, ILP design (no group sep.), 5% error rate							
1	0.990	1.000	1.000	1.000	1.000	1.000	1.000
2	–	0.990	0.995	0.995	0.995	0.995	0.995
3	–	–	0.993	1.000	1.000	1.000	1.000
4	–	–	–	0.917	0.938	0.978	0.993
5	–	–	–	–	0.712	0.894	0.950

approach surpasses previous optimization approaches to probe selection (e.g. [11]), as we can cope with multiple targets simultaneously present in a sample. This will almost always be the case in real applications.

Furthermore, experiments with our group separation implementation show that enforcing the separability between small groups adds only a small number of probes while improving the decoding capabilities (as conjectured by Klau et al. [4]).

The software is available to the community (see http://www.inf.fu-berlin.de/inst/ag-bio).

Acknowledgments

The authors thank Dietmar Ebner for help with implementing the experimental setup used in Section 3.3, Diana Poensgen as well as Ralf Borndörfer for helpful discussions, Ole Schulz-Trieglaff who conducted the new computational experiments described in Section 3.3, and two anonymous referees for valuable comments.

References

[1] J. Cole, B. Chai, T. Marsh, R. Farris, Q. Wang, S. Kulam, S. Chandra, D. McGarrell, T. Schmidt, G. Garrity, J. Tiedje, The ribosomal database project (RDP-II): previewing a new autoaligner that allows regular updates and the new prokaryotic taxonomy, Nucleic Acids Res. 31 (2003) 442–443.
[2] ILOG, Inc., 1987–2004. CPLEX, ⟨http://www.ilog.com/products/cplex⟩.
[3] T.H. Jukes, C.R. Cantor, Evolution of protein molecules, in: H.N. Munro (Ed.), Mammalian Protein Metabolism, Academic Press, New York, 1969, pp. 21–132.
[4] G.W. Klau, S. Rahmann, A. Schliep, M. Vingron, K. Reinert, Optimal robust non-unique probe selection using integer linear programming, in: Proceedings of the 12th International Conference on Intelligent Systems for Molecular Biology (ISMB-04), 2004.
[5] E. Knill, A. Schliep, D.C. Torney, Fall, Interpretation of pooling experiments using the Markov chain Monte Carlo method, J. Comput. Biol. 3 (3) (1996) 395–406.
[6] M. Markmann, Entwicklung und Anwendung einer 28S rDNA-Sequenzdatenbank zur Aufschlüsselung der Artenvielfalt limnischer Meiobenthosfauna im Hinblick auf den Einsatz moderner Chiptechnologie, Ph.D. Thesis, University of Munich, 2000.
[7] S. Rahmann, Rapid large-scale oligonucleotide selection for microarrays, in: Proceedings of the First IEEE Computer Society Bioinformatics Conference (CSB), IEEE, New York, 2002.
[8] S. Rahmann, Fast and sensitive probe selection for DNA chips using jumps in matching statistics, in: Proceedings of the Second IEEE Computational Systems Bioinformatics (CSB'03) Conference, IEEE, New York, 2003.
[9] S. Rahmann, Fast large scale oligonucleotide selection using the longest common factor approach, J. Bioinformatics and Comput. Biol. 1 (2) (2003) 343–361.
[10] S. Rahmann, REFORM (Random Evolutionary FORests Modeling software), 2003, Available at ⟨http://www.molgen.mpg.de/∼rahmann⟩.
[11] S. Rash, D. Gusfield, String barcoding: uncovering optimal virus signatures, in: Proceedings of RECOMB 2002, April 2002.
[12] J. SantaLucia, A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics, Proc. Nat. Acad. Sci. USA 95 (1998) 1460–1465.
[13] A. Schliep, D.C. Torney, S. Rahmann, Group testing with DNA chips: generating designs and decoding experiments, in: Proceedings of the Second IEEE Computer Society Bioinformatics Conference (CSB 2003), IEEE, New York, 2003.
[14] X. Wang, B. Seed, Selection of oligonucleotide probes for protein coding sequences, Bioinformatics 19 (2003) 796–802.
[15] L.A. Wolsey, Integer Programming, Wiley Interscience Series in Discrete Mathematics and Optimization, Wiley, New York, 1998.