IMPROVING THE DESIGN OF GENECHIP ARRAYS BY COMBINING PLACEMENT AND EMBEDDING

Sérgio A. de Carvalho Jr.* and Sven Rahmann

Computational Methods for Emerging Technologies (COMET),
Genome Informatics, Technische Fakultät, Bielefeld University, D-33594 Bielefeld, Germany;
DFG GK Bioinformatik and Institute for Bioinformatics, CeBiTec, Bielefeld University
Email: {Sergio.Carvalho,Sven.Rahmann}@cebitec.uni-bielefeld.de

The microarray layout problem is a generalization of the border length minimization problem and asks to distribute oligonucleotide probes on a microarray and to determine their embeddings in the deposition sequence in such a way that the overall quality of the resulting synthesized probes is maximized. Because of its inherent computational complexity, it is traditionally attacked in several phases: partitioning, placement, and re-embedding. We present the first algorithm, Greedy+, that combines placement and embedding and results in improved layouts in terms of border length and conflict index (a more realistic measure of probe quality), both on arrays of random probes and on existing Affymetrix GeneChip® arrays. We also present a large-scale study on how the layouts of GeneChip arrays have improved over time, and show how Greedy+ can further improve layout quality by as much as 8% in terms of border length and 34% in terms of conflict index.

1. INTRODUCTION

Microarrays are a ubiquitous tool in molecular biology with a wide range of applications on a whole-genome scale including high-throughput gene expression analysis, genotyping, and resequencing. This article is about improving the design of high-density oligonucleotide microarrays, sometimes called DNA *chips*. This type of microarray consists of relatively short DNA *probes* (20–30-mers) synthesized at specific locations, called *features* or *spots*, of a solid surface, that are usually built by light-directed combinatorial chemistry, nucleotide-by-nucleotide.

For example, Affymetrix GeneChip® arrays have up to 1.3 million spots on a fused silica substrate measuring a little over 1 cm^2. The spots are as narrow as 5 μm (0.005 mm), and are arranged in a regularly-spaced rectangular grid. GeneChip arrays are produced with techniques derived from micro-electronics and integrated circuits fabrication. Probes are usually 25 bases long and are synthesized on the chip, in parallel, in a series of repetitive steps. Each step appends the same kind of nucleotide to probes of selected regions of the chip. The sequence of nucleotides added in each step is called *deposition sequence*. The selection of which probes receive the nucleotide is achieved with the help of photolithographic masks[3]. The quartz wafer of a GeneChip array is initially coated with a chemical compound topped with a light-sensitive protecting group that is removed when exposed to ultraviolet light, activating the compound for chemical coupling. A mask is used to direct light and remove the protecting groups of only those positions that should receive the nucleotide of a particular synthesis step. A solution containing adenine (A), thymine (T), cytosine (C) or guanine (G) is then flushed over the chip surface, but the chemical coupling occurs only in those positions that have been previously deprotected. Each coupled nucleotide also bears another protecting group so that the process can be repeated until all probes have been fully synthesized.

An alternative method of *in situ* synthesis uses an array of miniature mirrors to direct or deflect the incidence of light on the chip[10].

Regardless of which method is used to direct light, it is possible that some probes are accidentally activated for chemical coupling because of light diffraction, scattering or internal reflection on the chip surface. The unwanted illumination introduces unexpected nucleotides that change the probe sequences, significantly reducing their chances of successful hybridization with their targets, and increasing the risk of cross-hybridization with unintended targets.

This problem can be (and has been) alleviated by

[*] Corresponding author.

improving the production process, which however is expensive. Here, we are interested in computational methods that re-arrange the probes on the chip in such a way that the problem is minimized.

Note that the problem of unintended illumination primarily occurs near the borders between masked and unmasked spots (in the case of maskless synthesis, between a spot that is receiving light and a spot that is not); we thus speak of a *border conflict.*

By carefully designing the *arrangement* of the probes on the chip and their *embeddings* (the sequences of masked and unmasked steps used to synthesize each probe), it is possible to reduce the risk of unintended illumination. The problem has received some attention in the past, mostly by Hannenhalli et al.[4], Kahng et al.[6–8], and ourselves[1, 2]. In this paper, we put forward a new idea: We efficiently combine probe placement with probe embedding in a single algorithm; previously, these task have been done in separate phases. We also present a large-scale layout-quality study on several old and recent GeneChip arrays and propose alternative layouts with reduced conflicts.

In the next section, we state the *microarray layout problem* formally and define two different objective functions to be minimized. Section 3 contains our study of GeneChip arrays and shows how their layouts can be improved. Section 4 explains our new Greedy+ algorithm that achieves these improvements. Since Greedy+ builds on previous work, we briefly review the relevant details in Section 4.1 before presenting Greedy+ in Section 4.2 and results on chips with random probes in Section 4.3. Section 5 contains a concluding discussion. Supplementary material is available at `http://gi.cebitec.uni-bielefeld.de/comet/chiplayout/affy/`.

2. THE MICROARRAY LAYOUT PROBLEM

Data. The data for the microarray layout problem (MLP) consists of

- a set of probes $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, where each $p_k \in \{\mathtt{A},\mathtt{C},\mathtt{G},\mathtt{T}\}^*$ with $1 \leq k \leq n$ is produced by a series of T synthesis steps. Frequently, but not necessarily, all probes have the same length ℓ.
- a geometry of spots, or sites, $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$, where each spot s accommodates many copies of a unique probe $p_k \in \mathcal{P}$. Each probe is synthesized at a unique spot, hence there is a one-to-one assignment between probes and spots (if we assume that there are as many spots as probes, i.e., $m = n$). Some microarrays may have complex physical structures but we assume that the spots are arranged in a rectangular grid.
- the *nucleotide deposition sequence* $N = N_1 N_2 \ldots N_T$ corresponding to the sequence of nucleotides added at each synthesis step. It is a supersequence of all $p \in \mathcal{P}$ and often a repeated permutation of the alphabet $\Sigma = \{\mathtt{A},\mathtt{C},\mathtt{G},\mathtt{T}\}$, mainly because of its regular structure and because such sequences maximize the number of distinct subsequences. Each synthesis step t uses a mask M_t to induce the addition of a particular nucleotide $N_t \in \Sigma$ to a subset of $\mathcal{P}$ (Figure 1).

A probe may be *embedded* within N in several ways. An embedding of p_k is a T-tuple $\varepsilon_k = (\varepsilon_{k,1}, \varepsilon_{k,2}, \ldots, \varepsilon_{k,T})$ in which $\varepsilon_{k,t} = 1$ if probe p_k receives nucleotide N_t (at step t), and 0 otherwise. In particular, a *left-most embedding* is an embedding in which the bases are added as early as possible (as in ε_1 in Figure 1). Finding good embeddings is part of the problem.

Problem statement. Given $\mathcal{P}$, $\mathcal{S}$, and N as specified above, the MLP asks to specify a chip layout (λ, ε) that consists of

(1) a bijective assignment $\lambda : \mathcal{S} \to \{1, \ldots, n\}$ that specifies a probe index $\lambda(s)$ for each spot s (meaning that $p_{\lambda(s)}$ will be synthesized at s),

(2) an assignment $\varepsilon : \{1, \ldots, n\} \to \{0,1\}^T$ specifying an embedding $\varepsilon_k = (\varepsilon_{k,1}, \ldots, \varepsilon_{k,T})$ for each probe index k, such that $N[\varepsilon_k] :\equiv (N_t)_{t:\varepsilon_{k,t}=1} = p_k$,

such that a given penalty function is minimized. We now describe two such penalty functions: total border length and total conflict index.

Objective functions. The *total border length* $B(\lambda, \varepsilon)$ of a chip layout (λ, ε) was first introduced by Hannenhalli et al.[4], who defined the *border length*

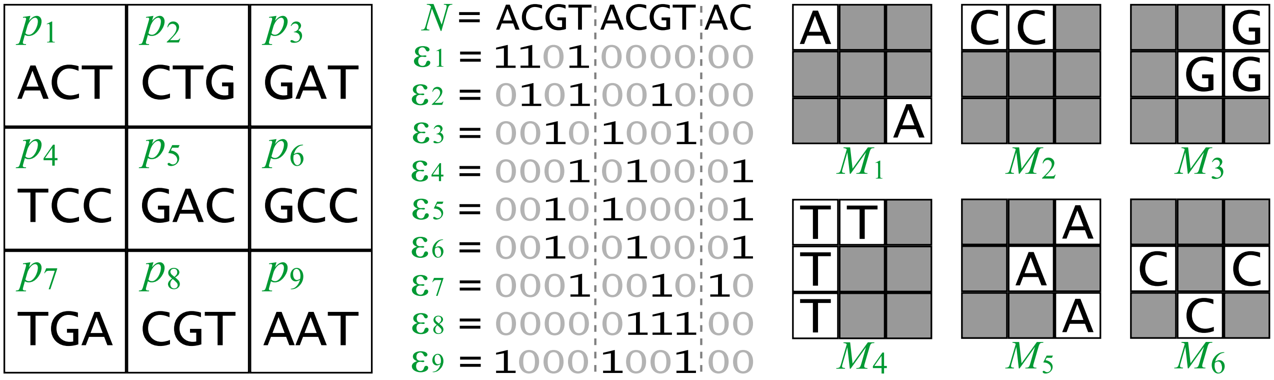

Fig. 1. Synthesis of a hypothetical 3×3 chip with photolithographic masks. Left: chip layout with 3-mer probe sequences. Center: deposition sequence with 2.5 cycles (delimited with dashed lines) and probe embeddings. Right: first six masks (masks 7 to 10 not shown).

$B_t(\lambda, \varepsilon)$ of a mask M_t as the number of borders separating masked and unmasked spots at synthesis step t. Then $B(\lambda, \varepsilon) = \sum_{t=1}^{T} B_t(\lambda, \varepsilon)$. As an example, the six masks shown in Figure 1 have $B_1 = 4$, $B_2 = 3$, $B_3 = 5$, $B_4 = 4$, $B_5 = 8$ and $B_6 = 9$. The total border length of that layout is 52 (masks M_7 to M_{10} are not shown).

Note that $B(\lambda, \varepsilon)$ can be expressed with the Hamming distance between embeddings of probes at adjacent spots: Let $H_\varepsilon(k, k')$ be the number of synthesis steps in which the embeddings ε_k and $\varepsilon_{k'}$ differ. Then $B(\lambda, \varepsilon) = \frac{1}{2} \sum_{s,s' \text{ adjacent}} H_\varepsilon(\lambda(s), \lambda(s'))$.

Ideally, all probes should have roughly the same risk of being damaged by unintended illumination, so that all hybridization signals are affected in approximately the same way. Total border length treats every conflict in the same way, which is reasonable without further information. However, it has been suggested previously[7] that stray light might activate not only adjacent neighbors but also spots that lie as far as three cells away from the targeted spot, and that imperfections produced in the middle of a probe are more harmful than in its extremities.

Therefore, as in Ref. 1, we define the *total conflict index* of a layout as $C(\lambda, \varepsilon) := \sum_s C(s)$, where $C(s) \equiv C(s; \lambda, \varepsilon)$ is the conflict index of a spot s defined as:

$$C(s) := \sum_{t=1}^{T} \Big(\mathbb{1}_{\{\varepsilon_{\lambda(s),t}=0\}} \cdot \omega(\varepsilon_{\lambda(s)}, t) \cdot \sum_{\substack{s':\text{ neighbor}\\ \text{of } s}} \mathbb{1}_{\{\varepsilon_{\lambda(s'),t}=1\}} \cdot \gamma(s, s') \Big). \quad (1)$$

The indicator functions ensure that there is a conflict at s during step t if and only if s is masked ($\varepsilon_{\lambda(s),t} = 0$) and a neighbor s' is unmasked ($\varepsilon_{\lambda(s'),t} = 1$).

Function $\gamma(s, s')$ is a "closeness" measure between s and s', defined as $\gamma(s, s') := (d(s, s'))^{-2}$, where $d(s, s')$ is the Euclidean distance between the spots s and s'. Note that, in (1), s' ranges over all neighboring spots that are at most three cells away from s.

The position-dependent weighting function $\omega(\varepsilon, t)$ accounts for the significance of the location inside the probe sequence where the undesired nucleotide is introduced in case of accidental illumination. It increases exponentially with the distance $\delta(\varepsilon, t)$ of the synthesized nucleotide from the probe's closer end, as motivated by thermodynamic considerations[1]: $\omega(\varepsilon, t) := c \cdot \exp(\theta \cdot \delta(\varepsilon, t))$, where $c > 0$ and $\theta > 0$ are constants. The parameter θ controls how steeply the exponential weighting function rises towards the middle of the probe. In our experiments, we use probes of length $\ell = 25$, and parameters $\theta = 5/\ell$ and $c = 1/\exp(\theta)$.

Problem variants and per-chip measures. We consider two variants of the MLP:

BLM *Border Length Minimization* (BLM) means that the objective is to minimize $B(\lambda, \varepsilon)$.

CIM *Conflict Index Minimization* (CIM) means that the objective is to minimize $C(\lambda, \varepsilon)$, which depends on the weighting functions γ and ω and their parameters, which we choose as described above.

In either case, we can measure both $B(\lambda, \varepsilon)$ and $C(\lambda, \varepsilon)$. Naturally, after BLM, $B(\lambda, \varepsilon)$ will be low, whereas $C(\lambda, \varepsilon)$ may be relatively large; the converse holds after CIM. In order to better compare chips of different size, we introduce normalized versions of these quantities.

NBL If the the chip is a rectangular grid with n_r rows and n_c columns, the number of internal borders is $n_b = n_r(n_c - 1) + n_c(n_r - 1) \approx 2n_r n_c = 2|\mathcal{S}|$, and we call $B(\lambda, \varepsilon)/n_b$ the *normalized border length* (NBL). We may also refer to the NBL of a particular mask M_t as B_t/n_b.

ABC Real arrays have a significant number of empty spots (as much as 11.94% on the Affymetrix Chicken Genome array). To better compare chips with different amounts of empty spots we use the *average number of border conflicts per probe* (ABC), defined as $B(\lambda, \varepsilon)/|\mathcal{P}|$. We roughly have ABC $\approx 2 \cdot$ NBL if $|\mathcal{S}| \approx |\mathcal{P}|$. The ABC of a particular mask M_t is $B_t/|\mathcal{P}|$.

ACI We define the *average conflict index* (ACI) of a layout as $C(\lambda, \varepsilon)/|\mathcal{P}|$.

3. ANALYSIS OF GENECHIP ARRAYS

We obtained the specification of several GeneChip arrays containing the list of probe sequences and their positions on the chip from Affymetrix's web site[a]. We make a few assumptions because some details such as the deposition sequence used to synthesize the probes, the probe embeddings, and the contents of "special" spots are not publicly available (some of the special spots contain *quality control probes* used to detect failures during the production of the chip). Not knowing the contents of these special spots barely interferes with our analysis because, in all arrays we examined, they amount to at most 1.22% of the total number of spots.

It has been reported that a fixed 74-step deposition sequence is used by Affymetrix[7]. All GeneChip arrays we analyzed, regardless of their size, can be synthesized in $N = (\texttt{TGCA})^{18}\texttt{TG}$, i.e., 18.5 cycles of `TGCA`, and a shorter deposition sequence is indeed unlikely. This suggests that only sub-sequences of this particular deposition sequence can be used as probes on Affymetrix chips. In principle, this should not be a problem as this sequence covers about 98.45% of all 25-mers[9].

Probes of GeneChip arrays appear in pairs: the perfect match (PM), which perfectly matches its target sequence, and the mismatch (MM) probe, which is used to quantify cross-hybridizations and unpredictable background signal variations. The MM probe is a copy of the PM probe except for the middle base (position 13 of the 25-mer), which is exchanged with its Watson-Crick complement. The layout of a GeneChip alternates rows of PM probes with rows of MM probes in such a way that the probes of a pair are always adjacent on the chip. Moreover, PM and MM probes are *pair-wise left-most embedded*. Informally, a pair-wise left-most embedding is obtained from left-most embeddings by shifting the second half of one embedding to the right until the two embeddings are "aligned" in the synthesis steps that follow the mismatched middle bases. This approach reduces border conflicts between the probes of a pair, but it leaves a conflict in the steps that add the middle bases. The fact that probes must appear in pairs restricts even more which sequences can be used as probes on GeneChip arrays because both PM and MM probes must "fit" in the deposition sequence.

Results. Figure 2 shows the ABC for each masking step of three GeneChip arrays (Yeast, Human and *E. coli*). We assume that the probes are pair-wise left-most embedded in $N = (\texttt{TGCA})^{18}\texttt{TG}$, and we consider all spots whose contents are not available as empty spots.

In all chips we analyzed, the ABC is higher in the steps that add the middle bases, a result of placing PM and MM probes in adjacent spots.

The Yeast Genome S98 array has the worst layout in terms of border conflicts, and most of the earlier GeneChip arrays such as the *E. coli* Antisense Genome have similar levels of conflicts. The layout of the Human Genome U95A2 array has significantly fewer border conflicts than the Yeast array, suggesting that it was designed with a better placement strategy. The curve of the *E. coli* Genome 2.0 array, with very low levels of conflicts in the first 10 masks, is typical of the latest generation of GeneChip arrays, including the Chicken Genome and the Wheat Genome (one of the largest GeneChip arrays currently available with 1 164 × 1 164 spots), which suggest yet another placement strategy.

Table 1 shows summary statistics on several

[a] `http://www.affymetrix.com/support/technical/byproduct.affx?cat=arrays`

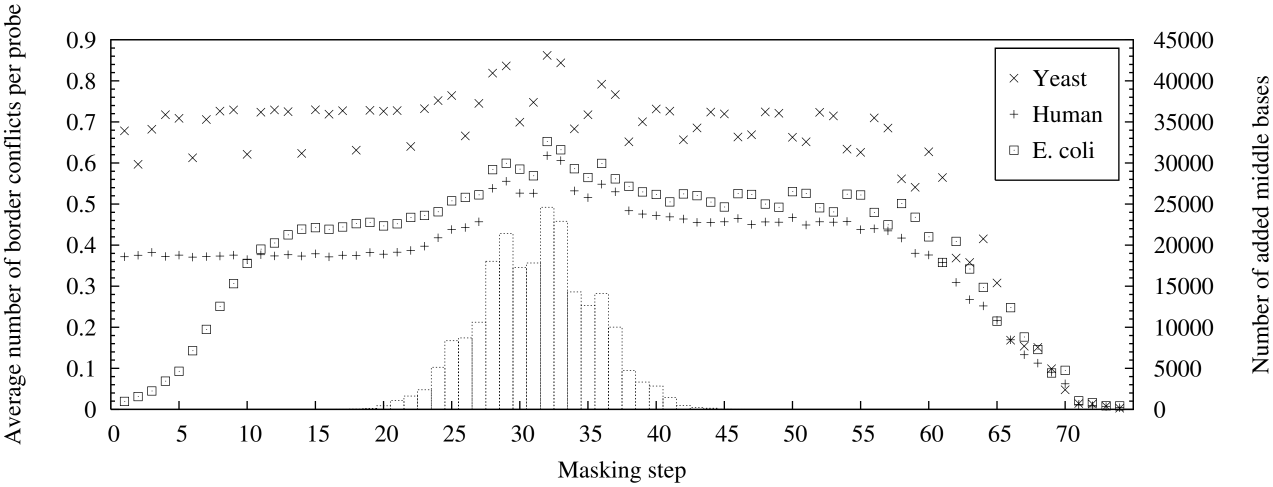

Fig. 2. Average number of border conflicts per probe (scale on the left y-axis) of selected GeneChip arrays: Yeast Genome S98, Human Genome U95A2, and *E. coli* Genome 2.0. The histogram shows the number of middle bases added per synthesis step on the *E. coli* 2.0 chip (scale on the right y-axis).

commercially available arrays. The layout of the Human Genome U95A2 array is one of the best in terms of NBL and the best in terms of ACI. This, however, has more to do with empty spots than with the placement strategy as this chip has about 1.83% of empty spots that are evenly distributed on the chip surface. In contrast, the Chicken Genome array has an exceptionally high percentage of empty spots (11.94%) that contribute to its low NBL but not equally to a low ABC in comparison with the Human Genome array because the empty spots are concentrated in the lower part of the chip (figures illustrating the distribution of empty spots on these chips are available in the supplementary web page).

GeneChip arrays exhibit relatively low levels of NBL and ABC when compared to layouts produced by the best algorithms for arrays of random probes of similar dimensions (see next section). This can be explained by the fact that each probe has a nearly identical copy next to it. However, they have relatively high ACIs because the conflicts are concentrated on the synthesis steps of the middle bases, which are expensive in the conflict index model.

Design improvements. We used our new algorithm Greedy+ with different parameters Q, and Sequential[8] re-embedding algorithm (see Section 4 for explanations; in general, larger Q gives better layouts, but also increases the running time), to create alternative layouts for two of the latest generation of GeneChip arrays: *E. coli* Genome 2.0 and Wheat Genome. Greedy+ was modified to avoid placing probes on special spots or empty spots that we believe might have a function on the chip. For each chip we separately run both BLM and CIM versions of the algorithms.

The main difference between our layouts and the original ones is that we do not require the arrays to alternate rows of PM and MM probes; hence, probes of a pair are not necessarily placed on adjacent spots. This is especially helpful for CIM since it avoids conflicts in the middle bases. With BLM, we observe that Greedy+ places between 90.7% and 95.2% of the PM probes adjacent to their corresponding MM probes. With CIM, this rate drops to between 12.9% and 21.3%.

Figure 3 shows the NBL for each masking step of the layout produced by Greedy+ and Sequential for the *E. coli* Genome 2.0 array in comparison with the original Affymetrix layout. It can be clearly seen that the CIM variant of our algorithm greatly reduces the number of border conflicts in the middle synthesis steps, where conflicts are expensive. In the BLM variant, the conflicts are distributed more evenly across all synthesis steps. To compare the new layout algorithm with re-embedding only, we also show the result of running a pair-wise version of Sequential on the original layout (this version ensures that the embeddings of PM-MM pairs remain pair-wise "aligned").

The total NBL and ACI values of these layouts are also shown in Table 2, together with several lay-

Table 1. Average number of border conflicts per probe (ABC), normalized border length (NBL) and average conflict index (ACI) of selected GeneChip arrays. The dimension of the chip, the percentage of spots with unknown content and the percentage of empty spots are also shown.

GeneChip Array	Dimension	Unknown	Empty	ABC	NBL	ACI
Yeast Genome S98	534 × 534	1.22%	1.70%	44.8168	21.7945	669.0663
E. coli Antisense Genome	544 × 544	1.17%	3.12%	43.3345	20.7772	663.7353
Human Genome U95A2	640 × 640	0.96%	1.83%	28.2489	13.7517	**510.3418**
E. coli Genome 2.0	478 × 478	1.08%	0.46%	29.2038	14.4079	550.2014
Chicken Genome	984 × 984	0.46%	11.94%	28.2087	**12.3680**	540.5022
Wheat Genome	1 164 × 1 164	0.38%	0.08%	**27.6569**	13.7771	539.9632

outs for the Wheat Genome array. Greedy+ with $Q = 10\text{K}$ produces a layout with 8.10% less border conflicts than the original layout for *E. coli* array (13.2406 versus 14.4079) in 218.3 minutes. With $Q = 2\text{K}$, the improvement is almost as good (7.15%), but requires only 46.9 minutes. For the larger Wheat array, Greedy+ with $Q = 2\text{K}$ generates a layout with 7.36% less border conflicts than the original layout (12.7622 versus 13.3771). In terms of CIM, our results show that Greedy+ can improve the quality of GeneChip arrays in as much as 34.31% (from 550.2014 to 361.4418 for the *E. coli* array).

4. ALGORITHMS

Traditionally, The MLP has been attacked heuristically in two phases, as exact solutions are computationally infeasible.

First, an initial embedding of the probes is fixed and an arrangement of these embeddings on the chip with minimum conflicts is sought. This is usually referred to as the *placement* phase. Placement algorithms typically assume that an initial embedding of the probes is given (which can be a left-most or otherwise pre-computed embedding), and do not change the given embeddings.

Second, a post-placement optimization phase *re-embeds* the probes considering their location on the chip, in such a way that the conflicts with neighboring spots are further reduced.

For superlinear placement algorithms, the chip is often *partitioned* into smaller sub-regions before the placement phase in order to reduce running times, especially on larger chips.

We briefly review the best known placement and re-embedding principles and then present a new algorithm, Greedy+, the first one to combine placement and embedding into a single phase. In addition to the results presented in the previous section, we show in Section 4.3 that Greedy+ compares favorably to the best known placement strategy (Row-Epitaxial). Partitioning algorithms such as Centroid-based Quadrisection[8] and Pivot Partitioning[1] are not discussed.

4.1. Review of Existing Placement and Re-Embedding Strategies

Placement. The following elements of placement strategies have proven successful in practice for large-scale chips.

Initial ordering The probe sequences (or their binary embeddings) are initially ordered, either lexicographically[7], which is easy, or to minimize the sum of distances of consecutive probes, which leads to an instance of the NP-hard traveling salesman problem (TSP) that is then solved heuristically[4].

***k*-threading** The sequence of ordered probes is threaded onto the chip. This can happen row-by-row, where the first row is filled left-to-right, the second one right-to-left, and so on. This leads to an arrangement where consecutive probes in the same row have few border conflicts, but probes in the same column may have a significant number of conflicts. An alternative is provided by k-threading[4], in which the right-to-left and left-to-right steps are interspaced with alternating upward and downward movements over k sites. Row-by-row threading can be seen as k-threading with $k = 0$.

Iterative refinement The Row-Epitaxial[7] algorithm refines an existing layout as follows: Spots are re-considered in a pre-defined order, from top to bottom, left to right. For each spot s, a user-defined number Q of probe candidates below and to the right of s is considered for an

Table 2. Normalized border length (NBL) and average conflict index (ACI) of layouts for the *E. coli* 2.0 and Wheat GeneChip arrays. Greedy+ used k-threading with $k = 5$ for BLM and $k = 0$ for CIM. Running times in minutes include placement and two passes of re-embedding with Sequential.

Array	Layout	NBL	ACI	Time
E. coli 2.0	Affymetrix with pair-wise left-most	14.4079	550.2014	—
	Affymetrix after "pair-aware" Sequential (BLM)	13.5005	541.0954	—
	Greedy+ with Q = 2K and Sequential (BLM)	13.3774	529.8129	46.9
	Greedy+ with Q = 10K and Sequential (BLM)	**13.2406**	515.5917	218.3
	Greedy+ with Q = 2K and Sequential (CIM)	17.6935	394.9905	54.9
	Greedy+ with Q = 10K and Sequential (CIM)	17.5575	**361.4418**	225.7
Wheat	Affymetrix with pair-wise left-most	13.7771	539.9632	—
	Affymetrix after "pair-aware" Sequential (BLM)	12.9151	531.2692	—
	Greedy+ with Q = 2K and Sequential (BLM)	12.7622	519.0869	279.2
	Greedy+ with Q = 5K and Sequential (BLM)	**12.6670**	511.7193	676.0
	Greedy+ with Q = 2K and Sequential (CIM)	17.1047	387.8430	322.7
	Greedy+ with Q = 5K and Sequential (CIM)	17.1144	**366.6045**	704.7

exchange with the probe p at s. Probe p is then swapped with the probe that generates the minimum number of border conflicts between s and its left and top neighbors.

In the experiments conducted by Kahng et al.[7], Row-Epitaxial was the best large-scale placement algorithm for the BLM problem.

We have adapted Row-Epitaxial to CIM by choosing the probe candidate that minimizes the sum of conflict indices in a region around s restricted to those neighboring spots that have been already refilled.

Re-embedding. Most current re-embedding strategies are based on the Optimum Single Probe Embedding algorithm (OSPE; see below) first introduced by Kahng et al.[6] and differ mainly in the order in which the spots are considered. Some of the proposed strategies are Chessboard, Greedy and Batched Greedy[6], and Sequential[8].

The Sequential strategy proceeds spot by spot, from top to bottom, left to right, re-embedding each probe optimally with regard to its neighbors using OSPE. Once the end of the array is reached, it is restarted at the top left corner of the array for the next iteration, until a local optimal solution is found, or until improvements drop below a given threshold, or until a given number of passes have been executed. Sequential is not only the simplest but also the fastest and most effective known strategy[8]. Therefore, we skip the discussion of other strategies.

OSPE is a dynamic programming algorithm (a variant of global sequence alignment) that computes an optimum embedding of a single probe p (of length ℓ) at a given spot s into the deposition sequence N (of length T) with respect to p's neighbors, whose embeddings are considered as fixed. The algorithm was originally developed for BLM but a more general form designed for conflict index minimization (CIM) was given by de Carvalho Jr. and Rahmann[1].

OSPE fills an $(\ell + 1) \times (T + 1)$ dynamic programming matrix D, where $D[i, t]$ is defined as the minimum cost of an embedding of $p_{1..i}$ into $N_{1..t}$ for $0 \leq i \leq \ell$, $0 \leq t \leq T$. The cost is the sum of conflicts induced by the embedding of $p_{1..t}$ on its neighbors (when s is unmasked and a neighbor is masked), plus the conflicts suffered by $p_{1..i}$ because of the embeddings of its neighbors (when s is masked and a neighbor is unmasked). The basic recurrence is

$$D[i,t] = \begin{cases} \min \begin{Bmatrix} D[i,t-1] + M_{i,t}, \\ D[i-1,t-1] + U_t \end{Bmatrix} & \text{if } p_i = N_t, \\ D[i,t-1] + M_{i,t} & \text{if } p_i \neq N_t. \end{cases}$$

In accordance with the conflict index model, the additional costs U_t (incurred at masked neighbors when s is unmasked, only possible if $p_i = N_t$) and $M_{i,t}$ (incurred at masked s because of unmasked neighbors)

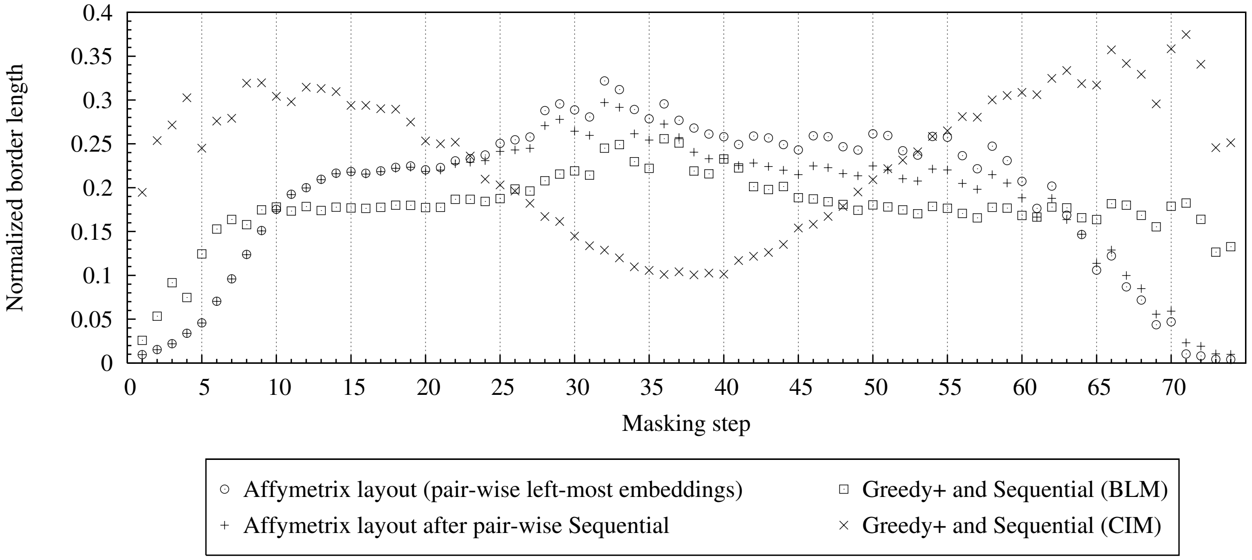

Fig. 3. NBL for each masking step of the original Affymetrix layout for the *E. coli* 2.0 GeneChip compared with alternative layouts produced by Greedy+ (with $Q = 10\text{K}$) and Sequential. The layout resulting from running Sequential on the original layout is also shown.

are

$$U_t := \sum_{\substack{s':\ \text{neighbor}\\ \text{of } s}} \mathbb{1}_{\{\varepsilon_{\lambda(s'),t}=0\}} \cdot \omega(\varepsilon_{\lambda(s')},t) \cdot \gamma(s',s),$$

$$M_{i,t} := c \cdot \exp(\theta \cdot (1 + \min\{i, \ell - i\})) \cdot \sum_{\substack{s':\ \text{neighbor}\\ \text{of } s}} \mathbb{1}_{\{\varepsilon_{\lambda(s'),t}=1\}} \cdot \gamma(s,s').$$

The initialization is given by $D[0,0] = 0$, $D[i,0] = \infty$ for $0 < i \leq \ell$, and $D[0,t] = D[0,t-1] + M_{0,t}$ for $0 < t \leq T$.

4.2. Greedy+: Merging Placement and Embedding

The problem with the "place first then re-embed" approach is that once the placement is fixed, there is usually little freedom for optimization by re-embedding the probes. Better results should be obtained when the placement and embedding phases are considered simultaneously instead of separately. However, because of the generally high number of possible embeddings of each probe, it is a challenge to design algorithms that efficiently use the additional freedom and run reasonably fast in practice. In this section, we propose Greedy+, the first placement algorithm that simultaneously places and embeds the probes. After the user has chosen two parameters Q and k, the overall strategy is as follows.

(1) Sort the probes lexicographically and store them, in sorted order, in a doubly linked list L.

(2) Place a randomly selected probe p at the first spot, using any reasonable embedding.

(3) Remove p from L, but remember its former position.

(4) For each following spot s of the array in a k-threading pattern:

  (a) For each of the Q probe candidates q closest to p's former position in L:

  - Compute q's optimal embedding with respect to the already-filled neighbors of s by temporarily placing q at s and using OSPE. Denote the best cost for q by $c(q)$.
  - Keep track of the minimum cost $c^* = \min_q c(q)$ and the corresponding best probe candidate q^*.

  (b) Place q^* at s with its optimal embedding.

  (c) Set $p \leftarrow q^*$. Remove p from L, but remember its former position.

(5) Optionally, run Sequential re-embedding over the whole array.

Compared to Row-Epitaxial, Greedy+ clearly spends more time evaluating each probe candidate. For this reason, we must use lower numbers Q of candidates per spot to achieve a running time comparable to Row-Epitaxial.

Table 3. Normalized border length (NBL) of layouts produced by Row-Epitaxial and Greedy+ with border length minimization (both using 0-threading) on random chips in approximately the same time (running times in minutes including two passes of Sequential re-embedding optimization). The relative difference in NBL and time between the two approaches is shown in percentage.

	Row-Epitaxial and Sequential			Greedy+ and Sequential			Relative	
Dim.	Q	NBL	Time	Q	NBL	Time	NBL	Time
300 × 300	10 000	18.0524	4.3	300	**17.9807**	4.2	−0.40%	−1.24%
	20 000	17.9430	9.5	700	**17.6746**	9.2	−1.50%	−2.85%
500 × 500	10 000	17.3584	16.0	450	**17.2216**	16.0	−0.79%	−0.40%
	20 000	17.2502	34.7	950	**16.9382**	30.4	−1.81%	−12.51%
800 × 800	10 000	16.7176	45.6	500	**16.6549**	41.7	−0.38%	−8.51%
	20 000	16.6012	100.1	1 130	**16.3175**	97.7	−1.71%	−2.41%

Three observations significantly reduce the time spent with OSPE computations when several probe candidates q are considered in succession for filling the same spot.

(1) The U_t and $M_{i,t}$ costs of OSPE need to be computed only once for a given spot s since they do not depend on the probe sequence placed at s: U_t depends solely on the existing neighbors of s, whereas $M_{i,t}$ depends on the neighbors of s and on the number i of bases already appended to q at synthesis step t (if all probes have the same length ℓ, then c and θ are constants).

(2) Once we know that there exists some q that can be placed at s with cost κ, we can stop the OSPE computation for other candidates as soon as all values in a row of the OSPE matrix D are greater than or equal to κ.

(3) If two candidates q and q' share a common prefix of length r, rows 0 through r of D are identical for q and q', so we can skip the re-computation. In order to fully exploit this fact, we examine the probes in lexicographical order so that we maximize the length of the common prefixes between two consecutive probe candidates. For this reason, Greedy+ uses the doubly-linked list L to maintain the probes in lexicographical order.

4.3. Results on Chips with Random Probes

We compare the layouts produced by Row-Epitaxial and Greedy+ when both algorithms are given approximately the same amount of time (the parameter Q is chosen differently for both algorithms so that the running times are comparable). For this experiment we use probes of length $\ell = 25$ i.i.d. randomly generated and left-most embedded in the standard Affymetrix deposition sequence (all results are averages over a set of ten arrays). Note that, although we use Affymetrix's deposition sequence, the probes on these arrays do not appear in pairs. For Row-Epitaxial, an initial placement is constructed by threading a lexicographically sorted list of probes using 0-threading, i.e., row-by-row. To be fair, since Row-Epitaxial is a traditional placement algorithm that does not change the probe embeddings, we need to compare the layouts obtained by both algorithms after a re-embedding phase. For this task we use the Sequential algorithm, performing two passes of re-embedding optimization.

The results are shown in Tables 3 (NBL after BLM) and 4 (ACI after CIM). For BLM, Greedy+ produces significantly better results in less time while looking at fewer probe candidates. For CIM, Greedy+ produces better layouts in approximately the same amount of time (or less), except for the smallest chips: On 300 × 300 arrays, Row-Epitaxial produces layouts with lower ACIs, but it quickly reaches its limit in terms of probe candidates per spot. Greedy+ examines fewer probe candidates to achieve similar results, and thus have a greater potential for producing better layouts. For instance, the largest value of Q for Row-Epitaxial on 300×300 chips ($Q = 90\,000$) produces a layout with 402.5457 ACI. Greedy+ produces a better layout (401.8089 ACI) already with $Q = 5\,500$ (although that takes more time than Row-Epitaxial with $Q = 90\,000$). Our results also suggest that the larger values of Q are used, the greater is the advantage of Greedy+.

In further experiments (details not shown), Row-Epitaxial often produces the best results (for both

Table 4. Average conflict index (ACI) of layouts produced by Row-Epitaxial and Greedy+ with conflict index minimization (both using 0-threading) on random chips in approximately the same time (running times in minutes including two passes of Sequential re-embedding optimization).

	Row-Epitaxial and Sequential			Greedy+ and Sequential			Relative	
Dim.	Q	ACI	Time	Q	ACI	Time	ACI	Time
300 × 300	10 000	**440.2397**	12.3	900	442.8057	12.4	+0.58%	+0.42%
	20 000	**423.4236**	21.2	1 900	423.9464	21.3	+0.12%	+0.60%
	90 000	402.5457	50.6	5 500	**401.8089**	53.9	−0.18%	+6.65%
500 × 500	10 000	434.9764	38.3	1 050	**432.9102**	38.1	−0.48%	−0.48%
	20 000	417.8499	68.7	2 150	**414.2703**	66.2	−0.86%	−3.67%
800 × 800	10 000	428.6301	106.6	1 150	**424.7285**	104.3	−0.91%	−2.12%
	20 000	412.4495	187.9	2 400	**405.6095**	184.4	−1.66%	−1.90%

BLM and CIM) with $k = 0$, although the best initial layouts are frequently produced with high values of k (e.g., $k = 4$), contradicting the results of Hannenhalli et al.[4] Greedy+ consistently achieves the best results with $k = 0$ for CIM, and with surprisingly high values of k (e.g., $k = 14$) for BLM. The results shown in Tables 3 and 4 use $k = 0$ (row-by-row threading), so the advantage of Greedy+ over Row-Epitaxial, in terms of BLM, is even greater in many cases.

5. DISCUSSION

We have presented a large-scale study on the layout of GeneChip arrays. Our analysis suggest that placing perfect match (PM) and mismatch (MM) probes on adjacent spots is responsible for the low border length on GeneChip arrays. However, this has the disadvantage of concentrating the conflicts on those synthesis steps that add the middle bases, precisely where an unintentionally added nucleotide results in the highest damage to the probes. Our results indicate that, if PM and MM probes are not regularly placed in alternating rows, the average conflict index (ACI) may be reduced by as much as 34%. However, other desired properties might be lost, e.g., the correlation of PM and MM signal due to spatial effects. We remark that several researchers in the past have proposed to ignore the MM signals altogether[5]. Of course, the exact numbers (such as the 34% above) depend on the parameters of the conflict index model, which are subject to debate. However, changing them does not qualitatively change the results: In fact, our estimate of the relative importance of the middle bases for the integrity of the probes is rather conservative.

We have also proposed the first layout algorithm, Greedy+, that combines the previously separate phases of placement and embedding. Also, in contrast to most previous work, we use two models to evaluate layout quality: border length minimization and conflict index minimization. For fair comparisons, we have adapted the existing methods to the conflict index model. As evident by the results in Section 4.3, Greedy+ is the best placement strategy for border length minimization. It is also the best for conflict index minimization, except for the smaller chips and when running time is limited. In fact, the advantage of Greedy+ becomes more apparent for larger chips and greater number of candidates per spot. This makes Greedy+ an ideal candidate for truly large designs. It should also be noted that Greedy+ outperforms previous algorithms regardless of how PM and MM probes are placed on the chip, as can be seen on the results with random chips (where there are no probes pairs).

Acknowledgments and availability. We thank Jens Stoye for many helpful discussions, Robert Giegerich for additional funding, and the reviewers for helping improving the presentation. Supplementary material is available online at `http://gi.cebitec.uni-bielefeld.de/comet/chiplayout/affy/`.

References

1. S. A. de Carvalho Jr. and S. Rahmann. Improving the layout of oligonucleotide microarrays: Pivot Partitioning. In P. Bucher et al., editors, *Proceedings of the 6th Workshop of Algorithms in Bioinformatics*, volume 4175 of *Lecture Notes in Computer Science*, pages 321–332. Springer, 2006.
2. S. A. de Carvalho Jr. and S. Rahmann. Microarray layout as a quadratic assignment problem. In

D. Huson et al., editors, *Proceedings of the German Conference on Bioinformatics*, volume P-83 of *Lecture Notes in Informatics (LNI)*, pages 11–20. Gesellschaft für Informatik, 2006.

3. S. P. Fodor, J. L. Read, M. C. Pirrung, L. Stryer, A. T. Lu, and D. Solas. Light-directed, spatially addressable parallel chemical synthesis. *Science*, 251(4995):767–773, 1991.
4. S. Hannenhalli, E. Hubell, R. Lipshutz, and P. A. Pevzner. Combinatorial algorithms for design of DNA arrays. *Advances in Biochemical Engineering Biotechnology*, 77:1–19, 2002.
5. R. A. Irizarry, B. M. Bolstad, F. Collin, L. M. Cope, B. Hobbs, and T. P. Speed. Summaries of Affymetrix GeneChip probe level data. *Nucleic Acids Research*, 31(4):e15, Feb 2003.
6. A. Kahng, I. Mandoiu, P. Pevzner, S. Reda, and A. Zelikovsky. Border length minimization in DNA array design. In R. Guigó et al., editors, *Algorithms in Bioinformatics (Proceedings of WABI)*, volume 2452 of *Lecture Notes in Computer Science*, pages 435–448. Springer, 2002.
7. A. B. Kahng, I. Mandoiu, P. Pevzner, S. Reda, and A. Zelikovsky. Engineering a scalable placement heuristic for DNA probe arrays. In *Proceedings of the seventh annual international conference on research in computational molecular biology (RECOMB)*, pages 148–156. ACM Press, 2003.
8. A. B. Kahng, I. Mandoiu, S. Reda, X. Xu, and A. Z. Zelikovsky. Evaluation of placement techniques for DNA probe array layout. In *Proceedings of the 2003 IEEE/ACM international conference on Computer-aided design (ICCAD)*, pages 262–269. IEEE Computer Society, 2003.
9. S. Rahmann. Subsequence combinatorics and applications to microarray production, DNA sequencing and chaining algorithms. In M. Lewenstein et al., editors, *Combinatorial Pattern Matching (CPM)*, volume 4009 of *LNCS*, pages 153–164, 2006.
10. S. Singh-Gasson, R. D. Green, Y. Yue, C. Nelson, F. Blattner, M. R. Sussman, and F. Cerrina. Maskless fabrication of light-directed oligonucleotide microarrays using a digital micromirror array. *Nat Biotechnol*, 17(10):974–978, Oct 1999.