

# A computational framework for optimal masking in the synthesis of oligonucleotide microarrays

Simon Kasif<sup>1,2,\*</sup>, Zhiping Weng<sup>1</sup>, Adnan Derti<sup>1</sup>, Richard Beigel<sup>3,4</sup> and Charles DeLisi<sup>1</sup>

<sup>1</sup>Center for Advanced Genomic Technology (CAGT), Bioinformatics Program and Biomedical Engineering Department, Boston University, Boston, MA, USA, <sup>2</sup>MIT Genome Center, Whitehead Institute, MIT, Cambridge, MA, USA and <sup>3</sup>Institute for Advanced Studies, Princeton, NJ, USA and <sup>4</sup>Temple University, Philadelphia, PA, USA

Received April 3, 2002; Revised August 5, 2002; Accepted August 15, 2002

## ABSTRACT

**High-throughput genomic technologies are revolutionizing modern biology. In particular, DNA microarrays have become one of the most powerful tools for profiling global mRNA expression in different tissues and environmental conditions, and for detecting single nucleotide polymorphisms. The broad applicability of gene expression profiling to the biological and medical realms has generated expanding demand for mass production of microarrays, which in turn has created considerable interest in improving the cost effectiveness of microarray fabrication techniques. We have developed the computational framework for an optimal synthesis strategy for oligonucleotide microarrays. The problem was introduced by Hubbell *et al.* Here, we formalize the problem, obtain precise bounds on its complexity and devise several computational solutions.**

## INTRODUCTION

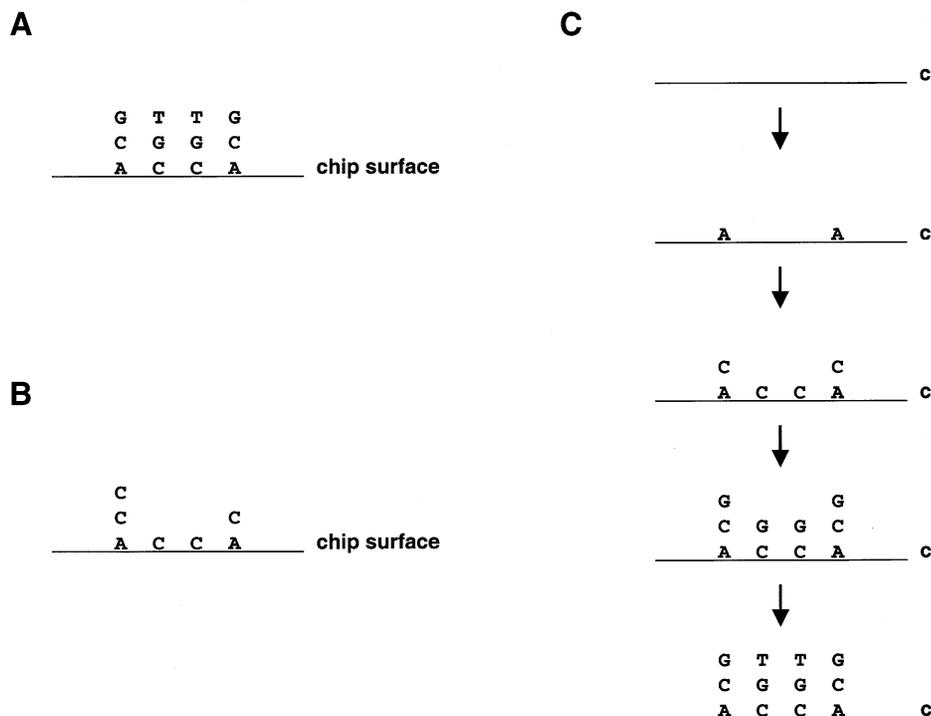
Oligonucleotide and cDNA microarrays can monitor mRNA expression levels for tens of thousands of genes simultaneously (1). While both types of arrays are applied to the elucidation of normal and pathological cellular mechanisms, the longer probes on cDNA arrays make them more susceptible to cross-hybridization, and oligo arrays are designed to reduce cross-hybridization and improve sensitivity (2). In addition, oligo microarrays can be used to detect polymorphisms (3) and, therefore, can greatly facilitate the research and diagnosis of genetic predisposition to diseases. Several large companies, such as Affymetrix, Corning, Motorola and Samsung, have established or are establishing the capability of manufacturing microarrays in large quantities. Thus, reductions in cost or time to manufacture can have a significant impact on biotechnology and medicine.

cDNA microarrays are produced by spotting pre-made cDNA solutions onto a glass or nylon surface via physical

contact or ink-jet deposition. While oligonucleotides can also be synthesized and then spotted, oligo microarrays are usually manufactured by *in situ* synthesis, primarily via photolithography (4), and more recently by ink-jet deposition (5). *In situ* synthesis involves the consecutive addition of A, C, G and T nucleotides to the appropriate spots on the microarray. An important advantage of photolithographic synthesis over ink-jet deposition is that in a single cycle of synthesis, a nucleotide can be added to all desired spots on the array. This is achieved via photodeprotection of the target spots on the array surface with UV light prior to the addition of the nucleotide. Meanwhile, non-target spots must be protected from the UV light using physical or virtual masks. The fabrication of physical masks is a laborious and costly process and one mask is needed for each cycle of synthesis for each variety of arrays. Singh-Gasson *et al.* (6) used a digital micro-mirror device to reflect light selectively onto the desired spots of an array, the 'virtual masking' strategy. Nonetheless, the deprotection step for each cycle lasts ~5 min and photolabile nucleosides are expensive. Therefore, decreasing the number of cycles required to synthesize a given set of sequences can reduce time and cost. Here, we address synthesis optimization, i.e., optimizing the order of nucleotide addition.

The simplest strategy for synthesizing a given set of sequences is to add A bases wherever appropriate as the first base, then C, G and T bases, repeating this process for the second base, and so on. Chee *et al.* (3) noted that if  $K$  is the length of the longest oligonucleotide to be synthesized, maximally  $4K$  cycles are required. Hubbell *et al.* (7) observed that it would be possible to skip a synthesis cycle if a base is not needed by any oligonucleotides, or if the oligonucleotides that require the base can still be synthesized when that base is presented again later. In a parallel publication, Tolonen *et al.* (8) observe that synthesis could be accelerated, even for a large set of oligonucleotides, if the order of base addition is tailored to the oligonucleotide sequences. Consequently, oligonucleotides can vary in length by more than one base at the end of every synthesis cycle. This observation has motivated the development of the optimal base addition strategy described in this paper.

\*To whom correspondence should be addressed at 44 Cummington Street, Boston University, Boston, MA 02215, USA. Tel: +1 617 358 1845; Fax: +1 617 353 6766; Email: kasif@bu.edu



**Figure 1.** Formulation of synthesis strategy. (A) Four 3mer oligonucleotides. (B) Four partially constructed oligos, defining frontier  $F = [3,1,1,2]$ . (C) The synthesis strategy [A,C,G,T] can synthesize the array in four cycles, instead of six cycles required by the traditional approach.

## A COMPUTATIONAL FRAMEWORK FOR OPTIMAL SYNTHESIS STRATEGY

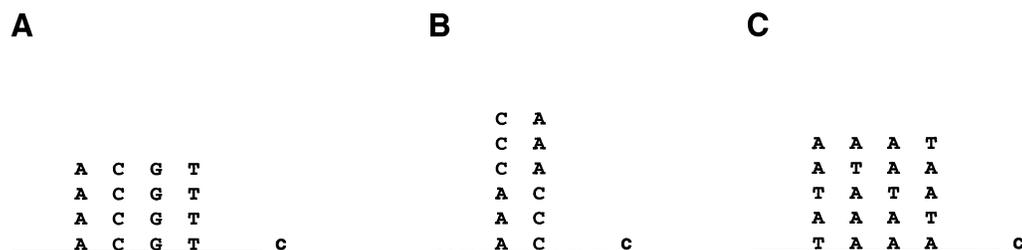
We formulate the question of devising an optimal synthesis strategy in oligo microarrays as a combinatorial state space search problem. This computational formulation provides insight into the complexity of the problem and enables a range of discrete optimization and heuristic search solutions.

In this paper, we assume that the input to the optimization software is a collection of  $N$  oligo sequences of arbitrary length, which have been pre-selected in a probe selection process. For simplicity, we discuss the case of uniform length  $K$ mers, but our framework is readily applicable to the more general case. An optimal synthesis strategy involves  $L$  cycles of synthesis where, in each cycle, a single and identical nucleotide is added to all unmasked oligos. The exact spatial location of each oligo on the array is not important, as long as it can be retrieved during actual synthesis. Therefore, we assume that the input to the optimization code is a list of oligos arranged in one dimension, such as shown in Figure 1A. We define a strategy for constructing  $K$ mers in  $L$  cycles as an  $L$  long vector  $S$ , consisting of elements A, C, G and T.  $S[j]$  is the nucleotide added in cycle  $j$ . For instance, the vector [A,C,A,T,G] corresponds to using A, C, A, T and G in cycles 1–5, respectively. We define the height of each partially constructed oligo as the number of nucleotides that have been added thus far by the synthesis strategy. We define a frontier  $F$  of a partially synthesized array to be an integer vector of size  $N$  where  $F[i]$  is the height of the  $i$ th oligo thus far. For example, the frontier of the four oligos in Figure 1B is  $F = [3,1,1,2]$ .

The ‘traditional’ way to create a chip of  $N$  oligos, each of height  $K$ , is to perform  $4K$  cycles of synthesis. After the addition of A, C, G and T to the appropriate spots of the array, all oligos will be one base long. We then proceed to synthesize layer two in four cycles and all oligos will be two bases long. We continue until the entire chip is synthesized in  $4K$  cycles. It is easy to observe that by a slight modification of the order of base addition (8), we can expedite the above process. As a simplified example (Fig. 1C), we can synthesize an array of  $K = 3$  in four cycles using a modified synthesis strategy, compared to six cycles with the ‘traditional’ approach.

In order to introduce the optimization framework for masking, we need to measure the ‘work’ that has been accomplished after several cycles of synthesis. We therefore give two definitions of ‘frontier height’: (i) the min height of a frontier constructed after  $L$  cycles is the length of the shortest oligo [in Fig. 1B, min height ( $L$ ) = 1]; (ii) the sum height of a frontier is the sum of the lengths of all oligos constructed so far [in Fig. 1B, sum height ( $L$ ) = 3 + 1 + 1 + 2 = 7].

The objective optimization criterion we desire to minimize is the number of cycles required to create a frontier of min height =  $K$ , i.e., we seek the shortest length strategy vector that is sufficient to synthesize all oligos on the chip. It is obvious that the best possible strategy for a  $K$ mer oligo chip is of length between  $K$  and  $4K$ . It is easy to construct an example where the shortest strategy is of length  $4K$  (Fig. 2A), although genomic sequences typically do not exhibit such extremely low complexity. In general, as the number of oligos on a chip grows, the length of the optimal strategy vector is expected to grow as well.



**Figure 2.** Example arrays that challenge synthesis strategies. (A) A worst-case scenario requiring  $4K$  stages for Kmer synthesis. (B) The optimal solution for this chip requires 9 cycles whereas the greedy solution produces a 12-cycle strategy. However, the sum height-based greedy solution produces the optimal synthesis strategy. (C) For this example, sum height heuristics create an 8-cycle strategy: AATAATAA. The exhaustive search produces a 7-cycle strategy: ATAATAA.

#### Local Search Algorithm:

**Input: A Strategy of Length  $M$**

*Until No More Improvement is Observed Iterate the Following Steps:*

1. For each position  $l$  in the current strategy  $0 < l < M$ 
  - a. Try every possible NT in position  $l$
  - b. Compute the length of the resulting strategy
2. Accept the strategy if the strategy is best over  $4M$  perturbations (strategies) tried.

**Figure 3.** A local search algorithm.

### HEURISTIC SEARCH SOLUTIONS

An obvious heuristic solution to devising the optimal synthesis strategy is a greedy search. In each synthesis cycle, we consider the four different options to extend the current layout and compute the height of the resulting frontier for each option. We choose the nucleotide that maximizes the height of the frontier, which could be either min height or sum height. The min height heuristic extends the shortest oligo, while the sum height heuristic chooses the nucleotide that will add the largest number of nucleotides to the chip.

The simple examples in Figure 2B and C show that a greedy search is not guaranteed to produce an optimal solution using either the min height or the sum height heuristic. In the remainder of this section, we consider ways to improve the greedy search. In the next section, we show that an efficient polynomial time solution is unlikely to exist for this optimization problem. We then report simulation results that describe the effectiveness of the sum height heuristics.

#### Look-ahead solution

A natural way to extend the greedy algorithm is to consider a look-ahead strategy. (i) Generate all possible frontiers that can be generated in  $L$  cycles. The number of strategies is  $4^L$ . The number of frontiers might be smaller since different strategies may generate the same frontier. (ii) For each frontier, compute the height. (iii) Choose (for the first cycle) the strategy that maximizes the height after  $L$  cycles. (iv) Repeat until all oligos have been synthesized.

When  $L = 4K$ , this algorithm performs an exhaustive search. A rough upper bound on the running time of this algorithm is  $O(4^L N)$ , which makes it prohibitive for large values of  $L$ . An alternative approach would be to use a variant of best-first search such as  $A^*$ , a popular algorithm in the artificial

intelligence community. A more space-efficient alternative would be to use a branch-and-bound formulation, a standard approach in discrete optimization.

#### Local search solution

A local search attempts to improve a given solution by a series of local perturbations until a minimum is achieved. One obvious local search approach for our problem would be to repeatedly change a selected nucleotide in the strategy vector and accept the new strategy if it results in an improvement over the previous one. Here, we implement a variant (Fig. 3) based on steepest descent.

The steepest descent algorithm considers every possible local perturbation of a single nucleotide in all positions of the strategy vector and chooses the move that results in the greatest improvement. This algorithm terminates relatively quickly since there are at most  $M$  possible improvements to be made. Each iteration (steps 1–2) requires time  $O(M)$ , so the total running time is  $O(M^2)$ .

A simple classic variant of this algorithm is to accept a new strategy with some probability even if no improvement is observed. A Gibbs sampler is a special case of this solution when the probability of acceptance is a function of the degree of improvement, and positions for possible perturbations are selected at random rather than sequentially as described above. We describe simulation results using local search below.

### COMPUTATIONAL ANALYSIS OF OPTIMAL SYNTHESIS STRATEGY

In this section, we provide a set of computational reductions that allow us to obtain a precise characterization of the complexity of the optimal synthesis strategy. We see this part as the main contribution of this paper.

### Multiple sequence alignment formulation

We first observe that the optimal masking problem can be reduced to a special case of multiple sequence alignment. A precise description of multiple sequence alignment can be found in Gusfield (9) and Waterman (10). In particular, the best  $L$  cycle synthesis strategy directly corresponds to the optimal multiple alignment of the  $N$  oligos, where the costs of the alignment are defined as follows: (i) replacement cost =  $+\infty$ ; (ii) deletion cost =  $+\infty$ ; (iii) insertion cost =  $+1$ . That is, the only allowed 'editing' operation is the insertion of a gap. We first demonstrate this principle with an example. For the oligo design problem in Figure 2B, we first align the two oligos CCCAAA and AAACCC. An optimal alignment is given by

```
CCCAAA
  AAACCC
```

Walking across the alignment from left to right creates the following synthesis strategy: [CCCAAACCC]. Another optimal strategy is [AAACCCAAA].

The formal proof of this equivalence is not difficult. Each strategy corresponds to a multiple alignment obtained by aligning each sequence against the strategy sequence. Therefore, the shortest strategy corresponds to the shortest global alignment.

This observation enables the application of computational solutions developed in multiple sequence analysis such as dynamic programming, Gibbs sampling and iterative refinement (9,10). A common greedy solution is based on aligning each pair, then producing a strategy based on the best aligned pair and subsequently continuing to add oligos to the alignment in the best-first manner.

### Shortest super-sequence formulation

The above observation is useful for obtaining insight into the problem. By reducing our problem to a special case of multiple sequence alignment, we show that multiple alignment is 'harder', which does not preclude the possibility of an efficient solution to our specific problem. Here we show that the optimal synthesis strategy problem is exactly equivalent to the problem of computing the shortest super-sequence of a collection of strings. This two-way reduction establishes our problem to be as hard as the shortest super-sequence problem, which is known to be NP hard.

We informally define sequence  $X$  to be a super-sequence of sequence  $Y$  if every character of  $Y$  occurs in  $X$  in the same order as they occur in  $Y$ . Similarly, we define a super-sequence of a collection of sequences where the above condition has to hold for each of the sequences. For instance, AAAACCCCTTTT is a super-sequence of ACT, AAACCCCT and AAAATTT. Sequence  $X$  is the shortest super-sequence of a collection of sequences if and only if its length is the shortest among all super-sequences of the collection.

Since the synthesis strategy must be a super-sequence of each of the oligos, the optimal synthesis strategy vector is equivalent to the shortest super-sequence of all oligos. The shortest super-sequence problem is known to be NP hard (11) and therefore the reduction above formally establishes the optimal synthesis strategy problem to be NP hard. More

explicitly, the problem of finding a masking strategy of length  $L$  ( $L < 4K$ ), given a collection of  $N$  oligos of length  $K$ , is NP complete.

This is an important observation since it implies that the optimal synthesis strategy is unlikely to have efficient (sub-exponential) optimal solutions for a large number of oligos. It is easy, of course, to devise relatively efficient dynamic programming solutions when the number of oligos is constant (e.g. less than 10).

### SIMULATIONS WITH RANDOM OLIGOS

We have conducted a large number of simulations to estimate the performance of several heuristic approaches to devising an optimal synthesis strategy. Here we report our results with three heuristic approaches. (i) Oblivious strategy: we simply repeat synthesizing ACGTACGT... independent of the input sequences. (ii) Max sum height heuristics: we choose the nucleotide that maximizes the sum height in the next cycle (as outlined above). (iii) Randomized local search improvement: once a solution is obtained by the above two methods, we attempt to improve the solution using local search.

Our results comparing the oblivious and max sum height heuristics are summarized in Table 1. It is clear that for random oligos there is no significant difference in performance between max height and oblivious heuristics. It is not particularly surprising for random oligos since, roughly speaking, every layer in the chip contains an approximately equal number of nucleotides of each type (A, C, G and T). Moreover, our results for 'real' oligos appear to be consistent with this performance (data not shown). Note that one of the criteria for selecting oligos aims to prevent cross-hybridization between mRNA and multiple oligos. This puts 'selective pressure' on the design to ensure that oligos are as different as possible. As the number of oligos on the chips grows, they behave more and more like random oligos.

We have interpolated the expected length of a strategy for 10 000 oligos and it appears to fit the following function well:

$$f(K) = 2.5K = 4.04\sqrt{K}$$

where  $K$  is the height of the oligos.

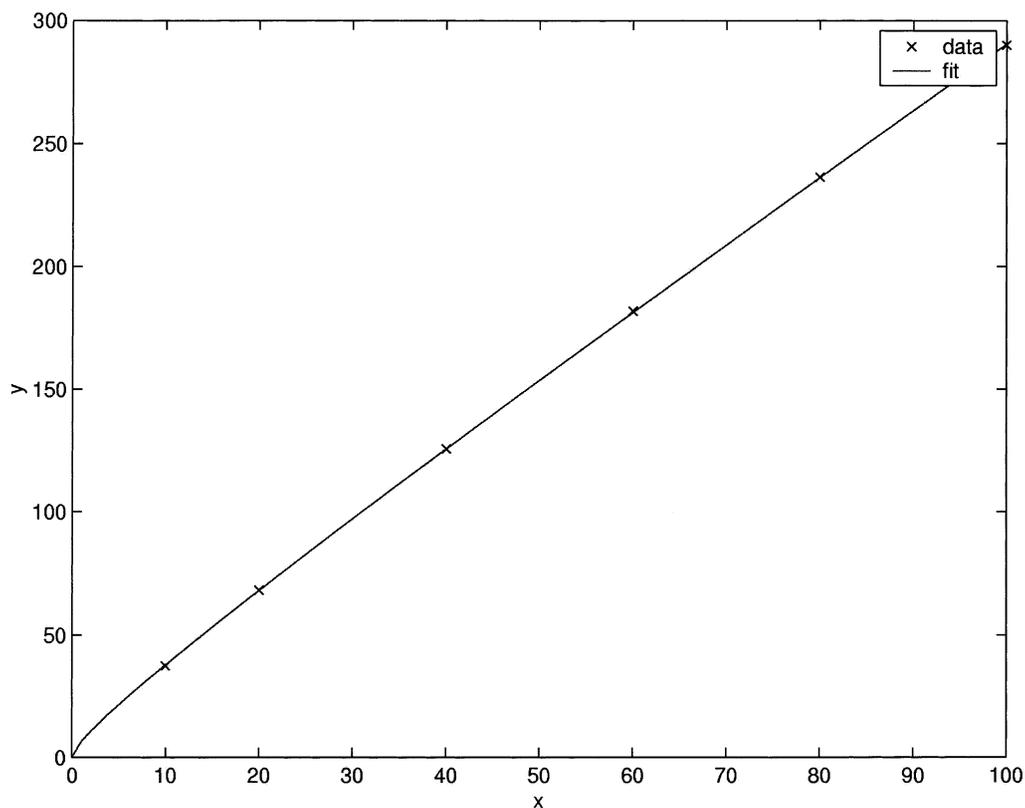
Figure 4 shows an essentially linear fit of the data. The graph was produced by fitting the function  $f(K) = 2.5K + C\sqrt{K}$ , where  $C$  is the single adjustable parameter. As a result, the fit is linear. The formal derivation that proves this expectation for max sum height is implied by the analysis in Jiang and Li (11).

Now we provide a brief motivation for the above interpolating function. When we find the shortest alignment of a single random oligo  $X_1X_2...X_K$  with ACGTACGT...,  $X_1$  aligns with the first, second, third or fourth base,  $X_2$  aligns with the first, second, third or fourth base after  $X_1$ ,  $X_3$  aligns with the first, second, third or fourth base after  $X_2$ , etc. Thus the expected distance between  $X_i$  and  $X_{i+1}$  is  $(1 + 2 + 3 + 4)/4 = 2.5$ . For example, if  $X_i = C$ , then the distance between  $X_i$  and  $X_{i+1}$  is 1 if  $X_{i+1} = T$ , 2 if  $X_{i+1} = G$ , 3 if  $X_{i+1} = A$  or 4 if  $X_{i+1} = C$ . The variance of the above possible distances is 1.25. Therefore, we expect  $X_1X_2...X_K$  to require an alignment of length  $2.5K$  and, by the law of large numbers, a random oligo

**Table 1.** Comparison of the oblivious strategy with the max sum height heuristic

<i>K</i>	<i>N</i>	Max sum height	Standard deviation	ACGT	Standard deviation	ADV
10	10 000	37.4	0.70	37.7	0.95	-3
	20 000	37.9	0.74	38.2	1.14	-3
	30 000	37.7	0.67	37.9	0.32	-2
	40 000	38.0	0.67	38.1	0.57	-1
20	10 000	68.1	0.57	67.7	1.25	4
	20 000	69.0	0.67	68.7	0.67	3
	30 000	69.7	0.67	70.7	0.82	-10
	40 000	69.7	1.06	70.1	1.20	-4
40	10 000	125.6	1.26	127.1	2.47	-15
	20 000	126.9	1.20	127.8	1.32	-9
	30 000	127.5	1.08	129.5	2.92	-20
	40 000	128.2	1.14	129.5	2.51	-13
60	10 000	181.3	1.49	183.2	2.94	-19
	20 000	183.3	1.42	184.0	1.33	-7
	30 000	183.8	1.69	184.3	1.77	-5
	40 000	184.8	1.81	185.4	2.72	-6
80	10 000	235.1	1.66	238.7	2.50	-36
	20 000	237.8	1.87	240.9	3.63	-31
	30 000	238.2	1.55	240.1	3.21	-19
	40 000	240.3	1.89	242.7	3.16	-24
100	10 000	290.6	2.01	291.8	2.10	-12
	20 000	292.6	2.17	294.1	2.33	-15
	30 000	293.3	1.49	295.8	2.57	-25
	40 000	294.6	1.17	296.6	2.27	-20

For each approach, we list the length of a strategy averaged over 10 experiments and the standard deviation. We also give the cumulative savings in nucleotides over 10 experiments (the ADV column). Note that the standard deviation is higher for the oblivious strategy (ACGT). *K* and *N* are the length and number of oligos on the microarray, respectively.



**Figure 4.** A linear fit of the function  $f(K) = 2.5K + 4.04\sqrt{K}$  to the simulation data produced by the oblivious strategy for 1000 oligos. The scaling constant (4.04) depends on the total number of oligos.

**Table 2.** Results of local search improvement (LS) for both max sum height (MSH) and oblivious solutions (ACGT)

$K$	$N$	MSH	MSH-LS	ACGT	ACGT-LS
10	100	33.1	32.6	34.6	32.5
	1000	35.9	35.7	36.1	35.7
20	100	62.3	61.1	63.3	60.6
	1000	65.4	65.3	65.9	64.7
40	100	115.4	114.6	117.3	114.2

$K$  and  $N$  are the length and number of oligos on the microarray, respectively.

$X_1X_2\dots X_K$  requires an alignment of length at most  $2.5K + O(\sqrt{K})$  with high probability. In order to align  $N$  oligos, length  $2.5K + O(\log N \sqrt{K})$  suffices with high probability. For a fixed number of oligos, the logarithm term is constant. Therefore, we can use the formula  $2.5K + O(\sqrt{K})$ .

We also used a simple local search to improve the solutions produced by the max sum height and oblivious strategies. The results are given in Table 2. We found that when the number of oligos is small (e.g. 100), the improvement was better (~3–5%). However, as the number of oligos grows (e.g. 1000), the percentage of improvement was reduced to ~1–1.5%. This is interesting, since Gibbs sampling is a close relative of local search and is a popular algorithm for multiple sequence alignment. However, in typical multiple alignments of proteins, we often align tens to hundreds of sequences. In this paper, we need to ‘align’ thousands to hundreds of thousands of oligos and it appears to have an impact on the degree of improvement obtained with this approach.

## CONCLUSIONS

In this paper, we have presented a computational formalization of the optimal synthesis strategy for oligonucleotide microarrays. We have shown that the problem is computationally intractable (NP complete). We have provided several simulation results that shed light on its practical complexity. As the number of applications of oligo microarrays increases and their use in diagnostic medical applications becomes a common practice, we expect the design of DNA chips to become more sophisticated and efficient. Our main conclusion from both the theoretical and simulation analyses provided in this paper is that the problem of optimal masking appears to be computationally difficult. Moreover, the simplest possible

solution appears to work almost as well as more sophisticated approaches that include heuristic greedy approaches and local search. It would be interesting to see if more exhaustive approaches based on best-first search, branch-and-bound or Gibbs sampling methods will generate a more dramatic improvement in performance. Naturally, these results must be confirmed in the context of practically used DNA chips.

## ACKNOWLEDGEMENTS

S.K. is supported in part by NSF grants IRI-9616254 and KDI-9980088. Z.W. is supported in part by NSF grants DBI-0078194 and DBI-0078194. R.B. is supported in part by NSF grants CCR-9996021 and CCR-0019019 and a grant from the State of New Jersey. A.D. was supported by the Paul and Daisy Soros Fellowship for New Americans.

## REFERENCES

- Schena, M., Shalon, D., Davis, R.W. and Brown, P.O. (1995) Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science*, **270**, 467–470.
- 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. *et al.* (1996) Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nat. Biotechnol.*, **14**, 1675–1680.
- Chee, M., Yang, R., Hubbell, E., Berno, A., Huang, X.C., Stern, D., Winkler, J., Lockhart, D.J., Morris, M.S. and Fodor, S.P. (1996) Accessing genetic information with high-density DNA arrays. *Science*, **274**, 610–614.
- Lipshutz, R.J., Fodor, S.P., Gingeras, T.R. and Lockhart, D.J. (1999) High density synthetic oligonucleotide arrays. *Nature Genet.*, **21** (suppl. 1), 20–24.
- Blanchard, A. (1998) Synthetic DNA arrays. *Genet. Eng.*, **20**, 111–123.
- Singh-Gasson, S., Green, R.D., Yue, Y., Nelson, C., Blattner, F., Sussman, M.R. and Cerrina, F. (1999) Maskless fabrication of light-directed oligonucleotide microarrays using a digital micromirror array. *Nat. Biotechnol.*, **17**, 974–978.
- Hubbell, E.A., Morris, M.S. and Winkler, J.L. (1999) Computer-aided engineering system for design of sequence arrays and lithographic masks. US Patent no. 5 856 101.
- Tolonen, A.C., Albeanu, D.F., Corbett, J.F., Handley, H., Henson, C. and Malik, P. (2002) Optimized *in situ* construction of oligomers on an array surface. *Nucleic Acids Res.*, **30**, e107.
- Gusfield, D. (1997) *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, New York, NY.
- Waterman, M.S. (1995) *Introduction to Computational Biology: Maps, Sequences and Genomes*. Chapman and Hall, New York, NY.
- Jiang, T. and Li, M. (1997) On the approximation of shortest common supersequences and longest common subsequences. *SIAM J. Comput.*, **24**, 1122–1139.