Approximating Border Length for DNA Microarray Synthesis

Cindy Y. Li[1] Prudence W.H. Wong[1] Qin Xin[2] Fencol C.C. Yung[3]

[1] Department of Computer Science, University of Liverpool, UK.
{cindyli,pwong}@liverpool.ac.uk
[2] Simula Research Lab, Norway.
xin@simula.no
[3] Department of Computer Science, University of Hong Kong, Hong Kong.
ccyung@graduate.hku.hk

Abstract. We study the border minimization problem (BMP), which arises in microarray synthesis to place and embed probes in the array. The synthesis is based on a light-directed chemical process in which unintended illumination may contaminate the quality of the experiments. Border length is a measure of the amount of unintended illumination and the objective of BMP is to find a placement and embedding of probes such that the border length is minimized. The problem is believed to be NP-hard. In this paper we show that BMP admits an $O(\sqrt{n}\log^2 n)$-approximation, where n is the number of probes to be synthesized. In the case where the placement is given in advance, we show that the problem is $O(\log^2 n)$-approximable. We also study a related problem called agreement maximization problem (AMP). In contrast to BMP, we show that AMP admits a constant approximation even when placement is not given in advance.

1 Introduction

DNA microarrays [9] 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 [21] in a cell, microarray is 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*. DNA microarray design raises a number of challenging combinatorial problems, such as probe selection [10, 14, 18, 22], deposition sequence design [17, 19] and probe placement and synthesis [3–5, 12, 15, 16]. In this paper, we focus on the probe placement and synthesis problem.

Probes are synthesized on the microarray through the process called *very large-scale immobilized polymer synthesis* (VLSIPS) [8]. In each step, light is selectively allowed through a *mask* to expose *spots* in the microarray in order to activate the nucleotides in the spots. The patterns of the masks used and the sequence of the deposition nucleotides in the illumination define the ultimate sequence of nucleotides of the array spot. A mask consists of masked (blocking light) and unmasked (allowing light) regions and induces deposition of a particular nucleotide (A, C, G or T) at its exposed array *spots*. The *deposition sequence* D corresponding to the sequence of masks is a supersequence of all probes in the array (see example in Figure 1).

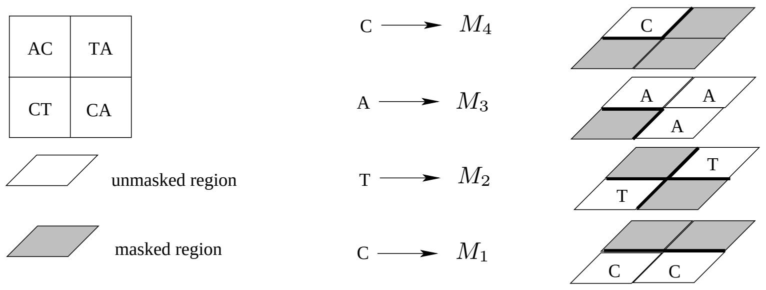

Fig. 1. Synthesis of a 2×2 microarray. The deposition sequence D = CTAC corresponds to the sequence of four masks M_1, M_2, M_3, and M_4. The masked regions are shaded. The borders between the masked and unmasked regions are represented by bold lines.

DNA microarray synthesis consists of two components, namely *probe placement* and *probe embedding*. Given a set of probes to be synthesized, probe placement is to place each probe to a unique spot in the microarray and probe embedding is the sequence of masked and unmasked steps used in the synthesis. For example, in Figure 2, the deposition sequence is $(\text{ACGT})^3$ and the sequence (a) $\text{A}(-)^4\text{C}(-)^5\text{T}$ is a possible embedding of the probe ACT, where "$-$" represents a space.

We distinguish two types of synthesis, namely, *synchronous* and *asynchronous* synthesis. In synchronous synthesis, each deposition nucleotide can only be deposited to the i-th position of the probes for a particular i. In asynchronous synthesis, there is no such restriction, allowing arbitrary embeddings. For example, Figure 1 shows an asynchronous synthesis in which M_2 deposits a nucleotide to the second position of the sequence CT and the first position of TA. Asynchronous synthesis is more flexible, yet more difficult to optimize. In this paper we focus on asynchronous synthesis.

Due to diffraction, internal reflection and scattering, spots on the *border* between masked and unmasked regions are often subject to unintended illumination [8]. This uncertainty produces unpredicted probes that can compromise experimental results. As microarray chip is expensive to synthesize, it is usual that as many probes as possible are placed in a chip (i.e., as many entries are used), while unintended illumination has to be minimized. The magnitude of unintended illumination can be measured by the *border length* of the masks used, which is the number of borders shared between masked and unmasked regions, e.g., in Figure 1, the border length of M_1, M_3, M_4 is 2 and M_2 is 4.

To reduce the amount of unintended illumination, one can exploit freedom in placing probes in the microarray during probe placement and choosing different probe embeddings. The *Border Minimization Problem (BMP)* [12] is to find a placement of the probes on the microarray together with their embeddings in such a way that the sum of border lengths over all masks is minimized. It has been stated in [3, 4] that the problem is believed to be NP-hard because of the exponential number of possible placements (although we are not aware of an NP-hardness proof). For this reason, we focus on approximation algorithms for BMP in this paper.

Previous work. The BMP problem has attracted a lot of attention [3–5, 12, 15, 16] and most work is experimental in nature. As far as we know, no polynomial time approximation algorithm is known for BMP with non-trivial performance guarantee.

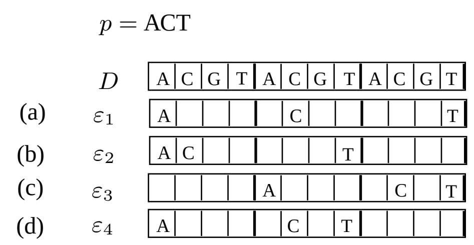

Fig. 2. Different embeddings of probe $p =$ ACT into deposition sequence $D = (\text{ACGT})^3$.

BMP was first formally defined by Hannenhalli et al. [12]. They focused on synchronous synthesis and the only concern becomes probe placement. Their algorithm computes an approximated travelling salesman path (TSP) in the complete graph with nodes representing probes and edge costs representing the Hamming distance between the probes. The TSP path is then placed on the microarray in a certain way called *threading*. Experiments shows that threading is effective in reducing border length. Since then, other algorithms [4, 15, 16] have been proposed to improve the experimental results.

Asynchronous probe embedding was introduced by Kahng et al. [15]. They studied a special case that the deposition sequence D is given and the embeddings of all but one probes are known. A polynomial time dynamic programming algorithm was proposed to compute the optimal embedding of this single probe whose neighbors are already embedded. This algorithm is used as the basis for several heuristics [3–5, 15, 16] that are shown experimentally to reduce unintended illumination in terms of border length.

On the other hand, there are few theoretical results. In [15], lower bounds on the total border length for synchronous and asynchronous BMP problem were given, which are based on Hamming distance, and Longest Common Subsequence (LCS), respectively. The asynchronous dynamic programming mentioned above computes the optimal embedding of a single probe in time $O(\ell|D|)$, where ℓ is the length of a probe and D is the deposition sequence. The algorithm can be extended to an exponential time algorithm to find the optimal embedding of all n probes in $O(2^n \ell^n |D|)$ time.

Our contribution. In this paper, we study approximation of BMP in asynchronous synthesis. This is the first result with proved performance guarantee. The main result is an $O(\sqrt{n}\log^2 n)$-approximation, where n is the number of probes in the microarray. This is based on an approximation algorithm for the variant when the placement of probes is given in advance (called P-BMP problem). We show that P-BMP is $O(\log^2 n)$-approximable. We further show that if the array is one-dimensional, P-BMP can be solved optimally in polynomial time and there is a constant approximation for BMP. On the other hand, we show that BMP can be defined as the maximum agreement problem (AMP) with a different objective called "agreement". Minimizing the border length is equivalent to maximizing the agreement. Yet we are able to devise $O(1)$-approximation algorithms for AMP regardless of whether the placement is given in advance or not.

Organization of the paper. In Section 2, we give some definitions and notations. In Sections 3 and 4, we present and analyze approximation algorithms for BMP and AMP, respectively. Finally we give a conclusion and discuss future work in Section 5.

2 Preliminaries

We are given a set of n length-ℓ probes $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, a $\sqrt{n} \times \sqrt{n}$ array (for simplicity, we assume that $\sqrt{n}$ is an integer). For any sequence p_i, we denote the t-th character of a sequence p_i by $p_i[t]$. The probes in $\mathcal{P}$ are to be placed on the $\sqrt{n} \times \sqrt{n}$ array. We represent this array by a grid graph $G = (V, E)$. Two grid vertices (x_1, y_1) and (x_2, y_2) are said to be *neighbor* if $|x_1 - x_2| + |y_1 - y_2| = 1$. For each vertex $v \in V$, we denote the set of neighbors of v by $\mathcal{N}(v)$.

Placement and embedding. A *placement* of the probes is a bijective function $\phi : \mathcal{P} \to V$ that maps each probe to a unique vertex in the grid G. An *embedding* of a set of probes $\mathcal{P}$ into a deposition sequence D is denoted by $\varepsilon = \{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n\}$. For $1 \leq i \leq n$, ε_i is a length-$|D|$ sequence such that (1) $\varepsilon_i[t]$ is either $D[t]$ or a space "$-$"; and (2) removing all spaces from ε_i gives p_i. The hamming distance between ε_i and ε_j measures the border length between p_i and p_j if they are neighbors in a certain placement. We define this quantity as the *conflict* between the embeddings of p_i and p_j, denoted by $\text{conf}_\varepsilon(p_i, p_j)$. Note that $\text{conf}_\varepsilon(p_i, p_j) \leq 2\ell$. We define the *share* between the embeddings of p_i and p_j as $2\ell - \text{conf}_\varepsilon(p_i, p_j)$, and denote it by $\text{share}_\varepsilon(p_i, p_j)$.

Border length and agreement. The *border length* of a placement ϕ and an embedding ε is defined as the sum of conflicts between the embeddings of probes that are neighbors in the placement ϕ in G:

$$\text{BL}(\phi, \varepsilon) = \frac{1}{2} \sum_{\substack{p_i, p_j : \\ \phi(p_j) \in \mathcal{N}(\phi(p_i))}} \text{conf}_\varepsilon(p_i, p_j) \, . \tag{1}$$

The objective of the BMP problem is to find a placement ϕ and an embedding ε, so that $\text{BL}(\phi, \varepsilon)$ is minimized. We denote the optimal placement and the corresponding optimal embedding by ϕ^* and ε^*, respectively. We further define the counter part of border length, the *agreement*, which is the sum of shares between the embeddings of probes that are neighbors in the placement ϕ in G:

$$\text{A}(\phi, \varepsilon) = \frac{1}{2} \sum_{\substack{p_i, p_j : \\ \phi(p_j) \in \mathcal{N}(\phi(p_i))}} \text{share}_\varepsilon(p_i, p_j) \tag{2}$$

The *Maximum Agreement Problem* (AMP) is to find a placement ϕ and an embedding ε, so that $\text{A}(\phi, \varepsilon)$ is maximized. Since $\text{A}(\phi, \varepsilon) = 4\ell(n - \sqrt{n}) - \text{BL}(\phi, \varepsilon)$, minimizing the border length $\text{BL}(\phi, \varepsilon)$ is equivalent to maximizing the agreement $\text{A}(\phi, \varepsilon)$.

Common subsequence and common supersequence. The border length is closely related to the common subsequence and common supersequence between neighboring sequences in the placement. Consider any two length-ℓ sequences p, q. We denote the longest common subsequence and shortest common supersequence of two sequences p and q by $LCS(p, q)$ and $SCS(p, q)$, respectively, and the corresponding length as $|LCS(p, q)|$ and $|SCS(p, q)|$, respectively. $SCS(p, q)$ can be obtained by finding $LCS(p, q)$ and inserting into p the characters in q that are not in $LCS(p, q)$ while preserving the order in q. Therefore, $|SCS(p, q)| = 2\ell - |LCS(p, q)|$. For any embedding ε, the maximum number of common deposition nucleotides between p and q is $|LCS(p, q)|$, in other words, $\text{conf}_\varepsilon(p, q) \geq 2(\ell - |LCS(p, q)|)$ and $\text{share}_\varepsilon(p, q) \leq 2|LCS(p, q)|$. We define the *LCS distance* to be $2(\ell - |LCS(p, q)|)$, denoted by $\text{dist}(p, q)$. In other words, $\text{dist}(p, q)$ is a lower bound of $\text{conf}_\varepsilon(p, q)$ for any embedding ε.

Multiple sequence alignment (MSA) and Weighted MSA (WMSA). As we will see in later sections, a variant of BMP problem, named P-BMP (BMP problem in which the placement is given), can be polynomial time reducible to WMSA. As a consequence, we can apply the approximation results on WMSA to P-BMP, which we can further use as a building block for the approximation for BMP. We first review the MSA and WMSA problems. MSA and WMSA have been studied extensively [2, 7, 11, 20]. Let Σ be the set of characters and $S = \{S_1, S_2, \ldots, S_k\}$ be a set of k sequences, with maximum length m, over Σ. An alignment of S is a matrix $S' = (S'_1, S'_2, \ldots, S'_k)$ such that $|S'_i| = m'$ and S'_i is formed by inserting spaces into S_i. For a given distance function $\delta(a, b)$ where $a, b \in \Sigma \cup \{-\}$, the *pair-wise score* of S'_i and S'_j is defined as $\sum_{1 \leq y \leq m'} \delta(S'_i[y], S'_j[y])$. Given a weight function $w(i, j)$ for the pair of sequences S_i and S_j, the *weighted sum-of-pair* (SP) score SP$(S', w) = \frac{1}{2} \sum_{1 \leq i,j \leq k} w(i, j) \sum_{1 \leq y \leq m'} \delta(S'_i[y], S'_j[y])$. The WMSA problem is to find an alignment S' such that SP(S', w) is minimized. WMSA has been proved to be NP-complete. An $O(\log^2 n)$-approximation algorithm [23] has been given via a reduction to the minimum routing cost tree problem (MRCT) [1].

Minimum routing cost tree problem (MRCT). In this problem, a graph with weighted edges is given. For a spanning tree of the graph, the *routing cost* between two vertices is the sum of weights of the edges on the unique path between the two vertices in the spanning tree. The *routing cost* of the spanning tree is defined as the sum of routing cost between every pair of two vertices. The MRCT problem is to find a spanning tree whose routing cost is minimum. The results in [1] state that there is a polynomial time reduction from WMSA to MRCT. Each sequence in the input of WMSA corresponds to a vertex in the input graph of MRCT. The edge weight between two vertices is set to be the weighted edit distance between the two corresponding sequences. The reduction result states that (1) there is a routing spanning tree T whose routing cost is at most $O(\log^2 n)$ times $\sum_{i,j} w(i, j)d(i, j)$, where $d(i, j)$ is the edit distance between the two sequences i and j; and (2) there is an alignment S' whose SP(S', w) is at most the routing cost of T. Note that $\sum_{i,j} w(i, j)d(i, j)$ is a lower bound on the weighted SP score. Therefore, the following lemma follows.

Lemma 1. [23] *There is an $O(\log^2 n)$-approximation algorithm for the WMSA problem, where n is the number of sequences to be aligned.*

3 The BMP problem

In this section, we study the BMP problem. We are to find a placement and an embedding for the given probe set. An $O(\sqrt{n} \log^2 n)$-approximation algorithm is given for BMP (Section 3.2), which is based on an approximability result for a variant of BMP, named P-BMP (Section 3.1). At the end of this section, we also discuss the case when the array is one-dimensional and we show that BMP admits better results in this case.

3.1 P-BMP: finding embedding when placement is given

In this section, we study the P-BMP problem, a variant of BMP with a placement given in advance. The concern becomes to find an embedding. We show that P-BMP is

$O(\log^2 n)$-approximable by giving a reduction to the weighted multiple sequence alignment problem (WMSA), for which there is an $O(\log^2 n)$-approximation algorithm [23].

Lemma 2. *There is a polynomial time reduction from P-BMP to WMSA.*

Proof. Let I be an instance of the P-BMP problem with a given placement ϕ. We construct an instance I' for WMSA such that there is a solution for I with border length X if and only if there is a solution for I' with a weighted SP score of X.

Construction of I'. We first show the construction of I'. The input sequence for WMSA is the same as the input probe set $\mathcal{P}$. The weight $w(i,j)$ is defined as follows:

$$w(i,j) = \begin{cases} 1 & \text{if } \phi(p_j) \in \mathcal{N}(\phi(p_i)), \\ 0 & \text{otherwise.} \end{cases}$$

The distance function $\delta(a,b)$, for $a,b \in \Sigma \cup \{-\}$, is defined as follows:

$$\delta(a,b) = \begin{cases} 0 & \text{if } a = b, \\ 1 & \text{if } a \neq b \text{ and } (a = \text{“}-\text{”} \text{ or } b = \text{“}-\text{”}), \\ \infty & \text{otherwise.} \end{cases}$$

Note that the edit distance of p_i and p_j in WMSA is the same as dist(p_i, p_j) in BMP.

Solution for I implies solution for I'. Suppose we have an embedding ε for I. Note that $\varepsilon = \{\varepsilon_1 \cdots \varepsilon_n\}$ is an alignment for $\mathcal{P}$ and the pairwise score of ε_i and ε_j equals conf$_\varepsilon(p_i, p_j)$. So, SP$(\mathcal{P}', w) = \frac{1}{2}\sum_{1\le i,j\le n} w(i,j) \sum_{1\le y\le |D|} \delta(\varepsilon_i[y], \varepsilon_j[y]) = \frac{1}{2}\sum_{1\le i,j\le n} w(i,j)\text{conf}_\varepsilon(p_i,p_j) = \frac{1}{2}\sum_{p_i,p_j:\phi(p_j)\in\mathcal{N}(\phi(p_i))} \text{conf}_\varepsilon(p_i,p_j) = \text{BL}(\phi,\varepsilon)$. The second last equality is due to the definition of $w(i,j)$, which is based on ϕ.

Solution for I' implies solution for I. On the other hand, suppose we have a solution for I', i.e., an alignment $\mathcal{P}' = (p'_1 \cdots p'_n)$ for $\mathcal{P}$ and $|p'_i| = m'$, for some m'. In the alignment $\mathcal{P}'$, each column contains the same character or “ $-$ ” because of the definition of the distance function $\delta(a,b)$. We denote the resulting matrix as $\varepsilon = (\varepsilon_1 \cdots \varepsilon_n)$. It can be seen that ε is an embedding for $\mathcal{P}$ and the hamming distance between ε_i and ε_j equals the pair-wise score of p'_i and p'_j. Then BL$(\phi,\varepsilon) = \frac{1}{2}\sum_{p_i,p_j:\phi(p_j)\in\mathcal{N}(\phi(p_i))} \text{conf}_\varepsilon(p_i,p_j) = \frac{1}{2}\sum_{p_i,p_j:\phi(p_j)\in\mathcal{N}(\phi(p_i))} \sum_{1\le y\le|D|} \delta(p'_i[y], p'_j[y]) = \frac{1}{2}\sum_{1\le i,j\le n} w(i,j)\sum_{1\le y\le|D|}\delta(p'_i[y], p'_j[y]) = \text{SP}(\mathcal{P}', w)$. Note that the second last equality holds for the same reason as above. Therefore, the lemma follows. □

Corollary 1. *The P-BMP problem is $O(\log^2 n)$-approximable.*

3.2 BMP: finding placement and embedding

In this section, we study the BMP problem in which we are to find both the placement as well as the embedding. We give an $O(\sqrt{n}\log^2 n)$-approximation, which makes use of the approximability result for P-BMP (Section 3.1). To make use of the result for P-BMP, we need a certain placement, the choice of which is guided by some travelling salesman path (TSP) on a particular graph (to be defined). Note that finding the minimum TSP is NP-hard, yet there is a polynomial time $O(1)$-approximation [6].

The algorithm PLACE&EMBED**.** The approximation algorithm PLACE&EMBED is shown in Algorithm 1. The graph G_c constructed in the algorithm is a weighted

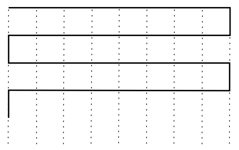

Fig. 3. Row-by-row threading of a TSP (solid edges) on a grid. Solid and dotted edges connect neighbors in the placement that are and are not, respectively, neighbors on the TSP.

complete graph with vertices representing $\mathcal{P}$ and edge weight representing dist() between the two vertices. A travelling salesman path (TSP) is obtained from G_c, which we *"thread"* on the grid G in a row-by-row fashion to form a placement [12]: the TSP is placed from left to right on the first row, right to left on the second, and then alternate in the same way in the remaining rows (see Figure 3 for an example). We then employ the approximation algorithm in Section 3.1. We denote the placement and embedding computed by PLACE&EMBED as $\tilde{\phi}$ and $\tilde{\varepsilon}$, respectively.

Algorithm 1 PLACE&EMBED: Approximation algorithm for BMP.

```
Input: Probe set $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ to be placed on a $\sqrt{n} \times \sqrt{n}$ array.
Output: A placement $\tilde{\phi}$ and an embedding $\tilde{\varepsilon}$ for $\mathcal{P}$.
1: Construct the weighted complete graph $G_c$.
2: Find an approximate TSP $\tilde{Q}$ for $G_c$ using algorithm in [6].
3: Thread $\tilde{Q}$ in a row-by-row fashion to obtain a placement $\tilde{\phi}$.
4: Run the approximation algorithm for P-BMP in Section 3.1 (i.e., by reducing the P-BMP
   instance to an WMSA instance) to obtain an embedding $\tilde{\varepsilon}$.
```

Theorem 1. *Algorithm* PLACE&EMBED *is an $O(\sqrt{n}\log^2 n)$-approximation for BMP.*

To analyze the performance of PLACE&EMBED, we need some notations. Recall that we define for any sequences p, q, dist$(p, q) = 2(\ell - |LCS(p, q)|)$. We overload the notation dist() for any subgraph of G_c. For any subgraph H of G_c, we define the LCS distance of H, denoted by dist(H), to be the sum of LCS distances of neighboring probes in H, i.e., dist$(H) = \frac{1}{2}\sum_{p,\, q\,:\, q \in \mathcal{N}(p) \text{ in } H}$ dist(p, q).

As mentioned before in Section 2, dist(p, q) is the minimum conflict between probes p and q. Yet the embeddings needed to achieve dist(p, q) may not be compatible with each other in a particular placement. For example, consider the placement ϕ in Figure 1, dist$(\phi) = 8$ since dist$(p, q) = 2$ for every neighboring pair p, q. Yet the minimum border length is 10 with CTAC as the deposition sequence, and embeddings $(- -\text{AC}, -\text{TA}-, \text{CT}- -, \text{C}-\text{A}-)$. We summarize this as follows.

Observation 1 *Given a placement ϕ, dist$(\phi) \leq$ BL(ϕ, ε), for any embedding ε.*

Observation 1 implies that for the optimal placement ϕ^* and embedding ε^*, dist$(\phi^*) \leq$ BL(ϕ^*, ε^*). To approximate BMP, it suffices to bound the border length by dist(ϕ^*). On the other hand, we make an observation about a graph H_1 and its subgraph H_2. The observation is true since any neighbors in H_2 are also neighbors in H_1.

Observation 2 *Consider any graph H_1 and a subgraph H_2 of it. dist$(H_2) \leq$ dist(H_1).*

Corollary 2. *Suppose Q^* is the optimal TSP for G_c. Then, we have dist$(Q^*) \leq$ dist(ϕ^*).*

Proof. ϕ^* can be viewed as threading a TSP Q in a row-by-row fashion. By Observation 2, dist$(Q) \leq$ dist(ϕ^*). As Q^* is the optimal TSP, dist$(Q^*) \leq$ dist$(Q) \leq$ dist(ϕ^*). □

It is known that TSP can be approximated by $3/2$ (Lemma 3). So, dist$(\tilde{Q}) \leq 3\,$dist$(Q^*)/2$.

Lemma 3. [6] *The travelling salesman problem admits a $3/2$-approximation if the weight satisfies the triangle inequality.*

Lemma 4. *(i) dist$(\tilde{\phi}) \leq 2\sqrt{n}\,$dist$(\tilde{Q})$; and (ii) BL$(\tilde{\phi}, \tilde{\varepsilon}) \leq O(\log^2 n)\,$dist$(\tilde{\phi})$.*

*Proof (*Sketch). (i) Suppose $\tilde{Q} = \{u_1, u_2, \ldots, u_n\}$. Note that the LCS distance dist() satisfies the triangular inequality, i.e., dist$(u_i, u_j) \leq \sum_{i \leq k < j}$ dist(u_k, u_{k+1}). Neighboring probes on $\tilde{Q}$ are also neighbors in $\tilde{\phi}$ but not vice versa. For any two probes u_i and u_j which are neighbors in $\tilde{\phi}$, we have $1 \leq |j-i| < 2\sqrt{n}$. When we sum up dist$(\tilde{\phi})$, dist(u_k, u_{k+1}), for any k, may be counted more than once, but no more than $2\sqrt{n}$ times. Therefore, dist$(\tilde{\phi}) \leq 2\sqrt{n}\,$dist$(\tilde{Q})$.

(ii) In Step 4 of PLACE&EMBED, we reduce the P-BMP instance with $\tilde{\phi}$ as the placement to an WMSA instance. Lemma 2 asserts that the border length of the embedding obtained is the same as the weighted SP score of the alignment. Furthermore, we have seen in Section 2 that approximation for WMSA can be found by the approximation for MRCT and the resulting routing tree has a routing cost, and thus, the weighted SP score, at most $O(\log^2 n)$ times the total weighted edit distance in WMSA. In the proof of Lemma 2, we note that the weighted edit distance of two sequences is the same as dist() of the two sequences. So, BL$(\tilde{\phi}, \tilde{\varepsilon}) \leq O(\log^2 n)\,$dist$(\tilde{\phi})$. □

Proof (Theorem 1). By Lemmas 4, 3, and Corollary 2, we have BL$(\tilde{\phi}, \tilde{\varepsilon}) \leq O(\sqrt{n}\log^2 n)$ dist$(\tilde{Q}) \leq O(\sqrt{n}\log^2 n)\,$dist$(Q^*) \leq O(\sqrt{n}\log^2 n)\,$dist$(\phi^*)$. Furthermore, Observation 1 holds for all placements, and hence for ϕ^*, in other words, dist$(\phi^*) \leq$ BL(ϕ^*, ε^*). Therefore, BL$(\tilde{\phi}, \tilde{\varepsilon}) \leq O(\sqrt{n}\log^2 n)\,BL(\phi^*, \varepsilon^*)$. □

3.3 One dimensional array

In this section, we study the special case on an 1D array. Intuitively, the problem is easier than the 2D case. We show that P-BMP on an 1D array can be solved optimally in polynomial time while BMP on an 1D array admits an $O(1)$-approximation.

P-BMP on 1D array. The algorithm EMBED1D shown in Algorithm 2 makes use of a procedure called EXTEND. EXTEND takes two sequences p and q, and a supersequence S of p as input and returns a supersequence of S and q. Let $c = |LCS(p, q)|$, $x_1, x_2, \ldots, x_c$ be the indices of S corresponding to p that belongs to $LCS(p, q)$, and $y_1, y_2, \ldots, y_c$ be the indices of q that belongs to $LCS(p, q)$. EXTEND then extends S by inserting characters in q but not in $LCS(p, q)$: characters between $q[y_{k-1}]$ and $q[y_k]$ are inserted right before $S[x_k]$ and characters beyond $q[y_c]$ are appended to the end of S. EXTEND keeps track of the indices of the new S that correspond to q (see Figure 4).

Theorem 2. EMBED1D *finds an optimal embedding for the P-BMP problem on 1D array in polynomial time.*

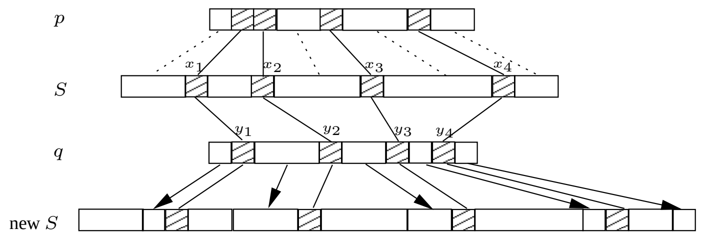

Fig. 4. An illustration of EXTEND. Shaded squares refer to characters in $LCS(p, q)$. Characters in q but not in $LCS(p, q)$ are inserted into S so that the order preserves as in q (see the arrows).

Algorithm 2 EMBED1D: Optimal embedding for P-BMP on 1D array.

```
Input: Probe set $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, placed on a 1D array in that order.
Output: An embedding $\varepsilon$ with minimum border length.
1: Set $D = p_1$.
2: For $i > 1$, call the procedure EXTEND with $p_{i-1}$, $p_i$ and $D$ as the input to obtain a new $D$.
3: For each $p_i$, set $\varepsilon_i$ such that $\varepsilon[y] = D[y]$ if $D[y]$ corresponds to a character in $p_i$ kept track
   by EXTEND, and $\varepsilon[y] =$ " $-$ " otherwise.
```

Proof. We first observe that D constructed in each iteration by EXTEND is a common supersequence of $p_1, \ldots, p_i$. This is clear from the way EXTEND finds $LCS(p_{i-1}, p_i)$ and inserts characters into D. It also implies that the number of nucleotides shared by p_{i-1} and p_i is maintained as $|LCS(p_{i-1}, p_i)|$, which is the maximum possible. Note that this property does not change by later steps. Hence, the border length of the final embedding is the minimum. As for time complexity, the bottleneck is finding the longest common subsequences of two sequences, which is known to take polynomial time [13]. This is done for $n-1$ times only. Therefore, EMBED1D also takes polynomial time. □

BMP on 1D array. Similar to the case on 2D array, we find a placement by finding an approximate TSP on the weighted complete graph G_c and then find an embedding by EMBED1D. This algorithm gives a $3/2$-approximation for BMP on 1D array.

Theorem 3. *There is a polynomial time algorithm for BMP on 1D array with approximation ratio* $3/2$.

4 The maximum agreement problem (AMP)

In this section, we study the counter part of BMP, which we called maximum agreement problem (AMP) (recall definition in Section 2). In contrast to BMP, AMP admits constant approximations, whether the placement is given in advance or not.

4.1 Approximation for P-AMP

We first study the P-AMP problem, a variant of AMP with a placement already given.

Algorithm AEMBED**.** The algorithm AEMBED (EMBED for Agreement) makes use of procedure EXTEND in Section 3.3. The order of probes to be considered is determined by a certain tree T with the bottom rightmost probe in G being the root. To

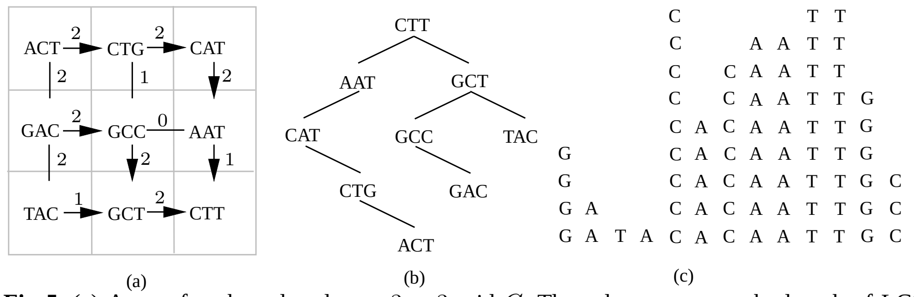

Fig. 5. (a) A set of probes placed on a 3×3 grid G. The values represent the length of LCS between the two neighboring probes. An arrow from p to q means $parent(p) = q$. (b) The tree constructed by AEMBED with root CTT. (c) How the deposition sequence D changes iteratively. The sequences are drawn in a way the characters align with the final D.

construct T, for each probe p, we assign a parent to the probe, denoted by $parent(p)$. We denote by $r(p)$ and $b(p)$ the right and bottom neighbors of probe p, respectively. The probes in the rightmost column and bottommost column has $r(p) = \text{NULL}$ and $b(p) = \text{NULL}$, respectively. We set $parent(p)$ to $r(p)$ or $b(p)$ depending on whether $|LCS(p, r(p))|$ or $|LCS(p, b(p))|$ is larger. Details of AEMBED is shown in Algorithm 3. The embedding found is denoted by $\hat{\varepsilon}$. Figure 5 shows an example.

Algorithm 3 AEMBED: Approximate algorithm for P-AMP.

```
Input: Probe set $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ placed on a $\sqrt{n} \times \sqrt{n}$ array according to a placement $\phi$.
Output: An embedding $\hat{\varepsilon}$ for $\mathcal{P}$.
1: Construct a tree $T$ by assigning parent to each probe $p$: if $|LCS(p, r(p))| \geq |LCS(p, b(p))|$
   set $parent(p) = r(p)$ else set $parent(p) = b(p)$.
2: Set $D$ to be the bottom rightmost probe in the grid $G$.
3: Traverse $T$ in a pre-order fashion: for each probe $p$ traversed, call the procedure EXTEND
   with $parent(p)$, $p$ and $D$ as input.
4: For each $p_i$, set $\hat{\varepsilon}_i$ such that $\hat{\varepsilon}[y] = D[y]$ if $D[y]$ corresponds to a character in $p_i$ kept track
   by EXTEND, and $\hat{\varepsilon}[y] =$ " $-$ " otherwise.
```

Analysis. To analyze the performance of AEMBED, we first observe that in the final embedding $\hat{\varepsilon}$, the number of nucleotides shared by a probe and its parent equals to the length of their LCS (by a similar argument as the proof of Theorem 2). We then bound the performance of AEMBED as follows.

Theorem 4. AEMBED *is a polynomial-time* 2*-approximation algorithm for P-AMP.*

Proof. For the given placement ϕ and the optimal embedding ε^*, the optimal agreement is: $\text{A}(\phi, \varepsilon^*) = \sum_{p \in \mathcal{P}} (\text{share}_{\varepsilon^*}(p, r(p)) + \text{share}_{\varepsilon^*}(p, b(p)))$. We assume $\text{share}_{\varepsilon^*}(p, q) = 0$ if $q = \text{NULL}$. As mentioned in Section 2, for any embedding, the share between the embeddings of probes p, q is at most $2|LCS(p, q)|$. Thus, $2|LCS(p, r(p))| \geq \text{share}_{\varepsilon^*}(p, r(p))$ and $2|LCS(p, b(p))| \geq \text{share}_{\varepsilon^*}(p, b(p))$. Note that $\text{share}_{\hat{\varepsilon}}(p, parent(p)) = 2 \max\{|LCS(p, r(p))|, |LCS(p, b(p))|\} \geq \frac{1}{2}(\text{share}_{\varepsilon^*}(p, r(p)) + \text{share}_{\varepsilon^*}(p, b(p)))$. Therefore, $\text{A}(\phi, \hat{\varepsilon}) = \sum_{p \in \mathcal{P}} \text{share}_{\hat{\varepsilon}}(p, parent(p)) \geq \frac{1}{2}\text{A}(\phi, \varepsilon^*)$. Finally, AEMBED runs in polynomial time as the bottleneck is finding LCS between two sequences. $\square$

4.2 Approximation for AMP

In this section, we study the general AMP problem to find both the placement and the embedding to maximize the agreement. We prove that the algorithm APLACE&EMBED as shown in Algorithm 4 has an asymptotic approximation ratio of 4.

Algorithm 4 APLACE&EMBED: Approximation algorithm for AMP.

```
Input: Probe set $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ to be placed on a $\sqrt{n} \times \sqrt{n}$ array.
Output: A placement $\check{\phi}$ and an embedding $\check{\varepsilon}$ for $\mathcal{P}$.
1: Partition $\mathcal{P}$ into four disjoint groups $\mathcal{A}$, $\mathcal{C}$, $\mathcal{G}$ and $\mathcal{T}$: a probe belongs to $\mathcal{A}$ if the number of A
   in the probe is the maximum over the number of other characters (similarly for $\mathcal{C}$, $\mathcal{G}$ and $\mathcal{T}$).
2: Thread the probes in group $\mathcal{A}$ on the array in a row-by-row fashion, followed by threading of
   probes in $\mathcal{C}$, $\mathcal{G}$, and $\mathcal{T}$ to form the placement $\check{\phi}$.
3: For probes in $\mathcal{A}$, align them such that the maximum number of A are aligned while different
   characters are not aligned. This forms a partial embedding $\check{\varepsilon}_a$ with deposition sequence $D_a$.
   Similarly, find $\check{\varepsilon}_c$, $\check{\varepsilon}_g$, $\check{\varepsilon}_t$ and $D_c$, $D_g$, $D_t$.
4: Combine $D_a$, $D_c$, $D_g$, and $D_t$ to form $D$ (append one after the other).
5: Extend the embeddings $\check{\varepsilon}_a$, $\check{\varepsilon}_c$, $\check{\varepsilon}_g$, $\check{\varepsilon}_t$ according to $D$ by inserting " $-$ " in the columns
   corresponding to other groups. The union of the extended embeddings is the resulting em-
   bedding $\check{\varepsilon}$.
```

Theorem 5. *The asymptotic approximation ratio of* APLACE&EMBED *is* 4.

Proof. Consider the optimal placement ϕ^* and embedding ε^*. For every pair of neighboring probes p, q, $\text{share}_{\varepsilon}(p, q) \leq 2\ell$. There are a total of $2(n - \sqrt{n})$ pairs of neighbors on the grid in total. So, the optimal agreement $\text{A}(\phi^*, \varepsilon^*) \leq 4\ell(n - \sqrt{n})$. On the other hand, consider $\check{\phi}$ and $\check{\varepsilon}$ returned by APLACE&EMBED. According to the way we partition the probes into group, for any two probes p, q in a group, the number of nucleotides that can be shared is at least $\ell/4$. Hence, $\text{share}_{\check{\varepsilon}}(p, q) \geq 2(\ell/4) = \ell/2$. As we seen above, there are altogether $2(n - \sqrt{n})$ pairs of neighbors in the grid. We may not share any nucleotide when the pair belongs to different groups. According to the way we thread the groups, there are at most $3\sqrt{n} + 3$ such pairs ($\sqrt{n}$ pairs of vertical neighbors between consecutive groups and 3 pairs of neighbors that are the last one in a group and the first one in the next group). As a result, we have at least $2n - 5\sqrt{n} - 3$ pairs each with $\text{share}_{\check{\varepsilon}}()$ at least $\ell/2$. Therefore, $\text{A}(\check{\phi}, \check{\varepsilon}) \geq \ell(n - 2.5\sqrt{n} - 1.5)$. Then $\text{A}(\check{\phi}, \check{\varepsilon})/\text{A}(\phi^*, \varepsilon^*)$ tends to 4 as $\text{A}(\phi^*, \varepsilon^*)$ tends to infinity. So, the asymptotic approximation ratio of APLACE&EMBED is 4. □

5 Concluding remarks

To summarize, we study the border minimization problem which is believed to be NP-hard with no known NP-hardness proof. An open question is to derive an NP-hardness proof. Another interesting open question is to improve the approximation ratio and/or derive inapproximability result. As mentioned before, there is an exponential time algorithm to compute the optimal BMP solution. Improving the exponential time algorithm could be useful in practice and is of theoretical interest.

References

1. Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *Proc. 37th FOCS*, pages 184–193, 1996.
2. P. Bonizzoni and G.D. Vedova. The complexity of multiple sequence alignment with SP-score that is a metric. *Theoretical Computer Science*, 259(1–2):63–79, 2001.
3. S.A. Carvalho Jr. and S. Rahmann. Improving the layout of oligonucleotide. microarrays: Pivot partitioning. In *Proc. 6th WABI*, pages 321–332, 2006.
4. S.A. Carvalho Jr. and S. Rahmann. Microarray layout as quadratic assignment problem. In *Proc. GCB*, pages 11–20, 2006.
5. S.A. Carvalho Jr. and S. Rahmann. Improving the design of genechip arrays by combining placement and embedding. In *Proc. 6th CSB*, pages 54–63, 2007.
6. N. Christofides. *Worst-case analysis of a new heuristic for the travelling salesman problem.* Technical Report 388, Carnegie-Mellon University, Pittsburgh, PA. *(ND33)*, 1976.
7. D.F. Feng and R.F. Doolittle. Approximation algorithms for multiple sequence alignment. *Theoretical Computer Science*, 182(1):233–244, 1987.
8. S. 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.
9. D. Gerhold, T. Rushmore, and C.T. Caskey. DNA chips: promising toys have become powerful tools. *Trends in Biochemical Sciences*, 24(5):168–173, 1999.
10. L. Gąsieniec, C.Y. Li, P. Sant, and P.W.H. Wong. Randomized probe selection algorithm for microarray design. *Journal of Theoretical Biology*, 248(3):512–521, 2007.
11. D. Gusfield. Efficient methods for multiple sequence alignment with guaranteed error bounds. *Bulletin of Mathematical Biology*, 55(1):141–154, 1993.
12. 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.
13. D.S. Hirschberg. A linear space algorithm for computing maximal common subsequences. *Communications of the ACM*, 18(6):341–343, 1975.
14. L. Kaderali and A. Schliep. Selecting signature oligonucleotides to identify organisms using DNA arrays. *Bioinformatics*, 18:1340–1349, 2002.
15. A.B. Kahng, I.I. Mandoiu, P.A. Pevzner, S. Reda, and A. Zelikovsky. Scalable heuristics for design of DNA probe arrays. *Journal of Computational Biology*, 11(2/3):429–447, 2004.
16. A.B. Kahng, I.I. Mandoiu, S. Reda, X. Xu, and A. Zelikovsky. Computer-aided optimization of DNA array design and manufacturing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(2):305–320, 2006.
17. S. Kasif, Z. Weng, A. Detri, R. Beigel, and C. DeLisi. A computational framework for optimal masking in the synthesis of oligonucleotide microarrays. *Nuc Acid Res*, 30(20):e106, 2002.
18. F. Li and G. Stormo. Selection of optimal DNA oligos for gene expression arrays. *Bioinformatics*, 17(11):1067–1076, 2001.
19. S. Rahmann. The shortest common supersequence problem in a microarray production setting. *Bioinformatics*, 19(suppl. 2):156–161, 2003.
20. K. Reinert, H.P. Lenhof, P. Mutzel, K. Mehlhorn, and J.D. Kececioglu. A branch-and-cut algorithm for multiple sequence alignment. In *Proc. 1st RECOMB*, pages 241–250, 1997.
21. D.K. Slonim, P. Tamayo, J.P. Mesirov, T.R. Golub, and E.S. Lander. Class prediction and discovery using gene expression data. In *Proc. 4th RECOMB*, pages 263–272, 2000.
22. W.K. Sung and W.H. Lee. Fast and accurate probe selection algorithm for large genomes. In *Proc. 2nd CSB*, pages 65–74, 2003.
23. B.Y. Wu, G. Lancia, V. Bafna, K.M. Chao, R. Ravi, and C.Y. Tang. A polynomial-time approximation scheme for minimum routing cost spanning trees. *SICOMP*, 29(3):761–778, 1999.