Efficient Algorithms for Model-Based Motif Discovery from Multiple Sequences

Bin Fu[1], Ming-Yang Kao[2], and Lusheng Wang[3]

[1] Dept. of Computer Science, University of Texas - Pan American
TX 78539, USA
binfu@cs.panam.edu

[2] Department of Electrical Engineering and Computer Science,
Northwestern University, Evanston, IL 60208, USA
kao@northwestern.edu

[3] Department of Computer Science, The City University of Hong Kong,
Kowloon, Hong Kong
lwang@cs.cityu.edu.hk

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 $G = g_1 g_2 \ldots g_m$ is a string of m characters. Each background sequence is implanted a randomly generated approximate copy of G. For a randomly generated approximate copy $b_1 b_2 \ldots b_m$ of G, every character is randomly 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 for finding subtle and faint motifs.

1 Introduction

Motif discovery is an important problem in computational biology and computer science. For instance, it has applications to coding theory [3,4], locating binding sites and conserved regions in unaligned sequences [18,10,6,17], genetic drug target identification [9], designing genetic probes [9], and universal PCR primer design [13,2,16,9].

This paper focuses on the application of motif discovery to finding conserved regions in a set of given 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 subset 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 distance at most r from a set of 0-1 sequence of length n. The decision problem associated with the covering radius for a set of binary sequences is NP-complete [3]. Another similar problem called closest string problem was also proved to be NP-hard [3,9]. Some approximation algorithms have also been proposed. Li et al. [12] gave an approximation scheme for the closest string and substring problems. The related consensus patterns problem is that give n sequences $s_1, \cdots, s_n$, it asks for a region of length L in each

M. Agrawal et al. (Eds.): TAMC 2008, LNCS 4978, pp. 234–245, 2008.

s_i, and a median 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 also reported in [11]. Furthermore, a number of heuristics and programs have been developed [15,7,8,19,1].

In many applications, motifs are faint and may not be apparent when two sequences alone are compared but may become clearer when more sequences are together [5]. For this reason, it has been conjectured that comparing more sequences together can help identifying faint motifs. In this paper, we give the first analytical proof for this conjecture.

In this paper, 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 $G = g_1g_2\ldots g_m$ is a string of m characters. Each background sequence is implanted a randomly generated approximate copy of G. For a randomly generated approximate copy $b_1b_2\ldots b_m$ of G, every character is randomly generated such that the probability for $b_i \neq g_i$ is at most α. This model was first proposed in [15] and has been widely used in experimentally testing motif discovery programs [7,8,19,1].

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 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 $t_0, \delta_0, \delta_1 > 0$ do not depend on x. We also show some lower bounds that imply 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 and is independent of the number of the input sequences. This is because that 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 an exact algorithm that has provable high probability to return the motif. The algorithm employs novel methods that extract 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. The algorithm allows the motif to appear any position at each sequence, and each mutation in a motif to be arbitrary (a mutation lets a character to be changed to an arbitrary character without any probabilistic condition). We also derive lower bounds that indicate the upper bounds are almost optimal.

We give a brief description about the algorithm as section 3. Before giving the algorithm, we set up a few parameters and constants that will affect the algorithm at section 4.1. Then entire Algorithm Find-Noisy-Motif is described.

We give the analysis and proof about Algorithm Find-Noisy-Motif and state it in our main theorem (Theorem 1). Two lower bounds are presented at section 5.

2 Notations

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 has 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 equal to 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 $b_1 \cdots b_m$ of S may start at an 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 only subject to the restriction that g_i will mutate with probability at most α.

A mutation converts a character g_i in the motif into an arbitrary different character b_i without probability restriction. This allows a character g_i in the motif to change into any character b_i in $\Sigma - \{g_i\}$ with even different probability.

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

Definition 1. *Assume that $S = a_1 a_2 \cdots a_n$ is a sequence. For its substring $S' = S[i_1, j_1]$ and $S'' = S[i_2, j_2]$, define* $\text{shift}_S(S', S'') = \min(|i_1 - i_2|, |j_1 - j_2|)$.

The analysis of our algorithm employs the well known Chernoff bound [14].

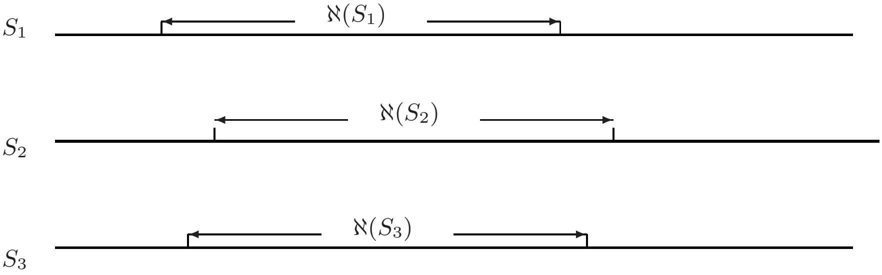

Fig. 1. The motif regions of S_1, S_2 and S_3 are not aligned

3 A Sketch of the Algorithm Find-Noisy-Motif

Our Algorithm Find-Noisy-Motif has two phases. The first phase exploits the fact that with high probability, the motif area in some sequences conserves the first and last characters. Furthermore, the middle area of the motif changes with a small ratio. We will select enough pairs of $\Theta_\alpha(n, G)$-sequences S', S'' and find their substrings G' and G'' of S' and S'', respectively such that G' and G'' match in their left and right most characters. Furthermore, G' and G'' only have a relatively small difference in the middle area. 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 $i = 1, 2, \cdots, k_2$. Assume that $G_1, \cdots, G_{k_2}$ are derived from matching G'' to all sequences $S_1, S_2, \cdots, S_{k_2}$. Some G_i may be an empty sequence if G'' can not 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 non-empty sequences among $G_1, \cdots, G_{k_2}$ is much larger than $\frac{k_2}{2}$ and the i-th character $G[i]$ of G can be recovered from voting among $G_1[i], \cdots, G_{k_2}[i]$. 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 conversion from figure 1 to figure 2 shows how we recover the motif via voting.

On the other hand, if $|G''| > |G|$ or G'' does not match G well, we can prove that the number of non-empty sequences among $G_1, \cdots, G_{k_2}$ is less than $\frac{k_2}{2}$. Our algorithm's time complexity depends on the length of the input sequences and is independent of the number of the input sequences. This is because that for a fixed x, $\Theta(\log n)$ sequences are sufficient to guarantee the probability of at least $1 - \frac{1}{2^x}$ the motif will be discovered. Additional sequences can improve the probability but are not needed for the high probability guarantee.

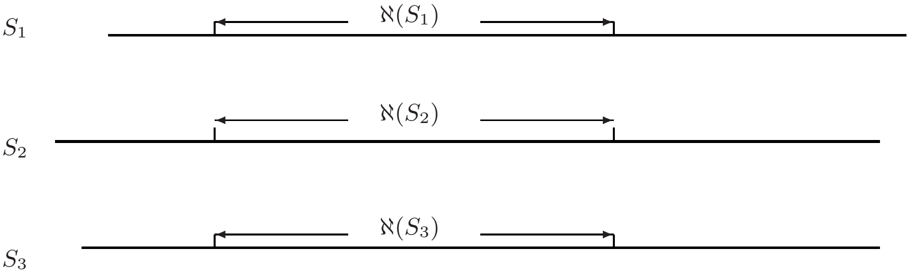

Fig. 2. S_1, S_2 and S_3 with their motif in the same column region

4 Algorithm Find-Noisy-Motif

In this section, we give an algorithm that any motif G can be discovered in $O(n^3)$ time. It requires that the size of alphabet is larger than a fixed constant.

Some parameters and constants will be used in Algorithm Find-Noisy-Motif . In section 4.1, we give a list of assignments for some parameters and constants that are used in the algorithm. The description of Algorithm Find-Noisy-Motif is given at section 4.2. The analysis of the algorithm is given at section 4.3.

4.1 Parameters

As multiple parameters affect the performance of Algorithm Find-Noisy-Motif, we list the parameters and discuss some useful inequalities here.

- Let x be any constant at least 8. We will prove that Algorithm Find-Noisy-Motif has probability at least $1 - \frac{1}{2^x}$ to output the motif G. Let α be any constant with $\alpha < 0.1771$. Note that $(1-\alpha)^2 - \alpha > \frac{1}{2}$. Let $\eta = \frac{1}{6}$ and let $\rho_0 = \frac{1}{24}$. Let $\epsilon > 0$ be any constant such that $(1-\alpha)^2 - \alpha - 3\epsilon > \frac{1}{2}$.
- Select any constant $r_0 > 0$ such that

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

- Let v be the least integer that satisfies the inequalities below:

$$1 \le v, \tag{2}$$

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

$$\frac{2c_2 v^3 c^v}{1-c} < \rho_0, \tag{4}$$

$$\frac{2c^v}{1-c} < \frac{r_0}{2}, \tag{5}$$

$$\frac{2c^v}{1-c} < \rho_0, \tag{6}$$

  where $c = e^{-\frac{\epsilon^2}{3}}$. Note that the existence of v for (3) follows from (1).
- We define the following Q_0. It will be first used in Lemma 1. Let $Q_0 = (1-\alpha)^2 - \frac{2c^v}{1-c}$.
- Let c_2 be a constant to be specified in Lemma 6.
- Let t_0 be any constant such that

$$\frac{2(v-1)}{t_0} \le \frac{r_0}{2}, \tag{7}$$

$$\frac{c_2 v^3}{t_0} \le \rho_0, \tag{8}$$

$$\frac{t_0 - 1}{t_0} - \beta > \epsilon, \tag{9}$$

  where R is to be defined in Lemma 8. In the remainder of this paper, we always assume the parameter $t \ge t_0$. Combining (3), (5), (7) and the definition of R, we have $Q_0 - \alpha - 3\epsilon - 2R > \frac{1}{2}$.

- Let $\beta = 2\alpha + 2\epsilon$.
- The constant z is selected so that $z \geq v$, and $\frac{4e^{-\frac{\epsilon^2}{3}z}}{1-e^{-\frac{\epsilon^2}{3}}} \leq \rho_0$.
- The number k_1 is selected such that

$$(1 - Q_1)^{k_1} \leq \frac{\eta}{2^x}, \tag{10}$$

  where Q_1 is defined in Lemma 6, and is at least $\frac{1}{12}$. Note that $k_1 = O(1)$ is a constant independent of the length of the input sequences.
- Select a constant $\delta_0 > 0$ and let $d = \delta_0 \log n$ such that $n^2 e^{-d} \leq \frac{\eta}{2^x}$, $n^2 e^{-\frac{\epsilon^2}{3}d} \leq \rho_0$.
- We require that the length of the motif G is at least d. Let n be the largest length of an input $\Theta_\alpha(n, G)$-sequence. 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 .
- Select a constant $\delta_1 > 0$ and let $k_2 = \delta_1 \log n_0 - 2k_1$ so that $n_0 k_2 e^{-\frac{\epsilon^2}{3}k_2} \leq \frac{\eta}{2^x}$, and $k_1 e^{-\frac{\epsilon^2}{3}k_2} \leq \frac{\eta}{2^x}$.

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

Let's imagine how a sequence S is generated in this model. 1). Generate a sequence S' with $n - |G|$ characters, in which each character is a random character Σ. 2). Generate G' such that with probability at most α, $G'[i] \neq G[i]$. For $G'[i] \neq G[i]$, it represents a mutation. Note that there is no restriction about how a character will change to in a mutation. 3). Insert G', which servers the motif region $\aleph(S)$ of S, into any 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 $\Theta_\alpha(n, G)$-sequences $\{S_1', S_1'', \cdots, S_{k_1}', S_{k_1}''\}$ in the k_1 pairs of sequences in Z_0, where k_1 is defined by inequality (10). Let Z_2 be a set of k_2 sequences 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 Algorithm Find-Noisy-Motif . In the remainder of this paper, we assume that the alphabet has $t \geq t_0$ characters.

4.2 Description of Algorithm Find-Noisy-Motif

Algorithm Find-Noisy-Motif has two phases. The input to Phase 1 is k_1 pairs of $\Theta_\alpha(n, G)$-sequences in the set Z_0. The input to Phase 2 is k_2 $\Theta_\alpha(n, G)$-sequences in the set Z_2 and the output result from Phase 1. All the $\Theta_\alpha(n, G)$-sequences are independent random $\Theta_\alpha(n, G)$-sequences. Note 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 according to the setting in Section 4.1. Algorithm Find-Noisy-Motif is a deterministic algorithm, which is based on the randomness of those sequences in both Z_0 and Z_2 and the independence in selecting them. Algorithm Find-Noisy-Motif is deterministic, but its input is generated by a probabilistic model. The following steps generate data sequenced for Algorithm Find-Noisy-Motif for Z_0 and Z_2.

Step 1. Randomly select $2k_1$ $\Theta_\alpha(n,G)$-sequences $S_1', S_1'', S_2', S_2'', \cdots, S_{k_1}', S_{k_1}''$ and let $Z_0 = \{(S_1', S_1''), (S_2', S_2''), \cdots, (S_{k_1}', S_{k_1}'')\}$.

Step 2. Randomly select k_2 $\Theta_\alpha(n,G)$-sequences $S_1, \cdots, S_{k_2}$ and let $Z_2 = \{S_1, \cdots, S_{k_2}\}$.

Definition 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) $\mathrm{diff}(X_1[1,i], X_2[1,i]) \le \beta$ for all integers i, $v \le i \le |X_1|$.*

- *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 $X = a_1 \cdots a_n$.*
- *Two sequences X_1 and X_2 are matched if X_1 and X_2 are both left and right matched.*

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 subsequence of $S_2$ which is similar to a subsequence of $S_1$.
Steps:
    for $h = \min(|S_1|, |S_2|)$ to $d$ (recall from Section 4.1 that $|G| \ge d$)
        for $i = 1$ to $|S_1|$
            for $j = 1$ to $|S_2|$
                let $i' = i + h - 1$ and $j' = j + h - 1$;
                if $S_1[i, i']$ and $S_2[j, j']$ are both left and right matched (see
                    Definition 2)
                then return $S_2[j, j']$;
    return $\emptyset$ (the empty sequence);
End of Extract
```

The following are the steps of Phase 1 of Algorithm Find-Noisy-Motif :

```
Phase 1:
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 in the initial stage of the algorithm.
Output: a set $W$ that contains a similar region of each pair in $Z_0$.
Steps:
    let $W = \emptyset$ (empty set);
    for each pair of sequence $(S, S') \in Z_0$
        let $G' = $ Extract$(S, S')$ and put $G'$ into $W$;
    return $W$, which will be used in Phase 2;
End of Phase 1
```

After a set of motif candidates W is produced from Phase 1 of Algorithm Find-Noisy-Motif, we use this set to match with another set of sequences to recover the hidden motif via voting.

```
Function Match$(G', S_i)$
Input: a motif candidate $G'$, which is returned from the function Extract(), and a sequence $S$ from the group $Z_2$;
Output: either a subsequence $G_i$ of $S_i$ of the same length as $G'$ or an empty sequence. $G_i$ will be considered 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 $G_i$ of $S_i$ with $|G| = |G_i|$ such that
        $G'$ and $G_i$ are matched (see Definition1)
    if such a $G_i$ does not exist, let $G_i = \emptyset$ (empty string).
    Output $G_i$;
End of Match
```

The function Vote$(G_1, G_2, \cdots, G_{k'})$ is to generate another sequence G' via 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 $k' \leq k_2$;
Output: a sequence $G'$, which is derived from voting at every position of the input sequences.
Steps:
    let $m = |G_1|$;
    for each $j = 1, \cdots, m$
        if strictly more than $\frac{k_2}{2}$ characters from $G_1[j], \cdots, G_{k'}[j]$ are equal
            to some character $a$
        then let $a_j = a$
        else return "failure";
    return $G' = a_1 \cdots a_m$;
End of Vote
```

The following are the steps of Phase 2 of Algorithm Find-Noisy-Motif . It uses the candidates of motif derived in the Phase 1 to extract the motif regions of another set Z_2 of sequences, and recover the motif via voting.

```
Phase 2:
    let $Z_2 = \{S_1, \cdots, S_{k_2}\}$ as defined in the begining of Section 4.2.
    for each $G' \in W$, let $G_i = \text{Match}(G', S_i)$ for $i = 1, \cdots, k_2$.
        let $G'_1, \cdots, G'_{k'_2}$ be the list of all non-empty sequences in the list
        $G_1, \cdots, G_{k_2}$ (Note: For every non-empty sequence that appears
        multiple times in the second list, it also appears the same number
        of times in the first list.)
        If $k'_2 \geq (Q_0 - 2R - 2\epsilon)k_2$
            then output Vote$(G'_1, G'_2, \cdots, G'_{k'_2})$ (which will be proven to be
        identical to $G$ with probability at least $1 - \frac{1}{2^x}$).
End of Phase 2
```

4.3 Analysis of Phase 1 of Algorithm Find-Noisy-Motif

We present Lemma 1 that shows that with high probability, the initial part and last part of motif region in a $\Theta_\alpha(n, G)$-sequence do not change much.

Lemma 1. *With probability at least $Q_0 = (1-\alpha)^2 - \frac{2c^v}{1-c}$, a $\Theta_\alpha(n, G)$-sequence S contains $G' = \aleph(S)$ satisfying the following conditions: (1) $G'[1] = G[1]$; (2) $G'[m] = G[m]$; (3) $\mathrm{diff}(G'[1, h], G[1, h]) \leq \frac{\beta}{2}$ for all $h = v, v+1, \cdots, m$; (4) $\mathrm{diff}(G'[m-h, m], G[m-h, m]) \leq \frac{\beta}{2}$ for $h = v-1, v+1, \cdots, m-1$, where $c = e^{-\frac{\epsilon^2}{3}}$ and $m = |G|$ as defined in Sections 4.1 and 2, respectively.*

Lemma 2 shows that with small probability, a sequence can match a random sequence. It will be used to prove that when two subsequences in two different $\Theta_\alpha(n, G)$-sequences are similar, they are unlikely to stay away the motif regions in the two $\Theta_\alpha(n, G)$-sequences, respectively.

Lemma 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*

1. *if $1 \leq |X_1| = |X_2| < v$, then the probability that X_1 and X_2 are matched is $\leq \frac{1}{t}$; and*
2. *if $v \leq |X_1| = |X_2|$, then the probability for $\mathrm{diff}(X_1, X_2) \leq \beta$ is at most $e^{-\frac{\epsilon^2 |X_1|}{3}}$.*

Function Extract(S_1, S_2) returns a subsequence of S_2. We expect that Extract($S_1, S_2)$ is the motif region $\aleph(S_2)$ in S_2. Lemma 3 shows that with small probability, the region for Extract(S_1, S_2) in S_2 has no overlap with the motif region $\aleph(S_2)$ of S_2.

Lemma 3. *With probability at most ρ_0,* Extract(S_1, S_2) *and $\aleph(S_2)$ are not overlaping substrings of S_2. In other words, with probability is at most ρ_0,* Extract$(S_1, S_2) = S_2[j, j']$, *$\aleph(S_2) = S_2[t, t']$, and the two intervals $[j, j']$ and $[f, f']$ have no overlap ($[j, j'] \cap [f, f'] = \emptyset$).*

In order to show that Extract(S_1, S_2) is efficient to find a motif region in S_2, we give Lemma 4 show that with small probability, the region to fetch Extract(S_1, S_2) in S_2 shift much from the motif region $\aleph(S_2)$ of S_2.

Lemma 4. *For every $z > 0$, the probability is at most $H_1 = 2\rho_0$ that for a pair of sequences (S_1, S_2) from Z_0, $\mathrm{shift}_{S_2}(M, \aleph(S_2)) \geq z$ and $|M| \geq |G|$, where $M = \mathrm{Extract}(S_1, S_2)$.*

We need the Lemma 5, which will be useful to give the upper bound of probability analysis. It is derived by the standard methods in calculus.

Lemma 5. *Let a be a real constant in interval $(0, 1)$ and j be an integer ≥ 1. Then, 1. $\sum_{i=j}^{\infty} i a^i = \frac{j a^j - (j-1) a^{j+1}}{(1-a)^2} < \frac{j a^j}{(1-a)^2}$; and*

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

Lemma 6 gives a lower bound for the probability that $\mathrm{Extract}(S_1, S_2)$ returns the motif region $\aleph(S_2)$ of S_2. Furthermore, the motif region $\aleph(S_2)$ of S_2 does not have much difference with the original motif G.

Lemma 6. *Given two independent $\Theta_\alpha(n, G)$-sequences S_1 and S_2, it has the probability at least $Q_1 = Q_0^2 - H_2 - H_1 \geq Q_0^2 - 4\rho_0$ that $G' = \mathrm{Extract}(S_1, S_2)$ is $\aleph(S_2)$, and $\aleph(S_2)$ satisfies the conditions of G' Lemma 1, where H_1 is defined in Lemma 4, $H_2 = c_2 v^3(\frac{1}{t} + c^v)$ and $c_2 = O(1)$ is a constant.*

By (3), we have $Q_0 \geq \frac{1}{2}$. By Lemma 6 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 (10)), the probability is at least $1 - (1 - Q_1)^{k_1} \geq 1 - \frac{\eta}{2^x}$ (by Lemma 6) that there is $G_0 = \mathrm{Extract}(S_1, S_2) = \aleph(S_2)$, where S_1 and S_2 satisfy the conditions of Lemma 1. We now assume there is such a G_0 that satisfies the conditions described above.

4.4 Analysis of Phase 2 of Algorithm Find-Noisy-Motif

Lemma 7 shows that with small probability, Z_1 generated in the initial stage (step 2) of Algorithm Find-Noisy-Motif has a sequence whose motif region has many mutations.

Lemma 7. *With probability at most $2k_1 e^{-\frac{\epsilon^2}{3}d}$, there is a sequence S in Z_1 that changes more than $\frac{\beta}{2}|G|$ characters in its motif region $\aleph(S)$.*

Lemma 8 shows that with high probability, phase 2 of Algorithm Find-Noisy-Motif extracts motif regions from the sequences in Z_1.

Lemma 8. *1. Assume that $G'' = \mathrm{Extract}(S_i', S_i'')$ with $|G| \leq |G''|$. Let S be a $\Theta_\alpha(n, G)$-sequence with $M = \mathrm{Match}(G'', S)$ and let w_0 be the number of characters of M that are not in the region of $\aleph(S)$. Then the probability is at most $R = 2(\frac{v-1}{t} + \frac{c^v}{1-c})$ that $w_0 \geq 1$.*

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

Lemma 9 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 and G_0 is defined right after Lemma 6).

Lemma 9. *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 $i = 1, \cdots, k_2$ (Recall that each sequence G_i is either an empty sequence or a sequence of the length $|G'|$).*

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

Theorem 1 (Main). *Assume that α is a constant 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 independent $\Theta_\alpha(n, G)$-sequences with $k \geq \delta_1 \log n_0$, Algorithm Find-Noisy-Motif outputs G with probability $\geq 1 - \frac{1}{2^x}$ 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.*

5 Lower Bounds on the Parameters

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.

Theorem 2 shows that when the motif is short, it is impossible to recover it with a small number $O(\log n)$ of sequences. Thus, the upper bounds of Algorithm Find-Noisy-Motif and the lower bounds here have constant factor multiplicative.

Theorem 2. *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 $1 - o(1)$ that given $n^{1-\epsilon}$ independent random $\Theta_\alpha(n, G)$-sequences $S_1, \cdots, S_{n^{1-\epsilon}}$, every sequence of length $m_0 = \lceil \delta \log n \rceil$ is a substrings of each S_i for $i = 1, 2, \cdots, n^{1-\epsilon}$.*

We consider the lower bound for the number of sequences needed for recovering the motif. Theorem 3 shows that if the number of sequences is $o(\log n)$, it is impossible to recover the motif correctly.

Theorem 3. *There exists a constant δ such that no algorithm can recover the motif G with at most $\delta \log n$ $\Theta_\alpha(n, G)$-sequences.*

Open Problems: An interesting open problem is whether there exists an algorithm to recover all the motifs for the alphabet with four characters.

Acknowledgements. We thank Miklós Csürös and Manan Sanghi for helpful discussions. Ming-Yang Kao is supported in part by National Science Foundation Grant CNS-0627751. Lusheng Wang is fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CityU 1196/03E).

References

1. Chin, F., Leung, H.: Voting algorithms for discovering long motifs. In: Proceedings of the 3rd Asia-Pacific Bioinformatics Conference, pp. 261–272 (2005)
2. Dopazo, J., Rodríguez, A., Sáiz, J.C., Sobrino, F.: Design of primers for PCR amplification of highly variable genomes. Computer Applications in the Biosciences 9, 123–125 (1993)
3. Frances, M., Litman, A.: On covering problems of codes. Theoretical Computer Science 30, 113–119 (1997)

4. Gąsieniec, L., Jansson, J., Lingas, A.: Efficient approximation algorithms for the Hamming center problem. In: Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. S905–S906 (1999)
5. Gusfield, D.: Algorithms on Strings, Trees, and Sequences. Cambridge University Press, Cambridge (1997)
6. Hertz, G., Stormo, G.: Identification of consensus patterns in unaligned DNA and protein sequences: a large-deviation statistical basis for penalizing gaps. In: Proceedings of the 3rd International Conference on Bioinformatics and Genome Research, pp. 201–216 (1995)
7. Keich, U., Pevzner, P.: Finding motifs in the twilight zone. Bioinformatics 18, 1374–1381 (2002)
8. Keich, U., Pevzner, P.: Subtle motifs: defining the limits of motif finding algorithms. Bioinformatics 18, 1382–1390 (2002)
9. Lanctot, J.K., Li, M., Ma, B., Wang, L., Zhang, L.: Distinguishing string selection problems. In: Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 633–642 (1999)
10. Lawrence, C., Reilly, A.: An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. Proteins 7, 41–51 (1990)
11. Li, M., Ma, B., Wang, L.: Finding similar regions in many strings. In: Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing, pp. 473–482 (1999)
12. Li, M., Ma, B., Wang, L.: On the closest string and substring problems. Journal of the ACM 49(2), 157–171 (2002)
13. Lucas, K., Busch, M., Mossinger, S., Thompson, J.: An improved microcomputer program for finding gene- or gene family-specific oligonucleotides suitable as primers for polymerase chain reactions or as probes. Computer Applications in the Biosciences 7, 525–529 (1991)
14. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambridge University Press, Cambridge (2000)
15. Pevzner, P., Sze, S.: Combinatorial approaches to finding subtle signals in DNA sequences. In: Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology, pp. 269–278 (2000)
16. Proutski, V., Holme, E.C.: Primer master: a new program for the design and analysis of PCR primers. Computer Applications in the Biosciences 12, 253–255 (1996)
17. Stormo, G.: Consensus patterns in DNA. In: Doolitle, R.F. (ed.) Molecular evolution: computer analysis of protein and nucleic acid sequences. Methods in Enzymolog, 183, 211–221 (1990)
18. Stormo, G., Hartzell III, G.: Identifying protein-binding sites from unaligned DNA fragments. Proceedings of the National Academy of Sciences of the United States of America 88, 5699–5703 (1991)
19. Wang, L., Dong, L.: Randomized algorithms for motif detection. Journal of Bioinformatics and Computational Biology 3(5), 1039–1052 (2005)