ALGORITHMIC APPROACHES TO SELECTING CONTROL CLONES IN DNA ARRAY HYBRIDIZATION EXPERIMENTS*

Q. FU†, E. BENT‡, J. BORNEMAN ‡, M. CHROBAK† and N. YOUNG†

We study the problem of selecting control clones in DNA array hybridization experiments. The problem arises in the OFRG method for analyzing microbial communities. The OFRG method performs classification of rRNA gene clones using binary fingerprints created from a series of hybridization experiments, where each experiment consists of hybridizing a collection of arrayed clones with a single oligonucleotide probe. This experiment produces analog signals, one for each clone, which then need to be classified, that is, converted into binary values 1 and 0 that represent hybridization and non-hybridization events. Besides the sample clones, the array contains a number of control clones needed to calibrate the classification procedure of the hybridization signals. These control clones must be selected with care to optimize the classification process. We formulate this as a combinatorial optimization problem called *Balanced Covering*. We prove that the problem is $\mathbb{NP}$-hard and we show some results on hardness of approximation. We propose an approximation algorithm based on randomized rounding and we show that, with high probability, it approximates well the optimum. The experimental results confirm that the algorithm finds high quality control clones. The algorithm has been implemented and is publicly available as part of the software package called CloneTools.

1. Introduction

Background. We study the problem of selecting control clones for DNA array hybridization experiments. The specific version of the problem that we address arises in the context of the OFRG (Oligonucleotide Fingerprinting of Ribosomal RNA Genes) method. OFRG ([8], [9], [10]) is a technique for analyzing microbial communities that classifies rRNA gene clones into taxonomic clusters based on binary fingerprints created from hybridizations with a collection of oligonucleotide probes. More specifically, in OFRG, clone libraries from a sample under study (e.g., fungi or bacteria from an environmental sample) are constructed using PCR primers. These cloned rRNA gene fragments are immobilized on nylon membranes and then subjected to a series of hybridization experiments, with each experiment using a single radiolabeled DNA oligonucleotide probe. This experiment produces analog signals, one for each clone, which then need to be classified, that is, converted into binary values 1 and 0 that represent hybridization and non-hybridization events. Overall, this process creates a hybridization fingerprint for each clone, which is a vector of binary values indicating which probes bind with this clone and which don't. The clones are then identified by clustering their hybridization fingerprints with those of known sequences and by nucleotide sequence analysis of representative clones within a cluster.

*Research supported by NSF grants CCR-0208856 and BD&I-0133265.
†Department of Computer Science, University of California, Riverside, CA 92521.
‡Department of Plant Pathology, University of California, Riverside, CA 92521.

Besides the sample clones, the array contains a number of *control clones*, with known nucleotide sequences, used to calibrate the classification of hybridization signals. For each probe, two distributions of signal intensities are obtained: one from control clones that match the probe (e.g., they contain its complement and thus should hybridize with it) and the other from clones that don't. This information is then used to determine the signal intensity threshold for clone-probe hybridization: signals above this threshold are interpreted as clone-probe hybridization events, while those below correspond to non-hybridizations.

The quality of information obtained from hybridizations depends critically on the accuracy of the signal classification process. Ideally, each probe should match (bind with) roughly half of the control clones. In prior OFRG work, control clones were selected arbitrarily, often producing control clones with skewed distribution of binding/non-binding with some probes. As an example, from a set of 100 control clones, only two might bind with a specific probe. The signal classification for this probe would be very unreliable, as it would be based on signal intensities from hybridization with only two control clones.

Problem formulation. Our control-clone selection problem can be then formulated as a combinatorial optimization problem called *Balanced Covering* [a]. The instance is given as a pair $\langle G, s\rangle$, where $G = (C, P, E)$ is a bipartite graph and $s \leq |C|$ is an integer. C represents the candidate control clone set, P is the probe set, and the edges in E represent potential hybridizations between clones and probes, that is, $(c, p) \in E$ iff p should hybridize with c. For $p \in P$ and $D \subseteq C$, let $\deg_D(p)$ be the number of neighbors of p in D (that is, the number of clones in D that hybridize with p). Throughout the paper, unless stated otherwise, by m and n we will denote the cardinality of C and P, respectively.

Generally, as indicated earlier, our goal is to find a set $D \subseteq C$ of cardinality s such that, for each $p \in P$, $\deg_D(p)$ is close to $s/2$. Several objective functions can be studied. To measure the deviation from the perfectly balanced cover, for a given probe p, we can compute either $\min\{\deg_D(p), s - \deg_D(p)\}$ or $|\deg_D(p) - s/2|$. The objective function can be obtained by considering the sum of these values or the worst case over all probes. This gives rise to four objective functions: (1) maximize $\mathcal{C}_{min}(D) = \min_{p\in P} \min\{\deg_D(p), s - \deg_D(p)\}$; (2) maximize $\mathcal{C}_{sum}(D) = \sum_{p\in P} \min\{\deg_D(p), s - \deg_D(p)\}$; (3) minimize $\mathcal{D}_{max}(D) = \max_{p\in P} |\deg_D(p) - s/2|$; (4) minimize $\mathcal{D}_{sum}(D) = \sum_{p\in P} |\deg_D(p) - s/2|$, where each function needs to be optimized over all choices of D. By $\mathsf{BCP}\_\mathcal{C}_{min}$, etc., we denote the four optimization problems corresponding to these functions. We focus on $\mathsf{BCP}\_\mathcal{C}_{min}$ and $\mathsf{BCP}\_\mathcal{D}_{sum}$.

Results. In this paper we show several analytical and experimental results on Balanced Covering. In Section 2 we prove that all versions of Balanced Covering are $\mathbb{NP}$-hard. In particular, it is $\mathbb{NP}$-complete to decide whether there is a perfectly balanced cover with s clones. This immediately implies that (unless $\mathbb{P}$=$\mathbb{NP}$), there is no polynomial-time approximation algorithm for $\mathsf{BCP}\_\mathcal{D}_{max}$ and $\mathsf{BCP}\_\mathcal{D}_{sum}$. For $\mathsf{BCP}\_\mathcal{D}_{sum}$, we further show that even if we allow an additional additive term $O((nm)^\epsilon)$ and randomization, approximating the

[a]There have been some discussions on the *Balanced Set Cover* (see [1], [6]) problem, however, they are not directly related to *Balanced Covering* problem discussed in this paper.

optimum is still hard. For BCP_$\mathcal{C}_{min}$, we prove that it cannot be approximated efficiently within the bound $\text{OPT}_{\min}(G,s) - \frac{1}{2}(1-\epsilon)\ln n$, unless $\mathbb{NP}$ has slightly superpolynomial time algorithms.

Then, in Section 3, we propose a polynomial-time randomized rounding algorithm RRBC for BCP_$\mathcal{C}_{min}$. The algorithm solves the linear relaxation of the integer program for BCP_$\mathcal{C}_{min}$, and then uses the solution to randomly pick an approximately balanced cover. We show that, with probability at least $\frac{1}{2}$, RRBC's solution deviates by no more than $O(\sqrt{z^* \ln n} + \sqrt{s})$ from the optimal solution z^* of the linear program.

In Section 5, we present an experimental study of RRBC's performance which shows that its solutions are close to the optimum. More specifically, in 92.8% cases of real data sets, RRBC found solutions at least as large as 97% of the linear relaxation's optimum. The implementation of RRBC is publicly available at the OFRG website as part of the CloneTools software package, see `http://algorithms.cs.ucr.edu/OFRG/`.

Algorithm RRBC performs well for input instances where the optimum is relatively large, but the performance bound given in Section 4 is not quite satisfactory when z^* is small compared to s, say for $z^* = o(\sqrt{s})$. In Section 6, we present another algorithm called RRBC2 which, with probability at least $\frac{1}{2}$, computes a solution that deviates by no more than $O(\sqrt{z^* \ln n})$ from the optimum.

Relation to other work. Note that OFRG differs from other array-based analysis approaches that, typically, involve a single microarray experiment where one clone of interest is hybridized against a collection of arrayed probes, each targeting a specific sequence. These experiments include control clones as well, but these control clones are used to test whether they bind as predicted to particular microarray probes (see [7], for example). In contrast, OFRG uses a small set of probes (roughly 30-50) to coordinately distinguish a much larger set of sequences (e.g., all bacterial rRNA genes). Each probe is used in one hybridization experiment, and the unknown DNA clones are immobilized on the array.

2. Hardness Results

In this section we prove several hardness results for Balanced Covering. We will first show that all versions of Balanced Covering are $\mathbb{NP}$-hard.

$\mathbb{NP}$-hardness. Given a bipartite graph $G = (C, P, E)$ and an even integer s, define a *perfectly balanced cover* in G to be a subset $D \subseteq C$ with $|D| = s$ such that $\deg_D(p) = s/2$ for each $p \in P$.

Theorem 2.1. *The following decision problem is $\mathbb{NP}$-complete: given a bipartite graph $G = (C, P, E)$ and an even integer s, is there a perfectly balanced cover in G? Consequently, Balanced Covering is $\mathbb{NP}$-hard for all objective functions $\mathcal{C}_{min}$, $\mathcal{C}_{sum}$, $\mathcal{D}_{max}$ and $\mathcal{D}_{sum}$.*

Proof. The proof is by a reduction from X3C (Exact Cover by 3-Sets) [5], which is known to be $\mathbb{NP}$-complete. The instance of X3C consists of a finite set X of $3m$ items, and a collection T of n 3-element subsets of X that we refer to as *triples*. We assume $n \geq m \geq 2$. The objective is to determine whether T contains an *exact cover of* X, i.e., a sub-collection $T' \subseteq T$ such that every element of X occurs in exactly one triple in T'.

The reduction is defined as follows. Given an instance $\langle X, T\rangle$ of X3C above, we construct an instance $\langle G = (T \cup W, X, E), s\rangle$ of Balanced Covering, where W is a set that contains $m - 2$ new vertices. For $t \in T$ and $x \in X$, we create an edge $(t, x) \in E$ if $x \in t$. Further, we create all edges $(w, x) \in E$ for $x \in X$ and $w \in W$. We let $s = 2m - 2$.

It remains to show that this construction is correct, namely that $\langle X, T\rangle$ has an exact cover iff $\langle G, s\rangle$ has a perfectly balanced cover.

($\Rightarrow$) If $\langle X, T\rangle$ has an exact cover T', we claim that $D = T' \cup W$ is a perfectly balanced cover for $\langle G, s\rangle$. To justify this, note first that $|T'| = m$ and $|W| = m - 2$, and thus $|D| = 2m - 2 = s$. Further, each vertex $x \in X$ has exactly one neighbor in T' and $m - 2$ neighbors in W, so x has $m - 1 = s/2$ neighbors in D, as required.

($\Leftarrow$) Suppose that $\langle G, s\rangle$ has a perfectly balanced cover $D \subseteq T \cup W$. Denoting $W' = D \cap W$, $k = |W'|$, and $T' = D \cap T$. We first show that D must contain all vertices in W.

We count the edges between D and X. There are $3km$ edges between W' and X, since each vertex in W' is connected to all $3m$ vertices in X. There are $3(s - k)$ edges between T' and X, since each vertex in T has degree 3. On the other hand, there must be $3m(m-1)$ edges between X and D, since each vertex in X must be connected to exactly $s/2 = m-1$ vertices in D. Together, this yields $3(2m - 2 - k) + 3km = 3m(m - 1)$. Solving this equation, we get $k = m-2$, which means that $W' = W$. So T' contains exactly $s-k = m$ vertices, as claimed. Each vertex $x \in X$ is adjacent to all vertices in W, so it has exactly $s/2 - (m - 2) = 1$ neighbor in T'. This means that T' is an exact cover of X. □

Approximation of BCP_$\mathcal{D}_{sum}$. Theorem 2.1 immediately implies that BCP_$\mathcal{D}_{sum}$ (as well as BCP_$\mathcal{D}_{max}$) cannot be approximated with any finite ratio. We show that even if we allow an additive term in the approximation bound, achieving such bound is still $\mathbb{NP}$-hard.

Given an algorithm $\mathcal{A}$ for BCP_$\mathcal{D}_{sum}$, denote by $\mathcal{D}^{\mathcal{A}}_{sum}(G, s)$ the value of the objective function computed by $\mathcal{A}$ on instance $\langle G, s\rangle$, that is, $\mathcal{D}^{\mathcal{A}}_{sum}(G, s) = \sum_{p\in P} |\deg_D(p) - s/2|$, where D is the set computed by $\mathcal{A}$. By $\mathcal{D}^{*}_{sum}(G, s)$ we denote the optimal value. Recall that $m = |C|$ and $n = |P|$.

Theorem 2.2. *Assuming $\mathbb{P} \neq \mathbb{NP}$, there is no polynomial-time algorithm $\mathcal{A}$ for BCP_$\mathcal{D}_{sum}$ that for any instance $\langle G, s\rangle$ satisfies $\mathcal{D}^{\mathcal{A}}_{sum}(G, s) \leq \alpha\cdot\mathcal{D}^{*}_{sum}(G, s)+\beta(mn)^{\epsilon}$, for any $\alpha, \beta > 0$ and $0 < \epsilon < 1$.*

Proof. Suppose, towards contradiction, that for some α, β and ϵ there is a polynomial-time algorithm $\mathcal{A}$ satisfying the theorem. We show that this would imply the existence of a polynomial-time algorithm that decides if there is a perfectly balanced covering, contradicting Theorem 2.1.

Given an instance $\langle G, s\rangle$ of BCP_$\mathcal{D}_{sum}$, where $G = (C, P, E)$, convert it into another instance $\langle G^r, s\rangle$ of BCP_$\mathcal{D}_{sum}$, where $G^r = (C, P', E')$ is obtained by creating r identical copies of each probe $p \in P$ (that is, with the same neighbors in C). Thus $|P'| = rn$. Without loss of generality, assume $\beta > 1$ and $mn \geq 2$. We choose $r = \lceil (mn)^{\delta}\rceil + 1$, for $\delta = (\epsilon + \log\beta)/(1 - \epsilon)$. For this r, we have $r > \beta(mnr)^{\epsilon}$. Therefore $\langle G^r, s\rangle$ satisfies:

• If $\langle G, s\rangle$ has a perfectly balanced cover (that is, $\mathcal{D}^*_{sum}(G, s) = 0$) then $\mathcal{D}^*_{sum}(G^r, s) = 0$ as well, and thus $\mathcal{D}^{\mathcal{A}}_{sum}(G^r, s) \leq \beta(mnr)^{\epsilon} < r$.

• If $\langle G, s\rangle$ does not have a perfectly balanced cover (that is, $\mathcal{D}^*_{sum}(G, s) \geq 1$) then $\mathcal{D}^{\mathcal{A}}_{sum}(G^r, s) \geq \mathcal{D}^*_{sum}(G^r, s) = r \cdot \mathcal{D}^*_{sum}(G, s) \geq r$.

So we get $\mathcal{D}^{\mathcal{A}}_{sum}(G^r, s) < r$ if and only if $\mathcal{D}^*_{sum}(G, s) = 0$. Since G^r can be obtained from G in polynomial time, this would yield a polynomial-time algorithm that determines the existence of a perfectly balanced cover, contradicting Theorem 2.1. $\square$

Next, we show that approximating BCP_$\mathcal{D}_{sum}$ is hard even with randomization. Recall that class $\mathbb{RP}$ (randomized polynomial time) is the complexity class of decision problems $\mathcal{P}$ which have polynomial-time probabilistic Turing machines M such that, for each input I, (i) if $I \in \mathcal{P}$ then M accepts I with probability at least $\frac{1}{2}$, and (ii) if $I \notin \mathcal{P}$ then M rejects I with probability 1. It is still open whether $\mathbb{RP} = \mathbb{NP}$.

Theorem 2.3. *Assuming $\mathbb{RP} \neq \mathbb{NP}$, there is no randomized polynomial-time algorithm $\mathcal{A}$ for BCP_$\mathcal{D}_{sum}$ that for any instance $\langle G, s\rangle$ satisfies Exp$[\mathcal{D}^{\mathcal{A}}_{sum}(G, s)] \leq \alpha \cdot \mathcal{D}^*_{sum}(G, s) + \beta(mn)^{\epsilon}$, for any $\alpha, \beta > 0$ and $0 < \epsilon < 1$.*

Proof. Suppose, towards contradiction, that there exists a randomized polynomial-time algorithm $\mathcal{A}$ that satisfies the theorem. Given an instance $\langle G, s\rangle$ of BCP_$\mathcal{D}_{sum}$, we again convert it into an instance $\langle G^r, s\rangle$ of BCP_$\mathcal{D}_{sum}$, as in the proof of Theorem 2.2. Without loss of generality, assume $\beta > 1$ and $mn \geq 2$. We choose $r = \lceil (mn)^{\delta} \rceil + 1$, for $\delta = (\epsilon + \log\beta + 1)/(1-\epsilon)$. Note that $r > 2\beta(mnr)^{\epsilon}$. We now make the following observations.

• If $\langle G, s\rangle$ has a perfectly balanced cover (that is, $\mathcal{D}^*_{sum}(G, s) = 0$) then $\mathcal{D}^*_{sum}(G^r, s) = 0$, and therefore 2Exp$[\mathcal{D}^{\mathcal{A}}_{sum}(G^r, s)] \leq 2\beta(mnr)^{\epsilon} < r$. Using Markov's inequality, this implies that Pr$[\mathcal{D}^{\mathcal{A}}_{sum}(G^r, s) < r] \geq \frac{1}{2}$.

• If $\langle G, s\rangle$ does not have a perfectly balanced cover (that is, $\mathcal{D}^*_{sum}(G, s) \geq 1$) then $\mathcal{D}^{\mathcal{A}}_{sum}(G^r, s) \geq \mathcal{D}^*_{sum}(G^r, s) \geq r \cdot \mathcal{D}^*_{sum}(G, s) \geq r$, with probability 1.

Thus from $\mathcal{A}$ we could obtain a randomized polynomial-time algorithm that determines the existence of a perfectly balanced cover, contradicting Theorem 2.1. $\square$

Approximation of BCP_$\mathcal{C}_{min}$. Next we show that BCP_$\mathcal{C}_{min}$ cannot be approximated efficiently with the bound $\mathcal{C}^*_{min}(G, s) - \frac{1-\epsilon}{2}\ln n$, unless $\mathbb{NP}$ has slightly superpolynomial time algorithms. Recall that $\mathcal{C}^*_{min}(G, s)$ is the optimal value of $\mathcal{C}_{min}$ on an instance $\langle G, s\rangle$ of BCP_$\mathcal{C}_{min}$. $\mathcal{C}^{\mathcal{A}}_{min}(G, s)$ is the value of the objective function computed by an algorithm $\mathcal{A}$.

Theorem 2.4. *Unless problems in $\mathbb{NP}$ have $n^{O(\log\log n)}$-time deterministic algorithms, there is no polynomial-time algorithm $\mathcal{A}$ for BCP_$\mathcal{C}_{min}$ that, for some $0 < \epsilon < 1$, for any instance $\langle G, s\rangle$, satisfies $\mathcal{C}^{\mathcal{A}}_{min}(G, s) \geq \mathcal{C}^*_{min}(G, s) - \frac{1-\epsilon}{2}\ln n$.*

Proof. Suppose, towards contradiction, that there exits a polynomial-time algorithm $\mathcal{A}$ that satisfies the theorem. We show that this would imply the existence of a polynomial-time $((1 - O(1))\ln n)$-approximation algorithm $\mathcal{B}$ for the Set Cover problem, which would imply in turn that problems in $\mathbb{NP}$ have $n^{O(\log\log n)}$-time deterministic algorithms [4].

Given an instance $\langle Q, X\rangle$ of set cover, where Q is a collection of sets over universe $X = \{1, 2, \ldots, n\}$, the algorithm $\mathcal{B}$ first reduces it to an instance $\langle G, s\rangle$ of $\mathsf{BCP}\_\mathcal{C}_{min}$, then calls algorithm $\mathcal{A}$ on input $\langle G, s\rangle$, and finally converts the solution of $\mathcal{A}$ to a set cover of X. We claim that this cover will be of size at most $(1 - \Omega(\epsilon)) \ln(n)b$, where b is the size of the minimum set cover of X. Below we assume that, without loss of generality, algorithm $\mathcal{B}$ knows the optimal value b of $\langle Q, X\rangle$. Otherwise, $\mathcal{B}$ can simply try each $b \in \{1, 2, ..., n\}$, and choose the smallest set cover.

$\mathcal{B}$ reduces $\langle Q, X\rangle$ to an instance $\langle G = (T \cup W, P, E), s\rangle$ of $\mathsf{BCP}\_\mathcal{C}_{min}$, where $P = X \cup \{0\}$, T contains k vertices $q_1, q_2, ..., q_k$ for each set $q \in Q$ and $k = \lceil \frac{\ln n}{2} \rceil$, and W is a set containing k new vertices. For each $q \in Q$ and $i = 1, 2, ..., k$, we create an edge $(q_i, 0) \in E$ and edges $(q_i, x) \in E$ if $x \in q$. We let $s = kb + k$.

We claim that $\mathcal{C}^*_{min}(G, s) \geq k$ if $\langle Q, X\rangle$ has a set cover Q' of size b. To justify this, from Q', we build the following balanced cover D: add k vertices $q_1, q_2, ..., q_k$ to D if $q \in Q'$, and all vertices in W to D; thus $|D| = kb + k = s$. For each $x \in X$, the k copies of Q' ensure that $\deg_D(x) \geq k$, while the k vertices in W ensure that $\deg_D(x) \leq s - k$.

If $\langle Q, X\rangle$ has a set cover of size b, then algorithm $\mathcal{A}$ on input $\langle G, s\rangle$ will return a balanced cover D with objective function value at least $\mathcal{C}^*_{min}(G, s) - \frac{1-\epsilon}{2} \ln n \geq k - (1 - \epsilon)k = \epsilon k$. We have $|D \cap W| \geq \epsilon k$, because $0 \in X$ is adjacent to every vertex in T and D has at least ϵk vertices not adjacent to 0. Therefore $|D \cap T| \leq s - \epsilon k = kb + (1 - \epsilon)k$. Thus, since each $x \in P$ is adjacent to at least one vertex in D (in fact, at least ϵk), the collection of sets $\{q : (\exists i) q_i \in D\}$ forms a set cover of X of size at most $kb + (1 - \epsilon)k \leq (2 - \epsilon)kb \leq (1 - \epsilon/2)(\ln(n) + O(1))b \leq (1 - \Omega(\epsilon)) \ln(n)b$, as claimed.

The algorithm $\mathcal{B}$ clearly runs in polynomial time, and is a $((1 - O(1)) \ln n)$-approximation algorithm for the Set Cover problem. Thus the theorem follows. □

3. A Randomized Rounding Algorithm

In this section we present our randomized algorithm RRBC for $\mathsf{BCP}\_\mathcal{C}_{min}$. Given $G = (C, P, E)$, let A be the $m \times n$ adjacency matrix of G, that is $a_{ij} = 1$ iff $(c_i, p_j) \in E$; otherwise $a_{ij} = 0$. Then $\mathsf{BCP}\_\mathcal{C}_{min}$ is equivalent to the integer linear program MinIP:

$$\begin{array}{rrcl} \text{maximize:} & z & & \\ \text{subject to:} & z & \leq & \sum_{i=1}^m a_{ij} x_i \quad \forall j = 1, ..., n \\ & z & \leq & \sum_{i=1}^m (1 - a_{ij}) x_i \quad \forall j = 1, ..., n \\ & \sum_{i=1}^m x_i & \leq & s \\ & x_i & \in & \{0, 1\} \quad \forall i = 1, ..., m \end{array}$$

RRBC first relaxes the last constraint to $0 \leq x_i \leq 1$ to obtain the linear program MinLP, and then computes an optimal solution x^*_i, $i = 1, 2, ..., m$, of MinLP. Next, applying randomized rounding, RRBC computes an integral solution $X_1, ..., X_m$ by choosing $X_i = 1$ with probability x^*_i and 0 otherwise. Note that this solution may not be feasible since $\sum_{i=1}^m X_i$ may exceed s. Let $L = \sum_{i=1}^m X_i - s$. If $L > 0$, RRBC changes L arbitrary variables $X_i = 1$ to 0, obtaining a feasible solution $\tilde{X}_1, ...\tilde{X}_m$.

4. Analysis of RRBC for $\mathsf{BCP\_}\mathcal{C}_{min}$

We denote by $\mathcal{C}^{\text{RRBC}}_{min}(G, s)$ the value of the objective function computed by RRBC, that is $\mathcal{C}^{\text{RRBC}}_{min}(G, s) = \tilde{Z} = \min_{j=1}^{n}\{\sum_{i=1}^{m} a_{ij}\tilde{X}_i, \sum_{i=1}^{m}(1 - a_{ij})\tilde{X}_i\}$.

Lemma 4.1. *For any instance $\langle G, s\rangle$ of $\mathsf{BCP\_}\mathcal{C}_{min}$, with probability at least $\frac{1}{2}$, $\mathcal{C}^{RRBC}_{min}(G, s) \;\geq\; \mathcal{C}^{*}_{min}(G, s) - O\left(\sqrt{\mathcal{C}^{*}_{min}(G, s)\ln n} + \sqrt{s}\right)$.*

Proof. Let $z^* = \min_{j=1}^{n}\{\sum_{i=1}^{m} a_{ij}x_i^*, \sum_{i=1}^{m}(1 - a_{ij})x_i^*\}$ be the optimum solution of MinLP. Let also $Z = \min_{j=1}^{n}\{\sum_{i=1}^{m} a_{ij}X_i, \sum_{i=1}^{m}(1 - a_{ij})X_i\}$.

The $\{X_i\}$ are independent random variables with Exp$[X_i] = x_i^*$ so, for each j, Exp$[\sum_{i=1}^{m} a_{ij}X_i] = \sum_{i=1}^{m} a_{ij}x_i^* \geq z^*$. By a standard Chernoff bound, we get Pr$[\sum_{i=1}^{m} a_{ij}X_i \leq (1 - \lambda)z^*] \leq e^{-\lambda^2 z^*/2}$, where $0 < \lambda \leq 1$. Similarly, for all j, Pr$[\sum_{i=1}^{m}(1 - a_{ij})X_i \leq (1 - \lambda)z^*] \leq e^{-\lambda^2 z^*/2}$. By the naive union bound, Pr$[Z \leq (1 - \lambda)z^*] \leq 2ne^{-\lambda^2 z^*/2}$.

Likewise, Exp$[\sum_{i=1}^{m} X_i] = \sum_{i=1}^{m} x_i^* \leq s$. Thus, by the Chernoff bound, Pr$[\sum_{i=1}^{m} X_i \geq (1 + \epsilon)s] \leq e^{-\epsilon^2 s/4}$, where $0 < \epsilon \leq 2e - 1$. Letting $L = \sum_{i=1}^{m} X_i - s$, we have Pr$[L \geq \delta\sqrt{s}] \leq e^{-\delta^2/4}$, where $0 < \delta \leq (2e - 1)\sqrt{s}$.

Since $\tilde{Z} \geq Z - L$, and using the above estimates, we get Pr$[\tilde{Z} \leq (1 - \lambda)z^* - \delta\sqrt{s}] \leq$ Pr$[Z \leq (1 - \lambda)z^*] +$ Pr$[L \geq \delta\sqrt{s}] \leq 2ne^{-\lambda^2 z^*/2} + e^{-\delta^2/4}$.

Choosing $\lambda = \sqrt{2\ln(8n)/z^*}$ and $\delta = \sqrt{4\ln 4}$, we get that $\tilde{Z} \geq z^* - \sqrt{2\ln(8n)z^*} - \sqrt{4\ln(4)s}$ with probability at least $\frac{1}{2}$, as long as when $z^* \geq 2\ln(8n)$. This inequality is also trivially true for $z^* < 2\ln(8n)$. This, together with the bound $\mathcal{C}^{*}_{min}(G, s) \leq z^*$, imply the lemma. □

5. Experimental Analysis

We implemented Algorithm RRBC using *LP_SOLVE* solver [2], and tested its performance on both synthetic and real data sets.

Synthetic data. We used random data sets composed of four adjacency matrices of sizes $(m, n) = (100, 30), (100, 100), (200, 60), (200, 200)$, respectively, where each element of the matrix is chosen to be 1 or 0 with probability $\frac{1}{2}$. We ran RRBC for $s = 20, 21, ..., 90$, and compared its solution to the optimal solution of the linear program MinLP.

Table 1. Performance of RRBC on synthetic data with $m = 100$ and $n = 30$.

s	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90
MinLP	10	12.5	15	17.5	20	22.5	25	27.5	29.82	31.89	33.88	35.76	37.54	39.11	40
RRBC	7	10	13	15	18	19	23	25	28	30	32	34	35	38	40

Table 1 shows some results on the comparison of RRBC's solution and the MinLP solution from the experiment in which $m = 100$ and $n = 30$. This table presents only the performance of a single run of RRBC, so the results are likely to be even better if we run RRBC several times and choose the best solution.

We also repeated our simulation test 10 times for each of the above settings and took the average of them. Figure 1 illustrates results of these experiments.

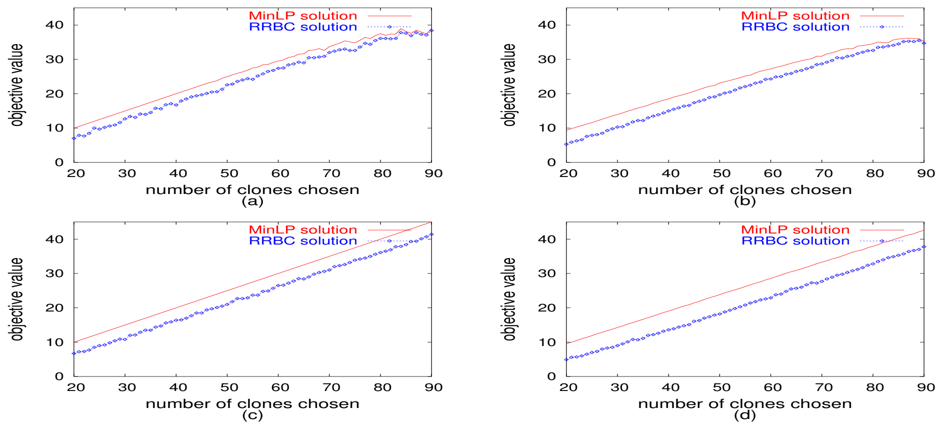

Figure 1. RRBC's performance on synthetic data for four matrices: (a) $(m, n) = (100, 30)$; (b) $(m, n) = (100, 100)$; (c) $(m, n) = (200, 60)$; (d) $(m, n) = (200, 200)$.

Note that (on average) RRBC was always able to find the solution close to that of MinLP. Furthermore, the true optimum (integral solution) could be smaller than the MinLP solution, so our approximation could be even closer to the true optimum than it appears.

Real data. To test the performance of RRBC with real data, we used four clone-probe adjacency matrices. The first two matrices were obtained from 500 bacterial clones extracted from rRNA genes analyzed in [10], along with two sets of 30 and 40 probes designed using the algorithm in [3], respectively. The other two matrices have similar settings, but used fungal clones of rRNA genes studied in [9]. For each of these four data sets we tested RRBC for $s = 200, 210, ..., 400$, running it 10 times and taking the average. We observe that in 92.8% cases, RRBC found a solution at least as large as 97% of MinLP's optimum. The results are shown in Figure 2.

Figure 2 shows that solutions found by RRBC are even closer to MinLP solution than those for synthetic data, for both bacterial and fungal data sets, sometimes even coinciding with MinLP solutions, i.e., RRBC achieved the optimum in some cases.

Our experiments were performed on a machine with Intel Pentium 4 2.4GHz CPU and 1GB RAM. The total running time for each single run of RRBC on these synthetic and real data sets was in the range of $20 - 80$ seconds, which is practically acceptable.

6. An Alternative Algorithm

The performance bound given for RRBC is not quite satisfactory for instances where the optimum is small compared to s. We now provide an alternative algorithm RRBC2, which

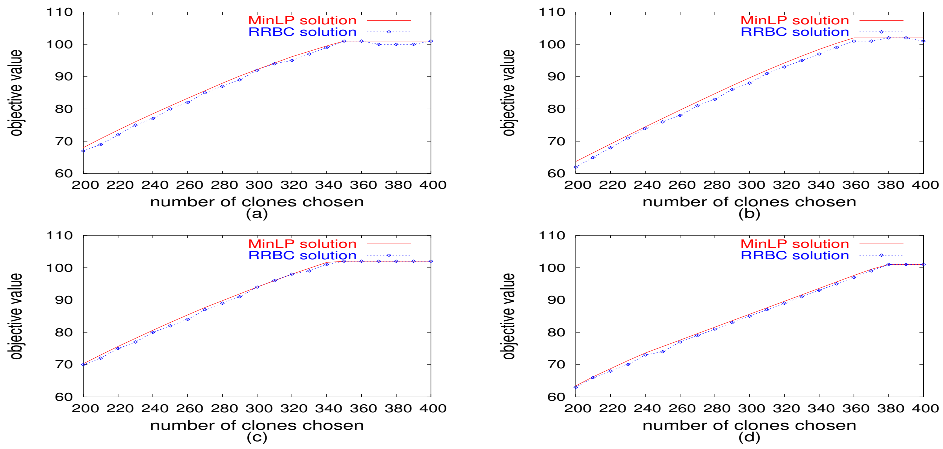

Figure 2. RRBC's performance for real data: (a) 500 bacterial clones and 30 probes; (b) 500 bacterial clones and 40 probes; (c) 500 fungal clones and 30 probes; (d) 500 fungal clones and 40 probes.

is identical to RRBC in all steps except for the rounding scheme: choose $X_i = 1$ with probability $(1-\epsilon)x_i^*$, and 0 otherwise, where $\epsilon = \min\left\{2\sqrt{\ln(4n+2)/z^*}, 1\right\}$. All notations are defined similarly as these in Section 4.

Lemma 6.1. *For any instance $\langle G, s\rangle$ of BCP_$\mathcal{C}_{min}$, with probability at least $\frac{1}{2}$, $\mathcal{C}_{min}^{RRBC2}(G,s) \geq \mathcal{C}_{min}^*(G,s) - O\left(\sqrt{\mathcal{C}_{min}^*(G,s)\ln n}\right)$.*

Proof. The $\{X_i\}$ are independent random variables with Exp$[X_i] = (1-\epsilon)x_i^*$, and $\sum_{i=1}^m x_i^* \leq s$. By linearity of expectation, Exp$[\sum_{i=1}^m X_i] \leq (1-\epsilon)s$. Thus, by the Chernoff bound, Pr$[\sum_{i=1}^m X_i \geq s] \leq$ Pr$[\sum_{i=1}^m X_i \geq (1+\epsilon)(1-\epsilon)s] \leq e^{-\epsilon^2(1-\epsilon)s/4}$.

Likewise, for each j, Exp$[\sum_{i=1}^m a_{ij}X_i] = \sum_{i=1}^m a_{ij}(1-\epsilon)x_i^* \geq (1-\epsilon)z^*$. Thus by the Chernoff bound, Pr$[\sum_{i=1}^m a_{ij}X_i \leq (1-\epsilon)^2 z^*] \leq e^{-\epsilon^2(1-\epsilon)z^*/2}$. Similarly, for all j, Pr$[\sum_{i=1}^m (1-a_{ij})X_i \leq (1-\epsilon)^2 z^*] \leq e^{-\epsilon^2(1-\epsilon)z^*/2}$.

As $z^* \leq s/2$, we have $s/4 \geq z^*/2$. Hence, by the union bound, the probability that any of these $2n+1$ events happens is at most $(2n+1)e^{-\epsilon^2(1-\epsilon)z^*/2}$. Since $(1-\epsilon)^2 \geq 1-2\epsilon$, for $\epsilon < \frac{1}{2}$, we have $\tilde{Z} \geq z^* - 4\sqrt{\ln(4n+2)z^*}$ with probability at lest $\frac{1}{2}$. This bound is also trivially true for $\epsilon \geq \frac{1}{2}$. Finally, since $\mathcal{C}_{min}^*(G,s) \leq z^*$, the lemma follows. □

7. Concluding Remarks

Our work demonstrated that randomized rounding is an effective method for solving the BCP_$\mathcal{C}_{min}$ version of Balanced Covering, especially on real data sets. In the actual implementation available at `http://algorithms.cs.ucr.edu/OFRG/`, the solution of RRBC is fed as an initial solution into a simulated annealing algorithm. We found out that the simulated annealing rarely produces any improvement of this initial solution, which provides further evidence for the effectiveness of randomized rounding in this case.

We have several additional results related to this work that are not described in the paper due to lack of space. For $\mathsf{BCP\_}\mathcal{C}_{\mathit{min}}$ we improved the hardness approximation bound to $\mathcal{C}^*_{\mathit{min}}(G,s) - O(\log n \log\log n)$. We can show that it is hard to approximate $\mathsf{BCP\_}\mathcal{C}_{\mathit{sum}}$ with the bound $\mathcal{C}^*_{\mathit{sum}}(G,s) - \beta(mn)^\epsilon$ for any constants $\beta > 0$ and $0 < \epsilon < 1$ even with randomization. Finally, we have a randomized algorithm for $\mathsf{BCP\_}\mathcal{D}_{\mathit{max}}$ whose solution deviates from the optimum by no more than $O(\sqrt{s \ln n})$ with probability at least $\frac{1}{2}$. These results will appear in the full version of this paper.

References

1. B. Berger, J. Rompel, and P. Shor. Efficient NC algorithms for set cover with applications to learning and geometry. *30th Annual Symposium on the Foundations of Computer Science*, pages 54–59, 1989.
2. M. Berkelaar, K. Eikland, and P. Notebaert. *Lp_solve mixed integer linear programming solver 5.5*, 2004. Available at `http://lpsolve.sourceforge.net/5.5`.
3. J. Borneman, M. Chrobak, G.D. Vedova, A. Figueroa, and T. Jiang. Probe selection algorithms with applications in the analysis of microbial communities. *Bioinformatics*, 17(1):S39–S48, 2001.
4. U. Feige. A threshold of ln n for approximating Set Cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.
5. M.R. Garey and D.S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. W.H.Freeman, New York, 1979.
6. L. Gargano, AA. Rescigno, and U. Vaccaro. Multicasting to groups in optical networks and related combinatorial optimization problems. *International Parallel and Distributed Processing Symposium*, 2003.
7. J. Schuchhardt, D. Beule, A. Malik, E. Wolski, H. Eickhoff, H. Lehrach, and H. Herzel. Normalization strategies for cDNA microarrays. *Nucleic Acids Res.*, 28(10):e47, 2000.
8. L. Valinsky, A. Scupham, G.D. Vedova, Z. Liu, A. Figueroa, K. Jampachaisri, B. Yin, E. Bent, R. Mancini-Jones, J. Press, T. Jiang, and J. Borneman. Oligonucleotide fingerprinting of ribosomal RNA genes (OFRG). In G.A. Kowalchuk, F.J. de Bruijn, I.M. Head, A.D. Akkermans, and J.D. Van Elsas, editors, *Molecular Microbial Ecology Manual*, pages 569–585. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2nd edition, 2004.
9. L. Valinsky, G. Della Vedova, T. Jiang, and J. Borneman. Oligonucleotide fingerprinting of ribosomal RNA genes for analysis of fungal community composition. *Applied and Environmental Microbiology*, 68(12):5999–6004, 2002.
10. L. Valinsky, G. Della Vedova, A. Scupham, S. Alvey, A. Figueroa, B. Yin, R. Hartin, M. Chrobak, D. Crowley, T. Jiang, and J. Borneman. Analysis of bacterial community composition by oligonucleotide fingerprinting of rRNA genes. *Applied and Environmental Microbiology*, 68(7):3243–3250, 2002.