Available online at www.sciencedirect.com
ScienceDirect
Journal of Theoretical Biology 248 (2007) 512–521
Journal of Theoretical Biology
www.elsevier.com/locate/yjtbi

# Randomized probe selection algorithm for microarray design ☆

Leszek Gąsieniec[a], Cindy Y. Li[a,*], Paul Sant[b,1], Prudence W.H. Wong[a]

[a]*Department of Computer Science, The University of Liverpool, Ashton Building, Ashton Street, Liverpool, L69 3BX, UK*
[b]*Department of Computing and Information Systems, University of Bedfordshire, UK*

Received 28 February 2007; received in revised form 11 May 2007; accepted 29 May 2007
Available online 11 June 2007

**Abstract**

DNA microarray technology, originally developed to measure the level of gene expression, has become one of the most widely used tools in genomic study. The crux of microarray design lies in how to select a unique probe that distinguishes a given genomic sequence from other sequences. Due to its significance, probe selection attracts a lot of attention. Various probe selection algorithms have been developed in recent years. Good probe selection algorithms should produce a small number of candidate probes. Efficiency is also crucial because the data involved are usually huge. Most existing algorithms are usually not sufficiently selective and quite a large number of probes are returned. We propose a new direction to tackle the problem and give an efficient algorithm based on randomization to select a small set of probes and demonstrate that such a small set of probes is sufficient to distinguish each sequence from all the other sequences. Based on the algorithm, we have developed probe selection software RandPS, which runs efficiently in practice. The software is available on our website (http://www.csc.liv.ac.uk/~cindy/RandPS/RandPS.htm). We test our algorithm via experiments on different genomes (*Escherichia coli*, *Saccharamyces cerevisiae*, etc.) and our algorithm is able to output unique probes for most of the genes efficiently. The other genes can be identified by a combination of at most two probes.


*Keywords:* Randomized algorithm; Probe selection; Microarray design

## 1. Introduction

DNA microarrays (Gerhold et al., 1999) have become a very important research tool which have proved to benefit areas including gene discovery, disease diagnosis, and multi-virus discovery. They are used for performing a large number of hybridization experiments simultaneously. Besides their prevalent use to measure the amount of gene expression (Slonim et al., 2000) in a cell, microarrays are an efficient tool for making a qualitative statement about the presence or absence of biological target sequences in a sample. A DNA microarray (“chip”) is a plastic or glass slide which consists of thousands of (about 60,000) short DNA sequences known as probes. A probe is a contiguous substring of a cDNA, which acts as its fingerprint (a.k.a signature). Fingerprinting is the technique of identifying or confirming specific DNA fragments by “cutting” them with special enzymes, observing the unique pattern of the fragment sizes that result, and then comparing this with the pattern of a known DNA fragment. Usually, a probe is 20–70 nucleotides (nt) long.

A typical application of microarrays is detection of different members of a virus family in a sample. In this case, we have a database of the DNA sequences (called targets) for a known family of viruses and we wish to identify an unspecified virus whose DNA sequence is present in the database. What we need is a set of hybridization tests based on good selection of probes such that on every known family, the set of answers (red, green, yellow or black signal on the microarray) that we receive is unique with respect to any other virus in the database.

☆A preliminary version of the paper appeared in Proceedings of IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 2006, pp. 247–254.

*Corresponding author. Tel.: +44 151 795 4277; fax: +44 151 795 4235.
*E-mail addresses:* leszek@csc.liv.ac.uk (L. Gąsieniec), cindy@csc.liv.ac.uk (C.Y. Li), paul.sant@beds.ac.uk (P. Sant), pwong@csc.liv.ac.uk (P.W.H. Wong).

[1]Part of this research was performed while this author was a research associate at The University of Liverpool.

0022-5193/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jtbi.2007.05.036

Therefore, the probe should bind only to its corresponding sequence, and not to any other sequence available in the database. If this is the case, we say that the probe is unique. The quality of the probe selection process can be expressed by the proportion of DNA sequences in the database possessing unique probes.

Depending upon the application, the hybridization experiments are conducted using either single or multiple probes and very often under the assumption that there is only one target present in the sample. The *probe selection problem* we studied is to find a small number of good probes with specified length for every gene in the genome, that satisfies (1) *quantitative criteria*; (2) *homogeneity*; (3) *sensitivity* and (4) *specificity*. For more detailed definitions of these criteria, see Section 2.1).

The specificity check based on Hamming Distance[2] (Hamming, 1999) as the similarity measure is computationally expensive and takes the most time in probe selection process. The brute force approach for specificity checking scans through the whole length-$n$ genome for every length-$m$ probe and determines if the Hamming distances are large enough. Such a process is expensive and requires $O(mn^2)$ time. For example, brute force specificity checking would take about 72 h for *S. pombe* genome of length $7.1 \times 10^6$ nt and is thus impractical for large genomes. A good probe selection algorithm should be both time and space efficient.

### 1.1. Probe selection problem

To summarize, given a set $S$ of gene sequences $g$ also called *targets* or *target sequences*, the objective is to find for each gene sequence $g$ in $S$ a probe $p$ which hybridizes only to $g$. The probe $p$ is said to be a unique probe of gene $g$. If such a probe $p$ does not exist, i.e., $p$ cross-hybridizes to other sequences in $S$, then find a small collection of probes that uniquely identifies $g$.

### 1.2. Previous work

*Selection criteria*: Lockhart et al. (1996) were among the first to study the probe selection problem. The quantitative criteria they proposed are widely used (Li and Stormo, 2001; Relogio et al., 2002; Tolonen et al., 2002; Sung and Lee, 2003; Rouillard et al., 2003; Bozdech et al., 2003), with some minor variations. Homogeneity and specificity were also used in their algorithm, though the exact algorithm has not been published. Homogeneity is used in almost all existing algorithms, in which is usually measured by the nearest-neighbor model (NNM). Kaderali and Schliep (2002) focus on melting temperature ($T_m$) and compute the optimal (the best) probe using suffix trees and dynamic programming. However, this is too slow, especially for large genomes, e.g., it takes 2 weeks to design a probe set for the whole yeast genome. A different formula was also used in Wright and Church (2002), Bozdech et al. (2003) to calculate $T_m$. Other work like Matveeva et al. (2003) also only focuses on criteria related to thermodynamic evaluation. It is generally agreed that $T_m$ and free energy can be used as parameters to evaluate probe hybridization behavior and have been shown to be useful (Li and Stormo, 2001).[3]

As for specificity, there are two major measurements: Hamming distance (Li and Stormo, 2001; Rahmann, 2002; Sung and Lee, 2003) and BLAST search (Relogio et al., 2002; Tolonen et al., 2002; Rouillard et al., 2003; Wright and Church, 2002; Bozdech et al., 2003). Using BLAST (Altschul et al., 1997) (http://www.ncbi.nih.gov/blast/), the algorithms assume the search is done in advance and the results passed as input. The computation time, thus, depends on the number of sequences in the BLAST database; e.g., the algorithm by Rouillard et al. (2003) takes from 4 to 12 h to design up to three 45 mer probes per gene for most of the bacterial genome.

Sensitivity is also a popular consideration to avoid self-binding of probes selected. This may be done by checking the stability of the secondary structure formed (stable means not a good candidate). MFOLD (Zuker et al., 1999), Vienna RNAfold (Hofacker, 2003) and Smith–Waterman (Smith and Waterman, 1981) algorithms have been used in Bozdech et al. (2003), Rouillard et al. (2003), Matveeva et al. (2003), Wright and Church (2002) for this purpose.[4] Other algorithms (Li and Stormo, 2001; Tolonen et al., 2002; Relogio et al., 2002; Sung and Lee, 2003; Rahmann, 2002) directly check sensitivity by eliminating probes that are self-complementary.

*Existing software*: Based on the above three criteria, a number of algorithms have been proposed. Li and Stormo (2001) used a fast approximate matching search algorithm Myersgrep (Myers, 1950) for uniqueness checking. However, the algorithm is still not fast enough for computing probes of large genome sets. It takes almost 4 days to design a length-24 probe set for *Saccharomyces cerevisiae* genome (12M nt with about 6000 genes). Rahmann (2002) presented a fast algorithm eliminating candidates that have a long common factor with other genes. This algorithm allows selection of probes for large genomes like *Neurospora crassa* with total size 43 MB in 4 h on a Compaq ES40 (833 MHz) with 16 GB memory. However, the approach only designs short probes and requires a lot of space during computation.

[2]For two strings $s$ and $t$, the Hamming distance $H(s,t)$ is the number of positions where the characters at corresponding positions of the two strings differ. For example, if $s = 00010101$, $t = 00011010$, then $H(s,t) = 4$.

[3]Some researchers (Naef and Magnasco, 2003; Wu and Irizarry, 2005) argue that thermodynamic criteria may not be adequate for microarray analysis, we leave this decision to biologists while we mainly provide a computational tool to design probes using thermodynamic criteria.

[4]It is worth mentioning that recently there have been other softwares developed for predicting secondary structure, e.g., Sfold (Ding et al., 2004), UNAFOLD (Markham and Zuker, 2005), though they are not yet employed directly in the context of probe selection.

Sung and Lee (2003) attempted to reduce the time complexity by using several filtering steps and exploiting the Pigeon Hole Principle (Cameron, 1994) to avoid redundant comparisons. A length 50 mer probe set for *N. crassa* can be generated in 3.5 h on SunFire Workstations (700 MHz) with 4 GB memory.

Relogio et al. (2002) proposed a modified version of the Gene Skipper software; the specificity check only considers perfect matches ignoring possible mismatches which may still result in probes that are non-specific and bind to other sequences in addition to the target.

Tolonen et al. (2002) also only considered perfect match; specificity checking requires no region of self-complementarity of five or more bases at either end.

Wright and Church (2002) proposed an algorithm which terminates once good probes (not necessary optimal) are found. They also introduced an interesting concept to define probe sequence complexity based on the Lempel-Ziv (LZ) compression algorithm (Lempel, 1977). Independently, this idea was also employed by Bozdech et al. (2003).

Recently, Klau et al. (2004) presented 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. Their approach is based on Integer Programming mixed with a branch-and-cut algorithm. Their preliminary implementation is capable of separating all pairs of targets optimally in a reasonable time and achieves a considerable reduction on the numbers of probes needed compared to previous greedy algorithms.

### 1.3. Our result

We propose a new approach that takes as input a set of known gene sequences and builds a small cardinality set of probes allowing us to identify the unknown target in the sample. Instead of checking all possible probes, we exploit randomization. We randomly pick probes with some minimal criteria checking. All probes are far (in terms of Hamming distance) from each other. Our algorithm performs efficient probe selection providing unique probes for almost all target sequences in the considered genomes. More detailed discussion on the selection of our procedure can be found in Section 2.2. Our algorithm is quick because exhaustive search is not required. Also, we do not rely on external software.

The experimental results show that our algorithm is much faster than existing algorithms especially for large genomes. For more commonly tested data sets, Table 1 summaries the relative performance of our algorithm with some algorithms mentioned in Section 1.2. Our randomized procedure selects probes efficiently from short (24 bases) through long (64 bases) probes for large genomes. Furthermore, our approach significantly reduces the number of probes needed in microarray design.

Table 1
Comparison of our algorithm and other algorithms

| | Li and Stormo (2001) | Rahmann (2002) | Our algorithm |
|---|---|---|---|
| *E. coli* | 23 nt, 1.5 days | 24 nt, 32 min | 64 nt, 20 min |
| *S. cerevisiae* | 24 nt, 4 days | 24 nt, 116 min | 64 nt, 40 min |
| *N. crassa* | More than a week | 24 nt, 240 min | 24 nt, 155 min |
| Human chromosome 1 | A few weeks | Space exhausted | 64 nt, 740 min |

The length of the probes designed by existing software ranges from 20 to 70: around 20 (Lockhart et al., 1996; Li and Stormo, 2001; Kaderali and Schliep, 2002; Sung and Lee, 2003; Tolonen et al., 2002), around 30 (Kaderali and Schliep, 2002; Rahmann, 2002) around 50 (Li and Stormo, 2001; Sung and Lee, 2003; Rouillard et al., 2003), and around 70 (Li and Stormo, 2001; Sung and Lee, 2003; Wright and Church, 2002; Bozdech et al., 2003). Our software is able to design probes of various length in this range (see Section 3.2).

As for the number of probes returned, some algorithms returned all probes (Sung and Lee, 2003) requiring longer computational time while most of the other software return a small number of probes. We follow the approach adopted by most software and report a small number.

## 2. Material and method

In the section, we present our approach for the probe selection problem. In Section 2.1, we first specify the exact criteria for a probe. In Section 2.2, we describe our randomized algorithm. In Section 2.3, we discuss the issue of speeding up our algorithm by some combinatorial structure.

### 2.1. Probe selection criteria

Every length-$m$ substring of a gene sequence is called a *candidate*. For every candidate, we check whether it satisfies fundamental probe selection criteria: (1) quantitative criteria; (2) homogeneity; (3) sensitivity. Any candidate that passes all these three criteria is called a *probe*.

*Quantitative criteria* are described by Lockhart et al. (1996) and are used in Affymetrix probe selection criteria: (1) the content of any single base (As, Ts, Cs or Gs) does not exceed 50% of the candidate size; (2) the length of any contiguous As and Ts or Cs and Gs region is less than 25% of the candidate size; (3) GC-content is between 40% and 60% of the candidate (GC-content is the percentage of nucleotides which are G or C in the sequence).

*Homogeneity* criterion requires that the melting temperature of candidates should be within some pre-defined range, because a good probe set needs to hybridize to their intended targets at about the same temperature in experiments.

Melting temperature (Rychlik et al., 1990) of a probe is the temperature at which 50% of the oligonucleotides and its perfect complement are in duplex. Since it is impossible to know the target DNA concentration, the calculation is approximate, but still useful. Melting temperature $T_m$ of each candidate in our approach is calculated as

$$T_m = \frac{\Delta H}{\Delta S + R \times \ln(c/4)} - 273.15, \quad (1)$$

where $\Delta H$ and $\Delta S$ are the enthalpy and entropy for the helix formation, respectively, $R$ is the molar gas constant (1.987 cal/(K mol)), and $c$ is the total molar concentration of the annealing oligonucleotides when oligonucleotides are not self-complementary.[5]

*Sensitivity* criterion filters out candidates prone to self-complementarity (see Fig. 1). This is to reject all candidates who may fold back on themselves rather than on target sequences. Consider every segment of a candidate of length $\ell$. If its reversal forms a consecutive length $\ell$ complementary segment within itself, the candidate is considered prone to fold back on itself.

Another useful measure for sensitivity is the free energy. The total difference in the free energy of the folded and unfolded states of a DNA duplex is approximated by a NNM:

$$\Delta G_i(total) = \sum_j n_{ij}\Delta G_j + \Delta G_i(init) + \Delta G_i(sym), \quad (2)$$

where each different oligonucleotide duplex is given the subscript $i$, $\Delta G_j$ is the free energy for the 10 possible Watson–Crick nearest-neighbor stacking interactions, $n_{ij}$ is the number of occurrences of each nearest neighbor $j$, in each sequence $i$, $\Delta G_i(init)$ is the initiation free energy, and $\Delta G_i(sym)$ equals $+0.4$ kcal/mol if duplex $i$ is self-complementary and zero if it is non-self-complementary (Cantor and Schimmel, 1980). DNA oligonucleotide nearest-neighbor thermodynamic parameters are available (SantaLucia et al., 1996) and they allow prediction of oligonucleotide DNA hybridization energies.

The thermodynamic parameters used in our melting temperature and free energy calculation were estimated from experimental measurements on short probes. Therefore, although we used both to model long probe binding stability, the free energy values should be viewed as a function of binding stability on a relative scale, rather than be interpreted as the absolute free energy generated during DNA duplex formation.

[5]The NNM is well adapted to compute the $T_m$ for short sequences, but may lead to an overestimate of the $T_m$ of probes longer than 50 nt. Other methods compute $T_m$ by the formula (Wetmur, 1991) $T_m = 81.5 + (16.6 \log([Na^+]) + 41[(G+C)/length] - (500/length)$ where $[Na^+]$ is the sodium ion concentration. However, evidence for size limitation of the NNM and parameters is sparse (Bozdech et al., 2003). For 70-mer probes, the difference between the $T_m$ values calculated using this method is negligible (Wright and Church, 2002).

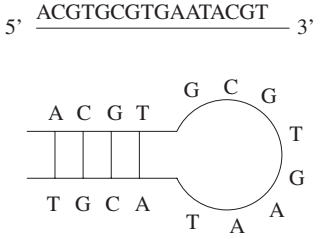


Fig. 1. A candidate prone to self-complementarity.

In this work, we are mainly interested in efficient selection of *unique probes*, playing a role of gene signatures. We say that probe $p$ is a *unique probe* for gene $g$ in a genome if and only if $p$ occurs in $g$ and there is no close occurrence (in terms of Hamming distance, see Specificity criterion) of $p$ in any other gene of the genome.

*Specificity* identifies probes that are unique to each gene in the genome. This condition minimizes cross-hybridization of the probes with other gene sequences. Hamming distance has been used as the basis for coding theoretic approaches (Frutos et al., 1997; Li et al., 2002) to the DNA word design problem. In particular, Hamming distance becomes a powerful tool for determining closeness/similarity and recently has been adopted as the specificity measure (Li and Stormo, 2001; Rahmann, 2002; Sung and Lee, 2003). Thus, if the Hamming distance between a probe and every candidate (excluding those candidates from the gene where the probe belongs to) is greater than some constant, the probe is said to be specific enough.[6]

### 2.2. Randomized probe selection algorithm

In this section, we present a new algorithm to select probes for DNA microarrays. Initially, our algorithm exploits several filters (based on probe selection criteria) to reduce the search space for probes. However, the main idea used here is to explore randomization to reduce the time complexity of the search. And indeed, randomly generated sequences are expected to possess properties of unique probes. E.g., probe selection criteria enforce balanced distribution of base pairs in probes which is naturally satisfied by random sequences. Moreover, the Hamming distance between two randomly chosen sequences of length $m$ over a four letter alphabet is about $3m/4$, which is also highly desired property of a system of probes.

**Algorithm 1.** Probe selection ($m$: length of probe; $S$: genome; $d$: Hamming distance threshold, default is 5).

$i \Leftarrow 0$ and $not\_found \Leftarrow true$;
**for** every gene $g \in S$: **do**
  **while** $i < 5$ and $not\_found$ is $true$ **do**
    generate a random sequence $r_i$ of length $m$;

[6]Our approach is independent from any particular specificity criterion (whether Hamming distance or BLAST search) is used. Our algorithm can be adopted any other specificity criteria as a black box.

find the closest probe $p_i$ in gene $g$;
**if** $H(p_i, q) \geqslant d$ for all candidates $q$ in other genes in $S-\{g\}$ **then**
$p_i$ is chosen as the unique probe for $g$, report $p_i$, $not\_found \Leftarrow false$;
**end if**
$i \Leftarrow i + 1$;
**end while**
**end for**

Our probe selection algorithm starts with the filtering stage applied on the whole genome. For each candidate, we test whether it passes the probe selection criteria (1), (2) and (3) and we eliminate all candidates who fail the test. For (2) homogeneity, we require that the melting temperature lies between 78 and 90; for (3) sensitivity, we reject all candidates with a self-complementary segment of length greater than or equal to 4.

When the filtering is completed, we iterate a probe selection procedure which acts on all genes in the genome. The probe selection procedure, see Algorithm 1, runs with gene $g \in S$, generates a unique (if it is able to find it) probe $p$ for gene $g$. This is done as follows: (a) generate a random sequence $r$ of length $m$; (b) find the closest match $p$ of $r$ among probes in the target; (c) check whether $p$ satisfies specificity criterion. This process is iterated at most five times which allows us to obtain a good trade-off between the accuracy of the search procedure and its running time. We have fixed the number of iterations to five times by testing the performance against the number of iterations. We observed that the percentage of targets identified by a single probe becomes stable after five iterations (see Fig. 2). The code of the procedure could be easily modified to incorporate the case when a unique probe is not found, in this case, we check whether a combination of any two (and very rarely three) already selected probes uniquely identifies the considered gene $g$.

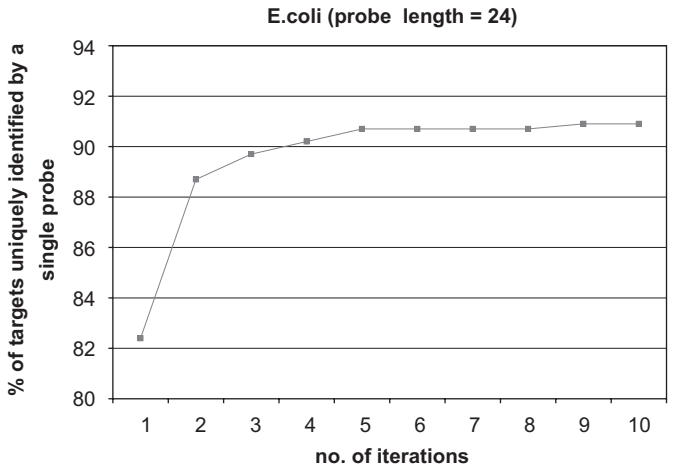


Fig. 2. Percentage of targets identified by a single probe becomes stable after five iterations.

It should be pointed out that our algorithm terminates once probes have been found to satisfy the probe selection criteria, rather than searching for optimal probes. In this end, we are in line with Rahmann (2002), Sung and Lee (2003), Relogio et al. (2002), Tolonen et al. (2002), Rouillard et al. (2003), Wright and Church (2002), Bozdech et al. (2003). Using this strategy, our algorithm can select probes for large genomes for which algorithms demanding optimality are unsuccessful (Li and Stormo, 2001; Kaderali and Schliep, 2002).

### 2.3. Speeding up methods

To speed up our probe selection procedure, we exploit an "encoding" method to test self-complementarity and specificity. Consider every segment of a candidate of length 4, if its reversal forms complementary segment within itself, the candidate is prone to form a secondary structure. In particular, every segment of a candidate of length $\ell$ is encoded as follows:

$$\sum_{i=0}^{\ell-1} c_i \times 4^{(\ell-i-1)}, \tag{3}$$

where $c_i$ is either 0, 1, 2 or 3 (standing for A, C, G, T, respectively) representing the $i$th base of the segment. For example, a sequence ATCG is encoded as $0 \times 4^3 + 3 \times 4^2 + 1 \times 4^1 + 2 \times 4^0 = 54$. Furthermore, we exploit the tabling method to speed up the specificity checking process. We pre-compute a matrix $D = [D_{ij}]$ in which the rows and columns are indexed by numerical values obtained (by Formula 3) from all possible DNA sequences of length 4. Each entry $D_{ij}$ is the Hamming distance between two DNA sequences with numerical value $i$ and $j$. For example, if $i = 0$, representing AAAA, and $j = 255$, representing TTTT, then $D_{0,255} = 4$. By looking up the appropriate entry in the table, Hamming distance between two probes of length-$m$ can be quickly determined.

## 3. Result and discussion

### 3.1. Time complexity

The brute force approach for specificity checking scans through the whole length-$n$ genome for every length-$m$ probe and determines if the Hamming distances are large enough. Such a process is computationally expensive, requiring $O(mn^2)$ time. In comparison, we pick up a probe of length $m$ by using randomization for every gene in the genome, then scan through the whole genome for specificity checking. By doing this, we do not need to check every probe in each gene which greatly reduce the time complexity. Thus, the time complexity of our algorithm is $O(kmn)$ where $k$ (usually much smaller than $n$) is the number of genes in the whole genome, $m$ is the length of probe and $n$ is length of the whole genome.

Table 2
Information of the data sets and time used for RandPS of probe length 64

| | *E. coli* | *S. cerevisiae* | *S. pombe* | *N. crassa* | *A. thaliana* | Mouse chromosome 2 | Human chromosome 1 |
|---|---|---|---|---|---|---|---|
| Total length | 4,752,411 | 8,783,280 | 7,272,320 | 17,484,362 | 33,581,216 | 182,887,278 | 197,317,844 |
| No. of genes | 5253 | 5888 | 5471 | 10,633 | 26,186 | 1302 | 2017 |
| Avg. length per gene | 905 | 1492 | 1329 | 1644 | 1282 | 140,466 | 97,827 |
| Time (minutes) | 20 | 40 | 60 | 310 | 1520 | 470 | 740 |

Table 3
Results of RandPS for *E. coli*

| Genome | *E. coli* | | |
|---|---|---|---|
| Length | 4,752,411 | | |
| No. of genes | 5253 | | |
| Probe length | Number of genes requiring | | |
| | 1 probe | 2 probes | No probe returned |
| 24 | 4759 (90.7%) | 490 (9.3%) | 4 |
| 32 | 4791 (91.3%) | 457 (8.7%) | 5 |
| 40 | 4805 (91.6%) | 442 (8.4%) | 6 |
| 48 | 4808 (91.7%) | 436 (8.3%) | 9 |
| 56 | 4827 (92.1%) | 413 (7.9%) | 13 |
| 64 | 4832 (92.3%) | 405 (7.7%) | 16 |

Table 4
Results of RandPS for *S. cerevisiae*

| Genome | *S. cerevisiae* | | |
|---|---|---|---|
| Length | 8,783,280 | | |
| No. of genes | 5888 | | |
| Probe length | Number of genes requiring | | |
| | 1 probe | 2 probes | No probe returned |
| 24 | 5481 (93.2%) | 401 (6.8%) | 6 |
| 32 | 5516 (93.9%) | 361 (6.1%) | 11 |
| 40 | 5525 (94.2%) | 341 (5.8%) | 22 |
| 48 | 5549 (94.7%) | 313 (5.3%) | 26 |
| 56 | 5560 (95.0%) | 292 (5.0%) | 36 |
| 64 | 5560 (95.1%) | 288 (4.9%) | 40 |

Table 5
Results of RandPS for *S. pombe*

| Genome | *S. pombe* | | |
|---|---|---|---|
| Length | 7,272,320 | | |
| No. of genes | 5471 | | |
| Probe length | Number of genes requiring | | |
| | 1 probe | 2 probes | No probe returned |
| 24 | 5061 (92.6%) | 407 (7.4%) | 3 |
| 32 | 5064 (92.6%) | 404 (7.4%) | 3 |
| 40 | 5131 (94.1%) | 321 (5.9%) | 19 |
| 48 | 5141 (94.3%) | 308 (5.7%) | 22 |
| 56 | 5154 (94.6%) | 294 (5.4%) | 23 |
| 64 | 5152 (94.7%) | 287 (5.3%) | 32 |

Table 6
Results of RandPS for *N. crassa*

| Genome | *N. crassa* | | |
|---|---|---|---|
| Length | 17,484,362 | | |
| No. of genes | 10,633 | | |
| Probe length | Number of genes requiring | | |
| | 1 probe | 2 probes | No probe returned |
| 24 | 10530 (99.2%) | 90 (0.8%) | 13 |
| 32 | 10551 (99.5%) | 57 (0.5%) | 25 |
| 40 | 10557 (99.5%) | 50 (0.5%) | 26 |
| 48 | 10558 (99.6%) | 45 (0.4%) | 30 |
| 56 | 10559 (99.6%) | 42 (0.4%) | 32 |
| 64 | 10544 (99.6%) | 40 (0.4%) | 49 |

Table 7
Results of RandPS for *A. thaliana*

| Genome | *A. thaliana* | | |
|---|---|---|---|
| Length | 33,581,216 | | |
| No. of genes | 26,186 | | |
| Probe length | Number of genes requiring | | |
| | 1 probe | 2 probes | No probe returned |
| 24 | 22407 (85.6%) | 3773 (14.4%) | 6 |
| 32 | 24400 (93.2%) | 1777 (6.8%) | 9 |
| 40 | 24813 (94.8%) | 1358 (5.2%) | 15 |
| 48 | 25094 (95.9%) | 1063 (4.1%) | 29 |
| 56 | 25238 (96.5%) | 910 (3.5%) | 38 |
| 64 | 25327 (96.9%) | 807 (3.1%) | 52 |

Table 8
Results of RandPS for Mouse chromosome 2

| Genome | Mouse chromosome 2 | | |
|---|---|---|---|
| Length | 182,887,278 | | |
| No. of genes | 1302 | | |
| Probe length | Number of genes requiring | | |
| | 1 probe | 2 probes | No probe returned |
| 24 | 1194 (91.7%) | 108 (8.3%) | 0 |
| 32 | 1229 (94.4%) | 73 (5.6%) | 0 |
| 40 | 1231 (94.5%) | 71 (5.5%) | 0 |
| 48 | 1235 (94.9%) | 67 (5.1%) | 0 |
| 56 | 1239 (95.2%) | 63 (4.8%) | 0 |
| 64 | 1240 (95.2%) | 62 (4.8%) | 0 |

### 3.2. Analysis of experimental results

Our software RandPS is written in C and is developed and tested on Athlon XP2000 + Cluster with 2 GB memory. The software is available on our website (http://www.csc.liv.ac.uk/~cindy/RandPS/RandPS.htm). The size of RandPS code is 25 KB which is simple and clean while being efficient and effective. Inputs of RandPS are FASTA formatted gene sequences, downloaded from the NCBI website (http://www.ncbi.nlm.nih.gov/). RandPS uses a size-$n$ array, where $n$ is the concatenated length of gene sequences of a genome, to store the inputs, together with another two size-$n$ arrays to store the corresponding numerical value of each base in the genome and the status (*candidate* or *probe*) of each position in the concatenated sequence.

Table 9
Results of RandPS for Human chromosome 1

| Genome | Human chromosome 1 | | |
|---|---|---|---|
| Length | 197,317,844 | | |
| No. of genes | 2017 | | |
| Probe length | Number of genes requiring | | |
| | 1 probe | 2 probes | No probe returned |
| 24 | 1718 (85.2%) | 299 (14.8%) | 0 |
| 32 | 1914 (94.9%) | 103 (5.1%) | 0 |
| 40 | 1918 (95.1%) | 99 (4.9%) | 0 |
| 48 | 1926 (95.5%) | 91 (4.5%) | 0 |
| 56 | 1931 (95.7%) | 86 (4.3%) | 0 |
| 64 | 1932 (95.8%) | 85 (4.2%) | 0 |

The experiments were undertaken in order to evaluate the performance of our software on various types of genomes. We report our results using several genomes that have been widely used for the probe selection problem. These data sets have been used in experiments in Kaderali and Schliep (2002), Li and Stormo (2001), Rahmann (2002), Rouillard et al. (2003), Sung and Lee (2003), Tolonen et al. (2002). In terms of time consumption, for probe length 64, it takes about 20 min to process the *Escherichia coli* genome, 40 min to process the *S. cerevisiae* genome, 60 min for *S. pombe*, 310 min for *N. crassa*, 470 min for Mouse chromosome 2, about 740 min for Human chromosome 1 and 1520 min for *Arabidopsis thaliana*. The genomes involved in the experiments and corresponding time used are listed and in Table 2.

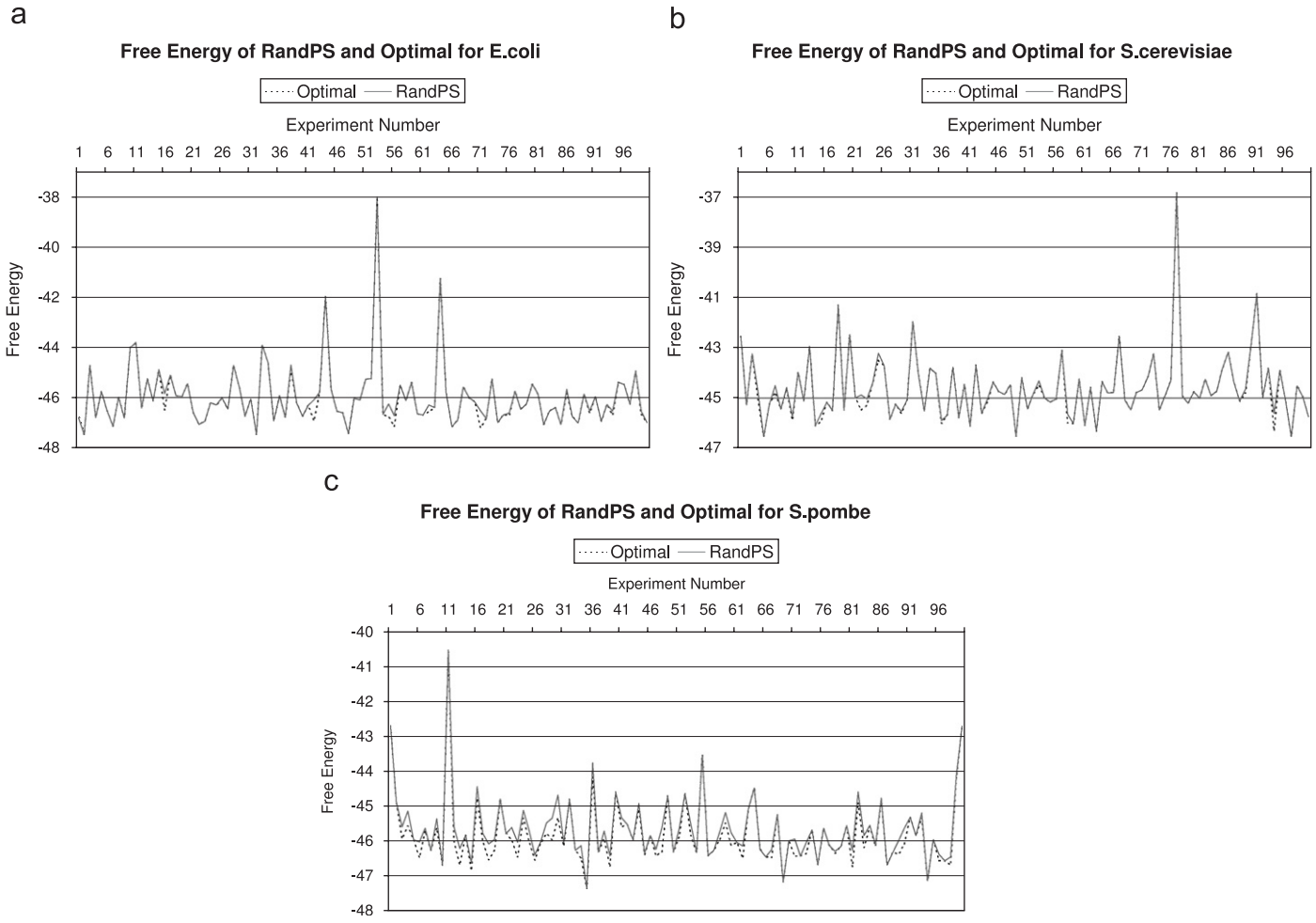


Fig. 3. Comparison of free energy between the optimal probe and the probe chosen by RandPS. (a) *E. coli*; (b) *S. cerevisiae*; (c) *S. pombe*.

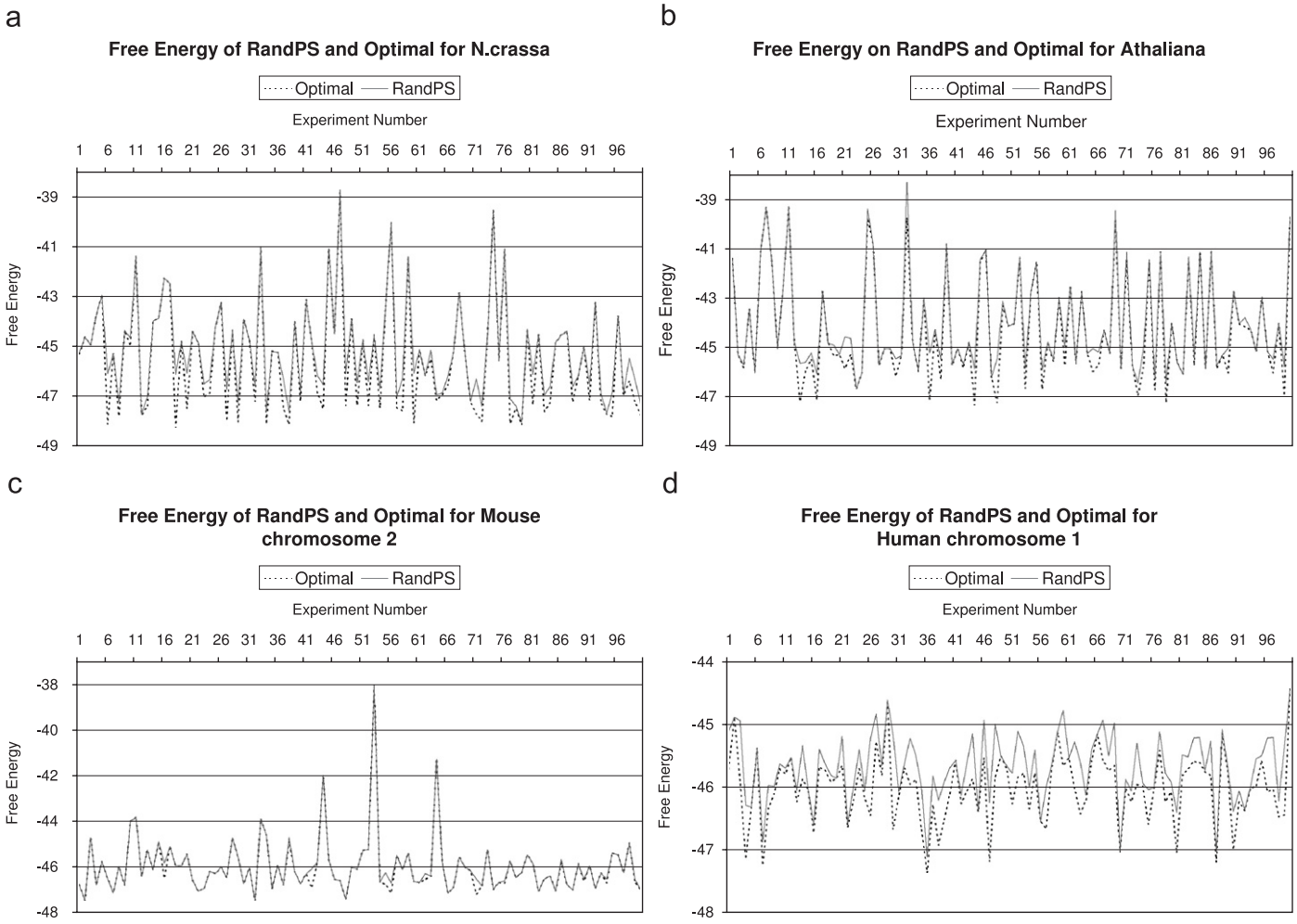


Fig. 4. Comparison of free energy between the optimal probe and the probe chosen by RandPS. (a) *N. crassa*; (b) *A. thaliana*; (c) Mouse chromosome 2; (d) Human chromosome 1.

In terms of accuracy of probe selection, we are able to find unique probes for up to 99% of genes in the whole genome. The full details of the experimental results are shown in Tables 3–9.[7] We have run experiments 30 times on each data set for each probe length. In these tables, the first three rows are basic information about the data sets, which are the name of the genome, the length of the genome and the number of genes in the genome. The column "Probe length" lists the different lengths we used to test the performance of our software. The column "1 probe" shows the number of genes which can be identified by a unique probe, while "2 probes" column shows the number of genes which require a combination of two probes for unique identification. The percentages in brackets are calculated on the basis of the number of genes with probes (i.e., total number of genes minus number of genes without probes). The "no probe returned" column shows the number of genes where our software did not find feasible probes.

[7]The melting temperature range has been slightly modified for longer probe lengths 48, 56, and 64.

The experimental results in Table 3 show that RandPS is able to find a unique probe for over 90% of *E. coli* with different probe lengths. The remaining genes can be identified by a combination of two probes. There are only around 10 genes where our algorithm did not find feasible probes. For other genomes with similar number of genes (*S. cerevisiae* and *S. pombe*), around 95% genes can be identified by using a single probe. The results can be found in Tables 4 and 5. Tables 6 and 7 illustrate that for genomes with larger number of genes (*N. crassa* and *A. thaliana*), up to 99% genes can be identified by one probe. Finally, for larger data sets of length over 180 M (Mouse chromosome 2 and Human chromosome 1), results are shown in Tables 8 and 9. In this case, RandPS is able to select unique probes for over 95% of the data sets.

In our experiments, we have noticed that there are some genes with no probe. An investigation of these genes revealed that some of these genes are duplicated or very similar to some other genes in the genome. Another reason is that the lengths of some of these genes are too short. Apart from these cases, our software is able to select probes for all genes.

Table 10
Comparison of mean and standard deviation of free energy between the optimal probe and the probe chosen by RandPS

| | OPT | RandPS | Absolute difference |
|---|---|---|---|
| *E. coli* | −45.991 (1.288) | −45.949 (1.272) | 0.042 (0.016) |
| *S. cerevisiae* | −44.625 (1.347) | −44.586 (1.329) | 0.039 (0.018) |
| *S. pombe* | −45.989 (0.863) | −45.609 (0.805) | 0.380 (0.058) |
| *N. crassa* | −45.490 (2.196) | −45.149 (1.915) | 0.341 (0.281) |
| *A. thaliana* | −44.439 (2.200) | −44.098 (2.025) | 0.341 (0.175) |
| Mouse chromosome 2 | −46.363 (0.579) | −45.937 (0.516) | 0.426 (0.063) |
| Human chromosome 1 | −46.020 (0.544) | −45.676 (0.534) | 0.344 (0.010) |

The values are represent by mean (standard deviation).

As further illustration of our software in terms of accuracy of the probe set, we compare the free energy of a group of our probes with the optimal probes with minimum free energy, which is found by using a brute force approach. This is shown in Figs. 3–4 on samples of 100 arbitrarily chosen genes for each genome. A closer look into the mean and standard deviation (Table 10) of hybridization free energy between the optimal probes and the probes chosen by RandPS reveals that the probes we found are very close to the optimal one. Thus, our software is able to find high quality probes.

## 4. Conclusion

We have proposed a new approach to select (randomly) a small set of probes and demonstrated that such a small set of probes is sufficient to distinguish each gene from all the other genes in the genome. Almost all genes can be identified by a unique probe, the others need at most two probes. We have implemented a probe selection software RandPS, which runs efficiently. The software is available on line at http://www.csc.liv.ac.uk/~cindy/RandPS/RandPS.htm.

We believe that our approach should prove to be useful also in the design of multiple probes. Multiple probes might be needed for several reasons. E.g., to accommodate a lack of accuracy in experimental work, a fault-tolerant system is desirable. In some experimental situations, the mRNA is broken into random fragments, which thus require multiple probes per gene.

Therefore, one of our future direction would be on identification and classification of genes by multiple probes. This requires adaptation of our algorithm. We expect the running time to increase, yet this is worthwhile for the scenario we described above.

In future research, it would be interesting to improve performance of our algorithm on more complex organisms, since the structure of higher organism differs from that of bacteria and viruses. This would lead to a more challenging combinatorial problem.

Another direction would be further studies on sensitivity. There have been several improvements in the calculation of minimum free energy in recent software UNAFOLD (Markham and Zuker, 2005). Although UNAFOLD is not yet used directly into probe selection, it is important to consider UNAFOLD in probe selection as future work.

## Acknowledgments

We would like to thank Mia Persson (Lund University, Sweden) for helpful discussions in the initial stages of this work. We are grateful to David Peleg (Weizmann Institute of Science, Israel) for very useful comments on probability aspects of this work. We would also like to thank Andrew Cossins, Derek Gardener, Christine Gosden, Dawn Jones (University of Liverpool, UK) for useful discussions on human genome study.

## References

Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Miller, W., Lipman, D.J., 1997. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 25 (17), 3389–3402.

Bozdech, Z., Zhu, J., Joachimiak, M.P., Cohen, F.E., Pulliam, B., DeRisi, J.L., 2003. Expression profiling of the schizont and trophozoite stages of *plasmodium falciparum* with a long-oligonucleotide microarray. Genome Biol. 4, R9.

Cameron, P.J., 1994. Combinatorics: Topics, Techniques, Algorithms. Cambridge University Press, Cambridge, MA.

Cantor, C.R., Schimmel, P.R., 1980. Biophysical Chemistry Part III: The Behavior of Biological Macromolecules. W.H. Freeman, San Francisco, CA.

Ding, Y., Chan, C.Y., Lawrence, C.E., 2004. Sfold web server for statistical folding and rational design of nucleic acids. Nucleic Acids Res. 32, W135–W141.

Frutos, A.G., Liu, Q., Thiel, A.J., Sanner, A.M.W., Condon, A.E., Smith, L.M., Corn, R.M., 1997. Demonstration of a word design strategy for DNA computing on surfaces. Nucleic Acids Res. 25 (23), 4748–4757.

Gerhold, D., Rushmore, T., Caskey, C.T., 1999. DNA chips: promising toys have become powerful tools. Trends Biochem. Sci. 24 (5), 168–173.

Hamming, R.W., 1999. Error-detecting and error-correcting codes. J. ACM (JACM) 46 (3), 395–415.

Hofacker, I.L., 2003. Vienna RNA secondary structure server. Nucleic Acids Res. 31 (13), 3429–3431.

Kaderali, L., Schliep, A., 2002. Selecting signature oligonucleotides to identify organisms using DNA arrays. Bioinformatics 18, 1340–1349.

Klau, G.W., Rahmann, S., Schliep, A., Vingron, M., Reinert, K., 2004. Optimal robust non-unique probe selection using Integer Linear Programming. Bioinformatics 20, i186–i193.

Lempel, Z.J., 1977. A universal algorithm for sequential data compression. IEEE Trans. Inf. Theory 23, 337–343.

Li, F., Stormo, G., 2001. Selection of optimal DNA oligos for gene expression arrays. Bioinformatics 17 (11), 1067–1076.

Li, M., Lee, H.J., Condon, A.E., Corn, R.M., 2002. DNA word design strategy for creating sets of non-interacting sets of oligonucleotides for DNA microarrays. Langmuir 18 (3), 805–812.

Lockhart, D.J., Dong, H., Byrne, M.C., Follettie, M.T., Gallo, M.V., Chee, M.S., Mittmann, M., Wang, C., Kobayashi, M., Horton, H., Brown, E.L., 1996. Expression monitoring by hybridization to high-density oligonucleotide arrays. Nature Biotechnol. 14, 1675–1680.

Markham, N.R., Zuker, M., 2005. DINAMelt web server for nucleic acid melting prediction. Nucleic Acids Res. 33, W577–W581.

Matveeva, O.V., Shabalina, S.A., Nemtsov, V.A., Tsodikov, A.D., Gesteland, R.F., Atkins, J.F., 2003. Thermodynamic calculation and statistical correlations for oligo-probes design. Nucleic Acids Res. 31 (14), 4211–4217.

Myers, E.W., 1950. A fast bit-vector algorithm for approximate string matching based on dynamic programming. Bell Syst. Tech. J. 29 (2), 147–160.

Naef, F., Magnasco, M.O., 2003. Solving the riddle of the bright mismatches: labeling and effective binding in oligonucleotide arrays. Phys. Rev. E 68, 011906.

Rahmann, S., 2002. Rapid large-scale oligonucleotide selection for microarrays. In: Proceedings of the First Computational Systems Bioinformatics (CSB), pp. 54–63.

Relogio, A., Schwager, C., Richter, A., Ansorge, W., Valcarcel, J., 2002. Optimization of oligonucleotide-based DNA microarrays. Nucleic Acids Res. 30 (11), e51.

Rouillard, J.-M., Zuker, M., Gulari, E., 2003. OligoArray2.0: design of oligonucleotide probes for DNA microarrays using a thermodynamic approach. Nucleic Acids Res. 31 (12), 3057–3062.

Rychlik, W., Spencer, W.J., Rhoads, R.E., 1990. Optimization of the annealing temperature for DNA amplification in vitro. Nucleic Acids Res. 18 (21), 6409–6412.

SantaLucia, J.J., Allawi, H.T., Seneviratne, P.A., 1996. Improved nearest-neighbor parameters for predicting DNA duplex stability. Biochemistry 35 (11), 3555–3562.

Slonim, D.K., Tamayo, P., Mesirov, J.P., Golub, T.R., Lander, E.S., 2000. Class prediction and discovery using gene expression data. In: Proceedings of the Fourth Annual International Conference on Research in Computational Molecular Biology (RECOMB), pp. 263–272.

Smith, T.F., Waterman, M.S., 1981. Identification of common molecular subsequences. J. Molecular Biol. 147, 195–197.

Sung, W.K., Lee, W.H., 2003. Fast and accurate probe selection algorithm for large genomes. In: Proceedings of the Second Computational Systems Bioinformatics (CSB), pp. 65–74.

Tolonen, A.C., Albeanu, D.F., Corbett, J.F., Handley, H., 2002. Optimized in situ construction of oligomers on an array surface. Nucleic Acids Res. 30 (20), e107.

Wetmur, J.G., 1991. DNA probes: applications of the principles of nucleic acid hybridization. Critical Rev. Biochem. Mol. Biol. 26 (3–4), 227–259.

Wright, M.A., Church, G.M., 2002. An open-source oligomicroarray standard for human and mouse. Nature Biotechnol. 20, 1082–1083.

Wu, Z., Irizarry, R.A., 2005. Stochastic models inspired by hybridization theory for short oligonucleotide microarrays. J. Comput. Biol. 12, 882–893.

Zuker, M., Mathews, D.H., Turner, D.H., 1999. Algorithms and Thermodynamics for RNA Secondary Structure Prediction: A Practical Guide. NATO ASI Series, Kluwer Academic publishers, Dordrecht, NL.