Optimal Probing Patterns for Sequencing By Hybridization

Dekel Tsur*

Abstract

Sequencing by Hybridization (SBH) is a method for reconstructing a DNA sequence based on its k-mer content. This content, called the *spectrum* of the sequence, can be obtained from hybridization with a universal DNA chip. The main shortcoming of SBH is that it reliably reconstructs only sequences of length at most square root of the size of the chip. Frieze et al. [9] showed that by using gapped probes, SBH can reconstruct sequences with length that is linear in the size of the chip. In this work we investigate the optimal placement of the gaps in the probes, and give an algorithm for finding nearly optimal gap placement. Using our algorithm, we obtain a chip design which is more efficient than the chip of Frieze et al.

1 Introduction

Sequencing by Hybridization (SBH) [3, 16] is a method for sequencing DNA molecules. In this method, the target sequence is hybridized to a universal chip containing all 4^k sequences of length k. For each k-long sequence (or *probe*) in the chip, if its reverse complement appears in the target, then the two sequences will bind (or hybridize), and this hybridization can be detected. Thus the hybridization experiment gives the set of all k-long substrings of the target sequence. This set is called the *spectrum* of the target.

Currently, SBH is not considered competitive in comparison with standard gel-based sequencing technologies. The main shortcoming of SBH is that several sequences can have the same spectrum. Thus, if, for example, we wish to reconstruct at least 0.9 fraction of the sequences of length n, then n must be less than roughly 2^k [2, 8, 20, 23]. Several methods for overcoming this limitation of SBH were proposed: interactive protocols [10, 17,25,28], using location information [1,4,5,7,11,23], using a known homologous string [18, 19, 27], and using restriction enzymes [24, 26].

Another method for enhancing SBH was proposed by Pevzner et al. [20]. They suggested using *gaps* (or *universal bases*) in the probes that can match to any of the four bases. For example, the probe A$\phi\phi$G matches the sequences TAAC, TACC, TAGC, etc. A gapped probe can be implemented using a uniform mixture of the sequences that match the probe with length the same as the probe. Frieze et al. [9] showed that for every k, there is a chip with 4^k probes that can reconstruct sequences of length $\Theta(4^k)$. This result is

*Department of Computer Science, Ben-Gurion University of the Negev.

optimal up to constants. For a fixed k, the chip of Frieze et al. consists of all probes of the form $X^{\lceil k/2 \rceil} \left(\phi^{\lceil k/2 \rceil - 1} X\right)^{\lfloor k/2 \rfloor}$, where the X symbols represent definite bases (namely, each X symbol is replaced by one of the four bases). Preparata and Upfal [22] considered the same probing pattern of Frieze et al., and gave an improved algorithm for reconstructing the sequence from its spectrum. This algorithm allows reconstructing longer sequences than the algorithm of Frieze et al. An even more efficient algorithm was given in [12]. Heath et al. [13] showed that chips containing probes of the form $X^{\lceil k/2 \rceil} \left(\phi^{\lceil k/2 \rceil - 1} X\right)^{\lfloor k/2 \rfloor}$ and of the form $\left(X\phi^{\lceil k/2 \rceil - 1}\right)^{\lfloor k/2 \rfloor} X^{\lceil k/2 \rceil}$ are more efficient than the Frieze et al. chips. A *semi-degenerate base* is a base that matches either A/T or G/C. Probes containing semi-degenerate bases were studied in [20, 21].

In this paper we study the problem of designing optimal probing patterns. More precisely, for a given length and number of definite bases, the goal is to find the probing pattern that allows reconstructing longest sequences (it is desired to use probes with small length and small number of definite bases, since each of these parameters affects the number of molecules on the chip). We give an algorithm for this problem, and show that the probing patterns obtained by this algorithm are about 3 times more efficient than the probing patterns of Frieze et al. For simplicity, we shall restrict our study to a single probing pattern consisting of definite bases and gaps. However, our method can also be used for multiple probing patterns, and for probes containing semi-degenerate bases. We note that our work is somewhat similar to the research on seed design for similarity search (see, for example, [6, 14, 15]).

Due to lack of space, some details are omitted from this extended abstract.

2 Finding optimal probing patterns

We first give some definitions. A *probing pattern* is binary string whose first and last characters are 1. The *weight* of a probing pattern P is the number of ones in P. The set of *probes* that corresponds to a probing pattern P is the set of all strings that are obtained by replacing every 0 in P by the character ϕ, and every 1 in P by one of the characters from $\Sigma = \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}$. A probe Q appears in a string S if the string Q matches to a substring of S, where the character ϕ is a don't care symbol (namely, it matches to every letter of Σ). The P-spectrum of a string S is the set of all probes from the set of probes of P that appear in S.

As an example for the definitions above, consider the probing pattern $P = 1101$. The set of probes corresponding to P is $\{\mathrm{AA}\phi\mathrm{A}, \mathrm{AA}\phi\mathrm{C}, \mathrm{AA}\phi\mathrm{G}, \mathrm{AA}\phi\mathrm{T}, \mathrm{AC}\phi\mathrm{A}, \ldots, \mathrm{TT}\phi\mathrm{T}\}$, and the P-spectrum of the string ACGATAC is $\{\mathrm{AC}\phi\mathrm{A}, \mathrm{AT}\phi\mathrm{C}, \mathrm{CG}\phi\mathrm{T}, \mathrm{GA}\phi\mathrm{A}\}$.

Our goal is to find optimal probing patterns, so we first need to define the notion of optimality. A string S is *unambiguously reconstructable* from its P-spectrum if there is no $S' \neq S$ whose P-spectrum is equal to the P-spectrum of S. For a probing pattern P, the *resolution power* $r(P, n)$ of P is the fraction of the strings of length n that are unambiguously reconstructable from their P-spectra. The resolution power is a natural measure for comparison between different probing patterns. However, this measure ignores the issue of the time complexity of reconstructing a string from its spectrum. Therefore, instead of the resolution power, we will use the following measure: Let R be a reconstruction algorithm, that is, R receives as input a P-spectrum of a string A

and outputs either the string A or 'failure'. The *success probability* of algorithm R on a probing pattern P, denoted $\mathrm{sp}(P, n, R)$, is the fraction of the strings of length n that are reconstructed correctly from their P-spectra by algorithm R. The *failure probability* of R is $1 - \mathrm{sp}(P, n, R)$.

In this extended abstract, we will concentrate on the reconstruction algorithm of Preparata and Upfal [22], which will be denoted R_{PU}. Our goal is to find a probing pattern P of a given length and weight, that maximizes $\mathrm{sp}(P, n, R_{\mathrm{PU}})$ for some given n. Efficiently computing $\mathrm{sp}(P, n, R_{\mathrm{PU}})$ seems a difficult task, so we will show how to compute a value $\widetilde{\mathrm{sp}}(P, n)$ that approximates $\mathrm{sp}(P, n, R_{\mathrm{PU}})$. A probing pattern P^{OPT} that maximizes $\widetilde{\mathrm{sp}}(P, n)$ is almost optimal with respect to $\mathrm{sp}(P, n, R_{\mathrm{PU}})$. An alternative way to approximate $\mathrm{sp}(P, n, R_{\mathrm{PU}})$ is by Monte Carlo simulations (running algorithm R_{PU} on a large set of random strings and computing the fraction of the runs in which the algorithm succeeds). However, this approach is much more computationally intensive than our approach, which makes it infeasible if the number of probing patterns that needs to be considered is large. Moreover, our approach gives insight on what makes a probing pattern efficient.

In the following, we will use $A = a_1 \cdots a_n$ to denote the target string. Let P be some probing pattern of length L and weight k, and let H be some constant. We assume that the first and last $L - 1$ letters of A are known. Algorithm R_{PU} reconstructs the first $n - H + 1$ letters of A as follows (reconstructing the last $H - 1$ letters is performed in a similar manner by reconstructing the sequence backwards):

1. Let $s_1, \ldots, s_{L-1}$ be the first $L - 1$ letters of A.
2. For $t = L, L + 1, \ldots, n - H + 1$ do:
   (a) Let $\mathcal{B}_t$ be the set of all strings B of length H such that the string $s_1 \cdots s_{t-1}B$ is consistent with the P-spectrum of A (i.e., the P-spectrum of $s_1 \cdots s_{t-1}B$ is a subset of the P-spectrum of A).
   (b) If all the strings in $\mathcal{B}_t$ have a common first letter a, then set $s_t \leftarrow a$. Otherwise, return 'failure'.
3. Return $s_1 \cdots s_{n-H+1}$.

For the rest of this section, we show how to compute an approximation $\widetilde{\mathrm{fp}}(P, n) = 1 - \widetilde{\mathrm{sp}}(P, n)$ of the failure probability of algorithm R_{PU}. Our analysis is similar to the analysis of Heath and Preparata [12]. However, the analysis of Heath and Preparata is specific to the probing pattern of Frieze et al. In particular, they omitted several cases from the analysis which are negligible for that probing pattern. In our analysis, we consider more cases. Moreover, we handle the time complexity for computing $\widetilde{\mathrm{fp}}(P, n)$, which is not done in [12].

It is easy to verify that if algorithm R_{PU} does not return 'failure', then $s_1 \cdots s_{n-H+1} = a_1 \cdots a_{n-H+1}$. Moreover, the algorithm stops at some t if and only if there is a string $B \in \mathcal{B}_t$ whose first letter is not equal to a_t. Such a string B will be called a *bad extension*. The string $a_t \cdots a_{t+H-1} \in \mathcal{B}_t$ is called the *correct extension*.

Suppose that the algorithm failed at some t, and let $B = b_1 \cdots b_H$ be the corresponding bad extension. Denote $B' = a_{t-l+1} \cdots a_{t-1} b_1 \cdots b_H$. By definition, the H probes that

appear in B' also appear in A. That is, for every $i = 1, \ldots, H$, there is an index r_i such that $B'[i + j - 1] = A[r_i + j - 1]$ for all $1 \leq j \leq L$ for which $P[j] = 1$. The probe that corresponds to the index r_i is called *supporting probe* i. Supporting probe i is called a *fooling probe* if $r_i \neq t - L + i$. Fooling probe i is called *close* if $r_i \in \{t - L + 2, \ldots, t\}$. Two supporting probes i and j will be called *adjacent* if $r_j - r_i = j - i$, and they will be called *overlapping* if $|r_j - r_i| < L$ and they are not adjacent. Fooling probe i is *simple* if it is not close, and it is not adjacent or overlapping with another fooling probe.

Let J be the minimum index such that the probes $J, J+1, \ldots, H$ are pairwise adjacent. Note that some of the supporting probes can appear more than once in A, and therefore, there may be several ways to choose the values of $r_1, \ldots, r_H$. We assume that these values are chosen in a way that minimizes the number of fooling probes and the value of J.

2.1 Simple probes

We first assume that fooling probes $1, \ldots, J-1$ are simple, and that probe J is a fooling probe. We will analyze the case of non-simple fooling probes in Section 2.2. Let α denote the probability that a random probe appears in the string A. Using the Chen-Stein method, it is easy to show that the number of occurrences of a random probes in A is approximated by a Poisson distribution with mean $(n-L+1)/4^k$. In particular, we have that $\alpha \approx 1 - e^{-(n-L+1)/4^k}$. We consider several cases:

Case 1 $J = 1$. In this case we have that $a_{r_1} \cdots a_{r_1+L-1} = a_{t-L+1} \cdots a_{t-1} b_1$. This event is composed of $L-1$ character equalities in A ($a_{r_1+j-1} = a_{t-L+j}$ for $j = 1, \ldots, L-1$), and one character inequality ($a_{r_1+L-1} = b_1 \neq a_t$). Thus, this event happens with probability $3/4^L$ for fixed t and r_1. Since there are approximately $\binom{n}{2}$ ways to choose t and r_1, it follows that the contribution of case 1 to $\widetilde{\text{fp}}(P, n)$ is by

$$b_1 = \frac{n^2}{2} \cdot \frac{3}{4^L}.$$

Case 2 $2 \leq J \leq L$. In this case we have $a_{r_J} \cdots a_{r_J+L-J-1} = a_{t-L+J} \cdots a_{t-1}$, and $a_{r_J+L-J} \neq a_t$. Moreover, from the minimality of J we have that $a_{r_J-1} \neq a_{t-L+J-1}$. Which of the probes $1, \ldots, J-1$ are fooling probes? If for a probe i, $b_{i-L+j} = a_{t+i-L+j-1}$ for all j for which $L - i + 1 \leq j \leq L$ and $P[j] = 1$ (in words, the characters sampled by probe i are equal in the bad extension and the correct extension) then the probe is not a fooling probe. Therefore, the number of fooling probes depends on the mismatches between the strings $a_t \cdots a_{t+J-2}$ and $b_1 \cdots b_{J-1}$. Let C be a binary string of length $J-2$, where $C[i] = 1$ if $a_{t+i} \neq b_{i+1}$, and $C[i] = 0$ otherwise. Let $\hat{C} = 0^{L-1}1C$, namely, a string with $L-1$ zeros followed by one is concatenated to C (the leftmost 1 is due to the fact that we always have $a_t \neq b_1$). We say that the pattern P *hits* a string S of length L if there is an index i such that $P[i] = S[i] = 1$. Thus, the number of fooling probes among probes $1, \ldots, J-1$ is equal to the number of substrings of $\hat{C}$ of length L that are hit by P, which will be denoted $\text{hits}(\hat{C})$.

Since probes $J, J+1, \ldots$ are pairwise adjacent, we have that $b_i = a_{r_J+L-j+i-1}$ for $i = 1, \ldots, L-1$. Therefore, $C[i] = 0$ forces the equality $a_{t+i} = a_{r_J+L-J+i}$ (which happens with probability $1/4$), and $C[1] = 1$ forces the inequality $a_{t+i} \neq a_{r_J+L-J+i}$ (which happens

with probability 3/4). Thus, for fixed t, r_i, and C, the probability that the equalities between the symbols a_{t+i} and b_{i+1} are according to C is $3^{\mathrm{ones(C)}}/4^{J-2}$, where ones$(C)$ is the number of ones in C. It follows that the contribution of case 2 to $\widetilde{\mathrm{fp}}(P,n)$ is $\sum_{J=2}^{L} b_J$, where

$$b_J = n^2 \left(\frac{3}{4}\right)^2 \frac{1}{4^{L-J}} \sum_{C\in\{0,1\}^{J-2}} \frac{3^{\mathrm{ones(C)}}\alpha^{\mathrm{hits}(0^{\mathrm{L}-1}1\mathrm{C})}}{4^{J-2}}$$
$$= n^2 \frac{9}{4^L} \sum_{C\in\{0,1\}^{J-2}} 3^{\mathrm{ones(C)}}\alpha^{\mathrm{hits}(0^{\mathrm{L}-1}1\mathrm{C})}.$$

Case 3 $L+1 \le J \le H-L+2$. This case is similar to case 2, so we omit the details. The contribution of this case to $\widetilde{\mathrm{fp}}(P,n)$ is $\sum_{J=L+1}^{H-L+2} b_J$, where

$$b_J = n^2 \frac{9}{4^L} \sum_{C\in\{0,1\}^{J-2}} 3^{\mathrm{ones(C)}}\alpha^{\mathrm{hits}(0^{\mathrm{L}-1}1\mathrm{C})}.$$

Case 4 $J > H-L+2$. The contribution of this case to $\widetilde{\mathrm{fp}}(P,n)$ is negligible (we omit the details).

We now handle the time complexity of computing $b_2,\ldots,b_{H-L+2}$. A straightforward computation of $b_2,\ldots,b_{H-L+2}$ takes $O(\sum_{J=2}^{H-L+2} LJ\cdot 2^J) = O(LH\cdot 2^H)$ time. We now show a dynamic programming algorithm for computing $b_2,\ldots,b_{H-L+2}$ in $O(H\cdot 2^L)$ time. For $J=2,\ldots,H-L+2$ and a binary string C of length $\min(J-2,L-1)$, define

$$b(J,C) = \sum_{C'\in\{0,1\}^{J-2}:C' \text{ is a suffix of } C} 3^{\mathrm{ones(C')}}\alpha^{\mathrm{hits}(0^{\mathrm{L}-1}1\mathrm{C'})}.$$

Clearly,

$$b_J = n^2 \frac{9}{4^L} \sum_{C\in\{0,1\}^{\min(J-2,L-1)}} b(J,C),$$

and the following recurrence is used to compute $b(J,C)$: For $J < L+2$,

$$b(J,C) = b(J-1, C[1]C[2]\cdots C[|C|-1])\cdot 3^{C[|C|]}\cdot\alpha^{\mathrm{hits}(0^{L-|C|-1}1C)},$$

and for $J \ge L+2$,

$$b(J,C) = \sum_{x\in\{0,1\}} b(J-1, xC[1]C[2]\cdots C[|C|-1])\cdot 3^{C[|C|]}\cdot\alpha^{\mathrm{hits}(xC)}.$$

The computation of $\mathrm{hits}(0^{L-|C|-1}1C)$ or $\mathrm{hits}(xC)$ is done in $O(1)$ time by computing a table that stores the value of $\mathrm{hits}(C')$ for every string C' of length L.

2.2 Non-simple probes

Consider cases 2 and 3 above.

Case 2 Fix some $2 \leq J \leq L$. In this extended abstract, we only handle the case when some of the probes $1, \ldots, J-1$ are adjacent to probe J, and the rest of the probes from $1, \ldots, J-1$ are pairwise non-adjacent. Consider some fixed $C \in \{0,1\}^{J-2}$, and let I_C be the set of fooling probes that correspond to substrings of $0^{L-1}1C$ that are hit by P. We say that a probe $i \in I_C$ *samples* position $r_J - j$ if $1 \leq j \leq J-i$ and $P[(J-i)+1-j] = 1$, or in other words, probe i contains the character a_{r_J-j} if it is adjacent to probe J.

By the definition of J, $a_{r_J-1} \neq a_{t-L+J-1}$. Therefore, the probes that sample position $r_J - 1$ cannot be adjacent to probe J. Let I'_C be the set of the probes in I_C that do not sample $r_J - 1$. Let $S_C = \{r_J - j_1, \ldots, r_J - j_{|S_C|}\}$ be the set of all the positions $r_J - j$ that are sampled by at least one probe from I'_C. If a probe $i \in I'_C$ is adjacent to probe J then $a_{r_J-j} = a_{t-L+J-j}$ for every position $r_J - j$ that is sampled by probe i. For a target string A, the equalities of the form $a_{r_J-j} = a_{t-L+J-j}$ that are satisfied for positions $r_J - j \in S_C$ can be represented by a binary string C' of length $|S_C|$: $C'[l] = 1$ if $a_{r_J-j_l} \neq a_{t-L+J-j_l}$ and $C'[l] = 0$ otherwise. The probes in I'_C that are adjacent to probe J can be determined from the string C': For each such probe, $C'[l] = 0$ for every l such that position $r_J - j_l$ is sampled by the probe. We define fooling(P, I_C, C') to be the number of probes in I'_C that sample some position $r_J - j_l$ with $C'[l] = 1$. To account for non-simple probes, we change the definition of b_J from Section 2.1 to

$$b_J = n^2 \frac{9}{4^L} \sum_{C \in \{0,1\}^{J-2}} \beta(P, I_C),$$

where

$$\beta(P, I_C) = 3^{\mathrm{ones(C)}} \alpha^{|\mathrm{I_C} - \mathrm{I'_C}|} \frac{1}{4^{|S_C|}} \sum_{C' \in \{0,1\}^{|S_C|}} 3^{\mathrm{ones}(C')} \alpha^{\mathrm{fooling}(P, I_C, C')}.$$

A naive computation of $\beta(P, I_C)$ is time consuming. To compute $\beta(P, I_C)$ more efficiently, we use the following idea: Let $S \subseteq S_C$ be the set of all positions $r_J - j_l \in S_C$ such that the set of probes that sample $r_J - j_l$ is equal to the set of probes that sample $r_J - j_1$. The positions in S can be collapsed into a single positions, namely instead of representing a configuration by a binary string C' of length S_C, we can represent a configuration using a string C'' of length $1 + |S_C - S|$. This can be repeated with the other positions in S_C.

Another speedup follows from the following observation:

Claim 1. *For two probing patterns P and P', if $P[i] \geq P'[i]$ for all i, then $\beta(P, I_C) \leq \beta(P', I_C)$ for every set I_C.*

We use the claim as follows. When searching for the optimal probing pattern of length L and weight k, we first compute $\beta(P', I_C)$ for all sets I_C and for all probing patterns with length L and weight at most k, in which all the ones are in first 8 positions of the pattern or the last position. Then, when computing the failure probability for some pattern P, we choose the pattern P' whose prefix of length 8 is equal to the prefix of length 8 of P, and we use $\beta(P', I_C)$ instead of $\beta(P, I_C)$.

Case 3 In this we need to consider two sub-cases. The first case is when probe J is a fooling probe. The analysis of this case is similar to the analysis of the previous case. The second case is when probe J is not a fooling probe, namely $r_J = t - L + J$.

We have that $b_1 \neq a_t$ and from the minimality of J, $b_{J-L} \neq a_{t-1+J-L}$. Recall that C is a binary string of length $J-2$, where $C[i] = 1$ if $a_{t+i} \neq b_{i+1}$, and $C[i] = 0$ otherwise. From the fact that $r_{J+i} = t-L+J+i$ for $i \geq 0$ it follows that $C[J-L] = C[J-L+1] = \cdots = C[J-2] = 0$. From the minimality of J, $C[J-L-1] = 1$ when $J > L+1$. Therefore, for $J > L+1$, we add the term

$$b'_J = 9n \sum_C 3^{\mathrm{ones(C)}} \alpha^{\mathrm{hits}(0^{\mathrm{L}-1}1\mathrm{C})}$$

to b_J, where the sum is over all strings $C \in \{0,1\}^{J-2}$ that satisfy $C[J-L] = C[J-L+1] = \cdots = C[J-2] = 0$ and $C[J-L-1] = 1$. For $J = L+1$ we have that only one string C satisfies the requirements (the string $C = 0^{J-2}$) and we have the term

$$b'_{L+1} = 3n \cdot 3^{\mathrm{ones}(0^{\mathrm{J}-2})} \alpha^{\mathrm{hits}(0^{\mathrm{L}-1}10^{\mathrm{J}-2})} = 3n \cdot \alpha^k.$$

The case of probe J not being a fooling probe for $J = L+1$ was called "Mode 1" in [12].

3 Results

The s,r-*probing pattern* of Frieze et al. [9] is the pattern $1^s(0^{s-1}1)^r$. For a fixed weight k, the optimal s,r-probing pattern is the pattern with $s = \lceil k/2 \rceil$ (and $r = \lfloor k/2 \rfloor$). Denote this pattern by P_k^{FPU}. We run the algorithm of Section 2 with $k = 7$, $L = 15, 16, 17$, and $n = 4000$. The best patterns found by the algorithm for $L = 15, 16, 17$ are $P_{15,7}^{\mathrm{OPT}} = 111001000001011$, $P_{16,7}^{\mathrm{OPT}} = 1101000100001011$, and $P_{17,7}^{\mathrm{OPT}} = 11010001000001011$, respectively. For each probing pattern, we ran algorithm R_{PU} on 1000 random target strings (with the probing patterns P_7^{FPU}, $P_{15,7}^{\mathrm{OPT}}$, $P_{16,7}^{\mathrm{OPT}}$, and $P_{17,7}^{\mathrm{OPT}}$), and computed the success rate of the algorithm. The results are given in Figures 1 and 2. The failure rate of R_{PU} for the pattern $P_{16,7}^{\mathrm{OPT}}$ (whose length is the same as the length of P_7^{FPU}) is about 3 times smaller than the failure rate for P_7^{FPU}.

Recall that Mode 1 refers to the case when the bad extension differs from the correct extension only in the first letter. The bad extension is supported by k fooling probes. Using Poisson approximations, the probability that a failure due to Mode 1 occurs is approximately $1 - e^{-3(n-L+1)\alpha^k}$. Note that this probability depends only on the length L of the probing pattern, but not on the pattern itself. Therefore, $e^{-3(n-19)\alpha^k}$ is an upper bound on the success probability of all probing patterns of length at most 20. This upper bound is shown as a gray solid line in Figures 1 and 2. An analysis of the failures show that most (about two thirds) of the failures in the runs of $P_{16,7}^{\mathrm{OPT}}$ are due to Mode 1, while only small part of the failures in the runs of P_7^{FPU} are due to Mode 1. Since Mode 1 failure is unavoidable, we have that the probing pattern $P_{16,7}^{\mathrm{OPT}}$ is very close to optimal.

It is clear from the analysis of Section 2 that longer probing patterns can achieve smaller failure probability. Indeed, the pattern $P_{17,7}^{\mathrm{OPT}}$ performs better than $P_{16,7}^{\mathrm{OPT}}$, and its failure probability is very close to the lower bound of Mode 1 failure probability. Moreover, while the pattern $P_{15,7}^{\mathrm{OPT}}$ is shorter than P_7^{FPU}, it has a smaller failure probability than P_7^{FPU} (for $n \geq 2000$).

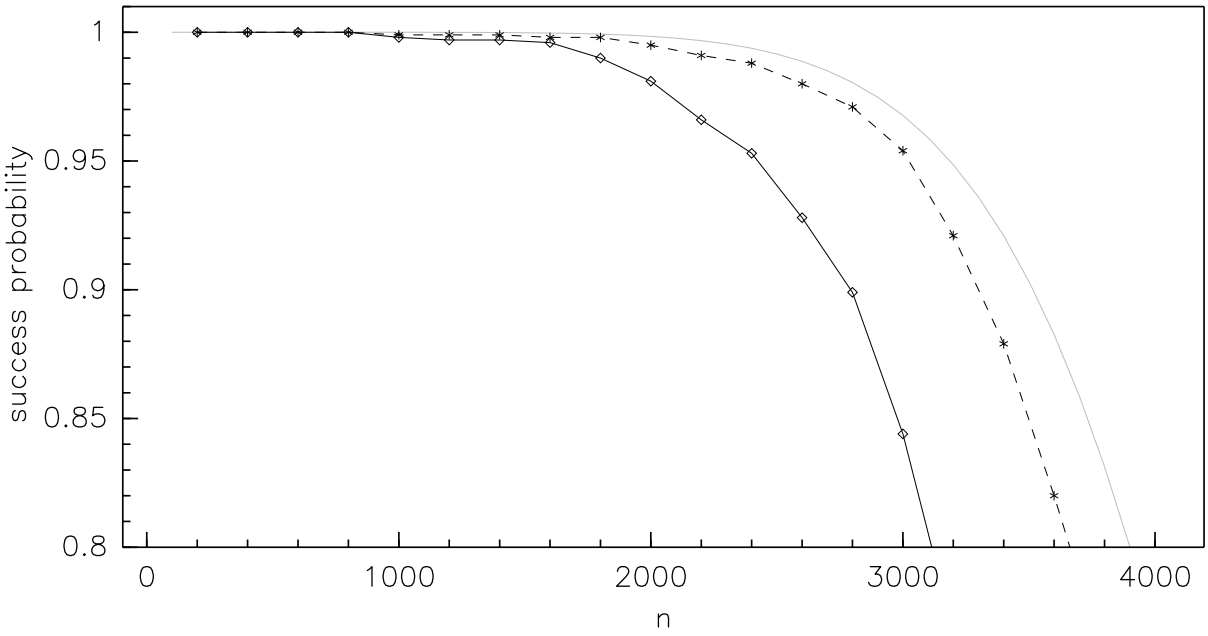

Figure 1: Success probability of algorithm R_{PU} on the patterns P_7^{FPU} (solid line) and $P_{16,7}^{\mathrm{OPT}}$ (dashed line) for various values of n. The gray solid line gives the probability that Mode 1 does not occur, which is an upper bound on the success probability for any pattern.

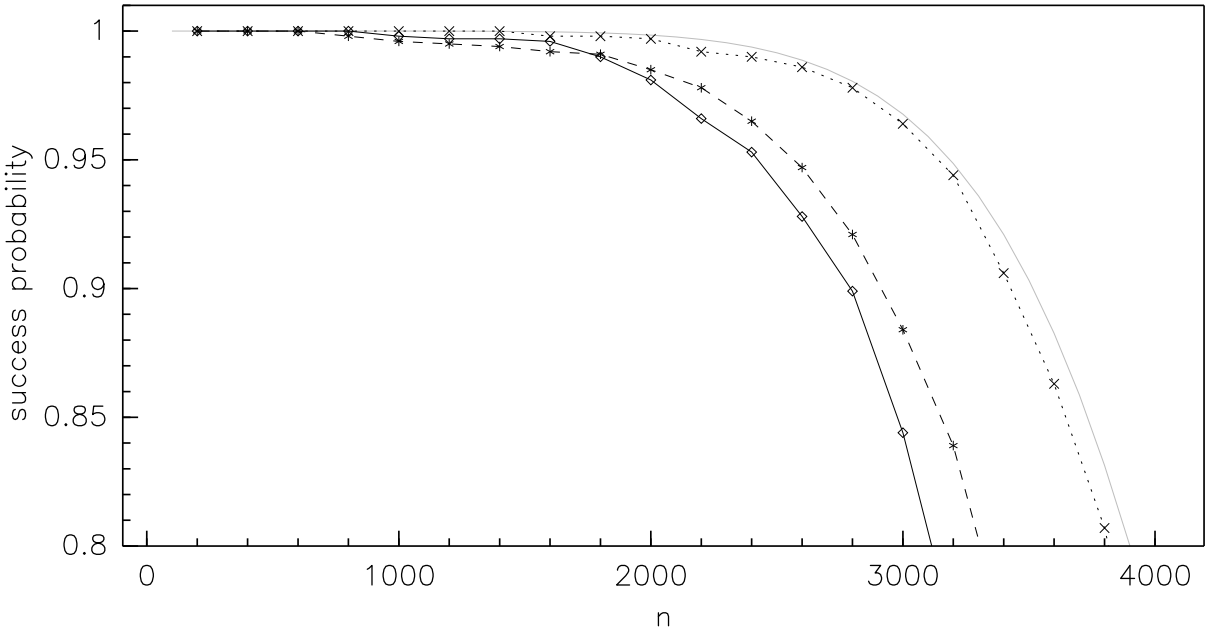

Figure 2: Success probability of algorithm R_{PU} on the patterns P_7^{FPU} (solid line), $P_{15,7}^{\mathrm{OPT}}$ (dashed line), and $P_{17,7}^{\mathrm{OPT}}$ (dotted line) for various values of n.

References

[1] L. M. Adleman. Location sensitive sequencing of DNA. Technical report, University of Southern California, 1998.

[2] R. Arratia, D. Martin, G. Reinert, and M. S. Waterman. Poisson process approximation for sequence repeats, and sequencing by hybridization. *J. of Computational Biology*, 3(3):425–463, 1996.

[3] W. Bains and G. C. Smith. A novel method for nucleic acid sequence determination. *J. Theor. Biology*, 135:303–307, 1988.

[4] A. Ben-Dor, I. Pe'er, R. Shamir, and R. Sharan. On the complexity of positional sequencing by hybridization. *J. Theor. Biology*, 8(4):88–100, 2001.

[5] S. D. Broude, T. Sano, C. S. Smith, and C. R. Cantor. Enhanced DNA sequencing by hybridization. *Proc. Nat. Acad. Sci. USA*, 91:3072–3076, 1994.

[6] J. Buhler, U. Keich, and Y. Sun. Designing seeds for similarity search in genomic DNA. *J. of Computer and System Sciences*, 70(3):342–363, 2005.

[7] R. Drmanac, I. Labat, I. Brukner, and R. Crkvenjakov. Sequencing of megabase plus DNA by hybridization: theory of the method. *Genomics*, 4:114–128, 1989.

[8] M. E. Dyer, A. M. Frieze, and S. Suen. The probability of unique solutions of sequencing by hybridization. *J. of Computational Biology*, 1:105–110, 1994.

[9] A. Frieze, F. P. Preparata, and E. Upfal. Optimal reconstruction of a sequence from its probes. *J. of Computational Biology*, 6:361–368, 1999.

[10] A. M. Frieze and B. V. Halldórsson. Optimal sequencing by hybridization in rounds. *J. of Computational Biology*, 9(2):355–369, 2002.

[11] S. Hannenhalli, P. A. Pevzner, H. Lewis, and S. Skiena. Positional sequencing by hybridization. *Computer Applications in the Biosciences*, 12:19–24, 1996.

[12] S. A. Heath and F. P. Preparata. Enhanced sequence reconstruction with DNA microarray application. In *COCOON '01*, pages 64–74, 2001.

[13] S. A. Heath, F. P. Preparata, and J. Young. Sequencing by hybridization using direct and reverse cooperating spectra. *J. of Computational Biology*, 10(3/4):499–508, 2003.

[14] U. Keich, M. Li, B. Ma, and J. Tromp. On spaced seeds for similarity search. *Discrete Applied Mathematics*, 138(3):253–263, 2004.

[15] G. Kucherov, L. Noé, and M. Roytberg. A unifying framework for seed sensitivity and its application to subset seeds. In *Proc. 5th Workshop on Algorithms in Bioinformatics (WABI '05)*, pages 251–263, 2005.

[16] Y. Lysov, V. Floretiev, A. Khorlyn, K. Khrapko, V. Shick, and A. Mirzabekov. DNA sequencing by hybridization with oligonucleotides. *Dokl. Acad. Sci. USSR*, 303:1508–1511, 1988.

[17] D. Margaritis and S. Skiena. Reconstructing strings from substrings in rounds. In *Proc. 36th Symposium on Foundation of Computer Science (FOCS 95)*, pages 613–620, 1995.

[18] I. Pe'er, N. Arbili, and R. Shamir. A computational method for resequencing long DNA targets by universal oligonucleotide arrays. *Proc. National Academy of Science USA*, 99:15497–15500, 2002.

[19] I. Pe'er and R. Shamir. Spectrum alignment: Efficient resequencing by hybridization. In *Proc. 8th International Conference on Intelligent Systems in Molecular Biology (ISMB '00)*, pages 260–268, 2000.

[20] P. A. Pevzner, Y. P. Lysov, K. R. Khrapko, A. V. Belyavsky, V. L. Florentiev, and A. D. Mirzabekov. Improved chips for sequencing by hybridization. *J. Biomolecular Structure and Dynamics*, 9:399–410, 1991.

[21] F. P. Preparata and J. S. Oliver. DNA sequencing by hybridization using semi-degenerate bases. *J. of Computational Biology*, 11(4):753–765, 2004.

[22] F. P. Preparata and E. Upfal. Sequencing by hybridization at the information theory bound: an optimal algorithm. *J. of Computational Biology*, 7:621–630, 2000.

[23] R. Shamir and D. Tsur. Large scale sequencing by hybridization. *J. of Computational Biology*, 9(2):413–428, 2002.

[24] S. Skiena and S. Snir. Restricting SBH ambiguity via restriction enzymes. In *Proc. 2nd Workshop on Algorithms in Bioinformatics (WABI '02)*, pages 404–417, 2002.

[25] S. Skiena and G. Sundaram. Reconstructing strings from substrings. *J. of Computational Biology*, 2:333–353, 1995.

[26] S. Snir, E. Yeger-Lotem, B. Chor, and Z. Yakhini. Using restriction enzymes to improve sequencing by hybridization. Technical Report CS-2002-14, Technion, Haifa, Israel, 2002.

[27] D. Tsur. Bounds for resequencing by hybridization. In *Proc. 3rd Workshop on Algorithms in Bioinformatics (WABI '03)*, LNCS 2812, pages 498–511, 2003.

[28] D. Tsur. Sequencing by hybridization in few rounds. In *Proc. 11th European Symposium on Algorithms (ESA '03)*, LNCS 2832, pages 506–516, 2003.