Hardness and Approximation of The Asynchronous Border Minimization Problem (Extended Abstract)

Alexandru Popa[1], Prudence W.H. Wong[2], and Fencol C.C. Yung[2]

[1] Department of Communications & Networking,
Aalto University School of Electrical Engineering, Aalto, Finland
`alexandru.popa@aalto.fi`
[2] Department of Computer Science, University of Liverpool, UK
`pwong@liverpool.ac.uk, ccyung@graduate.hku.hk`

Abstract. We study a combinatorial problem arising from the microarrays synthesis. The objective of the BMP is to place a set of sequences in the array and to find an embedding of these sequences into a common supersequence such that the sum of the "border length" is minimized. A variant of the problem, called P-BMP, is that the placement is given and the concern is simply to find the embedding.
Approximation algorithms have been proposed for the problem [21] but it is unknown whether the problem is NP-hard or not. In this paper, we give a comprehensive study of different variations of BMP by presenting NP-hardness proofs and improved approximation algorithms. We show that P-BMP, 1D-BMP, and BMP are all NP-hard. In contrast with the result in [21] that 1D-P-BMP is polynomial time solvable, the interesting implications include (i) the array dimension (1D or 2D) differentiates the complexity of P-BMP; (ii) for 1D array, whether placement is given differentiates the complexity of BMP; (iii) BMP is NP-hard regardless of the dimension of the array. Another contribution of the paper is improving the approximation for BMP from $O(n^{1/2} \log^2 n)$ to $O(n^{1/4} \log^2 n)$, where n is the total number of sequences.

1 Introduction

In this paper, we study an optimization problem called (asynchronous) border minimization problem (BMP), arising from a biological problem of microarray synthesis. We first describe the BMP (formal definition is given in Section 2) and then explain its relation with the biological problem. The input is a set of sequences $\mathcal{S} = \{s_1, s_2, \cdots, s_n\}$. We want to find a common supersequence D of $\mathcal{S}$ and an embedding ε_i for each sequence s_i into D, where ε_i is obtained by inserting spaces into s_i up to length $|D|$ with a constraint that the j-th position of ε_i is either the character at the j-th position of D or a space. The border length of s_i with respect to s_j is the number of non-space positions of ε_i that are different from ε_j. We then have to "place" the sequences into a $\sqrt{n} \times \sqrt{n}$ array such that the total border length is minimized (the total border length is the sum of the border length between every two sequences that are neighbors in the array). We study the complexity of BMP and give an approximation algorithm.

Motivation. DNA and peptide microarrays [7, 12] are important research tools used in gene discovery, multi-virus discovery, disease and cancer diagnosis. Apart from measuring the amount of gene expression [27], microarrays are an efficient tool for making a qualitative statement about the presence or absence of biological target sequences in a sample, e.g., peptide microarrays are used for detecting tumor biomarkers [6,23,29]. Microarray design raises a number of challenging combinatorial problems, such as probe selection [15, 22, 28], deposition sequence design [18, 24] and probe placement and synthesis [3–5, 14, 16, 17].

A microarray is a plastic or glass slide consisting of thousands of sequences called *probes*. The synthesis process [11] consists of two components: *probe placement* and *probe embedding*. In the probe placement the goal is to place each probe to a unique array cell. In the probe embedding we want to find a common supersequence of all sequences, called the *deposition sequence*, and a sequence of 2D arrays, called *masks*. The cells of a mask can be either opaque or transparent allowing the deposition of the character associated with the mask. For any cell, concatenating the characters for which the cell is transparent has to be the same as the probe in that cell of the microarray. See Figure 1(a) for an example. The embedding of a probe placed in a cell c is a sequence in which the ith character is "$-$" if cell c is opaque in the ith mask, or the ith character of the deposition sequence if transparent (Figure 1(b)).

Due to diffraction, the cells on the *border* between the masked and the unmasked regions are often subject to unintended illumination [11], and can compromise experimental results. As the microarray chip is expensive to synthesize, unintended illumination should 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(a), the border length of $\mathcal{M}_1, \mathcal{M}_3, \mathcal{M}_4$ is 2 and $\mathcal{M}_2$ is 4.

A synchronous variant of the problem was first studied [14] in which each deposition character can only be deposited to the i-th position of the probe sequences. Once the placement is fixed, the border length is unique and is proportional to the Hamming distance of neighboring probes. Thus the only problem is the placement of the probes. The synchronous version is NP-hard [19], $O(\sqrt{n})$-approximable [20] and there are also some experimental results [4,16,17].

Previous work on asynchronous BMP. The Asynchronous Border Minimization Problem (BMP) was introduced by Kahng et al. [16]. The problem appears to be difficult as they studied a special case in which the deposition sequence 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. This algorithm is used as the basis for several heuristics [3–5, 16, 17] that are shown experimentally to reduce unintended illumination. The dynamic programming [16] 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.

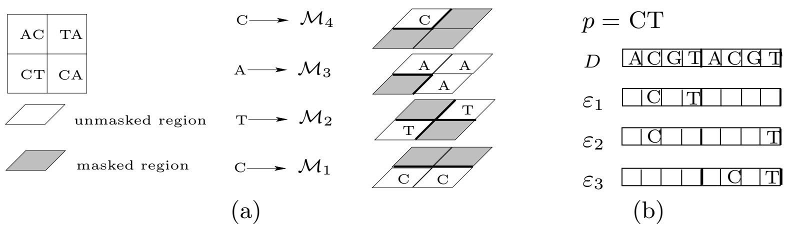


Fig. 1. (a) Asynchronous synthesis of a 2×2 microarray. The deposition sequence D = CTAC corresponds to four masks $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$, and $\mathcal{M}_4$. The corresponding embeddings are $--$AC, $-$TA$-$, CT$--$, and C$-$A$-$. The masked regions are shaded. The borders between the masked and unmasked regions are represented by bold lines. (b) Different embeddings of probe p = CT into deposition sequence $D = (\text{ACGT})^2$.

To find both the placement and the embedding, Li et al. [21] proposed the first randomized approximation algorithm with approximation ratio $O(\sqrt{n}\log^2 n)$, based on their $O(\log^2 n)$-approximation when placement is given. On a one-dimensional array, they improved the approximation ratio to 3/2. If the placement is given, the one-dimensional problem can be solved optimally in polynomial time. It is however unknown whether the general problem is NP-Hard or not. This leaves several open questions. Let us denote by P-BMP the problem with placement already given.

- So far, only approximation algorithms for BMP have been proposed. An open question is whether BMP is NP-Hard.
- P-BMP problem on 1D array can be solved optimally in polynomial time [21] while approximation algorithms and exponential time optimal algorithms have been proposed on 2D array. Two related questions are: Is 1D-BMP NP-Hard? Is P-BMP on 2D array NP-Hard?
- Is it possible to improve the approximation algorithms for BMP or P-BMP?

Our contributions. We give a comprehensive study of different variations of the asynchronous border minimization problem. We answer the above questions affirmatively by giving several NP-Hardness proofs and better approximation algorithms. Our contributions are listed below (see Table 1 also):

1. For P-BMP, we show that the Shortest Common Supersequence problem [25] can be reduced to P-BMP, implying that P-BMP is NP-Hard.
   This means that the dimension differentiates the complexity of P-BMP as we have seen in [21] that 1D-P-BMP is polynomial time solvable.
2. For 1D-BMP (placement not given), we give a reduction from the Hamming Traveling Salesman Problem [8], implying the NP-Hardness of 1D-BMP.
   This result implies that when the array is one dimensional, whether placement is given differentiates the complexity of BMP (as 1D-P-BMP is polynomial time solvable [21]).
3. We then show that 1D-BMP can be reduced to BMP, i.e. BMP is NP-Hard. This means that BMP is NP-Hard regardless of the dimension of the array.
4. We observe that the randomized approximation ratio for P-BMP can be improved from $O(\log^2 n)$ to $O(\log n)$. More interestingly, we improve the ratio for BMP from $O(n^{\frac{1}{2}}\log^2 n)$ to $O(n^{\frac{1}{4}}\log^2 n)$.

Setting	2D	1D
BMP	NP-Hard* $O(n^{1/4} \log n)$-approximate*	NP-Hard* $\frac{3}{2}$-approximate [21]
P-BMP	NP-Hard* $O(\log n)$-approximate*	polynomial time solvable [21]

Table 1. Results on BMP and P-BMP. Results in this paper are marked with an *.

We note that the reductions for (1) and (2) work for constant alphabet size. An interesting implication of (1) is that with placement already given, the synchronous problem [14] is trivial as the border length equals the Hamming distance. Nevertheless, the asynchronous problem is NP-hard. This indicates that the difficulty of the asynchronous problem is due to both the asynchronicity and the need to find a placement. Furthermore, our approximation algorithm also gives a $O(n^{\frac{1}{4}})$ approximation for the synchronous problem.

Technically speaking, the results for (1) and (4) are more challenging. The reduction for the NP-hardness proof of P-BMP proves that the Shortest Common Supersequence problem on binary alphabets can be solved with polynomially many calls to P-BMP. As for the approximation algorithm for BMP, we continue to use the observation in [21] that if we can find a good placement, then we can find a good embedding. Our improvement stems from a better placement, by defining a metric and using the randomized algorithm in [9] for "embedding" the metric into a tree distribution. This is a crucial step, since in this way we can control both the border length on the rows and the border length on the columns. An idea is to use an embedding in other metrics (e.g. Euclidean), but it is not at all clear how this can yield a better approximation algorithm.

Organization of the paper. Section 2 gives definitions and preliminaries. Sections 3 and 4 give the hardness results for P-BMP and BMP, respectively. Section 5 discusses approximation for BMP. We conclude in Section 6.

2 Preliminaries

We give the definition of the abstracted problem. We are given a set of n sequences $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ to be placed on a $\sqrt{n} \times \sqrt{n}$ array, where $\sqrt{n}$ is an integer. We denote the t-th character of a sequence s_i by $s_i[t]$. Two cells in the array (x_1, y_1) and (x_2, y_2) are said to be *neighbors* if $|x_1 - x_2| + |y_1 - y_2| = 1$, i.e., they are on the left/right/top/bottom of each other (diagonal cells are not neighbors). For each cell v, we denote the set of neighbors of v by $\mathcal{N}(v)$.

Deposition sequence, placement and embedding. A *placement* of $\mathcal{S}$ is a bijective function ϕ that maps each sequence to a unique cell in the array. A *deposition sequence* D is a common supersequence of the sequences in $\mathcal{S}$. An *embedding* of $\mathcal{S}$ into D is denoted by $\varepsilon = \{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n\}$, where 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 s_i. For example, there are four possible embeddings of the sequence ACT into the deposition sequence ACGTACGT: AC$-$T$----$, AC$-----$T, A$----$C$-$T, and $----$AC$-$T.

The border length of s_i with respect to s_j, denoted by $\text{border}_\varepsilon(s_i, s_j)$, is the number of non-space positions p's of ε_i that are different from ε_j, i.e., (i) $\varepsilon_i[p] \neq$ '$-$', and (ii) $\varepsilon_i[p] \neq \varepsilon_j[p]$. Condition (ii) means that $\varepsilon_j[p] =$ '$-$'. Note that $\text{border}_\varepsilon(s_i, s_j) \neq \text{border}_\varepsilon(s_j, s_i)$.

Border length and BMP. The *border length* of a placement ϕ and an embedding ε is defined as the sum of borders over all pairs of probe sequences

$$\text{BL}(\phi, \varepsilon) = \sum_{\substack{s_i, s_j : \\ \phi(s_j) \in \mathcal{N}(\phi(s_i))}} \text{border}_\varepsilon(s_i, s_j)\,. \tag{1}$$

The BMP is to find a placement ϕ and an embedding ε that minimizes $\text{BL}(\phi, \varepsilon)$. When the placement is given, we call the problem P-BMP. We also consider the BMP when the array is one dimensional, named 1D-BMP.

WMSA and MRCT. As shown in [21], P-BMP can be reduced to the *weighted multiple sequence alignment problem* (WMSA), which in turn can be reduced to the *minimum routing cost tree problem* (MRCT). In WMSA [2, 10, 13, 26], we are given k sequences $\mathcal{S} = \{s_1, s_2, \cdots, s_k\}$. An alignment is $\mathcal{S}' = \{s'_1, s'_2, \cdots, s'_k\}$ such that all s'_i have the same length and s'_i is formed by inserting spaces into s_i. The problem is to minimize the *weighted sum-of-pair score*. In MRCT [1], we are given a graph with weighted edges. In a spanning tree, 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 MRCT problem is to find a spanning tree with minimum routing cost, which is defined as the sum of routing cost between every pair of two vertices. The reduction results in [21] imply the following lemma.

Lemma 1 ([21]). *A c-approximation for* MRCT *implies a c-approximation for* P-BMP.

It is stated in [21] that Bartal's algorithm [1] finds a routing spanning tree by embedding a metric space into a distribution of trees with expected distortion $O(\log^2 n)$, implying MRCT is $O(\log^2 n)$-approximable [1]. Meanwhile, the ratio is improved to $O(\log n)$ by Fakcharoenphol, Rao and Talwar [9]. With Lemma 1, we have the following corollary. (Notice that we use the term embedding in two contexts, probe embedding refers to finding the deposition sequence while embedding a metric to trees is to obtain an approximation. This should be clear from the context and should not cause confusion.)

Corollary 1. *There is a randomized algorithm that is $O(\log n)$-approximate for the* P-BMP.

3 P-BMP: Finding embedding when placement is given

We give a reduction from the Shortest Common Supersequence (SCS) to the P-BMP.

Shortest Common Supersequence problem. Given n sequences of characters, a common supersequence is a sequence containing all n sequences as subsequences. The Shortest Common Supersequence problem is to find a minimum-length common supersequence.

(a)

\$	\$	\$	\$	\$	\$	\$
000	000	000	000	000	000	000
\$	010	\$	100	\$	00	\$
111	111	111	111	111	111	111
\$	\$	\$	\$	\$	\$	\$
\$	\$	\$	\$	\$	\$	\$
\$	\$	\$	\$	\$	\$	\$

(b)

\$	\$	\$	\$	\$	\$	\$
0	0	0	0	0	0	0
\$	010	\$	100	\$	00	\$
1	1	1	1	1	1	1
\$	\$	\$	\$	\$	\$	\$
\$	\$	\$	\$	\$	\$	\$
\$	\$	\$	\$	\$	\$	\$

Table 2. s_1 = "010", s_2 = "100", s_3 = "00". (a) The supersequence D = "010011" is an optimal deposition sequence for $I(3,3)$. Ignoring the mask for the dummy strings "\$", the optimal border length equals $2(p^* + q^*)(2k+1) + 2L = 100$, where $p^* = q^* = k = 3$ and $L = 8$. (b) The shortest common supersequence D = "0100" is an optimal deposition for $I(1,1)$. The optimal border length equals to $(2 \times 7 + 2 \times 7 + 2 \times 3 + 2 \times 2) + 2 \times 8 = 54$ (the first four terms refer to border length with top and bottom boundaries and the last term with left and right). On the other hand, $2(p^* + q^*)(2k+1) + 2L = 44 < 54$, where $p^* = q^* = 1$, $k = 3$ and $L = 8$.

The reduction is from the SCS problem over binary alphabets, which is known to be NP-Hard [25]. Suppose that the binary alphabet is $\{0,1\}$. Consider an instance of the SCS problem with a set $\mathcal{S}$ of k binary strings $s_1, \cdots, s_k$. Let ℓ_i be the length of s_i, $\ell = \max_{1 \le i \le k} \ell_i$ and $L = \sum_{1 \le i \le k} \ell_i$. For any $1 \le p, q \le \ell$, we define an instance of P-BMP, denoted by $I(p,q)$. As we show later, a shortest common supersequence can be found by computing the optimal solutions for a polynomial number of instances $I(p,q)$.

The input $\boldsymbol{I(p,q)}$. We construct a $(2k+1) \times (2k+1)$ array. The probe sequences are over the alphabet $\{0,1,\$\}$, where \$ is a character different from 0 or 1. (Tables 2 (a) and (b) show examples of $I(3,3)$ and $I(1,1)$, respectively.)

- Except for row 2-4, each cell of rows 1, 5, 6, 7, 8, $\cdots$, $(2k+1)$ of the array contains the string "\$". We call these rows *dummy-rows.*
- All the cells of row 2 contain the same string "0^p". We call this row *all-0-row.*
- All the cells of row 4 contain the same string "1^q". We call this row *all-1-row.*
- Row 3 contains s_1, s_2, $\cdots$, s_k in alternate cells, and the rest of the cells contain the string "\$", precisely, row 3 contains "\$", s_1, "\$", s_2, "\$", $\cdots$, "\$", s_k, "\$". We call this row *seq-row.*

Common supersequence and deposition sequence. Given an instance $I(p,q)$, we need at least one mask for the dummy strings "\$", and the best is to use precisely one mask, say $\mathcal{M}_\$$ for all these strings. We compute the border length induced by $\mathcal{M}_\$$. Row 1 (dummy-row) incurs a border length of $2k+1$ on the bottom boundary with all-0-row, and row 5 (dummy-row) incurs $2k+1$ on the top boundary with all-1-row. For seq-row, the border length on top boundary with all-0-row is $k+1$, on bottom boundary with all-1-row is also $k+1$, and within the seq-row on the left and right boundaries is $2k$. Therefore, the border length of $\mathcal{M}_\$$ is $4(2k+1)$. The total border length for $I(p,q)$ equals to the border length of $\mathcal{M}_\$$ plus that of the remaining deposition sequence, which in turn is related to a common supersequence of the sequences in $\mathcal{S}$. Since the border length of $\mathcal{M}_\$$ is present in all embeddings, we ignore this quantity when we discuss the border length for $I(p,q)$. Lemma 2 states a relationship between a common supersequence and an embedding of the probe sequences. Table 2(a) gives an example.

Lemma 2. *If D is a common supersequence of the sequences in $\mathcal{S}$ and the number of* 0*'s and* 1*'s in D is p^* and q^*, respectively, then D is an optimal deposition sequence for $I(p^*, q^*)$ and the resulting optimal embedding has a border length of $2(p^* + q^*)(2k + 1) + 2L$.*

Proof (Sketch). First of all, it is not difficult to observe that D is a deposition sequence for $I(p^*, q^*)$ since it is a common supersequence and has the same number of 0's and 1's in the all-0-row and all-1-row of the array in $I(p^*, q^*)$, respectively. Notice that p^* is at least the number of 0's in each of s_i and similarly q^* is at least the number of 1's. By examining each row, one can show that the total border length equals $2(p^* + q^*)(2k + 1) + 2L$.

We then argue that this is the minimum border length for $I(p^*, q^*)$. In any deposition sequence, the number of 0's and 1's is at least p^* and q^*, respectively. Therefore, the all-0-row and the cells with '0' on the seq-row together incur a border length of at least $2p^*(2k + 1)$, and similarly, the all-1-row and the cells with '1' on the seq-row incur at least $2q^*(2k + 1)$. The cell on the seq-row incurs $2L$ with the left and right boundaries. Therefore, no matter how we deposit characters, the total border length is at least $2(p^* + q^*)(2k + 1) + 2L$. □

Lemma 2 implies that if $p + q$ is large enough, we have a formula for the optimal border length of the instance $I(p, q)$ in terms of p, q, and L. The following lemma considers the situation when $p + q$ is small. Table 2(b) gives an example. Due to space limit, we leave the proof in the full paper.

Lemma 3. *If D is a shortest common supersequence of the sequences in $\mathcal{S}$ and the number of* 0*'s and* 1*'s in D is p^* and q^*, respectively, then for any p_1, q_1 such that $p_1 + q_1 < p^* + q^*$, the optimal embedding for $I(p_1, q_1)$ has a border length greater than $2(p_1 + q_1)(2k + 1) + 2L$.*

Using Lemmas 2 and 3, we can find the optimal solution for SCS from optimal solutions for P-BMP as follows. For all pairs of $1 \leq p \leq \ell$ and $1 \leq q \leq \ell$, we find the optimal solution to $I(p, q)$. If the border length of the optimal solution equals to $2(p + q)(2k + 1) + 2L$, then there is a common supersequence of length $p + q$. Among all such pairs of p and q, those with the minimum $p + q$ correspond to shortest common supersequences. Notice that there polynomially many, precisely ℓ^2, pairs of p and q to be checked. We then have the following theorem.

Theorem 1. *The* P-BMP *is NP-Hard.*

4 BMP: Finding placement and embedding

4.1 1D-BMP: BMP on a 1D array

The Hamming TSP. The input consists of a set of strings $s_1, s_2, \ldots s_n$ over the alphabet $\{0, 1\}$. We denote by $ham(s_1, s_2)$ the Hamming distance between s_1

and s_2 (i.e. the number of positions on which s_1 and s_2 differ). The goal is to find a permutation (we also call this permutation a *tour*) $\pi : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ such that the sum $\sum_{i=1}^{n-1} ham(s_{\pi(i)}, s_{\pi(i+1)})$ is minimized. Ernvall et al. prove that the Hamming TSP problem is NP-Hard [8].

Reduction. Consider a Hamming TSP instance with n binary strings $s_1, \ldots s_n$. We construct an instance of 1D-BMP with n sequences to be placed on an array of size $1 \times n$. We now define the alphabet Σ and the probe sequences S.

1. Alphabet: $\Sigma = \{0, 1, \$\}$, where \$ is a special character serving as a delimiter.
2. Probe sequences: for each string $s = x_1 x_2 \ldots x_k$ in the Hamming TSP instance, where $x_i \in \{0, 1\}$, we construct the probe sequence $s' = x_1 \$^{2n\ell} x_2 \$^{2n\ell} \ldots \$^{2n\ell} x_k$, where ℓ is the length of the longest string s_i.

Theorem 2. *The* 1D-BMP *is NP-Hard if the size of the alphabet is at least* 3.

4.2 BMP on 2D array

In this section, we reduce the BMP on an $1 \times n$ array to BMP on an $n \times n$ array. This implies that the BMP is NP-Hard. Consider an instance I_1 for the 1D-BMP where there are n sequences $s_1, s_2, \cdots, s_n$ over an alphabet Σ, and the length of s_i is ℓ_i. Let $\ell = \max_{1 \le i \le n} \ell_i$ and let $k > \ell$ be a large integer to be determined later. We construct an instance I_2 for BMP which contains two types of sequences on the alphabet $\Sigma' = \Sigma \cup \{x_1, x_2, \cdots, x_n\} \cup \{\$\}$, namely, the given sequence and the dummy sequence.

- Dummy sequences: we create $n^2 - n$ dummy sequences each containing one character \$.
- Given sequences: for each s_i, we create a length k sequence $x_i^{k-\ell_i} \cdot s_i$.

We claim that the best way is to put the given sequences on the top row. The optimal solution for I_1 would give an optimal solution for I_2 and vice versa. Due to space limit, we leave the proof to the full paper.

Theorem 3. *The (two-dimensional)* BMP *is NP-Hard.*

5 A $O(n^{\frac{1}{4}} \log^2 n)$ approximation algorithm for the BMP

In this section we present a $O(n^{\frac{1}{4}} \log^2 n)$ randomized approximation algorithm for the BMP, improving the previous $O(n^{\frac{1}{2}} \log^2 n)$ approximation. As mentioned in Corollary 1 (Section 2), there is a $O(\log n)$ approximation for the P-BMP in which the placement of the sequences is given. Therefore, to obtain an approximation for the BMP, it suffices to find a "good" placement of the sequences.

The intuitive ideas of our approximation algorithm are as follows. We first define a distance function $d(s_i, s_j)$ for any pair of sequences s_i and s_j, and this gives a lower bound on border(s_i, s_j) + border(s_j, s_i) (this is similar to [21]). A placement can be viewed as a permutation π. We define a function $p(\pi)$ based on $d(s_i, s_j)$ and show that $p(\pi)$ is a lower bound on the border length of any embedding (including the optimal one) for the permutation π. Therefore, if we can find an embedding such that the border length is at most a certain factor of $p(\pi)$, then we have an approximation for BMP. We then observe that it is difficult to find in polynomial time a permutation optimizing the value $p(\pi)$ on the general metric and turn to embedding the metric into a tree (distribution) such that (in expectation) the distance on the tree $d_T(s_i, s_j)$ satisfies the property $d(s_i, s_j) \leq d_T(s_i, s_j) \leq O(\log n) d(s_i, s_j)$. Finally, we show that using an Euler tour on the embedded tree as a permutation to place the sequences on the array gives us a $O(n^{\frac{1}{4}})$ approximation on $p_T(\pi)$, which is the counter part of $p(\pi)$ with $d(\cdot)$ replaced by $d_T(\cdot)$. Combining all the arguments above, we obtain a $O(n^{\frac{1}{4}} \log^2 n)$ approximation for BMP. Details are as follows.

The function $\boldsymbol{p(\pi)}$. We first derive a lower bound on the border length between two sequences s_i and s_j of length ℓ_i and ℓ_j, respectively. Let $LCS(s_i, s_j)$ denote the longest common subsequence between s_i and s_j and $|LCS(s_i, s_j)|$ denote its length. For any embedding ε, the maximum number of common deposition nucleotides between s_i and s_j is $|LCS(s_i, s_j)|$. Then, the border length is at least $\ell_i + \ell_j - 2|LCS(s_i, s_j)|$ and we denote this quantity as $d(s_i, s_j)$. Therefore, the sum of distances $d(s_i, s_j)$ is a lower bound on the optimal border length of a given placement. We observe that this distance $d(\cdot)$ is a metric.

We further derive a lower bound on the overall border length of a placement. A placement can be viewed as a permutation $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that the sequences $\pi(1), \ldots, \pi(\sqrt{n})$ are placed on the first row of the array in this order, $\pi(\sqrt{n}+1), \ldots, \pi(2\sqrt{n})$ on the second row and so on. Then any embedding for a placement π has a border length at least $p(\pi)$, which is defined as:

$$\begin{aligned} p(\pi) = &\sum_{i=1}^{n-1} d(\pi(i), \pi(i+1)) - \sum_{i=1}^{\sqrt{n}-1} d(\pi(i\sqrt{n}), \pi(i\sqrt{n}+1)) \\ &+ \sum_{i=1}^{\sqrt{n}} \sum_{j=1}^{\sqrt{n}-1} d(\pi(i+(j-1)\sqrt{n}), \pi(i+j\sqrt{n})) \ . \end{aligned}$$

We name the problem to minimize this "proxy" value $p(\pi)$ the PROXY problem. Note that the border length for a placement π can be much larger than $p(\pi)$ as the embeddings needed to achieve $d(s_i, s_j)$ for all s_i and s_j may not be compatible with each other. Yet, using a similar argument as in [21], one can show that given a placement π, the P-BMP approximation algorithm returns an embedding with the border length less than $O(\log n) p(\pi)$ (c.f. Corollary 1). Therefore, if we can place the sequences into the array such that the sum of the distances between any neighbors is within a factor c of $p(\pi)$, then we can apply the $O(\log n)$ approximation algorithm for the P-BMP and obtain a $O(c \log n)$ approximation for the BMP. We summarize this in the following proposition.

Proposition 1. *A c-approximation algorithm for the* PROXY *problem implies a $O(c \log n)$ approximation algorithm for the* BMP.

Tree embedding and Euler tour to approximate $\boldsymbol{p(\pi)}$. We optimize the value $p(\pi)$ by embedding the metric into a distribution of trees, with $O(\log n)$ distortion using the algorithm of Fakcharoenphol, Rao and Talwar [9]. This randomized embedding algorithm takes the input sequences as tree vertices and returns a tree with a metric $d_T(\cdot)$ defined by a tree such that in expectation $d(s_i, s_j) \leq d_T(s_i, s_j) \leq O(\log n) d(s_i, s_j)$. The distance $d_T(s_i, s_j)$ on the tree is the sum of distances along the unique path between s_i and s_j. Notice that the resulting tree may have vertices in addition to the n input sequences. Using the metric $d_T(s_i, s_j)$, we define a counter part of $p_T(\pi)$ by replacing $d(s_i, s_j)$ with $d_T(s_i, s_j)$. Then a c-approximation to p_T leads to a $O(c \log n)$ approximation to p. Together with Proposition 1, we have the following proposition.

Proposition 2. *If we can approximate the* Proxy *problem on a tree (i.e., approximate p_T) within a factor of c, then we have a $O(c \log^2 n)$ approximation to the* BMP*.*

We now present how to approximate p_T. Our approximation algorithm for the Proxy problem on trees is very simple: we consider the ordering of the vertices given by an Euler tour of the tree (we ignore the additional vertices which do not correspond to input sequences). We then prove that this is a $O(n^{\frac{1}{4}})$ approximation algorithm for p_T. Then, by Proposition 2 we are guaranteed to have a $O(n^{\frac{1}{4}} \log^2 n)$ approximation algorithm for the BMP problem (see Algorithm 1).

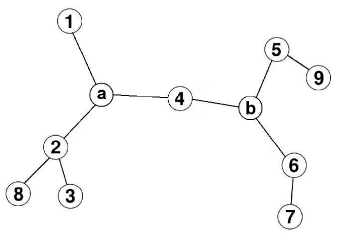
Algorithm 1 The $O(n^{\frac{1}{4}} \log^2 n)$ approximation algorithm for the BMP

```
1: Input: The strings $s_1, s_2, \ldots, s_n$.
2: Define $d(s_i, s_j) = \ell_i + \ell_j - 2|LCS(s_i, s_j)|$
3: Embed the metric given by this distance and the set of input vertices into a tree
   $T$ using the algorithm from [9].
4: Let $\pi : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ be the ordering of the sequences according to
   an Euler tour of the tree $T$ from which the additional vertices have been removed.
5: Place the sequences in the array according to $\pi$: $\pi(1), \ldots, \pi(\sqrt{n})$ are placed on the
   first row in this order, $\pi(\sqrt{n}+1), \ldots, \pi(2\sqrt{n})$ on the second row and so on. (See
   Figure 2.)
6: Apply the P-BMP approximation algorithm in [21].
7: Output: The placement of the sequences on the array based on the Euler tour and
   the embeddings given by the P-BMP approximation algorithm.
```

Theorem 4. *The placement of the sequences in the $\sqrt{n} \times \sqrt{n}$ array in the order given by the Euler tour gives a $O(n^{\frac{1}{4}} \log^2 n)$ approximation to the* BMP *problem.*

6 Concluding remarks

We give a comprehensive study of different variations of the *Border Minimization Problem* and present NP-hardness proofs and approximation algorithms. Contrasting with the previous result in [21] that the 1D-P-BMP is polynomial time solvable, our hardness results show that (i) the dimension differentiates the complexity of the P-BMP; (ii) for 1D array, whether placement is given differentiates the complexity of the BMP; (iii) the BMP is NP-Hard regardless of the dimension of the array.



(a)

s_1	s_2	s_8
s_3	s_4	s_5
s_9	s_6	s_7

(b)

Fig. 2. (a) Suppose the embedding in [9] returns such a tree for 9 sequences. The vertices that are mapped to input strings are labeled with numbers and the additional vertices introduced by the embedding algorithm are labeled with letters. (b) The placement of these sequences on the array according to an Euler tour of the tree, e.g., $1, a, 2, 8, 3, 4, b, 5, 9, 6, 7$. After removing the additional vertices a and b the ordering of n the vertices corresponding to input sequences is: $1, 2, 8, 3, 4, 5, 9, 6, 7$.

Moreover, our techniques can be used to improve the approximation ratio for the synchronous case from $O(n^{1/2})$ to $O(n^{1/4})$ using the placement method given by Algorithm 1 (where the metric is defined by the Hamming distance between the probes). Once a placement is found, the synchronous embedding can be computed exactly in polynomial time.

Note that the NP-hardness reduction for the P-BMP works for alphabets of size 3. In contrast, the hardness result for the BMP uses non-constant alphabets. An open problem is to prove that the BMP is hard also on constant alphabets (intuitively the BMP is harder than the P-BMP) but this does not seem to be easy.

Another natural open question is to further improve approximation algorithms for the BMP and the P-BMP and/or to derive inapproximability results.

References

1. Y. Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *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. *TCS*, 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. M. Chatterjee, S. Mohapatra, A. Ionan, G. Bawa, R. Ali-Fehmi, X. Wang, J. Nowak, B. Ye, F. A. Nahhas, K. Lu, S. S. Witkin, D. Fishman, A. Munkarah, R. Morris, N. K. Levin, N. N. Shirley, G. Tromp, J. Abrams, S. Draghici1, and M. A. Tainsky1. Diagnostic markers of ovarian cancer by high-throughput antigen cloning and detection on arrays. *Cancer Research*, 66(2):1181–1190, 2006.
7. M. Cretich and M. Chiari. *Peptide Microarrays Methods and Protocols*, volume 570 of *Methods in Molecular Biology.* Human Press, 2009.

8. J. Ernvall, J. Katajainen, and M. Penttonen. NP-completeness of the hamming salesman problem. *BIT Numerical Mathematics*, 25:289–292, 1985.
9. J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *STOC*, pages 448–455, 2003.
10. D. F. Feng and R. F. Doolittle. Approximation algorithms for multiple sequence alignment. *TCS*, 182(1):233–244, 1987.
11. S. Fodor, J. Read, M. Pirrung, L. Stryer, A. Lu, and D. Solas. Light-directed, spatially addressable parallel chemical synthesis. *Science*, 251(4995):767–773, 1991.
12. 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.
13. D. Gusfield. Efficient methods for multiple sequence alignment with guaranteed error bounds. *Bulletin of Mathematical Biology*, 55(1):141–154, 1993.
14. S. Hannenhalli, E. Hubell, R. Lipshutz, and P. A. Pevzner. Combinatorial algorithms for design of DNA arrays. *Adv in Biochem Eng/Biotech*, 77:1–19, 2002.
15. L. Kaderali and A. Schliep. Selecting signature oligonucleotides to identify organisms using DNA arrays. *Bioinformatics*, 18:1340–1349, 2002.
16. A. B. Kahng, I. I. Mandoiu, P. A. Pevzner, S. Reda, and A. Zelikovsky. Scalable heuristics for design of DNA probe arrays. *JCB*, 11(2/3):429–447, 2004. Preliminary versions in WABI 2002 and RECOMB 2003.
17. 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.
18. S. Kasif, Z. Weng, A. Detri, R. Beigel, and C. DeLisi. A computational framework for optimal masking in the synthesis of oligonucleotide microarrays. *Nucleic Acids Research*, 30(20):e106, 2002.
19. V. Kundeti and S. Rajasekaran. On the hardness of the border length minimization problem. In *BIBE*, pages 248–253, 2009.
20. V. Kundeti, S. Rajasekaran, and H. Dinh. On the border length minimization problem (BLMP) on a square array. *CoRR*, abs/1003.2839, 2010.
21. C. Li, P. Wong, F. Yung, and Q. Xin. Approximating border length for DNA microarray synthesis. In *Proc. 5th TAMC*, pages 410–422, 2008.
22. F. Li and G. Stormo. Selection of optimal DNA oligos for gene expression arrays. *Bioinformatics*, 17(11):1067–1076, 2001.
23. C. Melle, G. Ernst, B. Schimmel, A. Bleul, S. Koscielny, A. Wiesner, R. Bogumil, U. Möller, D. Osterloh, K.-J. Halbhuber, and F. von Eggeling. A technical triade for proteomic identification and characterization of cancer biomarkers. *Cancer Research*, 64(12):4099–4104, 2004.
24. S. Rahmann. The shortest common supersequence problem in a microarray production setting. *Bioinformatics*, 19(suppl. 2):156–161, 2003.
25. K.-J. Räihä. The shortest common supersequence problem over binary alphabet is NP-complete. *Theoretical Computer Science*, 16(2):187–198, 1981.
26. K. Reinert, H. P. Lenhof, P. Mutzel, K. Mehlhorn, and J. D. Kececioglu. A branch-and-cut algorithm for multiple sequence alignment. In *RECOMB*, 241–250, 1997.
27. D. K. Slonim, P. Tamayo, J. P. Mesirov, T. R. Golub, and E. S. Lander. Class prediction and discovery using gene expression data. In *RECOMB*, 263–272, 2000.
28. W. K. Sung and W. H. Lee. Fast and accurate probe selection algorithm for large genomes. In *Proc. 2nd CSB*, pages 65–74, 2003.
29. J. Welsh, L. Sapinoso, S. Kern, D. Brown, T. Liu, A. Bauskin, R. Ward, N. Hawkins, D. Quinn, P. Russell, R. Sutherland, S. Breit, C. Moskaluk, H. Frierson, Jr., and G. Hampton. Large-scale delineation of secreted protein biomarkers overexpressed in cancer tissue and serum. *PNAS*, 100(6):3410–3415, 2003.