2009

Probabilistic Analysis of a Motif Discovery Algorithm for Multiple Sequences

Bin Fu
The University of Texas Rio Grande Valley

Ming-Yang Kao

Lusheng Wang

Recommended Citation

Fu, Bin, Ming-Yang Kao, and Lusheng Wang. 2009. "Probabilistic Analysis of a Motif Discovery Algorithm for Multiple Sequences." SIAM Journal on Discrete Mathematics 23 (4): 1715–37. https://doi.org/10.1137/080720401.

SIAM J. DISCRETE MATH.
Vol. 23, No. 4, pp. 1715–1737

PROBABILISTIC ANALYSIS OF A MOTIF DISCOVERY ALGORITHM FOR MULTIPLE SEQUENCES*

BIN FU†, MING-YANG KAO‡, AND LUSHENG WANG§

Abstract. We study a natural probabilistic model for motif discovery that has been used to experimentally test the quality of motif discovery programs. In this model, there are k background sequences, and each character in a background sequence is a random character from an alphabet Σ. A motif $G = g_1g_2\cdots g_m$ is a string of m characters. Each background sequence is implanted into a probabilistically generated approximate copy of G. For an approximate copy $b_1b_2\cdots b_m$ of G, every character b_i is probabilistically generated such that the probability for $b_i \neq g_i$ is at most α. In this paper, we give the first analytical proof that multiple background sequences do help with finding subtle and faint motifs. This work is a theoretical approach with a rigorous probabilistic analysis. We develop an algorithm that under the probabilistic model can find the implanted motif with high probability when the number of background sequences is reasonably large. Specifically, we prove that for $\alpha < 0.1771$ and any constant $x \geq 8$, there exist constants $t_0, \delta_0, \delta_1 > 0$ such that if the length of the motif is at least $\delta_0 \log n$, the alphabet has at least t_0 characters, and there are at least $\delta_1 \log n_0$ input sequences, then in $O(n^3)$ time our algorithm finds the motif with probability at least $1 - \frac{1}{2^x}$, where n is the longest length of any input sequence and $n_0 \leq n$ is an upper bound for the length of the motif.

Key words. motif, probabilistic analysis, multiple sequences

AMS subject classification. 68W01

DOI. 10.1137/080720401

1. Introduction.

Motif discovery is an important problem in computational biology and computer science. For instance, it has applications to coding theory [3, 5], locating binding sites and conserved regions in unaligned sequences [7, 11, 18, 19], genetic drug target identification [10], designing genetic probes [10], and universal polymerase chain reaction (PCR) primer design [2, 10, 14, 17].

This paper focuses on the application of motif discovery to finding conserved regions in a set of DNA, RNA, or protein sequences. Such conserved regions may represent common biological functions or structures. Many performance measures have been proposed for motif discovery. Let C be a set of 0-1 sequences of length n. The *covering radius* of C is the smallest integer r such that each vector in $\{0,1\}^n$ is at a Hamming distance at most r from a string in C. The decision problem associated with the covering radius for a set of binary sequences is NP-complete [3].

*Received by the editors April 5, 2008; accepted for publication (in revised form) September 7, 2009; published electronically November 18, 2009. A preliminary version of this work appeared as *Efficient algorithms for model-based motif discovery from multiple sequences*, in Proceedings of the Fifth Annual Conference on Theory and Applications of Models of Computation, 2008, pp. 234–245.
http://www.siam.org/journals/sidma/23-4/72040.html

†Department of Computer Science, University of Texas-Pan American, Edinburg, TX 78539 (binfu@cs.panam.edu). The first author's research was supported in part by Louisiana Board of Regents fund under contract LEQSF(2004-07)-RD-A-35 and National Science Foundation Early Career Award 0845376.

‡Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208 (kao@northwestern.edu). The second author's research was supported in part by National Science Foundation grant CNS-0627751.

§Department of Computer Science, The City University of Hong Kong, Kowloon, Hong Kong (lwang@cs.cityu.edu.hk). The third author's research was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project City U 121207).

The similar closest string and substring problems were proved to be NP-hard [3, 10]. Some approximation algorithms have been proposed. Li, Ma, and Wang [13] gave an approximation scheme for the closest string and substring problems. The related consensus patterns problem is as follows: given n sequences $s_1, \cdots, s_n$, find a region of length L in each s_i and a string s of length L so that the total Hamming distance from s to these regions is minimized. Approximation algorithms for the consensus patterns problem were reported in [12]. Furthermore, a number of heuristics and programs have been developed [1, 8, 9, 16, 20].

In many applications, motifs are faint and may not be apparent when only two sequences are compared but may become clearer when more sequences are compared at the same time [6]. For this reason, it has been conjectured that comparing more sequences at the same time can help with identifying faint motifs. In this work, we give the first analytical proof for this conjecture. This is a theoretical approach with a rigorous probabilistic analysis.

We study a natural probabilistic model for motif discovery. In this model, there are k background sequences, and each character in the background sequence is a random character from an alphabet Σ. A motif $G = g_1g_2 \cdots g_m$ is a string of m characters. Each background sequence is implanted into a probabilistically generated approximate copy of G. For an approximate copy $b_1b_2 \cdots b_m$ of G, every character b_i is probabilistically generated such that the probability for $b_i \neq g_i$, which is called a *mutation*, is at most α. This model was first proposed in [16] and has been widely used to experimentally test motif discovery programs [1, 8, 9, 20]. We note that a mutation in our model converts a character g_i in the motif into a different character b_i with no further probability restriction than the upper bound of α. In particular, a character g_i in the motif may become any character b_i in $\Sigma - \{g_i\}$ with unequal probabilities.

We design an algorithm that for a reasonably large k can discover the implanted motif with high probability. Specifically, we prove that for $\alpha < 0.1771$ and any constant $x \geq 8$, there exist constants $t_0, \delta_0, \delta_1 > 0$ such that if the length of the motif is at least $\delta_0 \log n$, the alphabet has at least t_0 characters, and there are at least $\delta_1 \log n_0$ input sequences, then in $O(n^3)$ time the algorithm finds the motif with success probability at least $1 - \frac{1}{2^x}$, where n is the longest length of any input sequence and $n_0 \leq n$ is an upper bound for the length of the motif. When x is considered as a parameter of order $O(\log n)$, the parameters t_0, δ_0, δ_1 do not depend on x. We also show some lower bounds that imply that our conditions for the length of the motif and the number of input sequences are tight to within a constant multiplicative factor. This algorithm's time complexity depends on the length of input sequences but is independent of the number of the input sequences. This is because for a fixed x, $\Theta(\log n)$ sequences are sufficient to guarantee the probability of at least $1 - \frac{1}{2^x}$ to discover the motif. In contrast to the NP-hardness of other variants of the common substring problem, motif discovery is solvable in $O(n^3)$ time in this probabilistic model.

Our algorithm is a deterministic algorithm that has provable high probability to return the exact motif. The only source of randomness for the algorithm is the randomness in the input sequences. The algorithm extracts similar consecutive regions among multiple sequences while tolerating noises. The algorithm needs the motif to be long enough, but does not need to have the length of the motif as an input.

In section 2, we elaborate on our model of sequence generation and discuss some basics. We give a brief description of our main algorithm, Find-Noisy-Motif, in section 3. We set up some parameters and constants for the algorithm in section 4.1. The entire Find-Noisy-Motif is described in section 4.2. We analyze Algorithm Find-Noisy-Motif in section 5. Two lower bounds are presented in section 6. We conclude the paper with an open problem in section 7.

2. Notation and the model of sequence generation. For a set A, $|A|$ denotes the number of elements in A. Σ is an alphabet with $|\Sigma| = t \geq 2$. For an integer $n \geq 0$, Σ^n is the set of sequences of length n with characters from Σ. For a sequence $S = a_1 a_2 \cdots a_n$, $S[i]$ denotes the character a_i, and $S[i,j]$ denotes the substring $a_i \cdots a_j$ for $1 \leq i \leq j \leq n$. $|S|$ denotes the length of the sequence S. We use $\emptyset$ to represent the empty sequence, which has length 0.

Let $G = g_1 g_2 \cdots g_m$ be a fixed sequence of m characters. G is the motif to be discovered by our algorithm. A $\Theta_\alpha(n, G)$-sequence is defined to be a sequence S of the form $S = a_1 \cdots a_{n_1} b_1 \cdots b_m a_{n_1+1} \cdots a_{n_2}$, where $n_2 + m \leq n$, each a_i has probability $\frac{1}{t}$ to be π for each $\pi \in \Sigma$, and b_i has probability at most α not equal to g_i for $1 \leq i \leq m$, where $m = |G|$. $\aleph(S)$ denotes the motif region $b_1 \cdots b_m$ of S. The motif region of S may start at a probabilistic, arbitrary, or worst-case position in S. Also, a mutation may convert a character g_i in the motif into an arbitrary or worst-case different character b_i subject only to the restriction that g_i will mutate with probability at most α.

For two sequences $S_1 = a_1 \cdots a_m$ and $S_2 = b_1 \cdots b_m$ of the same length, let $\mathrm{diff}(S_1, S_2) = \frac{|\{i \mid a_i \neq b_i \text{ for } i=1,\ldots,m\}|}{m}$, i.e., the ratio of difference between the two sequences.

DEFINITION 2.1. *For two intervals* $[i_1, j_1]$ *and* $[i_2, j_2]$, *define* $\mathrm{shift}([i_1, j_1], [i_2, j_2]) = \min(|i_1 - i_2|, |j_1 - j_2|)$.

The analysis of our algorithm employs the Chernoff bound [15] and Corollary 2.3 below, which can be derived from that bound (see [13]).

THEOREM 2.2 (see [15]). *Let* $X_1, \cdots, X_n$ *be* n *independent random* 0-1 *variables, where* X_i *takes* 1 *with probability* p_i. *Let* $X = \sum_{i=1}^n X_i$, *and let* $\mu = E[X]$. *Then for any* $\delta > 0$,

(i) $\Pr(X < (1-\delta)\mu) < e^{-\frac{1}{2}\mu\delta^2}$, *and*

(ii) $\Pr(X > (1+\delta)\mu) < [\frac{e^\delta}{(1+\delta)^{(1+\delta)}}]^\mu$.

COROLLARY 2.3 (see [13]). *Let* $X_1, \cdots, X_n$ *be* n *independent random* 0-1 *variables and* $X = \sum_{i=1}^n X_i$.

(i) *If* X_i *takes* 1 *with probability at most* p, *then for any* $\frac{1}{3} > \epsilon > 0$, $\Pr(X > pn + \epsilon n) < e^{-\frac{1}{3}n\epsilon^2}$.

(ii) *If* X_i *takes* 1 *with probability at least* p, *then for any* $\frac{1}{3} > \epsilon > 0$, $\Pr(X < pn - \epsilon n) < e^{-\frac{1}{2}n\epsilon^2}$.

3. A sketch of Algorithm Find-Noisy-Motif. Our Algorithm Find-Noisy-Motif has two phases. The first phase exploits the fact that with high probability, the motif areas in some sequences conserve the first and last characters. Furthermore, the middle areas of the motif change with a small ratio. We will select enough pairs of $\Theta_\alpha(n, G)$-sequences S' and S'' and find their substrings G' and G'', respectively, such that G' and G'' match at their left- and rightmost characters. Furthermore, G' and G'' have only a relatively small difference in the middle areas. For each such pair S' and S'', the substring G'' of S'' is extracted.

During the second phase, a new set of $\Theta_\alpha(n, G)$-sequences $S_1, S_2, \cdots, S_{k_2}$ will be used. For each G'' extracted from a pair of sequences in the first phase, it is used to match a substring G_i of S_i for $i = 1, 2, \ldots, k_2$. Assume that $G_1, \cdots, G_{k_2}$ are derived

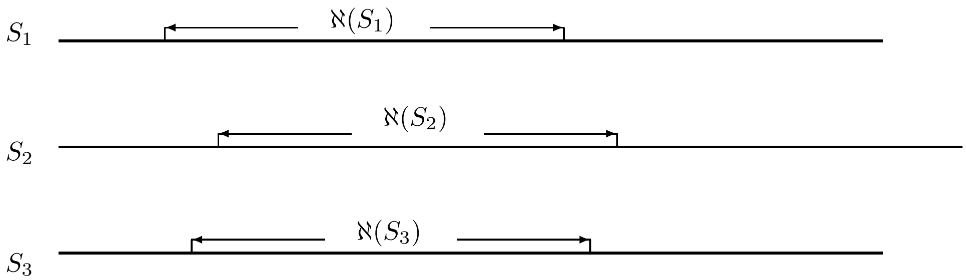

FIG. 1. *The motif regions of S_1, S_2, and S_3 are not aligned.*

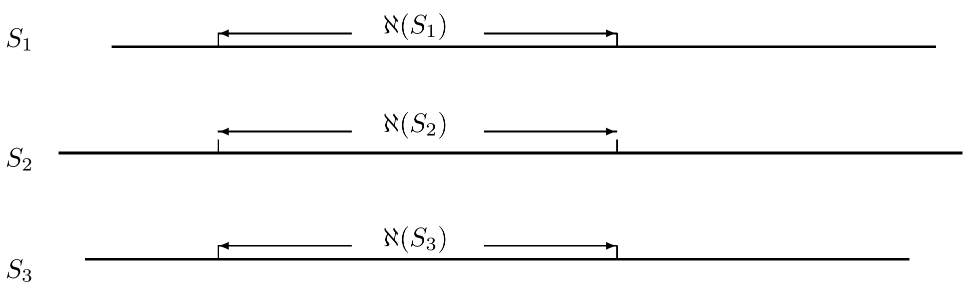

FIG. 2. *S_1, S_2, and S_3 have their motif in the same column region.*

from matching G'' to all sequences $S_1, S_2, \cdots, S_{k_2}$. Some G_i may be the empty sequence if G'' cannot match well to any substring of S_i. If G'' has the same length as that of motif G and is very similar to G, then the number of nonempty sequences among $G_1, \cdots, G_{k_2}$ is much larger than $\frac{k_2}{2}$ and the ith character $G[i]$ of G can be recovered by voting on $G_1[i], \cdots, G_{k_2}[i]$. In other words, $G[i]$ is the character that appears more than $\frac{k_2}{2}$ times in $G_1[i], \cdots, G_{k_2}[i]$. We prove that with high probability such a G'' exists. The rearrangement of $S_1, \cdots, S_{k_2}$ from Figure 1 to Figure 2 illustrates how we recover the motif via voting.

On the other hand, if $|G''| > |G|$ or G'' does not match G well, then we can prove that the number of nonempty sequences among $G_1, \cdots, G_{k_2}$ is less than $\frac{k_2}{2}$. Furthermore, if $|G''| < |G|$, G'' will be dropped since with high probability there exists a candidate G_0 with a good voting performance and the algorithm returns the result from the longest one. Our algorithm's time complexity depends on the length of the input sequences but is independent of the number of the input sequences. This is because for a fixed x, $\Theta(\log n)$ sequences are sufficient to guarantee that with probability at least $1 - \frac{1}{2^x}$ the motif will be discovered. Additional sequences can improve the success probability but are not needed for the high probability guarantee.

4. Algorithm Find-Noisy-Motif. In this section, we detail Algorithm Find-Noisy-Motif. The algorithm can find any hidden motif G in $O(n^3)$ time and with high success probability. It requires that the size of the alphabet is larger than a fixed constant. The performance of the algorithm is stated in the main theorem, Theorem 4.1. The proof of Theorem 4.1 is given in section 5.4.

THEOREM 4.1. *Assume that the mutation probability upper bound α is less than 0.1771. Then there exist constants t_0, δ_0, and δ_1 such that if the size t of the alphabet Σ is at least t_0 and the length of the motif G is at least $\delta_0 \log n$, then, given k independent $\Theta_\alpha(n, G)$-sequences with $k \geq \delta_1 \log n_0$, Algorithm Find-Noisy-Motif outputs G with probability at least $1 - \frac{1}{2^x}$ and runs in $O(n^3)$ time, where n is the longest length of any input sequences and $n_0 \leq n$ is a given upper bound for the length of G.*

Some parameters and constants will be used in Algorithm Find-Noisy-Motif . In section 4.1, we give a list of assignments for some such parameters and constants. The description of Algorithm Find-Noisy-Motif is given in section 4.2. The analysis of the algorithm is given in sections 5.2–5.4.

4.1. Parameters. Multiple parameters affect the performance of the main algorithm, Find-Noisy-Motif; we list them below and discuss some useful inequalities.

- Let x be any constant at least 8. The parameter x controls the failure probability of Find-Noisy-Motif to be at most $\frac{1}{2^x}$. We will prove that Find-Noisy-Motif has probability at least $1 - \frac{1}{2^x}$ to output the exact correct motif G.
- Let α be any constant with $0 \leq \alpha < 0.1771$. Note that

$$(1 - \alpha)^2 - \alpha > \frac{1}{2}. \tag{1}$$

  The parameter α is the upper bound for the mutation probability of each character in the motif region.
- Let $\eta = \frac{1}{6}$. The algorithm has five cases in which it may fail. In order to keep the total failure probability at most $\frac{1}{2^x}$, we ensure that each such case has failure probability at most $\frac{\eta}{2^x}$. As at most five cases can fail, the total failure probability is bounded by $5 \cdot \frac{\eta}{2^x} < \frac{1}{2^x}$.
- Let $\rho_0 = \frac{1}{24}$. We will design a function Extract(S_1, S_2) to output $\aleph(S_2)$ with probability greater than a fixed constant. This parameter controls the probability that Extract(S_1, S_2) derives a substring of S_2 without overlap with the motif region $\aleph(S_2)$ in S_2. It also affects the selection of d, a lower bound of the motif length.
- Let $\epsilon > 0$ be any constant such that

$$(1 - \alpha)^2 - \alpha - 3\epsilon > \frac{1}{2}. \tag{2}$$

  In order to find the motif, we often extract one of two similar substrings from two input sequences. The parameter ϵ controls the similarity of two substrings (see diff(S_1, S_2) in section 2) and appears in the probability that is derived from the Chernoff bound (see Corollary 2.3). The existence of ϵ follows from inequality (1). It also affects the selection of some other parameters.
- Let n be the largest length of an input sequence with $n \geq 3$. Let parameter $n_0 \in [d, n]$ be a given upper bound on the length of the motif G that will be discovered by Algorithm Find-Noisy-Motif. If n_0 is unknown, we just let $n_0 = n$.
- Select a constant $\delta_0 > 0$, and let $d = \delta_0 \log n$ such that

$$n^2 e^{-d} \leq \frac{\eta}{2^x} \tag{3}$$

  and

$$n^2 e^{-\frac{\epsilon^2}{3} d} \leq \rho_0. \tag{4}$$

To satisfy inequalities (3) and (4) above, select

$$\delta_0 = \max\left\{\left(2\left(1+\ln\left(\frac{\eta}{2^x}\right)\right)\right)/\ln 2, \left(\frac{6}{\epsilon^2}(1+\ln\rho_0)\right)/\ln 2\right\}.$$

Note that δ_0 is a constant since both η and x are fixed. We require that the length of the motif G is at least d as stated in Theorem 4.1.

The motif G is a pattern unknown to Algorithm Find-Noisy-Motif. Find-Noisy-Motif will attempt to recover G from a series of $\Theta_\alpha(n, G)$-sequences generated by the probabilistic model in section 2, which is controlled by the parameters α, n, and G. The source of randomness for Find-Noisy-Motif comes entirely from the input sequences.

Recall that a sequence S is generated as follows: (1) Generate a sequence S' with $n - |G|$ characters, in which each character is a random character in Σ. (2) Generate G' such that with probability at most α, $G'[i] \neq G[i]$. $G'[i] \neq G[i]$ represents a mutation. A mutation may create an arbitrary or worst-case $G'[i]$, with no probability restriction except that the mutation occurs with probability at most α. (3) Insert G', which serves as the motif region $\aleph(S)$ of S, into any arbitrary or worst-case position of S'.

Let Z_0 be a set of k_1 pairs of random $\Theta_\alpha(n, G)$-sequences $(S_1', S_1''), \cdots, (S_{k_1}', S_{k_1}'')$. Let Z_1 be the set of $\Theta_\alpha(n, G)$-sequences $\{S_1', S_1'', \cdots, S_{k_1}', S_{k_1}''\}$ in the k_1 pairs of sequences in Z_0. Let Z_2 be a set of k_2 sequences that will be used in the second phase of Algorithm Find-Noisy-Motif. Let $k = 2k_1 + k_2$ be the total number of $\Theta_\alpha(n, G)$-sequences that are used as the input to Find-Noisy-Motif. Both parameters k_1 and k_2 are determined later (see Definition 4.2).

4.2. Description of Algorithm Find-Noisy-Motif. The algorithm is detailed in this section. Before presenting the algorithm, we define some constants and notions.

DEFINITION 4.2.

1. *Select any constant $r_0 > 0$ such that*

$$(1-\alpha)^2 - \alpha - 3\epsilon - 2r_0 > \frac{1}{2}. \tag{5}$$

The constant r_0 will be used to select the constants v (which is defined below) and t_0 (which is the lower bound of the size of the alphabet and is defined in Definition 5.1*). The existence of r_0 follows from inequality* (2)*.*

2. *Let v be the least integer that satisfies the following inequalities:*

$$1 \leq v, \tag{6}$$

$$(1-\alpha)^2 - \frac{2c^v}{1-c} - \alpha - 3\epsilon - 2r_0 > \frac{1}{2}, \tag{7}$$

$$\frac{2c_2 v^3 c^v}{1-c} < \rho_0, \tag{8}$$

$$\frac{2c^v}{1-c} < \frac{r_0}{2}, \tag{9}$$

$$\frac{2c^v}{1-c} < \rho_0, \tag{10}$$

where $c = e^{-\frac{\epsilon^2}{3}}$. Note that the existence of v for inequality (7) *follows from inequality* (5)*.*

The function $\mathrm{Extract}(S_1, S_2)$ *tries to find* $\aleph(S_2)$ *by matching* $\aleph(S_1)$ *and* $\aleph(S_2)$ *without shifting (*$\aleph(S_1)[i]$ *is aligned to* $\aleph(S_2)[i]$*). The parameter* v *is a threshold for the number of characters shifted when matching two motifs from two input sequences (there is one shift if* $\aleph(S_1)[i]$ *is aligned to* $\aleph(S_2)[i+1]$*). For the case where the number of shifts is more than* v*, the Chernoff bound is used to show that the probability is small enough. For the case where the number of shifts is less than* v *but at least* 1*, the probability is still small due to the assumption that the size of the alphabet is large enough.*

3. *The number* k_1 *is selected such that*

$$\left(1 - \frac{1}{12}\right)^{k_1} \leq \frac{\eta}{2^x}. \tag{11}$$

Note that $k_1 = O(1)$ *is a constant independent of the length of the input sequences since both* η *and* x *are constants.*
The parameter k_1 *is the number of pairs of input sequences* $(S_1, S_1'), \cdots, (S_{k_1}', S_{k_1}'')$ *used to extract the motif candidates in the subroutine Phase-One of Find-Noisy-Motif.*

4. *Select a constant* $\delta_1 > 0$*, and let* $k_2 = \delta_1 \log n_0 - 2k_1$ *so that*

$$n_0 k_2 e^{-\frac{\epsilon^2}{3} k_2} \leq \frac{\eta}{2^x} \tag{12}$$

and

$$k_1 e^{-\frac{\epsilon^2}{3} k_2} \leq \frac{\eta}{2^x}. \tag{13}$$

The parameter k_2 *is the number of input sequences used in the subroutine Phase-Two of Find-Noisy-Motif. The candidates for the motif from Phase-One are used to match the motif regions of the* k_2 *sequences in Phase-Two. The original motif* G *is recovered via voting on the* k_2 *substrings.*

Definition 4.3.

1. *Let* $\beta = 2\alpha + 2\epsilon$*. The parameter* β *controls the similarity between* $\aleph(S)$ *and the original motif* G *(see Lemma* 5.8*).*
2. *Two sequences* X_1 *and* X_2 *are left matched if* (1) $|X_1| = |X_2|$, (2) $X_1[1] = X_2[1]$*, and* (3) $\mathrm{diff}(X_1[1,i], X_2[1,i]) \leq \beta$ *for all integers* i, $v \leq i \leq |X_1|$.
3. *Two sequences* X_1 *and* X_2 *are right matched if* X_1^R *and* X_2^R *are left matched, where* $X^R = a_n \cdots a_1$ *is the inverse sequence of* $X = a_1 \cdots a_n$.
4. *Two sequences* X_1 *and* X_2 *are matched if* X_1 *and* X_2 *are both left and right matched.*

Algorithm Find-Noisy-Motif has two phases. The two phases are organized as subroutines Phase-One and Phase-Two, respectively. The input to Phase-One is k_1 pairs of $\Theta_\alpha(n, G)$-sequences collected in the set Z_0. The input to Phase-Two consists of k_2 $\Theta_\alpha(n, G)$-sequences collected in the set Z_2 and the output from Phase-One. All the $\Theta_\alpha(n, G)$-sequences are independent $\Theta_\alpha(n, G)$-sequences. Recall that k_1 is constant, $k_2 = O(\log n_0)$, and n_0 ($\leq n$) is an upper bound for the length of the motif G as discussed in Definition 4.2 and section 4.1. Algorithm Find-Noisy-Motif is deterministic, and its probabilistic performance is based on the randomness of those sequences in both Z_0 and Z_2 and the independence in generating them.

The subroutine LoadInputSequences() below generates the input sequences for Find-Noisy-Motif using the probabilistic model in section 2.

LoadInputSequences()

Steps:

Independently generate $2k_1$ $\Theta_\alpha(n, G)$-sequences $S'_1, S''_1, \cdots, S'_{k_1}, S''_{k_1}$, and let $Z_0 = \{(S'_1, S''_1), (S'_2, S''_2), \cdots, (S'_{k_1}, S''_{k_1})\}$.

Independently generate k_2 $\Theta_\alpha(n, G)$-sequences $S_1, \cdots, S_{k_2}$, and let $Z_2 = \{S_1, \cdots, S_{k_2}\}$.

Return (Z_0, Z_2);

End of LoadInputSequences

The function Extract(S_1, S_2) below extracts the longest similar region between two sequences S_1 and S_2.

Function Extract(S_1, S_2)

Input: a pair of $\Theta_\alpha(n, G)$-sequences S_1 and S_2.

Output: a substring of S_2 which is similar to a substring of S_1.

Steps:

```
for h = min(|S1|, |S2|) to d (recall from section 4.1 that |G| ≥ d)
    for i = 1 to |S1|
        for j = 1 to |S2|
            let i' = i + h − 1 and j' = j + h − 1;
            if S1[i, i'] and S2[j, j'] are matched (see Definition 4.3),
            then return S2[j, j'] and end this function;
return ∅ (output the empty sequence when there is no match found);
```

End of Extract

The following function is Phase-One of Algorithm Find-Noisy-Motif.

Phase-One(Z_0)

Input: $Z_0 = \{(S'_1, S''_1), (S'_2, S''_2), \cdots, (S'_{k_1}, S''_{k_1})\}$, a set of pairs of sequences generated at step 1 of Find-Noisy-Motif.

Output: a set W that contains a similar region of each pair in Z_0.

Steps:

```
let W = ∅ (empty set);
for each pair of sequences (S', S'') ∈ Z0
    let G'' = Extract(S', S'') and put G'' into W;
return W, which will be used in Phase-Two;
```

End of Phase-One

After a set W of motif candidates is produced from Phase-One of Find-Noisy-Motif, we use this function Match(G'', S_i) below to match this set with the set Z_2 of input sequences to recover the hidden motif by voting.

Function Match(G'', S_i)

Input: a motif candidate G'', which is returned from the function Extract(), and a sequence S_i from the group Z_2.

Output: either a substring G_i of S_i of the same length as G'' or an empty sequence, where G_i will be considered as the motif region $\aleph(S_i)$ of S_i if it is not empty and the empty sequence means the failure in extracting the motif region $\aleph(S_i)$ of S_i.

Steps:

```
find a substring Gi of Si with |G''| = |Gi| such that
    G'' and Gi are matched (see Definition 4.3);
if such a Gi does not exist, let Gi = ∅ (empty string);
Output Gi;
```

End of Match

The function $\mathrm{Vote}(G_1, G_2, \cdots, G_{k'})$ below generates another sequence G' by voting, where $G'[i]$ is the most frequent character among $G_1[i], G_2[i], \cdots, G_{k'}[i]$.

Function Vote$(G_1, G_2, \cdots, G_{k'})$

Input: sequences $G_1, G_2, \cdots, G_{k'}$ of the same length with $k' \le k_2$.

Output: a sequence G', which is derived by voting on every position of the input sequences.

Steps:

let $m = |G_1|$;
for each $j = 1, \ldots, m$
if strictly more than $\frac{k_2}{2}$ characters from $G_1[j], \cdots, G_{k'}[j]$ are the same character a,
then let $a_j = a$
else return "failure" and end this function;
return $G' = a_1 \cdots a_m$;

End of Vote

The following function performs Phase-Two of Algorithm Find-Noisy-Motif. It uses the motif candidates for the motif derived in Phase-One to extract the motif regions of the set Z_2 of input sequences and recovers the motif by voting.

Phase-Two(Z_2, W**)**

Input: $Z_2 = \{S_1, \cdots, S_{k_2}\}$ as defined before and W from Phase-One.

Output: G', which is a recovery of motif G.

Steps:

for each $G'' \in W$, let $G_i = \mathrm{Match}(G'', S_i)$ for $i = 1, \ldots, k_2$.
If the number of nonempty sequences in $G_1, \cdots, G_{k_2}$ is at least $(Q_0 - 2R - 2\epsilon)k_2$,
then output $G' = \mathrm{Vote}(G_1, G_2, \cdots, G_{k_2})$ (which will be proved to be identical to G with probability at least $1 - \frac{1}{2^x}$) and end Phase-Two.
return "failure".

End of Phase-Two

The entire main algorithm is described as follows:

Algorithm Find-Noisy-Motif

Steps:

$(Z_0, Z_2) = \mathrm{LoadInputSequences}()$;
$W = \text{Phase-One}(Z_0)$;
$\text{Phase-Two}(Z_2, W)$;

End of Algorithm Find-Noisy-Motif

5. Analysis of Algorithm Find-Noisy-Motif.

Section 5.1 gives an overview of the analysis of Find-Noisy-Motif. Section 5.2 analyzes Phase-One. Section 5.3 analyzes Phase-Two. Section 5.4 gives an overall analysis of Algorithm Find-Noisy-Motif and proves the main theorem, Theorem 4.1.

5.1. Overview of the algorithm analysis.

Phase-One derives k_1 motif candidates via $G''_1 = \mathrm{Extract}(S'_1, S''_1), \cdots, G''_{k_1} = \mathrm{Extract}(S'_{k_1}, S''_{k_1})$. Lemmas 5.2, 5.3, 5.5, and 5.6 show that the probability is small for Phase-One returning a substring not from the motif region of an input sequence in Phase-One. Lemma 5.8 shows that each $\Theta_\alpha(n, G)$-sequence S has its motif region $\aleph(S)$ similar to the motif with high probability. In expecting $\mathrm{Extract}(S'_i, S''_i)$ to return the motif region $\aleph(S''_i)$, we compare the similarity between S'_i and S''_i in order to detect the location of $\aleph(S''_i)$ in S''_i. If the two substrings are $\aleph(S'_i)$ and $\aleph(S''_i)$, there is much similarity between them, and $\aleph(S''_i)$

is found in S_i''. Otherwise, they have high similarity only with a small probability according to Lemmas 5.2, 5.3, 5.5, and 5.6.

After showing that the probability is small for Phase-One returning a nonmotif region, we prove Lemma 5.9 to give an $\Omega(1)$ probability lower bound that a motif region $\aleph(S_i'')$ of S_i'' is returned via Extract(S_i', S_i''). Finally, the analysis of Phase-One will show that one of those $G_1'', \cdots, G_{k_1}''$ has the same length as G and is very similar to the true motif G with high probability. The number k_1 of pairs amplifies the success probability exponentially, as shown in Lemma 5.10. Now assume that G_0 is a relatively accurate motif produced by Phase-One.

Phase-Two uses such G_0 to detect the motif region $\aleph(S_i)$ for each S_i of the k_2 input sequences $S_1, \cdots, S_{k_2}$ via pattern matching between G_0 and a substring in S_i. This generates $G_1, \cdots, G_{k_2}$. Lemma 5.13 shows that G_i is $\aleph(S_i)$ for most of $i = 1, \ldots, k_2$. Therefore, $G[i]$ can be recovered via voting on $G_1[i], \cdots, G_{k_2}[i]$, as shown in Lemma 5.14.

Our main theorem about the correctness and complexity of the algorithm is Theorem 4.1. It combines the analyses for Phase-One and Phase-Two. If a G_i'' produced by Phase-One is longer than G, it will be dropped during the voting. If G_i'' is shorter than G, it will be also dropped since there G_0 is longer and has a better voting consensus than G_i''.

5.2. Analysis of Phase-One of Find-Noisy-Motif. Lemma 5.2 shows that with only a small probability, a sequence can match a random sequence. It will be used to prove that when two substrings in two $\Theta_\alpha(n, G)$-sequences are similar, they are likely to coincide with the motif regions in the two $\Theta_\alpha(n, G)$-sequences, respectively.

DEFINITION 5.1. *Let t_0 be any positive constant such that*

$$\frac{2(v-1)}{t_0} \leq \frac{r_0}{2}, \tag{14}$$

$$\frac{c_2 v^3}{t_0} \leq \rho_0, \tag{15}$$

and

$$\frac{t_0 - 1}{t_0} - \beta > \epsilon. \tag{16}$$

The parameter t_0 is used as a required lower bound of alphabet size. In the remainder of this paper, we always assume the alphabet size t is at least t_0.

LEMMA 5.2. *Assume that X_1 and X_2 are two independent sequences of the same length and that every character of X_2 is a random character from Σ. Then the following hold:*

(i) *If $1 \leq |X_1| = |X_2| < v$, then the probability that X_1 and X_2 are matched is $\leq \frac{1}{t}$, where $t = |\Sigma|$.*

(ii) *If $v \leq |X_1| = |X_2|$, then the probability for* $\text{diff}(X_1, X_2) \leq \beta$ *is at most $e^{-\frac{\epsilon^2 |X_1|}{3}}$.*

Proof. The two statements are proved as follows.

Statement (i). For every character $X_2[j]$ with $1 \leq j < v$, the probability is $\frac{1}{t}$ that $X_2[j] = X_1[j]$.

Statement (ii). For every character $X_2[j]$ with $1 \leq j \leq |X_2|$, the probability is $\frac{1}{t}$ for $X_2[j] = X_1[j]$. The expected number of positions where the two sequences X_1 and X_2 differ is $\frac{t-1}{t}|X_1|$. Since $\beta = \frac{t-1}{t} - (\frac{t-1}{t} - \beta)$, the probability for $\text{diff}(X_1, X_2) \leq \beta$

is at most $e^{-\frac{(\frac{t-1}{t}-\beta)^2}{3}|X_1|} \le e^{-\frac{\epsilon^2}{3}|X_1|}$ by inequality (16), Corollary 2.3, and the fact that $t \ge t_0$ (see Definition 5.1). $\square$

Function Extract(S_1, S_2) returns a substring of S_2. We expect that Extract(S_1, S_2) is the motif region $\aleph(S_2)$ in S_2. Lemma 5.3 shows that with a small probability, the region for Extract(S_1, S_2) in S_2 does not overlap the motif region $\aleph(S_2)$ of S_2.

LEMMA 5.3. *With probability at most ρ_0,* Extract(S_1, S_2) *and* $\aleph(S_2)$ *are not overlapping substrings of S_2. In other words, with probability at most ρ_0, $[j, j'] \cap [f, f'] = \emptyset$, where* Extract$(S_1, S_2) = S_2[j, j']$ *and* $\aleph(S_2) = S_2[f, f']$.

Proof. Assume that Extract(S_1, S_2) returns $M = S_2[j, j'] =$ Extract(S_1, S_2) such that $S_2[j, j']$ matches $S_1[i, i']$. Assume that $\aleph(S_2) = S_2[f, f']$. Further, assume that M and $\aleph(S_2)$ have no overlap in S_2; i.e., $[j, j'] \cap [f, f'] = \emptyset$. For the condition $\mathrm{diff}(S_1[i, i'], S_2[j, j']) \le \beta$, the probability is at most $e^{-\frac{\epsilon^2 d}{3}}$ by Lemma 5.2 (notice that the length of M is at least d according to Extract()). The probability of M and $\aleph(S_2)$ not overlapping for all possible $[j, j']$ is at most $n^2 \cdot e^{-\frac{\epsilon^2 d}{3}} \le \rho_0$ by inequality (4). $\square$

In order to show that Extract(S_1, S_2) can be used to effectively find a motif region in S_2, we give Lemma 5.5 to show that with only small probability, the region of Extract(S_1, S_2) in S_2 may shift far from the motif region $\aleph(S_2)$.

DEFINITION 5.4. *The constant z is selected so that*

$$z \ge v \tag{17}$$

and

$$\frac{4e^{-\frac{\epsilon^2}{3}z}}{1 - e^{-\frac{\epsilon^2}{3}}} \le \rho_0. \tag{18}$$

The parameter z is a threshold for controlling the shift in the analysis of Phase-One of Find-Noisy-Motif. See Lemma 5.5 *and Definition* 2.1.

LEMMA 5.5. *The probability is at most $H_1 = 2\rho_0$ that for a pair of sequences (S_1, S_2) from Z_0,* $\mathrm{shift}([i_2, j_2], [i_2', j_2']) \ge z$ *and $|S_2[i_2, j_2]| \ge |G|$, where $S_2[i_2, j_2] =$* Extract(S_1, S_2)*, $\aleph[S_2] = S_2[i_2', j_2']$, and z is as defined in Definition* 5.4.

Proof. Assume that $M = S_2[i_2, j_2] =$ Extract(S_1, S_2) is the matched sequence. By Lemma 5.3, the probability is at most ρ_0 that M does not intersect $\aleph(S_2)$.

Notice that $M = S_2[i_2, j_2]$ is a substring of S_2 according to the function Extract(). Assume that $M' = S_1[i_1, j_1]$ is the substring of S_1 such that M' and M are matched in the function Extract(S_1, S_2). If $\mathrm{shift}([i_2, j_2], [i_2', j_2']) = w \ge v$ and $|M| \ge |G| = |\aleph(S_2)|$, then M contains a substring N of length w outside $\aleph(S_2)$ and N is either a prefix or a suffix of M. Every character of N is outside $\aleph(S_2)$ and is a random character that has the probability $\frac{1}{t}$ to be equal to any character in Σ. By symmetry, we assume without loss of generality that N is a prefix of M. Then $M = NN_1$ and $M' = N'N_1'$, where N' and N have the same length and have a difference ratio at most β according to the conditions in Definition 4.3. By Lemma 5.2, the probability is at most $e^{-\frac{\epsilon^2}{3}w}$ that N' and N have the same length and have a difference of ratio at most β.

Therefore, the probability is at most $4e^{-\frac{\epsilon^2}{3}w}$ that $\mathrm{shift}([i_2, j_2], [i_2', j_2']) = w$. The probability for $\mathrm{shift}([i_2, j_2], [i_2', j_2']) \ge z$ is at most

$$4\sum_{w=z}^{\infty} e^{-\frac{\epsilon^2}{3}w} = \frac{4e^{-\frac{\epsilon^2}{3}z}}{1 - e^{-\frac{\epsilon^2}{3}}} = \frac{4c^z}{1-c}$$

(recall $c = e^{-\frac{\epsilon^2}{3}}$ from Definition 4.2). Therefore, the total probability that shift$([i_2, j_2], [i_2', j_2']) \geq z$ is at most $H_1 = \rho_0 + \frac{c^z}{1-c} \leq 2\rho_0$ by inequality (18). ◻

Lemma 5.6 will be used to give an upper bound in probability analysis. It is derived by standard methods in calculus.

LEMMA 5.6. *Assume that $0 < a < 1$ and j is a positive integer.*

(i) $\sum_{i=j}^{\infty} ia^i = \frac{ja^j - (j-1)a^{j+1}}{(1-a)^2} < \frac{ja^j}{(1-a)^2}$.

(ii) $\sum_{i=j}^{\infty} i^2a^i = a^j(\frac{(j^2-(j-1)(j+1)a)(1-a)-(j-(j-1)a)2(-a)}{(1-a)^3}) < \frac{2j^2a^j}{(1-a)^3}$.

Proof. Let θ be a negative parameter.

Statement (i). Let $f(y) = \sum_{i=j}^{\infty} e^{\theta iy}$. Therefore, we have the derivative $f'(y) = \theta \sum_{i=j}^{\infty} ie^{\theta iy}$. Alternatively, using the closed form $f(y) = \frac{e^{\theta jy}}{1-e^{\theta y}}$, we have

$$f'(y) = \frac{\theta je^{\theta jy}(1-e^{\theta y}) - e^{\theta jy}(-\theta e^{\theta y})}{(1-e^{\theta y})^2} \tag{19}$$

$$= \frac{\theta je^{\theta jy} - \theta(j-1)e^{\theta(j+1)y}}{(1-e^{\theta y})^2}. \tag{20}$$

Let $\theta = \ln a$ and $y = 1$. We have $\sum_{i=j}^{\infty} ia^i = \frac{ja^j-(j-1)a^{j+1}}{(1-a)^2} < \frac{ja^j}{(1-a)^2}$.

Statement (ii). By equality (20),

$$\begin{aligned}
f''(y) &= \left(\frac{\theta je^{\theta jy} - \theta(j-1)e^{\theta(j+1)y}}{(1-e^{\theta y})^2}\right)' \\
&= \frac{(\theta^2j^2e^{\theta jy} - \theta^2(j-1)(j+1)e^{\theta(j+1)y})(1-e^{\theta y})^2 - (\theta je^{\theta jy} - \theta(j-1)e^{\theta(j+1)y})2(-\theta e^{\theta y})(1-e^{\theta y})}{(1-e^{\theta y})^4} \\
&= \theta^2\left(\frac{(j^2e^{\theta jy} - (j-1)(j+1)e^{\theta(j+1)y})(1-e^{\theta y}) - (je^{\theta jy} - (j-1)e^{\theta(j+1)y})2(-e^{\theta y})}{(1-e^{\theta y})^3}\right) \\
&= \theta^2e^{\theta jy}\left(\frac{(j^2-(j-1)(j+1)e^{\theta y})(1-e^{\theta y}) - (j-(j-1)e^{\theta y})2(-e^{\theta y})}{(1-e^{\theta y})^3}\right).
\end{aligned}$$

We also have $f''(y) = \theta^2 \sum_{i=j}^{\infty} i^2e^{\theta iy}$. Let $\theta = \ln a$ and $y = 1$. We have

$$\begin{aligned}
\sum_{i=j}^{\infty} i^2a^i &= a^j\left(\frac{(j^2-(j-1)(j+1)a)(1-a) - (j-(j-1)a)2(-a)}{(1-a)^3}\right) \\
&< \frac{(j^2(1-a)+2ja)a^j}{(1-a)^3} \\
&< \frac{2j^2a^j}{(1-a)^3}. \qquad ◻
\end{aligned}$$

Lemma 5.8 below shows that with high probability, many prefixes and suffixes of the motif region in a $\Theta_\alpha(n, G)$-sequence do not change much. We define

$$Q_0 = (1-\alpha)^2 - \frac{2c^v}{1-c}. \tag{21}$$

The parameter Q_0 is used in Lemma 5.8 to give a lower bound on the probability that for a $\Theta_\alpha(n, G)$-sequence S, its $\aleph(S)$ will be similar enough to the original motif G according to the conditions in Lemma 5.8.

DEFINITION 5.7. *We say that a $\Theta_\alpha(n, G)$-sequence S contains a stable motif region $\aleph(S)$ if the following conditions hold:* (1) $G'[1] = G[1]$; (2) $G'[m] = G[m]$; (3) diff$(G'[1,h], G[1,h]) \leq \frac{\beta}{2}$ *for all* $h = v, v+1, \ldots, m$; *and* (4) diff$(G'[m-h, m], G[m-$

$h, m]) \le \frac{\beta}{2}$ for $h = v-1, v+1, \ldots, m-1$, where $G' = \aleph(S)$, $c = e^{-\frac{\epsilon^2}{3}}$, and $m = |G|$ as defined in Definition 4.2 *and section* 2.

LEMMA 5.8. *With probability at least Q_0, a $\Theta_\alpha(n, G)$-sequence S contains a stable motif region.*

Proof. The probability is $(1-\alpha)^2$ to satisfy conditions (1) and (2) in Definition 5.7. Consider condition (3). Since every character of $\aleph(S)[1, m]$ (notice that $m = |G|$) has probability at most α to mutate, by Corollary 2.3 the probability is at most $e^{-\frac{1}{3}\epsilon^2 r}$ that $\mathrm{diff}(G[1, h], G'[1, h]) > \frac{\beta}{2} = \alpha + \epsilon$. Let $V_3 = \sum_{r=v}^{\infty} e^{-\frac{1}{3}\epsilon^2 r} = \frac{c^v}{1-c}$, where $c = e^{-\frac{1}{3}\epsilon^2}$ as defined in Definition 4.2. Therefore, the probability is at most V_3 that $\mathrm{diff}(G[1, h], G'[1, h]) > \frac{\beta}{2} = \alpha + \epsilon$ for some $h \in \{v, v+1, \cdots, m\}$. Similarly we define $V_4 = \sum_{r=v}^{\infty} e^{-\frac{1}{3}\epsilon^2 r} \le \frac{c^v}{1-c}$. The probability is at most V_4 that $\mathrm{diff}(G[m-h, m], G'[m-h, m]) > \frac{\beta}{2} = \alpha + \epsilon$ for some $h \in \{v, v+1, \cdots, m\}$. In sum, the probability that S contains a stable motif region is at least $(1-\alpha)^2 - V_3 - V_4 = (1-\alpha)^2 - \frac{2c^v}{1-c} = Q_0$. ∎

Lemma 5.9 below gives a lower bound for the probability that $\mathrm{Extract}(S_1, S_2)$ returns the motif region $\aleph(S_2)$ of S_2, and that the motif region $\aleph(S_2)$ of S_2 does not differ much from the original motif G.

Define constants

$$c_1 = \frac{2}{(1-c)^4} \tag{22}$$

and

$$c_2 = 20c_1. \tag{23}$$

LEMMA 5.9. *Given two independently generated $\Theta_\alpha(n, G)$-sequences S_1 and S_2, the probability is at least $Q_1 = Q_0^2 - H_2 - H_1 \ge Q_0^2 - 4\rho_0$ that* $\mathrm{Extract}(S_1, S_2)$ *returns $\aleph(S_2)$, and $\aleph(S_2)$ contains a stable motif region, where H_1 is defined in Lemma* 5.5*, $H_2 = c_2 v^3(\frac{1}{t} + c^v)$, and c_2 is a constant defined in* (23).

Proof. Let M_1 be the substring of S_1 that matches the substring of M_2 of S_2, let $M_2 = \mathrm{Extract}(S_1, S_2)$, and let $h = |M_1| = |M_2|$. Assume $M_1 = S_1[i_1, j_1]$ and $M_2 = S_2[i_2, j_2]$. For two random $\Theta_\alpha(n, G)$-sequences S_1 and S_2, their motif regions $\aleph(S_1)$ and $\aleph(S_2)$ can match well with probability at least Q_0^2 by Lemma 5.8. We can assume that in the function $\mathrm{Extract}(S_1, S_2)$, we consider only the variable $h \ge |\aleph(S_1)| = |\aleph(S_2)|$. If $h < |\aleph(S_1)| = |\aleph(S_2)|$ happens, then $\aleph(S_1)$ and $\aleph(S_2)$ cannot match well.

Define $P_{L,R}(w_1, w_2, s)$ to be the probability that the following three conditions are satisfied.

Condition (1). There are w_1 characters outside $\aleph(S_1)$ on the left side of M_1. In other words, $S_1[i_1 + w_1]$ is the first character of $\aleph(S_1)$, and the w_1 characters $S_1[i_1, i_1 + w_1 - 1]$ are outside the region $\aleph(S_1)$.

Condition (2). There are w_2 characters outside $\aleph(S_2)$ in the right region of M_2. In other words, $M_2 = S_2[i_2, j_2]$, $S_1[j_2 - w_2]$ is the last character of $\aleph(S_2)$, and the w_2 characters $S_2[j_2 - w_2 + 1, j_2]$ are outside the region $\aleph(S_2)$.

Condition (3). The position of the first character of $\aleph(S_1)$ in M_1 and the position of the first character of $\aleph(S_2)$ in M_2 have shift s. In other words, if $S_1[t_1]$ is the first character of $\aleph(S_1)$ in S_1 and $S_2[t_2]$ is the first character of $\aleph(S_2)$ in S_2, then $s = |(t_1 - i_1) - (t_2 - i_2)|$. See Figure 3 for an illustration.

In a similar way, we define the probabilities $P_{L,L}(w_1, w_2, s)$, $P_{R,L}(w_1, w_2, s)$, and $P_{R,R}(w_1, w_2, s)$. In other words, for $P_{A,B}(w_1, w_2, s)$ with $A, B \in \{L, R\}$, $A = L$ (or $A = R$) represents the case that there are w_1 characters outside $\aleph(S_1)$ on the left

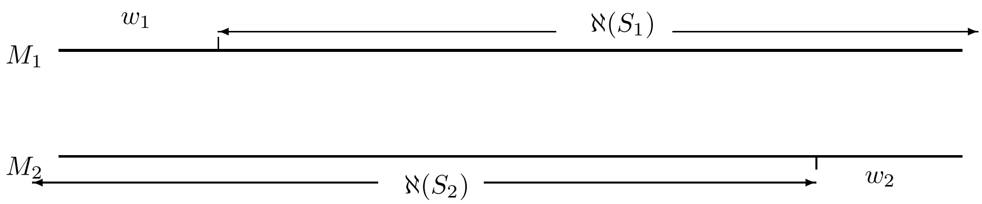

FIG. 3. M_1 *and* M_2 *for Case* 1 *of the proof of Lemma* 5.9.

(respectively, right) side of M_1, and $B = L$ (or $B = R$) represents the case that there are w_2 characters outside $\aleph(S_2)$ on the left (respectively, right) side of M_2. Furthermore, the parameter s indicates the shift defined in Condition (3) above.

By Lemma 5.5, with probability at most H_1, shift$([i_2, j_2], [i'_2, j'_2]) \geq z$, where $M = S_2[i_2, j_2]$ and $\aleph(S_2) = S_2[i'_2, j'_2]$. Therefore, the probability is at least $1-H_1$ that if $|M_2| \geq |G|$, then most parts of M_2 are in the region $\aleph(S_2)$ (recall that $z = O(1)$).

The probabilistic analysis below has 10 cases. Case $a.b$ is the bth subcase of Case a. Case $a.b.c$ is the cth subcase of Case $a.b$. We use $P_a, P_{a.b}$, and $P_{a.b.c}$ to denote the probabilities of Cases a, $a.b$, and $a.b.c$, respectively.

Case 1. $0 \leq w_2 < w_1$, M_1 has w_1 characters outside $\aleph(S_1)$ on the left side of M_1, the last character of $\aleph(S_2)$ is in M_2, and M_2 has w_2 characters outside $\aleph(S_2)$ on the right side outside M_2. See Figure 3.

We consider only the cases where $s = 1, 2, \ldots, w_1 + w_2$ and M_2 has fewer than w_1 characters outside $\aleph(S_2)$ on the left side. The cases where M_2 has at least w_1 characters outside $\aleph(S_2)$ are covered by Cases 5 and 9.

If $s > w_1 + w_2$, the matched region will be shorter than that of $\aleph(S_2)$. If $s \leq w_1$, the first character of $\aleph(S_2)$ is in M_2, and this case is included in Case 4. Therefore, we consider only the range $w_1 + 1 \leq s \leq w_1 + w_2$. For an upper bound for the probability of Case 1, we compute $P_1 = \sum_{w_2=0}^{\infty} \sum_{w_1=w_2+1}^{\infty} \sum_{s=w_1+1}^{w_1+w_2} P_{L,R}(w_1, w_2, s)$. There are some subcases.

- Case 1.1. $0 \leq w_2 < v$.
  Case 1.1.1. $0 \leq w_2 < w_1 < v$.
  By Lemma 5.2, $P_{L,R}(w_1, w_2, s) \leq \frac{1}{t}$ for fixed w_1, w_2, and s. $\sum_{s=1}^{w_1+w_2} P_{L,R}(w_1, w_2, s) \leq \frac{2v}{t}$ for a fixed w_1 and all $s = w_1 + 1, \ldots, w_1 + w_2$. Then

$$\sum_{w_1=w_2+1}^{v-1} \sum_{s=w_1+1}^{w_1+w_2} P_{L,R}(w_1, w_2, s) \leq \frac{2v^2}{t}.$$

  Case 1.1.2. $v \leq w_1$.
  By Lemma 5.2, $P_{L,R}(w_1, w_2, s) \leq e^{-\frac{\epsilon^2}{3} w_1}$ for a fixed w_1 and a fixed s. $\sum_{s=w_1+1}^{w_1+w_2} P_{L,R}(w_1, w_2, s) \leq w_2 e^{-\frac{\epsilon^2}{3} w_1}$. $\sum_{w_1=v}^{\infty} \sum_{s=w_1+1}^{w_1+w_2} P_{L,R}(w_1, w_2, s) \leq \sum_{w_1=v}^{\infty} w_2 e^{-\frac{\epsilon^2}{3} w_1} = w_2 \frac{c^v}{1-c}$.
  Combining Cases 1.1.1 and 1.1.2, we have

$$\begin{aligned} P_{1.1} &= \sum_{w_2=0}^{v-1} \sum_{w_1=w_2+1}^{\infty} \sum_{s=w_1+1}^{w_1+w_2} P_{L,R}(w_1, w_2, s) \\ &\leq \sum_{w_2=0}^{v-1} \left(\frac{2v^2}{t} + \frac{c^v w_2}{1-c} \right) < \frac{2v^3}{t} + \frac{v^2 c^v}{2(1-c)} \leq c_1 \cdot v^3 \left(\frac{1}{t} + c^v \right). \end{aligned}$$

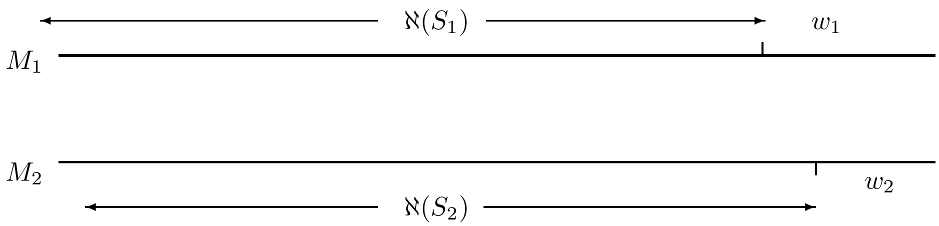

FIG. 4. *M_1 and M_2 for Case* 2 *of the proof of Lemma* 5.9.

- Case 1.2. $v \le w_2$.
  Assume that $v \le w_2 < w_1$. By Lemma 5.2, $P_{L,R}(w_1, w_2, s) \le e^{-\frac{\epsilon^2}{3}w_1}$ for fixed w_1, w_2, and s. $\sum_{s=w_1+1}^{w_1+w_2} P_{L,R}(w_1, w_2, s) \le w_2 e^{-\frac{\epsilon^2}{3}w_1}$ for fixed w_1 and w_2. $\sum_{w_1=w_2+1}^{\infty} \sum_{s=w_1+1}^{w_1+w_2} P_{L,R}(w_1, w_2, s) \le \sum_{w_1=w_2+1}^{\infty} w_2 e^{-\frac{\epsilon^2}{3}w_1} = w_2 \frac{c^{w_2+1}}{1-c}$.
  Therefore, by Lemma 5.6, $P_{1.2} = \sum_{w_2=v}^{\infty} \sum_{w_1=w_2+1}^{\infty} \sum_{s=w_1+1}^{w_1+w_2} P_{L,R}(w_1, w_2, s)$ $\le \sum_{w_2=v}^{\infty} w_2 \frac{c^{w_2+1}}{1-c} \le \frac{vc^{v+1}}{(1-c)^3} \le c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Therefore, we have derived a probability bound for Case 1: $P_1 = P_{1.1} + P_{1.2} \le 2c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Case 2. $0 \le w_2 < w_1$, the last character of $\aleph(S_2)$ is in M_2, M_2 has w_2 characters outside $\aleph(S_2)$ in the right side of the matched region, and M_1 has w_1 characters outside $\aleph(S_1)$ in the right side of the matched region. See Figure 4.

For a probability upper bound of Case 2, we compute

$$P_2 = \sum_{w_2=1}^{\infty} \sum_{w_1=w_2+1}^{\infty} P_{R,R}(w_1, w_2, w_1 - w_2).$$

There are some subcases.

- Case 2.1. $0 \le w_2 < v$.
  Case 2.1.1. $0 \le w_2 < w_1 < v$.
  By Lemma 5.2, $P_{R,R}(w_1, w_2, s) \le \frac{1}{t}$ for fixed w_1, w_2 and $s = w_1 - w_2$. The total probability of Case 2.1.1 for a fixed w_2 and all w_1 with $w_2 < w_1 \le v$ is at most $\frac{(v-w_2-1)}{t}$.
  Case 2.1.2. $v \le w_1$.
  By Lemma 5.2, $P_{R,R}(w_1, w_2, s) \le e^{-\frac{\epsilon^2}{3}w_1}$ for a fixed w_1 and a fixed $s = w_1 - w_2$. We have $\sum_{w_1=v}^{\infty} P_{R,R}(w_1, w_2, s) = \sum_{w_1=v}^{\infty} e^{-\frac{\epsilon^2}{3}w_1} = \frac{c^v}{1-c}$ for all $w_1 \ge v$.
  Therefore, $P_{2.1} = \sum_{w_2=0}^{v-1} \sum_{w_1=w_2+1}^{\infty} P_{R,R}(w_1, w_2, w_1 - w_2) \le \sum_{w_2=1}^{v-1} (\frac{(v-w_2-1)}{t}$ $+ \frac{c^v}{1-c}) < \frac{v^2}{2t} + \frac{vc^v}{1-c} \le c_1 \cdot v^3(\frac{1}{t} + c^v)$.
- Case 2.2. $v \le w_2$.
  Assume $v \le w_2 < w_1$. By Lemma 5.2, for a fixed w_1 and a fixed $s = w_1 - w_2$, the probability for Case 2.2 is at most $P_{R,R}(w_1, w_2, s) \le e^{-\frac{\epsilon^2}{3}w_1}$. The probability for Case 2.2 for all $w_1 > w_2$ is at most $\sum_{w_1=w_2+1}^{\infty} e^{-\frac{\epsilon^2}{3}w_1} = \frac{c^{w_2+1}}{1-c}$.
  Therefore, $P_{2.2} = \sum_{w_2=v}^{\infty} \sum_{w_1=w_2+1}^{\infty} P_{R,R}(w_1, w_2, w_1 - w_2) \le \sum_{w_2=v}^{\infty} (\frac{c^{w_2+1}}{1-c}) <$ $\frac{c^{v+1}}{(1-c)^2} \le c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Therefore, the probability for Case 2 is upper bounded as $P_2 = P_{2.1} + P_{2.2} \le 2c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Case 3. $0 \le w_2 < w_1$, M_1 has w_1 characters outside $\aleph(S_1)$ on the right side of M_1, the last character of $\aleph(S_2)$ is in M_2, and M_2 has w_2 characters outside $\aleph(S_2)$ on the left side of M_2.

For a probability upper bound of Case 3, we compute

$$P_3 = \sum_{w_2=0}^{\infty} \sum_{w_1=w_2+1}^{\infty} \sum_{s=1}^{\infty} P_{R,L}(w_1, w_2, s).$$

This case has the same analysis and probability as Case 1. Therefore, we have $P_3 = P_1 \le 2c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Case 4. $0 \le w_2 < w_1$, S_1 has w_1 characters outside $\aleph(S_1)$ on the left side of M_1, and S_2 has w_2 characters outside $\aleph(S_2)$ on the left side M_2.

For a probability upper bound of Case 4, we compute

$$P_4 = \sum_{w_2=1}^{\infty} \sum_{w_1=w_2+1}^{\infty} P_{L,L}(w_1, w_2, w_1 - w_2).$$

This case has the same analysis and probability as Case 2. Therefore, we have $P_4 = P_2 \le 2c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Case 5. $1 \le w_1 = w_2$, and the left sides of both M_1 and M_2 have the same number w_2 of characters outside $\aleph(S_1)$ and $\aleph(S_2)$, respectively.

For a probability upper bound of Case 5, we compute $P_5 = \sum_{w_2=1}^{\infty} P_{L,L}(w_2, w_2, 0)$. There are two subcases.

- Case 5.1. $1 \le w_2 < v$. By Lemma 5.2, the probability for this case is at most $\frac{1}{t}$ for a fixed w_2. The total probability for this case for $1 \le w_2 < v$ is $P_{5.1} \le \frac{v}{t}$.
- Case 5.2. $v \le w_2$. By Lemma 5.2, the probability for this case is at most $e^{-\frac{\epsilon^2}{3} w_2}$ for a fixed w_2. The total probability for this case for $v \le w_2$ is $P_{5.2} \le \sum_{w_2=v}^{\infty} e^{-\frac{\epsilon^2}{3} w_2} = \frac{c^v}{1-c}$.

The total probability bound for Case 5 is upper bounded as $P_5 = P_{5.1} + P_{5.2} \le \frac{v}{t} + \frac{c^v}{1-c} \le c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Case 6. $1 \le w_1 = w_2$, and the right sides of both M_1 and M_2 have the same number w_2 of characters outside $\aleph(S_1)$ and $\aleph(S_2)$, respectively.

For the probability upper bound of Case 6, we compute $P_6 = \sum_{w_2=1}^{\infty} P_{R,R}(w_2, w_2, 0)$. This case has the same analysis and probability as Case 5. Therefore, $P_6 = P_5 \le c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Case 7. $0 \le w_1 < w_2$, the first character of $\aleph(S_1)$ is in M_1, M_1 has w_1 characters outside $\aleph(S_1)$ on the left side of M_1, and M_2 has w_2 characters outside $\aleph(S_2)$ on the right side of M_2.

We have only $s = 1, 2, \ldots, w_1 + w_2$. For a probability upper bound of Case 7, we compute $\sum_{w_2=0}^{\infty} \sum_{w_1=1}^{w_2-1} \sum_{s=1}^{\infty} P_{L,R}(w_1, w_2, s)$.

- Case 7.1. $0 \le w_2 < v$.
  By Lemma 5.2, $P_{L,R}(w_1, w_2, s) \le \frac{1}{t}$ for fixed w_1, w_2, and s. The total probability for this case for all w_1 with $0 \le w_1 \le w_2$ is at most $\frac{w_1+w_2}{t}$. The probability is at most $\frac{2w_2^2}{t}$ for all w_1 with $0 \le w_1 < w_2$. Therefore, $P_{7.1} \le \sum_{w_2=0}^{v-1} \sum_{w_1=0}^{w_2-1} \sum_{s=1}^{w_1+w_2} P_{L,R}(w_1, w_2, s) \le \sum_{w_2=0}^{v-1} (\frac{2w_2^2}{t}) < \frac{2v^3}{t} \le c_1 \cdot v^3(\frac{1}{t} + c^v)$.
- Case 7.2. $v \le w_2$.
  By Lemma 5.2, $P_{L,R}(w_1, w_2, s) \le e^{-\frac{\epsilon^2}{3} w_2}$ for a fixed w_2 and a fixed s. The total probability for this case for fixed w_1 and w_2, and variable $s = 1, \ldots, w_1+w_2$

is $\sum_{s=1}^{w_1+w_2} P_{L,R}(w_1, w_2 s) \le (w_1 + w_2)e^{-\frac{\epsilon^2}{3}w_2}$. The total probability for this case for all $w_2 \ge v$ is at most $P_7 = \sum_{w_1=0}^{w_2-1}(w_1+w_2)e^{-\frac{\epsilon^2}{3}w_2} = \frac{2w_2^2 c^{w_2}}{1-c}$. Therefore, $P_{7.2} \le \sum_{w_2=v}^{\infty}\sum_{w_1=0}^{w_2-1}\sum_{s=1}^{w_1+w_2} P_{L,R}(w_1, w_2, s) \le \sum_{w_2=v}^{\infty}(\frac{2w_2^2 c^{w_2}}{1-c}) < \frac{2}{(1-c)}\frac{2v^2c^v}{(1-c)^3} = \frac{4v^2c^v}{(1-c)^4} \le c_1 \cdot v^3(\frac{1}{t} + c^v)$ (by Lemma 5.6).

In summary, the probability for Case 7 is upper bounded as $P_7 = P_{7.1} + P_{7.2} \le 2c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Case 8. $0 \le w_1 < w_2$, the last character of $\aleph(S_1)$ is in M_1, there are w_1 characters on the right side of M_1 and outside its $\aleph(S_1)$, and M_2 has w_2 characters on the right side of M_2 and outside its $\aleph(S_2)$.

For a probability upper bound of Case 8, we compute

$$P_8 = \sum_{w_2=1}^{\infty}\sum_{w_1=0}^{w_2-1} P_{R,R}(w_1, w_2, w_2 - w_1).$$

There are two subcases.

- Case 8.1. $0 < w_2 < v$.
  By Lemma 5.2, the probability for this case for a fixed w_1 and a fixed $s = w_2 - w_1$ is at most $\frac{1}{t}$. The total probability for this case for all w_1 with $0 \le w_1 \le w_2 - 1$ is at most $\frac{w_2}{t}$. Therefore, $P_{8.1} = \sum_{w_2=1}^{v-1}\sum_{w_1=0}^{w_2-1} P_{R,R}(w_1, w_2, w_2 - w_1) \le \sum_{w_2=1}^{v-1}\frac{w_2}{t} = \frac{v^2}{t} \le c_1 \cdot v^3(\frac{1}{t} + c^v)$.
- Case 8.2. $v \le w_2$.
  By Lemma 5.2, $P_{L,R}(w_1, w_2, s) \le e^{-\frac{\epsilon^2}{3}w_2}$ for a fixed w_2 and a fixed $s = w_2 - w_1$. The total probability for this case for all $0 \le w_1 \le w_2 - 1$ is at most $\sum_{w_1=0}^{w_2-1} e^{-\frac{\epsilon^2}{3}w_2} = \frac{w_2 c^{w_2}}{1-c}$. Therefore, $P_{8.2} = \sum_{w_2=v}^{\infty}\sum_{w_1=0}^{w_2-1} P_{R,R}(w_1, w_2, w_2 - w_1) \le \sum_{w_2=v}^{\infty}\frac{w_2c^{w_2}}{1-c} \le \frac{vc^v}{(1-c)^3} \le c_1 \cdot v^3(\frac{1}{t} + c^v)$ by Lemma 5.6.

We have $P_8 = P_{8.1} + P_{8.2} = O(v^2(\frac{1}{t} + c^v)) \le 2c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Case 9. $0 \le w_1 < w_2$, the last character of $\aleph(S_1)$ is in M_1, S_1 has w_1 characters outside $\aleph(S_1)$ on the right side of M_1, and S_2 has w_2 characters outside $\aleph(S_2)$ on the left side outside M_2.

For a probability upper bound of Case 9, we compute

$$P_9 = \sum_{w_2=1}^{\infty}\sum_{w_1=0}^{w_2-1}\sum_{s=1}^{\infty} P_{R,L}(w_1, w_2, s).$$

This case has the same analysis and probability as Case 7. Therefore, we have $P_9 = P_7 \le 2c_1 \cdot v^3(\frac{1}{t} + c^v)$.

Case 10. $0 \le w_1 < w_2$, the first character of $\aleph(S_1)$ is in M_1, M_1 has w_1 characters on the left side of M_1 outside $\aleph(S_1)$, and M_2 has w_2 characters on the left side of M_2 outside $\aleph(S_2)$.

For a probability upper bound of Case 10, we compute

$$P_{10} = \sum_{w_2=1}^{\infty}\sum_{w_1=0}^{w_2-1} P_{L,L}(w_1, w_2, w_2 - w_1).$$

This case has the same analysis and probability as Case 8. Therefore, we have $P_{10} = P_8 = O(v^2(\frac{1}{t} + c^v)) \le 2c_1 \cdot v^3(\frac{1}{t} + c^v)$.

The total probability for $w_2 \geq 1$ is at most $H_2 = \sum_{i=1}^{10} P_i \leq c_2(v^3(\frac{1}{t} + c^v))$. Therefore, we have a constant $c_2 > 0$ such that $H_2 \leq c_2(v^3(\frac{1}{t} + c^v)) \leq 2\rho_0$ by inequalities (8) and (15). Lemma 5.9 follows from the fact that S_2 has probability at least Q_0^2 for $\aleph(S_2)$ to match $\aleph(S_1)$ well, and probability $H_2 + H_1 \leq 4\rho_0$ for a bad match (i.e., $M_2 \neq \aleph(S_2)$). ∎

LEMMA 5.10. *With probability at most $\frac{\eta}{2^x}$, the set W outputted by Phase-One does not contain G_0 such that $G_0 = \text{Extract}(S_i', S_i'') = \aleph(S_i'')$ and S_i'' contains a stable motif region (see Definition 5.7).*

Proof. By inequality (7) and equality (21), we have $Q_0 \geq \frac{1}{2}$. By Lemma 5.9 and $\rho_0 = \frac{1}{24}$ defined in section 4.1, we have $Q_1 \geq Q_0^2 - 4\rho_0 \geq \frac{1}{12}$. Since the number k_1 is selected to be large enough that $(1-Q_1)^{k_1} \leq \frac{\eta}{2^x}$ (see inequality (11)), the probability is at most $(1 - Q_1)^{k_1} \leq \frac{\eta}{2^x}$ (by Lemma 5.9) such that there is no i with $1 \leq i \leq k_1$ such that both S_i' and S_i'' have stable motif regions and $\text{Extract}(S_i', S_i'')$ returns $\aleph(S_i'')$. ∎

DEFINITION 5.11. *Assume that $\aleph(S_i'')$ is a stable motif region as described in Lemma 5.10 and that $\text{Extract}(S_i', S_i'') = \aleph(S_i'')$ for some $1 \leq i \leq k_1$. Define $G_0 = \text{Extract}(S_i', S_i'') = \aleph(S_i'')$.*

5.3. Analysis of Phase-Two of Algorithm Find-Noisy-Motif. Lemma 5.12 below shows that with small probability, the input Z_1 (which is the set of sequences used to form the sequence pairs of Z_0 and is generated by LoadInputSequence) contains a sequence whose motif region has many mutations.

LEMMA 5.12. *With probability at most $2k_1 e^{-\frac{\epsilon^2}{3}d}$, at least one sequence S in Z_1 mutates at more than $\frac{\beta}{2}|G|$ characters in its motif region $\aleph(S)$.*

Proof. Every character in the $\aleph(S)$ region has probability at most α to mutate. Recall that $|\aleph(S)| = |G| \geq d$. By Corollary 2.3, with probability at most $e^{-\frac{\epsilon^2}{3}|G|} \leq e^{-\frac{\epsilon^2}{3}d}$, a sequence S in Z_1 has more than $(\alpha+\epsilon)|G| = \frac{\beta}{2}|G|$ mutations (recall the setting for β in Definition 4.3). Since there are $2k_1$ sequences in Z_1, the total probability is at most $2k_1 e^{-\frac{\epsilon^2}{3}d}$ that at least one sequence S in Z_1 mutates at more than $\frac{\beta}{2}|G|$ characters in its motif region $\aleph(S)$. ∎

Lemma 5.13 below shows that with high probability, Phase-Two of Algorithm Find-Noisy-Motif extracts the correct motif regions from the sequences in Z_1.

Let $R = 2(\frac{v-1}{t} + \frac{c^v}{1-c})$ as defined in Lemma 5.13. Combining inequalities (7), (9), (14), equation (21), and the definition of R, we have

$$Q_0 - \alpha - 3\epsilon - 2R > \frac{1}{2}. \tag{24}$$

Recall that Phase-One uses $\text{Extract}(S_i', S_i'')$ to obtain a motif candidate G''. Then Phase-Two uses G'' to match with $\aleph(S)$ in another sequences S. The parameter R is used as a small probability that the matched region between G'' and S is not in $\aleph(S)$. See Lemma 5.13 for more details.

LEMMA 5.13.

(i) *Assume that $G'' = \text{Extract}(S_i', S_i'')$ with $|G| \leq |G''|$. Let S be a $\Theta_\alpha(n, G)$-sequence with $M = \text{Match}(G'', S)$, and let w_0 be the number of characters of M that are not in the region $\aleph(S)$. Then the probability is at most $R = 2(\frac{v-1}{t} + \frac{c^v}{1-c})$ that $w_0 \geq 1$.*

(ii) *The probability is at least $Q_0 - R$ that, given a $\Theta_\alpha(n, G)$-sequence S, $\aleph(S) = \text{Match}(G_0, S)$.*

FIG. 5. *G'' and M for Lemma* 5.13.

Proof. Assume $w_0 \geq 1$. Let w be the number of characters outside $\aleph(S)$ on the left side of M, and let w' be the number of characters outside $\aleph(S)$ on the right side of M. Clearly, $w_0 = w + w'$. Since $w_0 \geq 1$, either $w \geq 1$ or $w' \geq 1$. See Figure 5. Without loss of generality, we assume $w \geq 1$.

Statement (i). There are two cases.

Case (a). $1 \leq w < v$. By Lemma 5.2, the probability for this case for a fixed w is at most $\frac{1}{t}$. Thus, the total probability for this case is at most $\frac{(v-1)}{t}$.

Case (b). $v \leq w$. By Lemma 5.2, the probability for this case for a fixed w is at most $e^{-\frac{\epsilon^2}{3}w}$. The total probability for this case is at most $\sum_{w=v}^{\infty} e^{-\frac{\epsilon^2}{3}w} = \frac{c^v}{1-c}$.

The probability analysis is similar when $w' \geq 1$. Therefore, the probability for $w_0 \geq 1$ is at most $R = 2(\frac{v-1}{t} + \frac{c^v}{1-c})$.

Statement (ii). By Lemma 5.8, with probability at least Q_0, S contains a stable motif region. By statement (i) of this lemma, we have probability at least $Q_0 - R$ that, given a random $\Theta_\alpha(n, G)$-sequence, S, $\aleph(S) = \mathrm{Match}(G_0, S)$. $\square$

Lemma 5.14 below shows that we can use G' to extract most of the motif regions for the sequences in Z_2 if $G' = G_0$ (recall that G_0 is close to the original motif G as defined in Definition 5.11).

LEMMA 5.14. *Assume that* $|G'| \geq |G|$ *and* $G_i = \mathrm{Match}(G', S_i)$ *for* $S_i \in Z_2 = \{S_1, \cdots, S_{k_2}\}$ *and* $i = 1, \ldots, k_2$ *(recall that each sequence* G_i *is either an empty sequence or a sequence of length* $|G'|$*).*

(i) *If* $G' = G_0$*, then the probability is at least* $1 - e^{-\frac{\epsilon^2 k_2}{3}}$ *that there are more than* $(Q_0 - R - \epsilon)k_2$ *sequences* G_i *with* $G_i = \aleph(S_i)$.

(ii) *The probability is at least* $1 - e^{-\frac{\epsilon^2 k_2}{3}}$ *that for every* G', $|\{i \mid G_i \neq \aleph(S_i), i = 1, \ldots, k_2\}| \leq (R + \epsilon)k_2$.

Proof. Recall that sequence G_0 is selected according to Definition 5.11. When G' is fixed, $G_i = \aleph(S_i) = \mathrm{Match}(G', S_i)$ and $G_j = \aleph(S_j) = \mathrm{Match}(G', S_j)$ are two independent events due to the independence of S_i and S_j. Thus, we can apply Chernoff bounds in the proof below.

Statement (i). By Lemma 5.13, for every $S_i \in Z_2$, the probability is at least $Q_0 - R$ that $G_i = \aleph(S_i)$. By Corollary 2.3, the probability is at most $e^{-\frac{\epsilon^2 k_2}{3}}$ that there are fewer than $(Q_0 - R - \epsilon)k_2$ sequences G_i with $G_i = \aleph(S_i)$.

Statement (ii). By Lemma 5.13, the probability is at most R that $G_i \neq \aleph(S_i)$. By Corollary 2.3, with probability at most $e^{\frac{\epsilon^2 k_2}{3}}$, $|\{i \mid G_i \neq \aleph(S_i), i = 1, \ldots, k_2\}| > (R + \epsilon)k_2$. $\square$

5.4. Proof of the main theorem. We now give the proof of Theorem 4.1.

Proof. The parameters $\alpha, \delta_0, \delta_1$, and t_0 are set as before. By Lemma 5.9, with probability at least Q_1, a pair (S', S'') from Z_0 gives that $\text{Extract}(S', S'') = \aleph(S'')$ and that $\aleph(S'')$ satisfies the conditions of Definition 5.7. The probability is at most $(1-Q_1)^{k_1} \le \frac{\eta}{2^x}$ (by inequality (11)) that the following statement (a) is false.

Statement (a). There exist sequences $(S', S'') \in Z_0$ such that $\text{Extract}(S', S'') = \aleph(S'')$ and S'' contains a stable motif region (see Definition 5.7 and Lemma 5.8).

As we select G_0 according to Definition 5.11, $G_0 = \text{Extract}(S', S'') = \aleph(S'')$ (we use the pair (S', S'') to represent the pair (S_1, S_2) of Z_0 right after Lemma 5.9). By Lemma 5.14, the probability is at most $e^{-\frac{\epsilon^2}{3}k_2} \le \frac{\eta}{2^x}$ (by inequality (12)) that the following statement (b) is false.

Statement (b). $|\{i \mid \text{Match}(G_0, S_i) = \aleph(S_i) \text{ for } S_i \in Z_2 = \{S_1, \cdots, S_{k_2}\}\}| \ge (Q_0 - R - \epsilon)k_2$.

Suppose G'' is one of the sequences in W returned by Phase-One of Algorithm Find-Noisy-Motif. If $|G''| > |G|$, then $M = \text{Extract}(G'', S)$ has $w_0 \ge 1$ (see Lemma 5.13). By Lemma 5.13, the probability is at most R that $\text{Extract}(G'', S)$ is not empty. By Corollary 2.3, the probability is at most $e^{-\frac{\epsilon^2}{3}k_2}$ that $|\{i \mid \text{Match}(G'', S_i) \ne \emptyset$ for $S_i \in Z_2\}| \ge (R+\epsilon)k_2$. Since Phase-One of Algorithm Find-Noisy-Motif returns at most k_1 sequences in W (because Z_0 has only k_1 pairs), the probability is at most $k_1 e^{-\frac{\epsilon^2}{3}k_2} \le \frac{\eta}{2^x}$ (by inequality (13)) that the following statement (c) is false.

Statement (c). $|\{i \mid \text{Match}(G'', S_i) \ne \emptyset \text{ for } S_i \in Z_2 = \{S_1, \cdots, S_{k_2}\}\}| \le (R+\epsilon)k_2$ for every $G'' \in W$ with $|G''| > |G|$.

Let G_1 be any of the longest sequences returned by $\text{Extract}(S_1', S_2')$ such that $|\{i \mid \text{Match}(G_1, S_i) \ne \emptyset \text{ for } S_i \in Z_2 = \{S_1, \cdots, S_{k_2}\}\}| \ge (Q_0 - R - \epsilon)k_2 > (R+\epsilon)k_2$ (by inequality (24)). If statements (a), (b), and (c) are all true, then $|G_1| = |G|$. By Lemma 5.14, the probability is at most $e^{-\frac{\epsilon^2 k_2}{3}}$ that $|\{i \mid \text{Match}(G_1, S_i) = \aleph(S_i)$ for $S_i \in Z_2 = \{S_1, \cdots, S_{k_2}\}\}| < (Q_0 - R - \epsilon - R - \epsilon)k_2 = (Q_0 - 2R - 2\epsilon)k_2$. Therefore, the probability is at most $k_1 e^{-\frac{\epsilon^2 k_2}{3}} \le \frac{\eta}{2^x}$ that the following statement (d) is false.

Statement (d). $|\{i \mid \text{Match}(G_1, S_i) \ne \aleph(S_i) \text{ for } S_i \in Z_2 = \{S_1, \cdots, S_{k_2}\}\}| \ge (Q_0 - 2R - 2\epsilon)k_2$ for every longest G_1 that satisfies $|\{i \mid \text{Match}(G_1, S_i) \ne \emptyset \text{ for } S_i \in Z_2 = \{S_1, \cdots, S_{k_2}\}\}| \ge (Q_0 - R - \epsilon)k_2$.

For a fixed j with $1 \le j \le |G_1| = |G|$ and k_2 sequences in Z_2, by Corollary 2.3, with probability at most $e^{-\frac{\epsilon^2 k_2}{3}}$, there are more than $(\alpha+\epsilon)k_2$ mutated characters $\aleph(S_i)[j]$ $(i = 1, \ldots, k_2)$. Thus, the probability is at most $n_0 e^{-\frac{\epsilon^2 k_2}{3}} \le \frac{\eta}{2^x}$ (by inequality (12)) that the following statement (e) is false.

Statement (e). For every j with $1 \le j \le |G_1|$, $|\{i \mid \aleph(S_i)[j] \ne G[i] \text{ for } S_i \in Z_2 = \{S_1, \cdots, S_{k_2}\}\}| \le (\alpha+\epsilon)k_2$.

We have probability at most $5 \cdot \frac{\eta}{2^x} \le \frac{1}{2^x}$ that at least one of statements (a)–(e) does not hold. In other words, we have probability at least $1 - \frac{1}{2^x}$ that statements (a)–(e) are all true. Now we assume that statements (a)–(e) all hold.

Therefore, by inequality (24), $(Q_0 - 2R - 2\epsilon - \alpha - \epsilon)k_2 > \frac{k_2}{2}$. For each j with $1 \le j \le |G_1| = |G|$, we have $|\{i \mid \text{Extract}(G_1, S_i)[j] = G[j] \text{ for } S_i \in Z_2 = \{S_1, \cdots, S_{k_2}\}\}| \ge (Q_0 - 2R - 2\epsilon - \alpha - \epsilon)k_2 > \frac{k_2}{2}$. Therefore, G can be recovered by voting.

The running time of Phase-One is $O(k_1 n^3)$, and the running time of Phase-Two is $O(k_1 k_2 n^2)$. The total time complexity for Find-Noisy-Motif is $O(n^3)$ since k_1 is constant for some fixed x and $k_2 = O(\log n)$. ∎

Since the length upper bound of motif G is no more than the length of an input sequence ($n_0 \le n$), we have the following simplified result, which does not involve n_0.

THEOREM 5.15. *Assume that the mutation probability upper bound α is less than 0.1771. There exist constants t_0, δ_0, and δ_1 such that if the size t of the alphabet Σ is at least t_0 and the length of the motif G is at least $\delta_0 \log n$, then, given k $\Theta_\alpha(n, G)$-sequences with $k \ge \delta_1 \log n$, Algorithm Find-Noisy-Motif outputs G with probability at least $1 - \frac{1}{2^x}$ and runs in $O(n^3)$ time.*

Proof. Set $n_0 = n$ and apply Theorem 4.1. ☐

6. Lower bounds on the parameters of motif discovery. In this section, we show some lower bounds for the length of the motif and the number of input sequences that are needed to recover the motif with high probability. Theorems 6.1 and 6.2 together show that the requirements for the motif length and the number of input sequences for Find-Noisy-Motif in the main theorem, Theorem 4.1, are optimal to within a multiplicative constant factor.

6.1. Lower bound for the motif length. Theorem 6.1 shows that when the motif is short relative to the lengths of input sequences, it is impossible to recover the motif with a small number $O(\log n)$ of sequences.

THEOREM 6.1. *Assume that constant $\epsilon > 0$ and the alphabet has constant number t characters. There is a constant $\delta > 0$ such that with probability at least $1 - o(1)$, given $n^{1-\epsilon}$ input $\Theta_\alpha(n, G)$-sequences $S_1, \cdots, S_{n^{1-\epsilon}}$, every sequence of length $m_0 = \lceil \delta \log n \rceil$ is a substring of each S_i for $i = 1, 2, \ldots, n^{1-\epsilon}$.*

Proof. We assume that n is sufficiently large. Assume that the length of the motif is $m_0 = \lceil \delta \log n \rceil \le 2\delta \log n$ for a small constant $\delta > 0$ such that $\delta \log t < \epsilon/8$. Thus, $t^{m_0} \le 2^{(\log t)(2\delta \log n)} < 2^{\frac{\epsilon}{4} \log n}$. Let S be a $\Theta_\alpha(n, G)$-sequence. We partition a substring S' of length $n/3$ of S such that S' does not intersect the motif region $\aleph(S)$ into $n' = \lfloor \frac{\frac{n}{3}}{m_0} \rfloor > \frac{n}{6\delta \log n}$ blocks of size m_0 each. The probability that a pattern G' of length m_0 does not occur in these n' blocks is $(1 - \frac{1}{t^{m_0}})^{n'} < 2^{\frac{-n'}{t^{m_0}}} < 2^{-\frac{n}{6\delta \log n} \frac{1}{2^{\frac{\epsilon}{4} \log n}}} < 2^{-\frac{n}{2^{\frac{\epsilon}{3} \log n}}} \le 2^{-n^{1-\frac{\epsilon}{3}}}$ for large n. The probability that at least one of those t^{m_0} patterns does not occur in S is at most $t^{m_0}(1 - \frac{1}{t^{m_0}})^{n'} < 2^{\frac{\epsilon}{4} \log n} 2^{-n^{1-\frac{\epsilon}{3}}} < 2^{-n^{1-\frac{\epsilon}{2}}}$.

The probability is at least $1 - 2^{-n^{1-\frac{\epsilon}{2}}}$ that the above sequence S' contains all the sequences of length m_0 as its sequences. If the number of input sequences is $k = n^{1-\epsilon}$, then the probability is at least $(1 - 2^{-n^{1-\frac{\epsilon}{2}}})^k = 1 - o(1)$ that each of the k input sequences of length n contains all sequences of length m_0 as its substrings. ☐

6.2. Lower bound for sample complexity. We consider the lower bound for the number of sequences needed for recovering the motif. Theorem 6.2 shows that if the number of input sequences is $o(\log n)$, then it is impossible to recover the motif correctly.

THEOREM 6.2. *There exists a constant δ such that no algorithm can recover the exact motif G with at most $\delta \log n$ $\Theta_\alpha(n, G)$-sequences.*

Proof. Assume that the motif region occupies each input sequence entirely. Thus, every character in the input sequence has probability α to mutate. We assume that α is a positive constant. Assume that we have k sequences $S_1, \cdots, S_k$. For a fixed i with $1 \le i \le n$, the probability that all the characters $S_1[i], \cdots, S_k[i]$ mutate is α^k. Therefore, the probability is $1 - (1 - \alpha^k)^n$ that for some i with $1 \le i \le n$, all the ith characters $S_1[i], \cdots, S_k[i]$ mutate. Note that $1 - (1 - \alpha^k)^n$ is very close to 1 when $k < \delta \log n$. When there is an i such that all characters $S_1[i], \cdots, S_k[i]$ mutate, it is impossible to recover the ith character of the motif. ☐

7. Conclusions. We have proved that if the mutation probability upper bound α is less than 0.1771, there exist constants $t_0 > 0$, $\delta_0 > 0$, and $\delta_1 > 0$ such that if the length of the motif is $n_0 > \delta_0 \log n$ and the alphabet has at least t_0 characters, then there exists an $O(n^3)$-time algorithm that, given at least $\delta_1 \log n_0$ input sequences, can find the motif with high probability, where n is the longest length of any input sequence. Very recently, we have also shown [4] that for any alphabet Σ with $|\Sigma| \geq 2$, for every motif $G \in \Sigma^\rho - \Psi_{\rho,h,\epsilon}(\Sigma)$, where $\Psi_{\rho,h,\epsilon}(\Sigma)$ is a small subset of Σ^ρ with $\frac{|\Psi_{\rho,h,\epsilon}(\Sigma)|}{|\Sigma^\rho|} \leq 2^{-\Theta(\epsilon^2 h)}$, if G has length at least $c_0 \log n$, it can be recovered with $O(n \log n)$ sequences with high probability. This second algorithm is applicable to DNA motif discovery. An interesting open problem is whether there exists an algorithm to recover all the motifs for an alphabet of four characters.

Acknowledgments. We thank Miklós Csürös and Manan Sanghi for helpful discussions. We are also grateful to Andrew Wei Xu, David Sankoff, and anonymous referees for providing comments to help us improve the presentation of this paper.

REFERENCES

[1] F. Chin and H. Leung, *Voting algorithms for discovering long motifs*, in Proceedings of the 3rd Asia-Pacific Bioinformatics Conference, Singapore, 2005, Imperial College Press, London, 2005, pp. 261–271.

[2] J. Dopazo, A. Rodríguez, J. C. Sáiz, and F. Sobrino, *Design of primers for PCR amplification of highly variable genomes*, Comput. Appl. Biosci., 9 (1993), pp. 123–125.

[3] M. Frances and A. Litman, *On covering problems of codes*, Theory Comput. Syst., 30 (1997), pp. 113–119.

[4] B. Fu, M.-Y. Kao, and L. Wang, *Discovering almost any hidden motif from multiple sequences in polynomial time with low sample complexity and high success probability*, in Proceedings of the Sixth Annual Conference on Theory and Applications of Models of Computation, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 231–240.

[5] L. Gąsieniec, J. Jansson, and A. Lingas, *Efficient approximation algorithms for the Hamming center problem*, in Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, 1999, pp. S905–S906.

[6] D. Gusfield, *Algorithms on Strings, Trees, and Sequences*, Cambridge University Press, Cambridge, UK, 1997.

[7] G. Hertz and G. Stormo, *Identification of consensus patterns in unaligned DNA and protein sequences: A large-deviation statistical basis for penalizing gaps*, in Proceedings of the Third International Conference on Bioinformatics and Genome Research, World Scientific, Singapore, 1995, pp. 201–216.

[8] U. Keich and P. Pevzner, *Finding motifs in the twilight zone*, Bioinformatics, 18 (2002), pp. 1374–1381.

[9] U. Keich and P. Pevzner, *Subtle motifs: Defining the limits of motif finding algorithms*, Bioinformatics, 18 (2002), pp. 1382–1390.

[10] J. K. Lanctot, M. Li, B. Ma, L. Wang, and L. Zhang, *Distinguishing string selection problems*, Inform. Comput., 185 (2003), pp. 41–55.

[11] C. Lawrence and A. Reilly, *An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences*, Proteins, 7 (1990), pp. 41–51.

[12] M. Li, B. Ma, and L. Wang, *Finding similar regions in many strings*, in Proceedings of the 31st Annual ACM Symposium on Theory of Computing, 1999, pp. 473–482.

[13] M. Li, B. Ma, and L. Wang, *On the closest string and substring problems*, J. ACM, 49 (2002), pp. 157–171.

[14] K. Lucas, M. Busch, S. Mossinger, and J. Thompson, *An improved microcomputer program for finding gene- or gene family-specific oligonucleotides suitable as primers for polymerase chain reactions or as probes*, Comput. Appl. Biosci., 7 (1991), pp. 525–529.

[15] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, Cambridge, UK, 1995.

[16] P. Pevzner and S. Sze, *Combinatorial approaches to finding subtle signals in DNA sequences*, in Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, AAAI Press, Menlo Park, CA, 2000, pp. 269–278.

[17] V. Proutski and E. C. Holme, *Primer master: A new program for the design and analysis of PCR primers*, Comput. Appl. Biosci., 12 (1996), pp. 253–255.

[18] G. Stormo, *Consensus patterns in DNA*, in Molecular Evolution: Computer Analysis of Protein and Nucleic Acid Sequences, Methods Enzymol. 183, R. F. Doolittle, ed., Academic Press, New York, 1990, pp. 211–221.

[19] G. Stormo and G. Hartzell III, *Identifying protein-binding sites from unaligned DNA fragments*, Proc. Natl. Acad. Sci. USA, 88 (1991), pp. 5699–5703.

[20] L. Wang and L. Dong, *Randomized algorithms for motif detection*, J. Bioinform. Comput. Biol., 3 (2005), pp. 1039–1052.