BIOINFORMATICS

Vol. 20 Suppl. 1 2004, pages i186–i193
DOI: 10.1093/bioinformatics/bth936

Optimal robust non-unique probe selection using Integer Linear Programming

Gunnar W. Klau[1], Sven Rahmann[2,3,†], Alexander Schliep[2], Martin Vingron[2] and Knut Reinert[3,*]

[1]Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstraße 9-11, 1040 Vienna, Austria, [2]Computational Molecular Biology, Max Planck Institute for Molecular Genetics, Ihnestraße 73, D-14195 Berlin, Germany and [3]Algorithmic Bioinformatics, Free University Berlin, Takustraße 9, 14195 Berlin, Germany

Received on January 15, 2004; accepted on March 1, 2004

ABSTRACT

Motivation: Besides their prevalent use for analyzing gene expression, microarrays are an efficient tool for biological, medical and industrial applications due to 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. Each hybridization experiment determines whether the probe binds to its corresponding sequence in the target. Depending on the problem, the experiments are conducted using either unique or non-unique probes and usually assume that only one target is present in the sample. The problem at hand is to compute a design, i.e. a minimal set of probes that allows to infer the targets in the sample from the result of the hybridization experiment. 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.

Results: 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 ILP formulation and a branch-and-cut algorithm. Our preliminary implementation greatly reduces the number of probes needed while preserving the decoding capabilities.

Availability: http://www.inf.fu-berlin.de/inst/ag-bio

Contact: reinert@inf.fu-berlin.de

1 INTRODUCTION

Microarrays are a widely used tool as 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 different and likewise important application, arising for example in medicine, environmental sciences, industrial quality control or biothreat reduction, is the identification of biological agents in a sample.

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

This wide range of applications leads to the same methodological problem: to determine the presence or absence of targets—viruses or bacteria—in a biological sample.

Our paper 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 8–25, hybridizes 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 simultaneously present viruses where we want to determine those which are indeed contained in the sample.

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. However, finding unique probes is often difficult, e.g. in the case of closely related virus subtypes. One way around this basic problem is to devise 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, which contains a 1 if probe

[*]To whom correspondence should be addressed.

[†]Present address: Genome Informatics, Faculty of Technology, University of Bielefeld, D-33595 Bielefeld, Germany.

j hybridizes to target i. Then we will observe the hybridization of all probes incident to any target present in the sample, i.e. the logical OR of the row vectors. If the probe set is not carefully chosen, this can easily lead to situations where we cannot resolve the experimental result.

We illustrate the problem using the example in Table 1. We have four targets $t_1, \ldots, t_4$ and a total of nine probes $p_1, \ldots, p_9$. The target–probe incidence matrix H indicates which probe hybridizes to which target.

Assume we are given a sample and only one of $t_1, \ldots, t_4$ is in the sample. The goal is now 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 very reasonable objective function, since it is proportional to the cost of the experiment.

Now assume that target t_2 and target t_3 are simultaneously in the sample. In this case all three probes p_1, p_2 and p_3 hybridize. This situation cannot be distinguished from the situation where only t_1 is in the sample. 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 the probes is, however, not necessary: we do not lose resolution using only probes p_1, p_4, p_5, p_6 and p_8.

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 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_p (i.e. the experiment reports a hybridization although there should be none) and the false-negative rate f_n (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.

Moreover, it is not trivial to compute the target–probe incidence matrix H in the beginning. Among the potentially very 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 (e.g. Wang and Seed, 2003). In this paper, we use the method proposed by Rahmann (2002) to derive the initial target–probe incidence matrix H. We emphasize, however, that our results do not depend on this choice. The aim of this paper is to compute the design matrix D given some target–probe incidence matrix H.

While we strive to minimize the number of probes, we do not want to lose the ability to decode, i.e. we want to be able to infer the original targets even in the presence of errors.

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 Schliep *et al.* (2003). In this work, we address the second step, the computation of the design, and use for the first and third step the methods proposed in the above-mentioned paper, adopting its notation to a large extent. Having illustrated the problem we formalize it now.

Problem Definition. We denote the m target sequences by t_i ($i \in M := \{1, \ldots, m\}$) and the n candidate probes by p_j ($j \in N := \{1, \ldots, n\}$), and define a 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. So $D_{ij} = 1$ if target t_i hybridizes to the selected probe p_j.

The set of probes hybridizing to target t_i, i.e. the index set of non-zero entries in row i of the incidence matrix D (or 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 non-zero entries in column j of D.

Schliep *et al*. (2003) describe a fast heuristic that allows the computation of a good design, and we describe it shortly:

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 $|P(S) \Delta P(T)| \geq d$.*

Consider the example in Table 1. According to the above definition the sets $S = \{t_1, t_2\}$ and $T = \{t_3, t_4\}$ are 2-separable using a subset of the nine probes (e.g. p_1 and p_9).

The following procedure is proposed to greedily compute a design that guarantees d-separability for all pairs of targets if possible. 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 $i \in M$.

(2) Ensure that all pairs of targets are separated by at least d oligos by calling Separate$(\{t_i\}, \{t_{i'}\}, d)$ for all $1 \leq i < i' \leq m$.

(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 $d' < d$.

```
Separate$(S, T, d)$
Add oligos to the current partial design $D$ to $d$-separate $S$ and $T$
(1) Let $C := P(S) \Delta P(T)$
(2) Partition $C$ into $C = C_D \cup C'$, where $C_D := C \cap D$, and $C'$ contains the separating oligos not yet included in $D$
(3) if $|C_D| \geq d$ then return (nothing to do)
(4) if $|C'| < (d - |C_D|)$ then warn 'Can only $(|C_D| + |C'|)$ separate $S$ and $T$'
(5) Add the $d - |C_D|$ highest-quality probes from $C'$ to $D$
```

This approach is simple and very practical. However, since a design for a microarray is only done once, the time spent to compute the design is far less crucial than the size and quality of the design, i.e. 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.

Note that this is quite different from the ILP approach taken by Rash and Gusfield (2002) that addresses a different problem and considers only a pairwise separation of targets.

2 ILP FORMULATION

PROBLEM 1. *Given a target–probe incidence matrix H with non-unique probes and two parameters* minimum coverage c_{min} *and* minimum Hamming distance h_{min}, *find a minimal set of probes, such that all targets are covered by at least c_{min} probes and such that all targets are separated with Hamming distance at least h_{min}*[1].

It can be shown that Problem 1 is NP-hard using a reduction from the set cover problem. We formulate the problem as a variation of a set cover ILP. Let x_j, $j \in N$, be a set of binary variables with $x_j = 1$ if probe p_j is chosen and 0 otherwise, and let $P := \binom{M}{2} = \{\{i, k\} \in \mathbb{Z} \times \mathbb{Z} \mid 1 \leq i < k \leq m\}$ be the set of 2-subsets of target indices. Then the problem can be formulated as the following integer linear program which we refer to as the master ILP:

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} x_j && \text{(master ILP)} \\ \text{s. t.} \quad & \sum_{j=1}^{n} H_{ij} x_j \geq c_{\min} && \forall i \in M, \\ & \sum_{j=1}^{n} |H_{ij} - H_{kj}| x_j \geq h_{\min} && \forall (i,k) \in P, \\ & x_j \in \{0,1\} && \forall j = 1, \ldots, n. \end{aligned}$$

Note that it can be easily checked whether it is possible to d-separate all pairs of targets. If not, then the solution set of the above ILP is empty. As a remedy, we consider a variation of the problem: we add a sufficiently large number $l := m \cdot \max\{c_{\min}, h_{\min}\}$ 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 C, i.e. we change the objective in the master ILP to

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

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

The master ILP guarantees the pairwise separation of all targets similar to the greedy heuristic. Solving the ILP, however, leads to the minimal number of oligonucleotides necessary to do this. In the experimental section we show that the difference in the number of oligonucleotides 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. Now, assume we have a collection $\mathcal{S} \subset \{S \mid S \subset M\}$ of subsets of the m targets. Then our goal is to guarantee the Hamming distance constraint for the ω-vectors of all pairs $\{S, T\} \in \binom{\mathcal{S}}{2}$ with $S \cap T = \emptyset$. We call these additional requirements 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.* (2003). 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 (e.g. Wolsey, 1998), a linear programming-based branch and bound technique for solving mixed integer linear programs by dynamically adding violated inequalities (cuts).

[1] The coverage constraints are dominated by the distance constraints if we add an empty target.

$$
\begin{array}{llrr}
\max & \sum_{j\in X}(\sigma_j^0+\sigma_j^1) & \text{(slave ILP)} & \\
\text{s.t.} & \sigma_j^0 \le 1 - s_i & & \\
 & \forall j \in X,\ \forall i \in M : H_{ij} \equiv 1 & & (2) \\
 & \sigma_j^0 \le 1 - t_i & & \\
 & \forall j \in X,\ \forall i \in M : H_{ij} \equiv 1 & & (3) \\
 & \sigma_j^1 \le \sum_{i\in M} H_{ij} s_i & \forall j \in X & (4) \\
 & \sigma_j^1 \le \sum_{i\in M} H_{ij} t_i & \forall j \in X & (5) \\
 & \sigma_j^0 \le \sum_{i:H_{ij}\equiv 0} s_i & \forall j \in X & (6) \\
 & \sigma_j^0 \le \sum_{i:H_{ij}\equiv 0} t_i & \forall j \in X & (7) \\
 & s_i + t_i \le 1 & \forall i \in M & (8) \\
 & 0 \le \sigma_j^0 \le 1 & \forall j \in X & (9) \\
 & 0 \le \sigma_j^1 \le 1 & \forall j \in X & (10) \\
 & s_i \in \{0,1\} & \forall i \in M & (11) \\
 & t_i \in \{0,1\} & \forall i \in M & (12)
\end{array}
$$

Fig. 1. The slave ILP.

2.1 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 $X = \{j \mid x_j^* \equiv 1\}$, i.e. the index set of the currently chosen oligonucleotides. Further, for a target set S, let $\omega^S|_X$ denote the restriction of ω^S to the columns in X. We solve another integer linear program, the slave ILP in Figure 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 $h_{\min}$.

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 after applying the logical OR operation.

Variables σ_j^0 and σ_j^1 model the similarity of S and T at position j (the column of H corresponding to the j-th oligonucleotide), i.e. $\sigma_j^0 = 1$ iff both ω_j^S and ω_j^T are equal to zero and $\sigma_j^1 = 1$ iff both values are equal to one. Besides the trivial constraints (9)–(12) and inequality (8), which keeps S and T disjoint, we have three main classes of inequalities: The first class, given by inequalities (2) and (3), models the fact that σ_j^0 cannot be one if S or T contain a target that hybridizes to oligonucleotide 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 $T = \emptyset$. Note that it is easy to limit the cardinalities of S and T by adding the inequalities $\sum_i s_i \le c_1$ and $\sum_i t_i \le c_2$ for some constants c_1 and c_2.

LEMMA 1. *A feasible solution $(s, t, \sigma^0, \sigma^1)$ of the slave ILP for a partial design characterized by X corresponds to two disjunct 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.

The proof of the above lemma is omitted in this extended abstract.

If h is smaller than the minimum required Hamming distance $h_{\min}$, we have found a violated group inequality, namely

$$\sum_{j=1}^{n+l} |\omega_j^S - \omega_j^T| x_j \ge h_{\min}.$$

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 oligonucleotides is well-suited to additionally distinguish between similar groups of targets that might be present in the sample.

3 EXPERIMENTAL VALIDATION

Schliep *et al.* (2003) tested their greedy heuristic on a set of 1230 28S rDNA sequences from different organisms present in the Meiobenthos (Markmann, 2000). 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 dataset and report on the results. Additionally, in order to evaluate the benefits of our new method more systematically, we also generate artificial datasets and compare the results of our method against the result of the greedy heuristic.

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 (http://www.molgen.mpg.de/~rahmann) 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

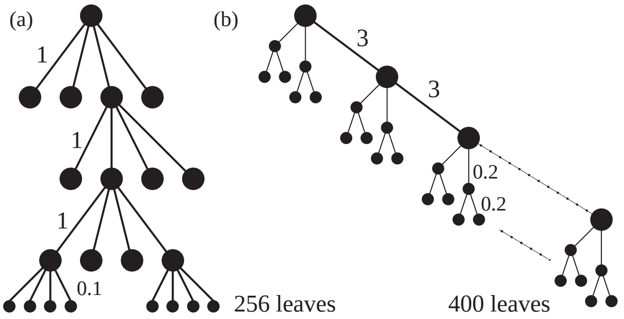

Fig. 2. The two evolutionary tree models used for sequence family generation. Branch lengths are indicated next to sample branches. The 256 respective 400 leaf sequences were taken as family members. In (a) not all children of the nodes are shown.

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); e.g. as the simple model by Jukes and Cantor (1969) that assigns equal probabilities to all mutation types. Alternatively it can be specified as any valid rate matrix $Q = (Q_{ij}) \geq 0$ with $Q_{ii} = -\sum_{j \neq i} Q_{ij}$ generating a Markov process (for $i \neq j$, Q_{ij} is the mutation rate $i \rightarrow j$, where i and j are different nucleotides: $|Q_{ii}|$ then measures the overall mutation rate away from i. Branch lengths are measured in percentage of expected mutations).

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 used two different forest models (Fig. 2). From each model, five independent test sets were generated.

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 five 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 $(8, 1, 4, 2, 1, 0.5, 0.25, 0.125, \dots)$. 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 four children. The distance between adjacent nodes corresponds to $t = 1\%$ 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 (0.2, 2.2, …). The leaf sequences are subsequently used for probe candidate selection.

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 (Rahmann, 2003a,b). Probe candidates are selected to be between 19 and 21 nt long and have a stability (Gibbs energy) of −20 to −19.5 kcal/mol at 40°C and 0.075 M [Na$^+$] according to the Nearest Neighbor model with parameters from SantaLucia (1998).

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 presence of errors the question is, however, whether or not we lose our capability to decode the experiment if we reduce the size of the design. In order to check this we use the method proposed by Schliep *et al*. (2003):

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 large. As iterating over all possible sets of targets is infeasible the following Monte Carlo approach was used.

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—recall, it specifies the incidence of targets and probes—the set of probes which all should hybridize to our sample. 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_n and for true negative probes to positive with probability f_p. This noisy result is used as input to the MCMC-based decoding procedure described by (Schliep *et al*., 2003).

The result of the decoding is a sorted list of the most probable true-positive 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 $1, 2, 3, \ldots$ 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.

Clearly, the performance must degrade as k grows. Even for small k maximal performance cannot be expected for two reasons. First, in the presence of reasonably 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 a perfect result.

We propose to use the proportion of true positives among the top $k+1$ targets for k-cardinality samples up to a maximal value of k suggested as realistic by the specific application.

4 RESULTS

We report on our results using the Markmann (2000) dataset (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.

Designs with minimum coverage 10 and minimum Hamming distance 5 are generated with the greedy heuristic (a part of the Promide package) and by the ILP approach described in Sect. 2, utilizing version 8.1 of the commercial CPLEX software with standard settings (http://www.ilog.com/products/cplex).

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 datasets. Our approach significantly reduces the amount of oligonucleotides 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.

Table 2. For each artificial dataset (a)1 to (b)5 and for the Markmann (2000) 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	15139	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 time are also shown.

Table 3. Decoding results for the greedy heuristic design and the ILP design on the Markmann (2000) dataset (M)

Pos.	1	2	3	4	5
Heuristic design for (M)					
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)					
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

At the time of submission we have not yet computed the group separations for all cases (this takes more implementational effort and will be completed in the near future), preliminary results indicate however only a moderate increase in the number of oligonucleotides needed.

The question remains, whether the reduction in the number of oligonucleotides has any impact on the ability to decode the experiments. We cannot 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.

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 3 shows the result of the decoding procedure for dataset (M), Table 4

Table 4. Decoding results for artificial dataset (a)1

Pos.	1	2	3	4	5
Heuristic design for (a)1					
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					
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

Table 5. Decoding results for artificial dataset (b)3

Pos.	1	2	3	4	5
Heuristic design for (b)3					
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					
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

for a representative of the first artificial dataset (a), and Table 5 for a representative of the second artifical dataset (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 bold.

It can be clearly seen that the ILP solution—remember that it contains often less than half of the oligonucleotides 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. This can be explained by the fact that we currently only conduct the pairwise separation. Hence for larger k we have more problems than the heuristic solution which has many more oligonucleotides. We conjecture that these values will become better once we implement the group separation.

5 CONCLUSIONS

We have presented an exact approach to the problem of selecting non-unique probes. We have formulated the problem as an integer linear program and have developed a branch-and-cut formulation for solving the group separation problem in the general case.

Our preliminary implementation is capable of separating all pairs of targets optimally in reasonable computation time and achieves a considerable reduction of the numbers of oligonucleotides needed compared to a previous greedy algorithm. The drastic size reduction has only a mild effect on the decoding capabilities of the design.

These results reinforce the findings of Schliep *et al*. (2003), namely that probe selection with non-unique probes is capable of accurately assessing the presence of small target sets even when minimizing the cardinality of the probe set. Our approach already surpasses optimization approaches to probe selection (Rash and Gusfield, 2002), as we can cope with multiple targets simultaneously present in a sample. This will almost always be the case for real biological applications.

Experiments with a prototypical group separation implementation let us conjecture that enforcing the separability between small groups will only add a small number of nucleotides while improving the decoding capabilities. We plan to finish the implementation of the group separation algorithm in the near future and to speed up our initial implementation. Especially, we believe that using the slave ILP within a real branch-and-cut framework—i.e. separating violated inequalities also from fractional solutions at each node of the branch-and-bound tree—will reduce the computation time considerably, making optimal group separation up to small cardinalities viable for practical use.

The software will be made available to the community.

ACKNOWLEDGEMENTS

The authors thank Dietmar Ebner for help with implementing the experimental set-up used in Section 3.2 and Diana Poensgen as well as Ralf Borndörfer for helpful discussions.

REFERENCES

ILOG, Inc. (1987–2004) CPLEX.

Jukes,T.H. and Cantor,C.R. (1969) Evolution of protein molecules. In Munro,H.N. (ed.), *Mammalian Protein Metabolism*. Academic Press, New York, pp. 21–132.

Markmann,M. (2000) 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, Munich, Germany.

Rahmann,S. (2002) Rapid large-scale oligonucleotide selection for microarrays. *Proceedings of the First IEEE Computer Society Bioinformatics Conference (CSB)*. Stanford, CA August 2002. IEEE, pp. 54–63.

Rahmann,S. (2003a) Fast and sensitive probe selection for DNA chips using jumps in matching statistics. *Proceedings of the 2nd IEEE Computational Systems Bioinformatics (CSB'03) Conference*. Stanford, CA, August 2003. IEEE, pp. 57–64.

Rahmann,S. (2003b) Fast large scale oligonucleotide selection using the Longest Common Factor Approach. *J. Bioinform. Comput. Biol.*, **1**, 343–361.

Rash,S. and Gusfield,D. (2002) String barcoding: uncovering optimal virus signatures. In Myers,G., Hannenballi,S., Istrail,S., Perzner,P. and Waterman,M. (eds), *Proceedings of the Sixth Annual International Conference on Computational Biology*, Washington DC, USA, 18–21 April 2002. pp. 254–261.

SantaLucia,J. (1998) A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics. *Proc. Natl Acad. Sci., USA*, **95**, 1460–1465.

Schliep,A., Torney,D.C. and Rahmann,S. (2003) Group testing with DNA chips: generating designs and decoding experiments. *Proceedings of the 2nd IEEE Computer Society Bioinformatics Conference (CSB 2003)*. Stanford, CA, August 2003. IEEE, pp. 84–93.

Wang,X. and Seed,B. (2003) Selection of oligonucleotide probes for protein coding sequences. *Bioinformatics*, **19**, 796–802.

Wolsey,L.A. (1998) *Integer Programming. Wiley Interscience Series in Discrete Mathematics and Optimization*. John Wiley & Sons, New York.