8-2013

Probabilistic Model and Algorithms Design for Motif Detection

Yuan Xue
University of Texas-Pan American

PROBABILISTIC MODEL AND ALGORITHMS DESIGN FOR MOTIF DETECTION

A Thesis

by

YUAN XUE

Submitted to the Graduate School of the
University of Texas-Pan American
In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2013

Major Subject: Computer Science

PROBABILISTIC MODEL AND ALGORITHMS DESIGN FOR MOTIF DETECTION

A Thesis
by
YUAN XUE

COMMITTEE MEMBERS

Dr.Bin Fu
Chair of Committee

Dr.Zhixiang Chen
Committee Member

Dr.Artem Chebotko
Committee Member

August 2013

ABSTRACT

Yuan Xue, Probabilistic Model and Algorithms Design For Motif Detection. Master of Science (MS), Aug, 2013, 89pp., 5 tables, 1 figure, references 22 titles.

In this thesis, a natural probabilistic model has been used to test the quality of motif discovery programs. In this model, there are k background sequences, in which each character is a random character from Σ. Motif is a string $G = g_1 g_2 \dots g_m$. Each background sequence is implanted a probabilistically generated approximate copy of G. For each copy $b_1 b_2 \dots b_m$ of G, every character b_i is probabilistically generated such that the probability for $b_i \neq g_i$ is at most α. Based on this model, two randomized algorithms ,one deterministic algorithm and one enumerative algorithm are designed, which can handle any motif patterns, and run much faster than those before, one can even run in sublinear time. These methods have been implemented in software.

ACKNOWLEDGMENTS

I would like to express the deepest appreciation to my advisor Prof.Bin Fu for the continuous support of my Master study and research, for his patience, encouragement, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

Special thanks go to rest of my thesis committee: Prof.Zhixiang Chen, Prof.Artem Chebotko, for their encouragement and insightful comments.

Finally, I would like to express my gratitude to all those who helped me during the writing of this thesis.

TABLE OF CONTENTS

Page

ABSTRACT..... iii

ACKNOWLEDGEMENTS..... iv

TABLE OF CONTENTS..... v

LIST OF TABLES..... viii

LIST OF FIGURES..... ix

CHAPTER I. INTRODUCTION..... 1

CHAPTER II. NOTATIONS AND THE MODEL OF SEQUENCE GENERATION..... 5

CHAPTER III. RANDOMIZED ALGORITHMS FOR MOTIF DETECTION..... 7

BRIEF INTRODUCTION TO ALGORITHM..... 7

ALGORITHM..... 7

AN EXAMPLE..... 8

OUR RESULTS..... 13

ALGORITHM RECOVER-MOTIF..... 16

SOME PARAMETERS..... 16

DESCRIPTION OF ALGORITHM RECOVER-MOTIF..... 20

BOUNDARY-PHASE OF ALGORITHM RECOVER-MOTIF..... 21

EXTRACT-PHASE OF ALGORITHM RECOVER-MOTIF..... 27

VOTING-PHASE..... 29

ENTIRE ALGORITHM RECOVER-MOTIF..... 30
DETERMINISTIC ALGORITHM..... 31
ANALYSIS OF ALGORITHM..... 34
REVIEW OF SOME CLASSICAL RESULTS IN PROBABILITY..... 34
ANALYSIS OF BOUNDARY-PHASE OF ALGORITHM RECOVER-MOTIF..... 36
ANALYSIS OF EXTRACT-PHASE AND VOTING-PHASE OF
ALGORITHM RECOVER-MOTIF..... 42
RANDOMIZED ALGORITHMS FOR MOTIF DETECTION..... 51
SUBLINEAR TIME ALGORITHM FOR $\frac{1}{(\log n)^{2+\mu}}$ MUTATION RATE..... 53
RANDOMIZED ALGORITHM FOR $\Omega(1)$ MUTATION RATE..... 58
DETERMINISTIC ALGORITHM FOR $\Omega(1)$ MUTATION RATE..... 62
SUMMARY..... 64
CHAPTER IV. ENUMERATIVE ALGORITHMS FOR MOTIF DETECTION..... 65
AN OVERVIEW OF ENUMERATIVE ALGORITHM..... 65
ENUMERATIVE ALGORITHM..... 67
PARAMETER SETTING..... 67
INITIAL MOTIF REGION..... 68
EXACT BOUNDARIES AND VOTING..... 69
COMBINING TOGETHER..... 70
ANALYSIS OF ALGORITHM..... 71
SUMMARY..... 80
CHAPTER V. EXPERIMENTS AND RESULTS..... 81
EXPERIMENTS ON SIMULATED DATA SETS..... 81

EXPERIMENTS ON BIOLOGICAL DATA SETS..... 82

CHAPTER VI. CONCLUSIONS AND FUTURE WORKS..... 85

REFERENCES..... 86

BIOGRAPHICAL SKETCH..... 89

LIST OF TABLES

Page

Table 1: RESULTS OF RANDOMIZED ALGORITHM ON SIMULATED DATA..... 82

Table 2: RESULTS OF ENUMERATIVE ALGORITHM ON SIMULATED DATA..... 82

Table 3: NUMBER OF SEQUENCES AND MOTIF LENGTH..... 83

Table 4: TOTAL NUMBER OF MISMATCH POSITIONS..... 84

Table 5: AVERAGE MISMATCH NUMBERS PER SEQUENCE..... 84

LIST OF FIGURES

Page

FIGURE 1: G'' and M 42

CHAPTER I

INTRODUCTION

Motif discovery is an important problem in computational biology and computer science. For instance, it has applications in coding theory [3, 6], locating binding sites and conserved regions in unaligned sequences [8, 12, 20, 21], genetic drug target identification [11], designing genetic probes [11], and universal PCR primer design [2, 11, 16, 19].

This thesis focuses on the application of motif discovery to find 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 string in C. The decision problem associated with the covering radius for a set of binary sequences is NP-complete [3]. The similar closest string and substring problems were proved to be NP-hard [3, 11]. Some approximation algorithms have been proposed. Li et al. [14] gave an approximation scheme for the closest string and substring problems. The related consensus patterns problem is that 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 [13]. Furthermore, a number of heuristics and programs have been developed [1, 9, 10, 18, 22].

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 compared together [7]. For this reason, it has been conjectured that comparing more sequences together can help with identifying faint motifs. This thesis gives 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 $G = g_1 g_2 \dots g_m$ is a string of m characters. Each background sequence is implanted a probabilistically generated approximate copy of G. For a probabilistically generated approximate copy $b_1 b_2 \dots b_m$ 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 [18] and has been widely used in experimentally testing motif discovery programs [1, 9, 10, 22]. We note that a mutation in our model converts a character g_i in the motif into a different character b_i without probability restriction. This means that a character g_i in the motif may not become any character b_i in $\Sigma - \{g_i\}$ with equal probability.

We develop four algorithms that under the probabilistic model, one can find the implanted motif with high probability via a tradeoff between computational time and the probability of mutation. For the first three algorithms, each algorithm has a preprocessing phase and the voting phase. We use a pair of functions $(t_1(n,k), t_2(n,k))$ to describe the computational complexity of motif detection algorithm, where n is the largest length of input sequence, and k is the number of sequences. Function $t_1(n,k)$ is the time complexity for the part for preprocessing, and $t_2(n,k)$ is the time complexity for recovering one character for motif after preprocessing. The total time is $O(t_1(n,k) + t_2(n,k)|G|)$.

(1) There exists a randomized algorithm such that there are positive constants c_0 and c_1

that if the alphabet size is at least 4, the number of sequences is at least $c_1 \log n$, the motif length is at least $c_0 \log n$, and each character in motif region has probability at most $\frac{1}{(\log n)^{2+\mu}}$ of mutation for some fixed $\mu > 0$, then motif can be recovered with probability at least $\frac{3}{4}$ in $(O(\frac{n}{\sqrt{h}}(\log n)^{\frac{7}{2}} + h^2 \log^2 n), O(\log n))$ time, where n is the longest length of any input sequences, and $h = \min(|G|, n^{\frac{2}{5}})$. The algorithm total time is sublinear if the motif length $|G|$ is in the range $[(\log n)^{7+\mu}, \frac{n}{(\log n)^{1+\mu}}]$. This is the first sublinear time algorithm with rigorous analysis in this model.

(2) There exists a randomized algorithm such that there are positive constants c_0, c_1, and α that if the alphabet size is at least 4, the number of sequences is at least $c_1 \log n$, the motif length is at least $c_0 \log n$, and each character in motif region has probability at most α of mutation, then motif can be recovered with probability at least $\frac{3}{4}$ in $(O(\frac{n^2}{|G|}(\log n)^{O(1)}), O(\log n))$ time.

(3) There exists a deterministic algorithm such that there are positive constants c_0, c_1, and α that if the alphabet size is at least 4, the number of sequences is at least $c_1 \log n$, the motif length is at least $c_0 \log n$, and each character in motif region has probability at most α of mutation, then motif can be recovered with probability at least $\frac{3}{4}$ in $(O(n^2(\log n)^{O(1)}), O(\log n))$ time.

(4) We also design an enumerative algorithm for motif detection. There exists constants $\sigma_1, \delta_1, \delta_2, \epsilon > 0$ such that given an input of $\Theta_\alpha(n, G)$-sequences $S_1, \cdots, S_k$, if the alphabet size is at least 2, and each character in motif region has probability at most α of mutation, our enumerative algorithm satisfies

i. If $G \in \Sigma^\rho - \Psi_{\rho,h,d}(\Sigma)$ and $\rho \geq h + l_0 = O(\log n)$, then with probability at most $e^{-\Omega(h)} + e^{-\Omega(k)}$, it fails to return a unique G' with $|G| = |G'|$, and G and G' are very similar.

ii. For every G, with probability at most $2c_3^k + \delta_1$, it fails to return at most $2^{O(k)}$ sequences

$G'_1, \cdots, G'_m$ such that at least one G'_i is similar to G.

iii. The algorithm takes $O(kn^2(\log n)^2)$ time.

The research in this model has been reported in [4, 5, 15]. In [4], Fu et al. developed an algorithm that needs the alphabet size to be a constant that is much larger than 4. In [5], our algorithm cannot handle all possible motif patterns. In [15], Liu et al. designed an algorithm that runs in $O(n^3)$ time and is lack of rigorous analysis about its performance. The motif recovery in this natural and simple model has not been fully understood and seems a complicated problem.

This thesis presents two new randomized algorithms, one new deterministic algorithm and one new enumerative algorithm. They make advancements in the following aspects: 1. The algorithms are much faster than those before. Our algorithms can even run in sublinear time. 2. They can handle any motif pattern. 3. The restriction for the alphabet size is small, giving them potential applications in practical problems since gene sequences have an alphabet size 4. 4. All algorithms have rigorous proofs about their performances.

The randomized algorithm for motif detection is named to be Recover-Motif(.). The entire Recover-Motif(.) is described and analyzed in CHAPTER III. The entire enumerative algorithm for motif detection is described and analyzed in CHAPTER IV. Experiments and results are given in CHAPTER V. CHAPTER VI includes conclusions and future works.

CHAPTER II

NOTATIONS 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 $\Theta(n, G, \alpha)$-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. 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. The motif region of S may start at an arbitrary position in S. Also, a mutation may convert a character g_i in the motif into an arbitrary different character b_i only subject to the restriction that g_i will mutate with probability at most α.

A $\Psi(n, G)$-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 there are at most $O(1)$ characters b_i not equal to g_i for $1 \leq i \leq m$ and each mutation occurs at a random position of G, where $m = |G|$.

$\aleph(S)$ denotes the motif region $b_1 \cdots b_m$ of S.

For two sequences $S_1 = a_1 \cdots a_m$ and $S_2 = b_1 \cdots b_m$ of the same length, let the *relative Hamming distance* $\mathrm{diff}(S_1, S_2) = \frac{|\{i|a_i \neq b_i (i=1,\cdots,m)\}|}{m}$.

Definition 1. For two intervals $[i_1, j_1]$ and $[i_2, j_2]$, define $\mathrm{shift}([i_1, j_1], [i_2, j_2]) = \min(|i_1 - i_2|, |j_1 - j_2|)$.

CHAPTER III

RANDOMIZED ALGORITHMS FOR MOTIF DETECTION

Brief Introduction to Algorithms

Every detection algorithm in this thesis has two phases. The first phase is preprocessing so that the motif regions from multiple sequences can be aligned in the same column region. The second phase is to recover the motif via voting. We use a pair of functions $(t_1(n,k), t_2(n,k))$ to describe the computational complexity of motif detection algorithm. Function $t_1(n,k)$ is the time complexity for the preprocessing phase and $t_2(n,k)$ is the time complexity for outputting one character for motif in the voting phase.

The motif G is a pattern unknown to algorithm Recover-Motif, and algorithm Recover-Motif will attempt to recover G from a series of $\Theta(n, G, \alpha)$-sequences generated by the probabilistic model.

Algorithm

The algorithm first detects a position that is close to the left motif boundary in a sequence. It finds such a position via sampling and collision between two sequences. After the rough left boundary of a sequence is found, it is used to find the rough boundaries of the rest of the sequences. Similarly, we find those right boundaries of motif among the input sequences. The exact left boundary of each motif region will be detected in the next phase via voting. Each character of the motif is recovered by voting among all the characters at the same positions in the motif regions of

input sequences. For a sequence S, a sample point is a random position i in S. For two sequences S and S' with two sample points i, and j, respectively, a rough motif boundary is detected by the similarity of $S[i, i + l]$ and $S'[j, j + l]$ for some reasonably large parameter l.

Descriptions of Algorithm

Input: $Z = Z_1 \cup Z_2$, where $Z_1 = \{S'_1, \cdots, S'_{2k_1}\}$ and $Z_2 = \{S''_1, \cdots, S''_{k_2}\}$ are two sets of input sequences.

Output: Planted motif in each sequence and consensus string

*S*tart:

Randomly select sample points from each sequence both in Z_1 and Z_2

For each pair of sequences selected from Z_1 and Z_2,

Find the rough left and rough right boundaries via the matching at sample points.

Improve rough boundaries.

If motif boundaries of each sequence in Z_2 are not empty,

Use Voting to get the planted motifs.

End of Algorithm

An Example

We provide the following example for the brief idea of our algorithm. Let the following input strings be defined as below. We assume that the original motif is **TTTTTAACGATTAGCS**. The motif part is displayed with bold font, and the mutated characters in the motif region have been marked by $*$ in their feet.

1. Input Sequences

It contains two groups $Z_1 = \{S'_1, S'_2\}$ and $Z_2 = \{S''_1, S''_2, S''_3, S''_4, S''_5\}$.

Z_1 :

$$S_1' = GTACCATGGA\mathbf{TTA}_*\mathbf{TTAACGATTAGCS}TAGAGGACCTA.$$

$$S_2' = AATCCTTA\mathbf{C}_*\mathbf{TTTTAACGATTAGCS}GTC.$$

The above two strings are used to detect the initial motif region and use them to deal with the motif in the second group below.

Z_2 :

$$S_1'' = ATTCGATCCAG\mathbf{TTTTTAACGG}_*\mathbf{TTAGCS}CAATTACTTAG.$$

$$S_2'' = GCATTGCAT\mathbf{TTTTTAACGATTAC}_*\mathbf{CS}GTACTTAGCTAGATC.$$

$$S_3'' = TCAGGGCATCGAGAC\mathbf{TTTTTAG}_*\mathbf{CGATTAGCS}CTAGAATCAGACCT.$$

$$S_4'' = GTACCTGGCATTGAACG\mathbf{TTTTTAACGATTAGCA}_*TGCAGATGGACCTTTA.$$

$$S_5'' = AATGGATCAGA\mathbf{TTTTTAACGATTC}_*\mathbf{GCS}CTAGATTCAG.$$

2. Select Sample Points

Some sample points of two sequences in Z_1 are selected randomly and marked with the little dots on the top.

$$S_1' = GT\dot{A}CC\dot{A}TG\dot{G}A\mathbf{T\dot{T}A}_*\mathbf{TTA\dot{A}CGA\dot{T}T\dot{A}GCS}TA\dot{G}AG\dot{G}ACC\dot{T}A.$$

$$S_2' = \dot{A}ATC\dot{C}CTT\dot{A}\mathbf{C}_*\mathbf{\dot{T}TTT\dot{A}ACG\dot{A}T\dot{T}TA\dot{G}CS}\dot{G}TC.$$

3. Collision Detection

In this step, the left and right rough boundaries of two sequences will be marked. The following shows the left collision, which happens nearby the left motif boundary and are marked by two overline $\overline{TATT}$ and $\overline{TTTT}$ subsequences.

$$S_1' = GT\dot{A}CC\dot{A}TG\dot{G}A\mathbf{T\dot{T}\overline{TA_*TT}AA\dot{A}CGA\dot{T}T\dot{A}GC\dot{S}}TA\dot{G}AG\dot{G}ACC\dot{T}A.$$

$$S_2' = \dot{A}AT\dot{C}C_*TT\dot{A}\mathbf{C}_*\overline{\mathbf{\dot{T}TTT\dot{A}ACG\dot{A}T\dot{T}TA\dot{G}CS}}\dot{G}TC.$$

The following shows the right collision, which happens nearby the right motif boundary and is marked by two overline $\overline{TTAG}$ subsequences.

$$S_1' = GT\dot{A}CC\dot{A}_*TG\dot{G}A\mathbf{T\dot{T}A_*TTA\dot{A}CGA\overline{\dot{T}T\dot{A}GC\dot{S}}}TA\dot{G}AG\dot{G}ACC\dot{T}A.$$

$$S_2' = \dot{A}AT\dot{C}C_*TT\dot{A}\mathbf{C}_*\mathbf{\dot{T}TTT\dot{A}ACG\dot{A}\overline{\dot{T}TA\dot{G}}CS}\dot{G}TC.$$

4. Improving the Boundaries

In the early phase of the algorithm, we first detect a small piece of motif in S_1' by comparing S_1' and S_2'. Assume "$\mathbf{TA_*TT}$" and "$\mathbf{TTAG}$" are found in the left and right motif region of S_1' respectively. The rough motif length will be calculated via the difference of the location of the first character 'T' of the first subsequence and the location of the last character 'G' of the second subsequence. The position marked by "$\underline{\text{A}}$" is the rough left boundary of motif and the position marked by "$\underline{\text{T}}$" is the rough right boundary of motif in S_1' below.

$$S_1' = GTACCATGG\underline{A}\mathbf{TTA_*TTAACGATTAGCS}\underline{T}AGAGGACCTA.$$

$$S_2' = AATCCTT\underline{A}\mathbf{C}_*\mathbf{TTTTAACGATTAGCS}\underline{G}TC.$$

5. Sample Points for the Sequences in Z_2

Some sample points near the motif boundaries of S_1' are selected.

$S_1' = GTACCATG\dot{G}A\mathbf{T\dot{T}A}_*\mathbf{\dot{T}TAACGATT\dot{A}G\dot{C}S}T\dot{A}GAGGACCTA.$

Sample points are selected in each sequence in Z_2.

$$S_1'' = A\dot{T}TC\dot{G}ATCC\dot{A}G\mathbf{T\dot{T}T\dot{T}TAACGG}_*\mathbf{TTAG\dot{C}S}C\dot{A}AT\dot{T}ACTT\dot{A}G.$$

$$S_2'' = G\dot{C}ATT\dot{G}CAT\mathbf{T\dot{T}TTTAACGATTAC}_*\mathbf{\dot{C}S}GT\dot{A}CTT\dot{A}GCT\dot{A}GA\dot{T}C.$$

$$S_3'' = \dot{T}CA\dot{G}GGCA\dot{T}CGA\dot{G}AC\mathbf{TTT\dot{T}TAG}_*\mathbf{CGATTAG\dot{C}S}CTA\dot{G}AATC\dot{A}GACC\dot{T}.$$

$$S_4'' = GT\dot{A}CCTG\dot{G}CAT\dot{T}GAACG\mathbf{T\dot{T}TTTAACGATT\dot{A}GCA}_*TGC\dot{A}GAT\dot{G}GACCT\dot{T}TA.$$

$$S_5'' = AA\dot{T}GGA\dot{T}CAGA\mathbf{T\dot{T}TTTAACGATTC}_*\mathbf{G\dot{C}S}CTA\dot{G}ATT\dot{C}AG.$$

6. Collision Detection Between S_1' with the Sequences in Z_2

Some sample points near the motif boundaries of S_1' are selected.

$S_1' = GTACCATG\dot{G}A\mathbf{T\dot{T}A}_*\mathbf{\dot{T}TAACGATT\dot{A}G\dot{C}S}T\dot{A}GAGGACCTA.$

Sample points are selected in each sequence in Z_2.

$$S_1'' = A\dot{T}TC\dot{G}ATCC\dot{A}G\mathbf{T\dot{T}T\dot{T}T}\overline{\mathbf{AACGG}_*\mathbf{TTAG\dot{C}S}}C\dot{A}AT\dot{T}ACTT\dot{A}G.$$

$$S_2'' = G\dot{C}ATT\dot{G}CAT\mathbf{T}\overline{\mathbf{\dot{T}TTTAACGATTAC}_*\mathbf{\dot{C}S}}GT\dot{A}CTT\dot{A}GCT\dot{A}GA\dot{T}C.$$

$$S_3'' = \dot{T}CA\dot{G}GGCA\dot{T}CGA\dot{G}AC\mathbf{TTT}\overline{\mathbf{\dot{T}TAG}_*\mathbf{CGATTAG\dot{C}S}}CTA\dot{G}AATC\dot{A}GACC\dot{T}.$$

$S_4'' = GT\dot{A}CCT\dot{G}GCAT\dot{T}GAACG\mathbf{T}\overline{\dot{\mathbf{T}}\mathbf{TTTT}}\mathbf{AACG}\overline{\mathbf{ATT}\dot{\mathbf{A}}\mathbf{GCA}}_*TGC\dot{A}GAT\dot{G}GACCT\dot{T}TA.$

$S_5'' = AA\dot{T}GGA\dot{T}CAGA\mathbf{T}\overline{\dot{\mathbf{T}}\mathbf{TTTT}}\mathbf{AACGATTC}\overline{\mathbf{G}\dot{\mathbf{C}}\mathbf{S}}CTA\dot{G}ATT\dot{C}AG.$

7. Improving the Motif Boundaries for the Sequences in Z_2

After the collision with the sequences in Z_2, we obtain the rough location of motifs of the sequences in Z_2. Their motif boundaries for the sequences in Z_2 are improved.

$S_1' = GTACCATGGA\overline{\mathbf{TTA}_*\mathbf{T}}\mathbf{TAACGATT}\overline{\mathbf{AGCS}}TAGAGGACCTA.$

The improved motif boundaries of the sequences in Z_2 are marked below.

$S_1'' = ATTCGATCCA\underline{G}\mathbf{TTTTTAACGG}_*\mathbf{TTAGCS}\underline{C}AATTACTTAG.$

$S_2'' = GCATTGC\underline{A}T\mathbf{TTTTTAACGATTAC}_*\mathbf{CS}\underline{G}TACTTAGCTAGATC.$

$S_3'' = TCAGGGCATCGAGA\underline{C}\mathbf{TTTTTAG}_*\mathbf{CGATTAGCS}\underline{C}TAGAATCAGACCT.$

$S_4'' = GTACCTGGCATTGAAC\underline{G}\mathbf{TTTTTAACGATTAGCA}_*\underline{T}GCAGATGGACCTTTA.$

$S_5'' = AATGGATCAG\underline{A}\mathbf{TTTTTAACGATTC}_*\mathbf{GCS}C\underline{T}AGATTCAG.$

8. Motif Boundaries for the Sequences in Z_2

$S_1' = GTACCATGGA\overline{\mathbf{TTA}_*\mathbf{T}}\mathbf{TAACGATT}\overline{\mathbf{AGCS}}TAGAGGACCTA.$

Use the pair (G_L, G_R) with $G_L = \overline{\mathbf{TTAT}}$ and $G_R = \overline{\mathbf{AGCS}}$ to find the motif boundaries in the sequences of Z_2. The rough boundaries of the second group are marked below with underlines.

$S_1'' = ATTCGATCCA\underline{G}\mathbf{TTTTTAACGG}_*\mathbf{TTAGCS}\underline{C}AATTACTTAG.$

$$S_2'' = GCATTGCA\underline{T}\mathbf{TTTTTAACGATTAC_*CS}\underline{G}TACTTAGCTAGATC.$$

$$S_3'' = TCAGGGCATCGAGA\underline{C}\mathbf{TTTTTAG_*CGATTAGCS}\underline{C}TAGAATCAGACCT.$$

$$S_4'' = GTACCTGGCATTGAAC\underline{G}\mathbf{TTTTTAACGATTAGCA_*}\underline{T}GCAGATGGACCTTTA.$$

$$S_5'' = AATGGATCAG\underline{A}\mathbf{TTTTTAACGATTC_*GCS}\underline{C}TAGATTCAG.$$

9. Extracting the Motif Regions

The motif regions of the second group will be extracted. The original motif is recovered via voting at each column.

$$G_1'' = \mathbf{TTTTTAACGG_*TTAGCS}$$

$$G_2'' = \mathbf{TTTTTAACGATTAC_*CS}$$

$$G_3'' = \mathbf{TTTTTAG_*CGATTAGCS}$$

$$G_4'' = \mathbf{TTTTTAACGATTAGCA_*}$$

$$G_5'' = \mathbf{TTTTTAACGATTC_*GCS}$$

10. Recovering Motif via Voting

The original motif **TTTTTAACGATTAGCS** is recovered via voting at all columns. For example, the last **S** in the motif is recovered via voting among the characters **S, S, S, A, S** in the last column.

Our Results

We give an algorithm for the case with at most $\frac{1}{(\log n)^{2+\mu}}$ mutation rate. The performance of the algorithm is stated in Theorem 2. Theorem 2 implies Corollary 3 by selecting $k = c_1 \log n$ with

some constant c_1 large enough.

Theorem 2. *Assume that μ is a fixed number in $(0, 1)$ and the alphabet size t is at least 4. There exists a randomized algorithm and a constant c_0 such that if the length of the motif G is at least $c_0 \log n$, then given k independent $\Theta(n, G, \frac{1}{(\log n)^{2+\mu}})$-sequences, the algorithm outputs G' such that*

1) with probability at most $e^{-\Omega(k)}$, $|G'| \neq |G|$, and

2) for each $1 \leq i \leq |G|$, with probability at most $e^{-\Omega(k)}$, $G'[i] \neq G[i]$, and

3) with probability at most $\frac{k}{n^3}$, the algorithm Recover-Motif does not stop in $(O(k(\frac{n}{\sqrt{h}}(\log n)^{\frac{5}{2}} + h^2 \log n)), O(k))$ time,

where n is the longest length of any input sequences, and $h = \min(|G|, n^{\frac{2}{5}})$.

Corollary 3. *There exists a randomized algorithm, and positive constants c_0, c_1 and μ such that if the alphabet size is at least 4, the number of sequences is at least $c_1 \log n$, the motif length is at least $c_0 \log n$, and each character in motif region has probability at most $\frac{1}{(\log n)^{2+\mu}}$ of mutation, then motif can be recovered with probability at least $\frac{3}{4}$ in $(O(\frac{n}{\sqrt{h}}(\log n)^{\frac{7}{2}} + h^2 \log^2 n), O(\log n))$ time, where n is the longest length of any input sequences, and $h = \min(|G|, n^{\frac{2}{5}})$.*

We give a randomized algorithm for the case with $\Omega(1)$ mutation rate. The performance of the algorithm is stated in Theorem 4. Theorem 4 implies Corollary 5 by selecting $k = c_1 \log n$ with some constant c_1 large enough..

Theorem 4. *Assume that the alphabet size t is at least 4. There exists a randomized algorithm and a constant c_0 such that if the length of the motif G is at least $c_0 \log n$, then given k independent $\Theta(n, G, \mu))$-sequences, the algorithm outputs G' such that*

1) with probability at most $e^{-\Omega(k)}$, $|G'| \neq |G|$, and

2) for each $1 \leq i \leq |G|$*, with probability at most* $e^{-\Omega(k)}$*,* $G'[i] \neq G[i]$*,*

3) with probability at most $\frac{k}{n^3}$*, the algorithm Recover-Motif does not stop in* $(O(k(\frac{n^2}{|G|}(\log n)^{O(1)} + h^2)), O(k))$*,*

where n *is the longest length of any input sequences, and* $h = \min(|G|, n^{\frac{2}{5}})$*.*

Corollary 5. *There exists a randomized algorithm, and positive constants* c_0, c_1*, and* α *such that if the alphabet size is at least* 4*, the number of sequences is at least* $c_1 \log n$*, the motif length is at least* $c_0 \log n$*, and each character in motif region has probability at most* α *of mutation, then motif can be recovered with probability at least* $\frac{3}{4}$ *in* $(O(\frac{n^2}{|G|}(\log n)^{O(1)}), O(\log n))$ *time.*

We give a deterministic algorithm for the case with $\Omega(1)$ mutation rate. The performance of the algorithm is stated in Theorem 6. Theorem 6 implies Corollary 7 by selecting $k = c_1 \log n$ with some constant c_1 large enough.

Theorem 6. *Assume that the alphabet size* t *is at least* 4*. There exists a deterministic algorithm and a constant* c_0 *such that if the length of the motif* G *is at least* $c_0 \log n$*, then given* k *independent* $\Theta(n, G, \mu))$*-sequences, the algorithm runs in* $(O(n^2(\log n)^{O(1)} + h^2 k), O(k))$*, and outputs* G' *such that*

1) with probability at most $e^{-\Omega(k)}$*,* $|G'| \neq |G|$*, and*

2) for each $1 \leq i \leq |G|$*, with probability at most* $e^{-\Omega(k)}$*,* $G'[i] \neq G[i]$*,*

3) with probability at most $\frac{k}{n^3}$*, the algorithm Recover-Motif does not stop in* $(O(k(n^2(\log n)^{O(1)} + h^2)), O(k))$ *time,*

where n *is the longest length of any input sequences, and* $h = \min(|G|, n^{\frac{2}{5}})$*.*

Corollary 7. *There exists a deterministic algorithm, and positive constants c_0, c_1, and α such that if the alphabet size is at least 4, the number of sequences is at least $c_1 \log n$, the motif length is at least $c_0 \log n$, and each character in motif region has probability at most α of mutation, then motif can be recovered with probability at least $\frac{3}{4}$ in $(O(n^2(\log n)^{O(1)}), O(\log n))$ time.*

Algorithm Recover-Motif

In this section, we give a unified approach to describe three algorithms. The performance of the algorithms is stated in the Theorems 2, 4, and 6.

Some Parameters

Definition 8.

i. Parameter x is selected to be 10. This parameter controls the failure probability of our algorithms to be at most $\frac{1}{2^x}$.

ii. The size of alphabet is $t \geq 4$.

iii. Select a constant $\rho_0 \in (0, 1)$ to have inequality (1)

$$\rho_0 < \frac{t-1}{2t}. \tag{1}$$

iv. The constant $\epsilon \in (0, 1)$ is selected to satisfy

$$\epsilon < \min((\frac{t-1}{t} - (2\rho_0 + 2\epsilon)),\ \ \frac{1}{5}(1 - \frac{2}{t-1} - \frac{4}{2^x}),\ \ \frac{1}{3}). \tag{2}$$

The existence of ϵ follows from inequality (1). The constant ϵ is used to control the mutation in the motif area. It is a part of parameter β defined in item (xiv) of this definition.

v. Let $c = e^{-\frac{\epsilon^2}{3}}$. The constant c is used to simplify probabilistic bounds which are derived from the applications of Chernoff bounds (See Corollary 18).

vi. Define $r(y) = (\frac{1}{t-1} + \frac{c^y}{1-c})$.

vii. Define u_1 to be a large constant that for all $v \geq 0$,

$$\frac{2(v+u_1)c^{v+u_1}}{(1-c)^2} \leq \frac{1}{5 \cdot 2^x}. \tag{3}$$

viii. Select constant $\rho_1 \in (0,1)$ such that

$$\frac{2}{t-1} + \frac{4}{2^x} + 5\epsilon + \rho_1 < 1. \tag{4}$$

The existence of ρ_1 follows from $\epsilon < \frac{1}{5}(1 - \frac{2}{t-1} - \frac{4}{2^x})$, which is implied by inequality (2).

ix. Select constant $\rho_2 \in (0,1)$ and constant positive integer v large enough such that

$$\frac{6(v+u_1)c^v}{1-c} + \rho_2 < \rho_1, \text{ and} \tag{5}$$

$$(\frac{1}{2^x} + (v+u_1)\frac{c^v}{1-c} + \frac{c^v}{1-c} + \frac{1}{5 \cdot 2^x}) \leq 1/2. \tag{6}$$

x. Define $\varsigma_0 = \frac{1}{2^x}$, and $\varphi(v) = (v+u_1)\frac{c^v}{1-c} + \frac{c^v}{1-c})$.

xi. Select constant α_0 such that

$$4(v-1)\alpha_0 + \alpha_0 < \rho_2, \text{ and} \tag{7}$$

$$\alpha_0 < \rho_0. \tag{8}$$

Adding inequalities (4), (5), and (7), we have inequality (9)

$$(\frac{2}{t-1}+\frac{4}{2^x}+5\epsilon)+\frac{6(v+u_1)c^v}{1-c}+(4(v-1)\alpha_0+\alpha_0)<1. \tag{9}$$

By arranging the terms in inequality (9) and the definitions of $r(v)$ and $\varphi(v)$, we have inequality (10)

$$2((2(v-1)\alpha_0+\frac{c^v}{1-c})+r(v)+2(\varsigma_0+\varphi(v))+2\epsilon)+(\alpha_0+\epsilon)<1. \tag{10}$$

xii. The maximal mutation rate α for the second algorithm (Theorem 4) and third algorithm (Theorem 6) are selected as α_0. Since the mutation rate of our sublinear time algorithm is bounded by $\frac{1}{(\log n)^{2+\mu}}$, the maximal mutation rate α for the first algorithm (Theorem 2) is less than α_0 when n is large enough. We always assume that all mutation rates α in our three algorithms are in the range $(0,\alpha_0]$.

xiii. Define $q(y)=2(v-1)\alpha+\frac{2c^y}{1-c}$. By inequality (10), the definition of $q(y)$, and the fact $\alpha\in(0,\alpha_0)$, we have

$$2(q(v)+r(v)+2(\varsigma_0+\varphi(v))+2\epsilon)+(\alpha_0+\epsilon) \quad < \quad 1. \tag{11}$$

Inequality (11) implies $q(v) \leq \frac{1}{2}$. By inequality (6), we have that

$$(\frac{1}{2^x} + (v + u_1)\frac{c^v}{1-c} + \frac{c^v}{1-c} + \frac{1}{5 \cdot 2^x}) + q(v) \leq 3/4 \tag{12}$$

xiv. Let $\beta = 2\alpha + 2\epsilon$. The parameter β controls the similarity of $\aleph(S)$ and the original motif G (see Lemma 27).

xv. Define $R = r(v)$.

xvi. We define the following Q_0.

$$Q_0 = q(v). \tag{13}$$

The parameter Q_0 used in Lemma 27 gives an upper bound of the probability that a $\Theta(n, G, \alpha)$-sequence S whose $\aleph(S)$ will not be similar enough to the original motif G according to the conditions in Lemma 27.

xvii. Select constant d_0 such that

$$n^3 c^{d_0 \log n} < \frac{1}{5 \cdot 2^x}. \tag{14}$$

xviii. Select constant d_1 such that $(v + u_1)c^{d_1 \log n} < \frac{1}{5 \cdot 2^x}$.

xix. Select number u_2 such that

$$(d_1 \log n)(v + u_1)\frac{c^{v+u_2}}{1-c} \leq \frac{1}{5 \cdot 2^x}. \text{ and} \tag{15}$$

$$(v+u_1)\frac{c^{v+u_2}}{1-c} \quad < \quad \frac{1}{5\cdot 2^x} \tag{16}$$

Since only n is variable, we can make $u_2 = O(\log\log n)$.

xx. For a fixed $c \in (0,1)$, define $\delta_c = \frac{\ln\frac{1}{c}}{2}$.

Description of Algorithm Recover-Motif

The algorithms are described in this section. The description combines three algorithms together. Before presenting the algorithm, we define some notions.

Definition 9.

- Two sequences X_1 and X_2 are *weakly left matched* if (1) both $|X_1|$ and $|X_2|$ are at least $d_0 \log n$, (2) $\mathrm{diff}(X_1[1,i], X_2[1,i]) \le \beta$ for all integers i, $v \le i \le d_0 \log n$.

- Two sequences X_1 and X_2 are *left matched* if (1) $d_0 \log n \le |X_1|, |X_2|$, (2) $X_1[i] = X_2[i]$ for $i = 1, \cdots, v-1$, and (3) $\mathrm{diff}(X_1[1,i], X_2[1,i]) \le \beta$ for all integers i, $v \le i \le d_0 \log n$.

- Two sequences X_1 and X_2 are *weakly right matched* if X_1^R and X_2^R are weakly 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 *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.

Variable L will be controlled in the range $L \in [(\log n)^{3+\epsilon_1}, n^{\frac{2}{5}-\epsilon_2}]$ in our algorithm with high probability. We define the following functions that depend on L.

Definition 10. Define $M(L) = \frac{\sqrt{3\log n+x}}{\sqrt{1-\gamma}}\sqrt{L}\log n$. Define $M_1(L) = \frac{\delta_{c_0}M(L)}{\log n}$ (see Definition 8 for δ_c), where $c_0 = \frac{1}{4}$.

We would like to minimize the function $(\frac{n}{L}M + L^2)\log n$. This selection can make the total time complexity sublinear.

Definition 11. For a $\Theta(n, G, \alpha)$ sequence S, define LB(S) to be the left boundary l of the motif region $\aleph(S)$ in S, and RB(S) to be the right boundary r of the motif region $\aleph(S)$ in S such that $\aleph(S) = S[l, r]$.

Boundary-Phase of Algorithm Recover-Motif

The first phase of Algorithm Recover-Motif finds the rough motif boundaries of all input sequences. It first detects the rough motif boundaries of one sequence via comparing two input sequences. Then the rough boundaries of the first sequence is used to find the rough motif boundaries of other input sequences.

Three algorithms share most of the functions. We have a unified approach to describe them. A special variable "algorithm-type" selects one of the three algorithms, respectively.

Definition 12. Let algorithm-type represent one of the three algorithm types, "RANDOMIZED-SUBLINEAR", "RANDOMIZED-SUBQUADRATIC", and "DETERMINISTIC-SUPERQUADRATIC".

Definition 13. Assume that A_1 is a set of positions in a $\Theta(n, G, \alpha)$ sequence S_1 and A_2 is a set of positions in a $\Theta(n, G, \alpha)$ sequence S_2. If there are positions $a_1 \in A_1$ and $a_2 \in A_2$ such that for some position j with $1 \leq j \leq |G|$, a_1 is the position of $\aleph(S_1)[j]$ in S_1 and a_2 is the position of $\aleph(S_2)[j]$ in S_2, then A_1 and A_2 have a *collision* at (a_1, a_2).

In the following function Collision-Detection, the parameter $\omega \leq \beta$ is defined below in the three algorithms.

$$\omega_{\text{algorithm-type}} = \begin{cases} 0 & \text{if algorithm-type=RANDOMIZED-SUBLINEAR;} \\ \beta & \text{if algorithm-type=RANDOMIZED-SUBQUADRATIC;} \\ \beta & \text{if algorithm-type=DETERMINISTIC-SUPERQUADRATIC.} \end{cases} \quad (17)$$

Function **Collision-Detection**(S_1, U_1, S_2, U_2) is used to detect a point $a_1 \in U_1$ in the motif area in S_1 and another $a'_1 \in U_1$ point in the motif area of S_1. The two points a_1 and a'_1 are close to the left and right motif boundaries of S_1, respectively. A similar pair of points e_1 and e'_1 in U_2 is also derived for S_2.

Collision-Detection(S_1, U_1, S_2, U_2)

Input: a pair of $\Theta(n, G, \alpha)$-sequences S_1 and S_2, U_i is a set of locations in S_i for $i = 1, 2$.

Output: the left and right rough boundaries of two sequences.

Let D_1 be all subsequences $S_1[a, a + d_0 \log n - 1]$ of S_1 of length $d_0 \log n$ with $a \in U_1$.

Let D_2 be all subsequences $S_2[b, b + d_0 \log n - 1]$ of S_2 of length $d_0 \log n$ with $b \in U_2$.

Find two subsequences $X_1 = S_1[a_1, a_1 + d_0 \log n - 1] \in D_1$ and

$X_2 = S_2[b_1, b_1 + d_0 \log n - 1] \in D_2$ such that a_1 is the least and $\text{diff}(X_1, X_2) \leq \omega_{\text{algorithm-type}}$.

Find two subsequences $X'_1 = S_1[a'_1, a'_1 + d_0 \log n - 1] \in D_1$ and

$X'_2 = S_2[b'_1, b'_1 + d_0 \log n - 1] \in D_2$ such that a'_1 is the largest and

$\text{diff}(X'_1, X'_2) \leq \omega_{\text{algorithm-type}}$.

Find two subsequences $Y_1 = S_1[f_1, f_1 + d_0 \log n - 1] \in D_1$ and

$Y_2 = S_2[e_1, e_1 + d_0 \log n - 1] \in D_2$ such that e_1 is the least and

$\text{diff}(Y_1, Y_2) \leq \omega_{\text{algorithm}-\text{type}}$.

Find two subsequences $Y_1' = S_1[f_1', f_1' + d_0 \log n - 1] \in D_1$ and

$Y_2' = S_2[e_1', e_1' + d_0 \log n - 1] \in D_2$ such that e_1' is the largest and

$\text{diff}(Y_1', Y_2') \leq \omega_{\text{algorithm}-\text{type}}$.

Return (a, a', e_1, e_1').

End of Collision-Detection

Definition 14. Let $[a, b]$ be an interval with two integers boundaries a and b, and l be a positive integer parameter. Define l-partition of $[a, b]$ to be $l - P([a, b])$ that contains the intervals $[a_1, b_1], [a_2, b_2], \cdots, [a_r, b_r]$ such that $a_1 = a, b_r = b, a_{i+1} = b_i + 1, b_i = a_i + l - 1$ for $i = 1, 2, \cdots, r - 1$, and $a_r \leq b_r \leq a_i + l - 1$.

For example, the 3-partition of the interval $[1, 10]$ is 3-$P([1, 10]) = \{[1, 3], [4, 6], [7, 9], [10, 10]\}$. Function **Point-Selection**(S, L, I) will be defined differently in three different algorithms, where I is an interval of positions in sequence S, and L is an positive integer parameter. For randomized algorithms, some random points are selected in L-$P(I)$. For deterministic algorithm, all points in I are selected.

Point-Selection(S, L, I)

Input: a pair of $\Theta(n, G, \alpha)$-sequences S, a size parameter L of partition, and an interval of positions I in S.

Output: a set U of positions from S respectively.

Steps:

Let $U = \emptyset$.

If algorithm-type=RANDOMIZED-SUBLINEAR or RANDOMIZED-SUBQUADRATIC

If ($L \geq \frac{(\log n)^{3+\tau}}{100}$)

For each interval I' in I, obtain L-partition of I' L-$P(I)$.

Sample $M(L)$ (see Definition 10) random positions at every

interval J in L-$P(I')$, and put them into U.

Else

Put every position of I into U_1.

If algorithm-type=DETERMINISTIC-SUPERQUADRATIC

Put every position of I into U.

Return U.

End of Point-Selection

The function **Improve-Boundaries**$(S_1, a_l, a_r, S_2, f_l, f_r, L)$ is used to improve the existing rough left and right boundaries a_l and a_r of S_1 respectively, and improve the existing rough left and right boundaries f_l and f_r of S_2 respectively. We assume $a_l \in [\mathrm{LB}(S_1) - L, \mathrm{LB}(S_1) + L]$, $a_r \in [\mathrm{RB}(S_1) - L, \mathrm{LB}(S_1) + L]$, $f_l \in [\mathrm{LB}(S_2) - L, \mathrm{LB}(S_2) + L]$, and $f_r \in [\mathrm{RB}(S_2) - L, \mathrm{RB}(S_2) + L]$. After calling this function, more accurate approximate boundaries will be derived. From the probabilistic analysis, we have good chance to get the exact motif boundaries for both S_1 and S_2.

Improve-Boundaries$(S_1, a_l, a_r, S_2, f_l, f_r, L)$

Input: a $\Theta(n, G, \alpha)$-sequence S_1 with rough left and right boundaries a_l and a_r, a $\Theta(n, G, \alpha)$-sequences S_2 with rough left and right boundaries f_l and f_r, and an approximate distance L to the

nearest motif boundary from those rough boundaries (The parameter L usually has the properties that $\mathrm{LB}(S_1) \in [a_l - L, a_l]$, $\mathrm{RB}(S_1) \in [a_r, a_r + L]$, $\mathrm{LB}(S_2) \in [f_l - L, f_l]$, and $\mathrm{RB}(S_2) \in [f_r, f_r + L]$).

Output: improved rough left and right boundaries for both S_1 and S_2.

Steps:

Find two subsequences $X_1 = S_1[a_1, a_1 + d_0 \log n - 1]$ and $X_2 = S_2[b_2, b_2 + d_0 \log n - 1]$ with $a_1 \in [a_l - L, a_l + L]$ and $b_2 \in [f_l - L, f_l + L]$ such that $\mathrm{diff}(X_1, X_2) \leq \beta$ and a_1 is the least.

Find two subsequences $X_1' = S_1[a_1', a_1' + d_0 \log n - 1]$ and $X_2' = S_2[b_2', b_2' + d_0 \log n - 1]$ with $a_1' \in [a_r - L, a_r + L]$ and $b_2 \in [f_r - L, f_r + L]$ such that $\mathrm{diff}(X_1', X_2') \leq \beta$ and a_1' is the largest.

Find two subsequences $Y_1 = S_1[e_1, e_1 + d_0 \log n - 1]$ and $Y_2 = S_2[f_2, f_2 + d_0 \log n - 1]$ with $e_1 \in [a_l - L, a_l + L]$ and $f_2 \in [f_l - L, f_l + L]$ such that $\mathrm{diff}(Y_1, Y_2) \leq \beta$ and f_2 is the least.

Find two subsequences $Y_1' = S_1[e_1', e_1' + d_0 \log n - 1]$ and $Y_2' = S_2[f_2', f_2' + d_0 \log n - 1]$ with $e_1' \in [a_r - L, a_r + L]$ and $f_2' \in [f_r - L, f_r + L]$ such that $\mathrm{diff}(Y_1', Y_2') \leq \beta$ and f_2' is the largest.

Return (a_1, a_1', f_2, f_2').

End of Improve-Boundaries

The function **Initial-Boundaries**(S_1, S_2) detects the motif boundaries for two sequences S_1 and S_2. It first detect rough motif boundaries that is controlled by parameter L. The rough boundaries will be improved to exact motif boundaries via calling **Improve-Boundaries**$(.)$.

Initial-Boundaries(S_1, S_2)

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

Output: rough left boundary $\mathrm{roughLeft}_{S_1}$ of S_1, right boundary $\mathrm{roughRight}_{S_1}$ of S_1, rough left boundary $\mathrm{roughLeft}_{S_2}$ of S_2, and right boundary $\mathrm{roughRight}_{S_2}$ of S_2.

Steps:

Let $U_1 = U_2 = \emptyset$.

Let $L = n^{2/5}$.

Repeat

Let U_1 =Point-Selection$(S_1, L, [1, |S_1|])$.

Let U_2 =Point-Selection$(S_2, L, [1, |S_2|])$.

Let $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ =Collision-Detection(S_1, U_1, S_2, U_2).

If ($L_{S_1} \neq \emptyset$ and $R_{S_1} \neq \emptyset$)

Then Goto H.

Else $L = L/2$.

Until $(L < \frac{1}{2}\frac{(\log n)^{3+\tau}}{100})$

H: Return Improve-Boundaries$(S_1, L_{S_l}, R_{S_1}, S_2, L_{S_2}, R_{S_2}, 2L)$.

End of Initial-Boundaries

If L_S and R_S are the left and right motif boundaries of a sequence S, then the motif length is $R_S - L_S + 1$. When we have the exact motif boundaries L'_{S_i} and R'_{S_i} for most sequences S_i, their motif length can be derived via the median in $\cup_i\{R'_{S_i} - L'_{S_i} + 1\}$. Therefore, we have the function **Motif-Length-And-Boundaries(**Z_1**)** to compute the length of motif region.

Motif-Length-And-Boundaries(Z_1**)**

Input: $Z_1 = \{S'_1, \cdots, S'_{2k_1}\}$ is a set of independent $\Theta(n, G, \alpha)$ sequences.

Steps:

For $i = 1$ to k_1

let $(\text{roughLeft}_{S'_{2i-1}}, \text{roughRight}_{S'_{2i}})$=Initial-Boundaries$(S'_{2i-1}, S'_{2i})$.

Let L_1 be the median of $\cup_{i=1}^{k_1}\{(\text{roughRight}_{S'_{2i-1}} - \text{roughLeft}_{S'_{2i-1}} + 1)\}$.

Return L_1.

End of Motif-Length-And-Boundaries

Extract-Phase of Algorithm Recover-Motif

After a set of motif candidates W is produced from Boundary-Phase of algorithm Recover-Motif, we use this set to match with another set of input sequences to recover the hidden motif by voting.

Match(G_l, G_r, S_i)

Input: a motif left part G_l (which can be derived from the rough left boundary of an input sequence S), a motif right part G_r, a sequence S''_i from the group Z_2, with known rough left and right boundaries.

Output: either a rough motif region of S''_i, or an empty sequence which means the failure in extracting the motif region $\aleph(S''_i)$ of S''_i.

Steps:

Find a position a in S''_i with $\text{roughLeft}_{S''_i} \leq a \leq \text{roughLeft}_{S''_i} + (v + u_2)$.

such that G_l and $S''_i[a, a + |G_l| - 1]$ are left matched (see Definition 9).

Find a position b in S''_i with $\text{roughRight}_{S''_i} - (v + u_2), \leq b \leq \text{roughRight}_{S''_i}$

such that G_r and $S''_i[b - |G_r| + 1, b]$ are right matched (see Definition 9).

If both a and b are found

Then output $S_i''[a, b]$

Else output $\emptyset$ (empty string).

End of Match

If the left G_l and right G_r motif parts are known, we extract all the motif regions for all sequences in the set Z_2 by the function **Extract**(G_l, G_r, Z_2)**.**

Extract(G_l, G_r, Z_2)**)**

Input $Z_2 = \{S_1'', S_2'', \cdots, S_{k_2}''\}$ and left and right motif parts G_l and G_r (see function **Match**(G_l, G_r, S_i)).

Steps:

For each S_i'' with $i = 1, 2, \cdots, k_2$,

let $G_i'' = \mathrm{Match}(G_l, G_r, S_i'')$.

Return $(G_1'', G_2'', \cdots, G_{k_2}'')$.

End of Extract

The following is Extract-Phase of algorithm Recover-Motif. It extracts the motif regions of another set Z_2 of input sequences. The function is based on the condition that exact motif boundaries can be derived for most sequences.

Extract-Phase(S', Z_2**):**

Input S' is an input sequence with known $\mathrm{roughLeft}_{S'}$ and $\mathrm{roughRight}_{S'}$ for its rough left and right boundaries respectively, and $Z_2 = \{S_1'', \cdots, S_{k_2}''\}$ is a set of input sequences.

Steps:

For each subsequence $G_l = S'[a, a + d_0 \log n - 1]$ with $a \in [\text{roughLeft}_{S'}, \text{roughLeft}_{S'} + (v + u_1)]$

and $G_r = S'[b - d_0 \log n + 1, b]$ with $b \in [\text{roughRight}_{S'} - (v + u_1), \text{roughRight}_{S'}]$

let $(G''_1, G''_2, \cdots, G''_{k_2})$ be the output from Extract(G_l, G_r, Z_2).

If the number of empty sequences in $G''_1, \cdots, G''_{k_2}$ is at most $(Q_0 + (R + 2\epsilon))k_2$

Then return $(G''_1, G''_2, \cdots, G''_{k_2})$.

Return $\emptyset$ (empty set).

End of Extract-Phase

Voting-Phase

The function Vote$(G''_1, G''_2, \cdots, G''_{k_2})$ is to generate another sequence G' by voting, where $G'[i]$ is the most frequent character among $G''_1[i], G''_2[i], \cdots, G''_{k_2}[i]$.

Voting-Phase$(G''_1, G''_2, \cdots, G''_{k_2})$

Input: $\Theta(n, G, \alpha)$ sequences $G''_1, G''_2, \cdots, G''_{k_2}$ of the same length m.

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

Steps:

For each $j = 1, \cdots, m$

let a_j be the most frequent character among $G''_1[j], \cdots, G''_{k_2}[j]$.

Return $G' = a_1 \cdots a_m$.

End of Vote

Entire Algorithm Recover-Motif

The entire algorithm is described below. The input has two sets of sequences Z_1 and Z_2. It detects the motif boundaries for the sequences in Z_1 via pair wise comparisons, and also the motif length. The motif regions of the sequences in Z_2 are detected in the next phase, and will be extracted. The original motif is recovered via voting for each column of characters among the extracted motif regions.

We maintain the sizes of Z_1 and Z_2 to be roughly equal, which implies

$$|Z_1| = \Theta(|Z_2|) \tag{18}$$

Algorithm Recover-Motif (Z)

Input: $Z = Z_1 \cup Z_2$, where $Z_1 = \{S'_1, \cdots, S'_{2k_1}\}$ and $Z_2 = \{S''_1, \cdots, S''_{k_2}\}$ are two sets of input sequences.

Steps:

Preprocessing Part:

For each $S \in Z_1 \cup Z_2$, let $\mathrm{roughLeft}_S = \mathrm{roughRight}_S = 0$ (the two boundaries are unknown).

l_{motif} =MotifLengthAndBoundaries(Z_1).

Let $L = l_{motif}/4$.

For $i = 1$ to k_1,

let $U_{S'_{2i-1}}$ =Point-Selection$(S'_{2i-1}, L, [\mathrm{roughLeft}_{S'_{2i-1}} - 2L, \mathrm{roughLeft}_{S'_{2i-1}} + 2L]) \cup$

Point-Selection$(S'_{2i-1}, L, [\mathrm{roughRight}_{S'_{2i-1}} - 2L, \mathrm{roughRight}_{S'_{2i-1}} + 2L])$.

For $j = 1$ to k_2

let $U_{S''_j}$ =Point-Selection$(S''_j, L, [1, |S''_j|])$.

For $i = 1$ to k_1

For each $S''_j \in Z_2$

Let $(L_{S'_{2i-1}}, R_{S'_{2i-1}}, L_{S''_j}, R_{S''_j})$ =Collision-Detection$(S'_{2i-1}, U_{S'_{2i-1}}, S''_j, U_{S''_j})$.

Let $(L_{S'_{2i-1}}, R_{S'_{2i-1}}, \text{roughLeft}_{S''_j}, \text{roughRight}_{S''_j})$=

Improve-Boundaries$(S'_{2i-1}, L_{S'_{2i-1}}, R_{S'_{2i-1}}, S''_j, L_{S''_j}, R_{S''_j}, 2L)$.

Let $(G''_1, G''_2, \cdots, G''_{k_2})$ be the output from Extract-Phase(S'_{2i-1}, Z_2).

If $(G''_1, G''_2, \cdots, G''_{k_2})$ is not empty

Then go to Voting Part.

Voting Part:

Return Voting-Phase$(G''_1, G''_2, \cdots, G''_{k_2})$.

End of Algorithm Recover-Motif

Deterministic Algorithm

In this section, we give a deterministic algorithm, which is a simplified version of the unified algorithm described before. It is simpler than the randomized versions. The first phase of Algorithm Recover-Motif (.) finds the rough motif boundaries of all input sequences. It first detects the rough motif boundaries of one sequence via comparing two input sequences. Then the rough boundaries of the first sequence is used to find the rough motif boundaries of other input sequences. We still let

$$\omega_{\text{DETERMINISTIC-SUPERQUADRATIC}} = \beta. \tag{19}$$

Collision-Detection(S_1, S_2)

Input: a pair of $\Theta(n, G, \alpha)$-sequences S_1 and S_2, U_i is a set of locations in S_i for $i = 1, 2$.

Output: the left and right rough boundaries of two sequences.

Let D_1 be all subsequences $S_1[a, a+d_0 \log n-1]$ of S_1 of length $d_0 \log n$ with $a \in [1, |S_1|]$.

Let D_2 be all subsequences $S_2[b, b+d_0 \log n-1]$ of S_2 of length $d_0 \log n$ with $b \in [1, |S_2|]$.

Find two subsequences $X_1 = S_1[a_1, a_1 + d_0 \log n - 1] \in D_1$ and

$X_2 = S_2[b_1, b_1 + d_0 \log n - 1] \in D_2$ such that a_1 is the least and $\text{diff}(X_1, X_2) \leq \omega_{\text{DETERMINISTIC-SUPERQUADRATIC}}$.

Find two subsequences $X_1' = S_1[a_1', a_1' + d_0 \log n - 1] \in D_1$ and

$X_2' = S_2[b_1', b_1' + d_0 \log n - 1] \in D_2$ such that a_1' is the largest and

$\text{diff}(X_1', X_2') \leq \omega_{\text{DETERMINISTIC-SUPERQUADRATIC}}$.

Find two subsequences $Y_1 = S_1[f_1, f_1 + d_0 \log n - 1] \in D_1$ and

$Y_2 = S_2[e_1, e_1 + d_0 \log n - 1] \in D_2$ such that e_1 is the least and

$\text{diff}(Y_1, Y_2) \leq \omega_{\text{DETERMINISTIC-SUPERQUADRATIC}}$.

Find two subsequences $Y_1' = S_1[f_1', f_1' + d_0 \log n - 1] \in D_1$ and

$Y_2' = S_2[e_1', e_1' + d_0 \log n - 1] \in D_2$ such that e_1' is the largest and

$\text{diff}(Y_1', Y_2') \leq \omega_{\text{DETERMINISTIC-SUPERQUADRATIC}}$.

Return (a, a', e_1, e_1').

End of Collision-Detection

Function **Point-Selection**(S_1, S_2, L) is not used in the deterministic algorithm.

Improve-Boundaries$(S_1, a_l, a_r, S_2, f_l, f_r, L)$ is the same as that in the randomized algorithms.

Initial-Boundaries(S_1, S_2)

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

Output: rough left boundary $\mathrm{roughLeft}_{S_1}$ of S_1, right boundary $\mathrm{roughRight}_{S_1}$ of S_1, rough left boundary $\mathrm{roughLeft}_{S_2}$ of S_2, and right boundary $\mathrm{roughRight}_{S_2}$ of S_2.

Steps:

Let $U_1 = U_2 = \emptyset$.

Let $L = n^{2/5}$.

Repeat

Let $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ =Collision-Detection(S_1, S_2).

If ($L_{S_1} \neq \emptyset$ and $R_{S_1} \neq \emptyset$)

Then Goto H.

Else $L = L/2$.

Until $(L < \frac{1}{2}\frac{(\log n)^{3+\tau}}{100})$

H: Return Improve-Boundaries$(S_1, L_{S_l}, R_{S_1}, S_2, L_{S_2}, R_{S_2}, 2L)$.

End of Initial-Boundaries

Motif-Length-And-Boundaries(Z_1) is the same as that before.

Match(G_l, G_r, S_i) is the same as that for the randomized algorithm.

Extract(G_l, G_r, Z_2) is the same as that for the randomized algorithm.

The following is Extract-Phase of algorithm Recover-Motif. It extracts the motif regions of another set Z_2 of input sequences.

Extract-Phase(S', Z_2) is the same as that for the randomized algorithm.

Voting-Phase$(G''_1, G''_2, \cdots, G''_{k_2})$ is the same as that for the randomized algorithm.

The entire deterministic algorithm is described below. We maintain the sizes of Z_1 and Z_2 to

be roughly equal.

Algorithm Recover-Motif (Z)

Input: $Z = Z_1 \cup Z_2$, where $Z_1 = \{S'_1, \cdots, S'_{2k_1}\}$ and $Z_2 = \{S''_1, \cdots, S''_{k_2}\}$ are two sets of input sequences.

Steps:

Preprocessing Part:

For each $S \in Z_1 \cup Z_2$, let $\mathrm{roughLeft}_S = \mathrm{roughRight}_S = 0$ (the two boundaries are unknown).

l_{motif} =MotifLengthAndBoundaries(Z_1).

Let $L = l_{motif}/4$.

For $i = 1$ to k_1

For each $S''_j \in Z_2$

Let $(L_{S'_{2i-1}}, R_{S'_{2i-1}}, L_{S''_j}, R_{S''_j})$ =Collision-Detection(S'_{2i-1}, S''_j).

Let $(L_{S'_{2i-1}}, R_{S'_{2i-1}}, \mathrm{roughLeft}_{S''_j}, \mathrm{roughRight}_{S''_j})$=

Improve-Boundaries($S'_{2i-1}, L_{S'_{2i-1}}, R_{S'_{2i-1}}, S''_j, L_{S''_j}, R_{S''_j}, 2L$).

Let $(G''_1, G''_2, \cdots, G''_{k_2})$ be the output from Extract-Phase(S'_{2i-1}, Z_2).

If $(G''_1, G''_2, \cdots, G''_{k_2})$ is not empty

Then go to Voting Part.

Voting Part:

Return Voting-Phase($G''_1, G''_2, \cdots, G''_{k_2}$).

End of Algorithm Recover-Motif

Analysis of Algorithms

Review of Some Classical Results in Probability

Some well known results in classical probability theory are listed. The readers can skip this

section if they understand them well. The inclusion of these results make the thesis self-contained.

- For a list of events $A_1, \cdots, A_m$, $\Pr[A_1 \cup A_2 \cup \cdots \cup A_m] \leq \Pr[A_1] + \Pr[A_2] + \cdots + \Pr[A_m]$.
- For two independent events A and B, $\Pr[A \cap B] = \Pr[A]\Pr[B]$.
- For a random variable Y, $\Pr[Y \geq t] \leq \frac{E[Y]}{t}$ for all positive real number t. This is called Markov inequality.

The analysis of our algorithm employs the Chernoff bound [17] and Corollary 18 below, which can be derived from it (see [14]).

Theorem 15 ([17]). *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 $\mu = E[X]$. Then for any $\delta > 0$,*

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

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

We follow the proof of Theorem 15 to make the following version of Chernoff bound so that it can be used in our algorithm analysis.

Theorem 16. *Let $X_1, \cdots, X_n$ be n independent random 0-1 variables, where X_i takes 1 with probability at most p. Let $X = \sum_{i=1}^{n} X_i$. Then for any $\delta > 0$, $\Pr(X > (1+\delta)pn) < \left[\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right]^{pn}$.*

Define $g(\delta) = \frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}$. We note that $g(\delta)$ is always strictly less than 1 for all $\delta > 0$, and $g(\delta)$ is fixed if δ is a constant. This can be verified by checking that the function $f(x) = \ln \frac{e^x}{(1+x)^{(1+x)}} = x - (1+x)\ln(1+x)$ is decreasing and $f(0) = 0$. This is because $f'(x) = -\ln(1+x)$, which is less than 0 for all $x > 0$.

Theorem 17. *Let $X_1, \cdots, X_n$ be n independent random 0-1 variables, where X_i takes 1 with probability at most p. Let $X = \sum_{i=1}^{n} X_i$. Then for any $\delta > 0$, $\Pr(X < (1-\delta)pn) < e^{-\frac{1}{2}pn\delta^2}$.*

Corollary 18 ([14]). *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 $\epsilon > 0$, $\Pr(X < pn - \epsilon n) < e^{-\frac{1}{2}n\epsilon^2}$.

Analysis of Boundary-Phase of Algorithm Recover-Motif

Lemma 55 shows that with only small probability, a sequence can match a random sequence. It will be used to prove that when two substrings in two different $\Theta(n, G, \alpha)$-sequences are similar, they are unlikely not to coincide with the motif regions in the two $\Theta(n, G, \alpha)$-sequences, respectively.

Lemma 19. *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*

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

ii. the probability for $\mathrm{diff}(X_1, X_2) \leq \beta$ 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 $X_2[j]$ to equal $X_1[j]$. If $\mathrm{diff}(X_1, X_2) \leq \beta$, the two sequences X_1 and X_2 are identical in at least $(1-\beta)|X_1|$ positions, but the expected number of positions where the two sequences are identical is

$\frac{1}{t}|X_1|$. The probability for $\mathrm{diff}(X_1, X_2) \leq \beta$ is at most $e^{-\frac{(1-\beta-\frac{1}{t})^2}{3}|X_1|} \leq e^{-\frac{\epsilon^2}{3}|X_1|}$ by Corollary 18, and Definitions 8 and 9. ∎

Lemma 20 shows that with small probability, an input $\Theta(n, G, \alpha)$ sequence contains motif region that has many mutations.

Lemma 20. *With probability at most $\frac{c^y}{1-c}$, a $\Theta(n, G, \alpha)$ sequence S changes more than $\frac{\beta}{2}t$ characters in its first left t motif region $\aleph(S)$ for some t with $y \leq t \leq |G|$, where $c = e^{-\frac{\epsilon^2}{3}}$.*

Proof: Every character in the $\aleph(S)$ region has probability at most α to mutate. We know that $|\aleph(S)| = |G| \geq d$. By Corollary 18, with probability at most $e^{-\frac{\epsilon^2}{3}t}$, a sequence S in Z_1 has more than $(\alpha + \epsilon)t$ mutations (recall the setting for β at Definition 9) among the first left t characters. The total is $\sum_{t=y}^{\infty} e^{-\frac{\epsilon^2}{3}t} = \frac{c^y}{1-c}$. ∎

Lemma 21 shows that Improve-Boundaries() has good chance to improve the accuracy of rough motif boundaries. Note that $\mathrm{LB}(S)$ and $\mathrm{RB}(S)$ are the left and right motif boundaries of S respectively (see Definition 11).

Lemma 21. *Assume that $\Theta(n, G, \alpha)$ sequence S_i has $L_{S_i} \in [\mathrm{LB}(S_i) - L, \mathrm{LB}(S_i) + L]$ and $R_{S_i} \in [\mathrm{RB}(S_i) - L, \mathrm{RB}(S_i) + L]$ for $i = 1, 2$. Then for* $(\mathrm{roughLeft}_{S_1}, \mathrm{roughRight}_{S_1}, \mathrm{roughLeft}_{S_2}, \mathrm{roughRight}_{S_2})$*=Improve-Boundaries*$(S_1, L_{S_1}, R_{S_1}, S_2, L_{S_2}, R_{S_2}, L)$*, we have the following two facts:*

i. With probablity at most $\frac{2c^v}{1-c} + \frac{2(v+u)c^{v+u}}{(1-c)^2} + \frac{1}{5 \cdot 2^x n}$, $\mathrm{roughLeft}_{S_i}$ is not in $[\mathrm{LB}(S_i) - (v + u), \mathrm{LB}(S_i)]$ for $i = 1, 2$.

ii. With probablity at most $\frac{2c^v}{1-c} + \frac{2(v+u)c^{v+u}}{(1-c)^2} + \frac{1}{5 \cdot 2^x n}$, $\mathrm{roughRight}_{S_i}$ is not in $[\mathrm{RB}(S_i), \mathrm{RB}(S_i) + (v + u)]$ for $i = 1, 2$.

iii. Improve-Boundaries$(S_1, L_{S_1}, R_{S_1}, S_2, L_{S_2}, R_{S_2}, L)$ *runs in* $O(L^2 \log n)$ *time.*

Proof: We need a bound for the following inequality:

$$\sum_{i=j}^{\infty} ia^i < \frac{ja^j}{(1-a)^2}. \tag{20}$$

Let $f(x) = \sum_{i=j}^{\infty} e^{\theta ix}$. Compute the derivative $f'(x) = \theta \sum_{i=j}^{\infty} ie^{\theta ix}$. We also have the closed form for the function $f(x) = \frac{e^{\theta jx}}{1-e^{\theta x}}$, which implies

$$\begin{aligned} f'(x) &= \frac{\theta j e^{\theta jx}(1-e^{\theta x}) - e^{\theta jx}(-\theta e^{\theta x})}{(1-e^{\theta x})^2} && (21) \\ &= \frac{\theta j e^{\theta jx} - \theta(j-1)e^{\theta(j+1)x}}{(1-e^{\theta x})^2}. && (22) \end{aligned}$$

Let $\theta = \ln a$ and $x = 1$. 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 i. By Lemma 20, with probability at most $2\frac{c^v}{1-c}$, one of the left motif first y characters region of S_i will change $\frac{\beta}{2}y$ characters. Therefore, with probability at most $P_1 = 2\frac{c^v}{1-c}$, $\mathrm{roughLeft}_{S_i} > \mathrm{LB}(S_i)$.

For a pair of positions p in S_1 and q in S_2, without loss generality, assume that p has larger distance to the left boundary $\mathrm{LB}(S_1)$ of S_1 than q to the left boundary $\mathrm{LB}(S_2)$ of S_2. Let $v + y$ be the distance from p to the left boundary $\mathrm{LB}(S_1)$ of S_1.

By Lemma 55, the probability is at most c^{v+y} that there will be a match. There are at most $(v + y)$ cases for q. With probability at most $P_2 = 2\sum_{y=u}^{\infty}(v + y)c^{v+y} < \frac{2(v+u)c^{v+u}}{(1-c)^2}$ by inequality (20), $\mathrm{roughLeft}_{S_1} < LB(S_1) - (v + u)$.

For the cases that one position is in random region and has distance more than $d_0 \log n$ from the left boundary, the probability is at most $P_3 = n^2 c^{d_0 \log n} < \frac{1}{5 \cdot 2^x n}$ by inequality (14).

Therefore, we have total probability at most $P_1 + P_2 + P_3$ that roughLeft_{S_1} is not in $[\text{LB}(S_1) - (v+u), \text{LB}(S_1)]$.

Statement ii. One can also provide a symmetric analogous proof for this statement.

Statement iii. The computation time easily follows from the implementation of Improve-Boundaries$(S_1, L_{S_1}, R_{S_1}, S_2, L_{S_2}, R_{S_2})$. ∎

Lemma 22. *Assume that for each L with $0 < L \leq \frac{|G|}{2}$, with probability at most $\varsigma(n)$, $L_{S_i} \notin [\text{LB}_{S_i} - L, \text{LB}_{S_i} + L]$ for $i = 1, 2$, where $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ =Collision-Detection(S_1, U_1, S_2, U_2), U_1 =Point-Selection(S_1, L), and U_2 =Point-Selection(S_2, L). Then with probability at most $\varsigma(n) + \frac{2(v+u_1)c^{v+u_1}}{(1-c)^2} + \frac{c^v}{1-c} + \frac{1}{5 \cdot 2^x n}$, Initial-Boundary$(S_1, S_2)$ returns $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ with $L_{S_i} \notin [\text{LB}(S_i) - (v+u_1), \text{LB}(S_i)]$ or $R_{S_i} \notin [\text{RB}(S_i), \text{RB}(S_i) + (v+u_1))]$ for $i = 1, 2$;*

Proof: It follows from Lemma 21. ∎

Lemma 23. *Assume that with probability $p < 0.5$, each S'_{2i-1} has its rough boundaries $\text{roughLeft}_{S'_{2i-1}} \notin [\text{LB}(S'_{2i-1}) - u, \text{LB}(S'_{2i-1})]$ or $\text{roughRight}_{S'_{2i-1}} \notin [\text{RB}(S'_{2i-1}), \text{RB}(S'_{2i-1}) + u]$, then with probability at most $e^{-(0.5-p-\epsilon)^2 k_1/3}$, l_{motif} is not in $[|G| - 2u, |G| + 2u]$, where l_{motif} is selected as median of $\cup_{i=1}^{k_1}\{(\text{roughRight}_{S'_{2i-1}} - \text{roughLeft}_{S'_{2i-1}})\}$.*

Proof: If both $\text{roughLeft}_{S'_{2i-1}} \in [\text{LB}(S'_{2i-1}) - u, \text{LB}(S'_{2i-1})]$ and $\text{roughRight}_{S'_{2i-1}} \in [\text{RB}(S'_{2i-1}), \text{RB}(S'_{2i-1}) + u]$, then $(\text{roughRight}_{S'_{2i-1}} - \text{roughLeft}_{S'_{2i-1}})$ is in $[|G| - 2u, |G| + 2u]$.

If the median of $\cup_{i=1}^{k_1}\{(\text{roughRight}_{S'_{2i-1}} - \text{roughLeft}_{S'_{2i-1}})\}$ is not in $[|G| - 2u, |G| + 2u]$, then there are at least $\lfloor k_1 \rfloor$ numbers i to have $\text{roughLeft}_{S'_{2i-1}} \notin [\text{LB}(S'_{2i-1}) - u, \text{LB}(S'_{2i-1})]$ or $\text{roughRight}_{S'_{2i-1}} \notin [\text{RB}(S'_{2i-1}), \text{RB}(S'_{2i-1}) + u]$.

On the other hand, the probability is at most p, $\mathrm{roughLeft}_{S'_{2i-1}} \notin [\mathrm{LB}(S'_{2i-1}) - u, \mathrm{LB}(S'_{2i-1})]$ or $\mathrm{roughRight}_{S'_{2i-1}} \notin [\mathrm{RB}(S'_{2i-1}), \mathrm{RB}(S'_{2i-1}) + u]$. So, this lemma follows from Corollary 18.

∎

For a $\Theta(n, G, \alpha)$-sequence S, we often obtain its left rough boundary with $\mathrm{roughLeft}_S \leq \mathrm{LB}(S)$. Some times its exact left boundary may be missed in the algorithm.

Definition 24.

- A $\Theta(n, G, \alpha)$-sequence S misses its left boundary if $\mathrm{roughLeft}_S > \mathrm{LB}(S)$.
- A $\Theta(n, G, \alpha)$-sequence S misses its right boundary if $\mathrm{roughRight}_S < \mathrm{RB}(S)$.

Definition 25.

- A $\Theta(n, G, \alpha)$-sequence S contains a *left half stable* motif region $\aleph(S)$ if $\mathrm{diff}(G'[1, h], G[1, h]) \leq \frac{\beta}{2}$ for all $h = v, v+1, \cdots, m$, where $G' = \aleph(S)$, $c = e^{-\frac{\epsilon^2}{3}}$ and $m = |G|$ as defined in Definition 8 .
- A $\Theta(n, G, \alpha)$-sequence S contains a *right half stable* motif region $\aleph(S)$ if $\mathrm{diff}(G'[m - h, m], G[m - h, m]) \leq \frac{\beta}{2}$ for $h = v - 1, v + 1, \cdots, m - 1$, where $G' = \aleph(S)$ and $m = |G|$.
- A $\Theta(n, G, \alpha)$-sequence S contains a *stable* motif region $\aleph(S)$ satisfying the following conditions: (1) $G'[i] = G[i]$ for $i = 1, \cdots, v - 1$; (2) $G'[m - i + 1] = G[m - i + 1]$ for $i = 1, \cdots, v - 1$; (3) S motif region is both left and right half stable, where $G' = \aleph(S)$ and $m = |G|$.

Lemma 26. *Assume that*

- $l_{motif} \in [|G| - 2(v+u_1), |G| + 2(v+u_1)]$;

- S *contains a both left half and right half stable motif region and* $\text{roughLeft}_S \in [\text{LB}(S) - (v+u_1), \text{LB}(S)]$ *and* $\text{roughRight}_S \in [\text{RB}(S), \text{RB}(S) + (v+u_1)]$ *(see Definition 8 for* u_1 *and* v*); and*

- *for each* L *with* $(v+u_1) < L \leq \frac{|G|}{2}$*, if* S_1 *has* $\text{roughLeft}_{S_1} \notin [\text{LB}_{S_1} - L, \text{LB}_{S_1} + L]$ *and* $\text{roughRight}_{S_1} \notin [\text{RB}_{S_1} - L, \text{RB}_{S_1} + L]$*, then with probability at most* $\varsigma(n)$*,* $L_{S_i''} \notin [\text{LB}_{S_i''} - 2L, \text{LB}_{S_i''} + 2L]$ *for* $i = 1, 2$*, where* $(L_{S_1}, R_{S_1}, L_{S_i''}, R_{S_i''})$ *=Collision-Detection(*S_1, U_1, S_i'', U_2*),* U_1 *=Point-Selection*$(S_1, L, [\text{roughLeft}_{S_1} - 2L, \text{roughLeft}_{S_1} + 2L]) \cup$ *Point-Selection*$(S_1, L, [\text{roughRight}_{S_1} - 2L, \text{roughRight}_{S_1} + 2L])$*, and* U_2 *=Point-Selection*$(S_i'', L, [1, |S_i''|])$.

- *The rough boundaries for all sequences* $S_i'' \in Z_2$ *are computed via* $(L_S, R_S, L_{S_i''}, R_{S_i''})$ *=Collision-Detection*$(S, U_S, S_i'', U_{S_i''})$*,*

  and $(L_S, R_S, \text{roughLeft}_{S_i''}, \text{roughRight}_{S_i''})$*=Improve-Boundaries*$(S, L_S, R_S, S_i'', L_{S_i''}, R_{S_i''}, 2L)$.

Then with probability at most $e^{-\frac{\epsilon^2 k_2}{3}}$*, there are more than* $(2(\varsigma(n) + (v+u_1)\frac{c^{v+u}}{1-c} + \frac{c^v}{1-c}) + \epsilon)k_2$ *sequences* S_i'' *in* $\{S_1'', \cdots, S_{k_2}''\}$ *with* $\text{roughLeft}(S_i'') \notin [\text{LB}(S_i'') - (v+u), \text{LB}(S_i'')]$ *or* $\text{roughRight}(S_i'') \notin [\text{RB}(S_i''), \text{RB}(S_i'') + (v+u)]$.

Proof: According to the condition of this lemma, with probability at most $P_1 = \varsigma(n)$, $L_{S_i''} \notin [\text{LB}_{S_i''} - 2L, \text{LB}_{S_i''} + 2L]$, where $(L_S, R_S, L_{S_i''}, R_{S_i''})$ =Collision-Detection(S, U_1, S_i'', U_2) and (U_1, U_2) =Point-Selection(S, S_i'', L).

For a fixed pattern from S, by Lemma 55, with probability at most $\sum_{y=v+u}^{\infty} c^y = \frac{c^{v+u}}{1-c}$, it has distance more than $v+u$ to the true left boundary. As we need to deal with $v+u_1$ possible patterns

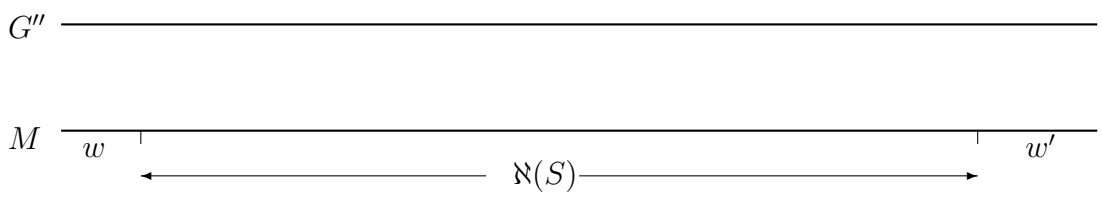

Figure 1: G'' and M

from S, with probability at most $P_{2,l} = (v+u_1)\frac{c^{v+u}}{1-c}$, $\text{roughLeft}_{S_i''} < \text{LB}(S_i'') - (v+u)$.

Similarly, with probability at most $P_{2,r} = (v+u_1)\frac{c^{v+u}}{1-c}$, $\text{roughRight}_{S_i''} < \text{RB}(S_i'') + (v+u)$. Let $P_2 = P_{2,l} + P_{2,r}$.

With probability at most $P_{3,l} = \frac{c^v}{1-c}$, S_i'' does not contain a left half stable motif region by Lemma 20. Similarly, with probability at most $P_{3,r} = \frac{c^v}{1-c}$, S_i'' does not contain a right half stable motif region. Let $P_3 = P_{3,l} + P_{3,r}$.

Although S is involved to search the left boundary with all other sequences. The non-missing condition is to let each sequence do not change too many characters in the motif region. Therefore, this is an independent event for each sequence. It is safe to use Chernoff bound to deal with it.

With probability at most $P = e^{-\frac{\epsilon^2 k_2}{3}}$, the are more than $(P_1 + P_2 + P_3 + \epsilon)k_2$ sequences S_i'' in $\{S_1'', \cdots, S_{k_2}''\}$ with $\text{roughLeft}(S_i'') \notin [\text{LB}(S_i'') - (v+u), \text{LB}(S_i'')]$ or $\text{roughRight}(S_i'') \notin [\text{RB}(S_i''), \text{RB}(S_i'') + (v+u)]$.

∎

Analysis of Extract-Phase and Voting-Phase of Algorithm Recover-Motif

Lemma 27 shows that with high probability, the left and last parts of the motif region in a $\Theta(n, G, \alpha)$-sequence do not change much.

Lemma 27. *With probability at most Q_0, a $\Theta(n, G, \alpha)$-sequence S does not contain a stable motif*

region.

Proof: The probability is $V_1 = 2(v-1)\alpha$ not to satisfy conditions (1) and (2) of Definition 25. Consider condition (3). Since every character of $\aleph(S)[1,m]$ (notice that $m = |G|$) has probability at most α to mutate, by Corollary 18, 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 8. 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 $V_4 = \sum_{r=v}^{\infty} e^{-\frac{1}{3}\epsilon^2 r} \leq \frac{c^v}{1-c}$ for the probability on the right-hand side. 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 $h \in \{v, v+1, \cdots, m\}$. The probability that S does not contain a stable motif region is at most $V_1 + V_3 + V_4 = Q_0$. ∎

Definition 28. Assume that $Z_1 = \{S'_1, \cdots, S_{2k_1}\}$ contains S'_{2i-1} that contains a stable motif region. We fix such a S'_{2i-1}.

- Define $G_L = \aleph(S'_{2i-1})[1, d_0 \log n - 1]$ to be the left part of the motif region $\aleph(S'_{2i-1})$.
- Define $G_R = \aleph(S'_{2i-1})[|G| - (d_0 \log n) + 1, |G|]$ to be the right part of the motif region $\aleph(S'_{2i-1})$.

Lemma 29 shows that with high probability, Extract-Phase of algorithm Recover-Motif extracts the correct motif regions from the sequences in Z_1. It uses G'' to match $\aleph(S)$ in another sequences S. The parameter R gives a small probability that the matched region between G'' and S is not in $\aleph(S)$.

Lemma 29.

i. Assume that G_l and G_r are fixed sequences of length $d_0 \log n$. Let S be a $\Theta(n, G, \alpha)$-sequence with $M \in \text{Match}(G_l, G_r, S)$ 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 that $w_0 \geq 1$, where R is defined in Definition 8.

ii. The probability is at most Q_0 that given a $\Theta(n, G, \alpha)$-sequence S, $\text{Match}(G_L, G_R, S) = \emptyset$.

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

Statement i: There are two cases.

Case (a): $1 \leq w < v$. By Lemma 55, the probability for this case is at most $\frac{1}{t}$ for a fixed w. The total probability for this case for $1 \leq w < v$ is at most $\sum_{i=1}^{v-1} \frac{1}{t^i} \leq \sum_{i=1}^{\infty} \frac{1}{t^i} = \frac{1}{t-1}$.

Case (b): $v \leq w$. By Lemma 55, the probability is at most $e^{-\frac{\epsilon^2}{3}w}$ for a fixed w. The total probability for $v \leq w$ 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 this case is at most $R = (\frac{1}{t-1} + \frac{c^v}{1-c})$ for $w_0 \geq 1$.

Statement ii: By Lemma 27, with probability at most Q_0, S does not contain a stable motif region. Therefore, we have probability at most Q_0 that given a random $\Theta(n, G, \alpha)$-sequence S, $\text{Match}(G_L, G_R, S) = \emptyset$. ∎

Lemma 30 shows that we can use G_l and G_r to extract most of the motif regions for the sequences in Z_2 if $G' = G_L$ (recall that G_L is defined right after Lemma 27).

Lemma 30. *Assume that G_l and G_r are two sequences of length $d_0 \log n$, and G_i =*

$\mathrm{Match}(G_l, G_r, S_i'')$ *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* $|G_l|$*).*

i. If $G_l = G_L$*,* $G_r = G_R$*, and there are no more than* yk_2 *(*$y \in [0, 1]$*) sequences* S_i'' *with* $\mathrm{roughLeft}_{S_i''} \notin [\mathrm{LB}(S_i'') - (v + u_2), \mathrm{LB}(S_i'')]$ *or* $\mathrm{roughRight}_{S_i''} \notin [\mathrm{RB}(S_i''), \mathrm{RB}(S_i'') + (v + u_2)]$*, then the probability is at most* $e^{-\frac{\epsilon^2 k_2}{3}}$ *that there are more than* $(Q_0 + y + \epsilon)k_2$ *sequences* G_i *with* $G_i = \emptyset$*.*

ii. For arbitrary G_l *and* G_r*, with probability at most* $e^{-\frac{\epsilon^2 k_2}{3}}$*,* $|\{i | G_i \neq \emptyset \text{ and } G_i \neq \aleph(S_i''), i = 1, \cdots, k_2\}| > (R + \epsilon)k_2$*, where* R *is defined in Definition 8.*

Proof: Recall that sequence G_{1L} is selected right after Lemma 27.

Statement i: By Lemma 29, for every $S_i'' \in Z_2$, the probability is at most Q_0 that S_i'' does not contain a stable motif region $\aleph(S_i'')$. By Corollary 18, we have probability at most $e^{-\frac{\epsilon^2 k_2}{3}}$ that there are more than $(Q_0 + y + \epsilon)k_2$ sequences G_i with $G_i = \emptyset$.

Statement ii: By Lemma 29, the probability is at most R that $G_i \neq \aleph(S_i'')$. By Corollary 18, with probability at most $e^{-\frac{\epsilon^2 k_2}{3}}$, $|\{i | G_i \neq \aleph(S_i''), i = 1, \cdots, k_2\}| > (R + \epsilon)k_2$. ∎

Definition 31.

- Given two sequences G_r and G_r, define

  $M(G_r, G_r) = \{G_i'' : G_i'' = \textbf{Match}(G_l, G_r, \mathrm{roughLeft}_{S_i''}, \mathrm{roughRight}_{S_i''}, S_i'')\ i = 1, \cdots, k_2\}$.

- For a $\Theta(n, G, \alpha)$ sequence S, define $G_{S,L}$ to be the $\aleph(S)[1, d_0 \log n]$, which is the leftmost subsequence of length $d_0 \log n$ in the motif region of S.

- For a $\Theta(n, G, \alpha)$ sequence S, define $G_{S,R}$ to be the $\aleph(S)[m - d_0 \log n + 1, m]$, which is the rightmost subsequence of length $d_0 \log n$ in the motif region of S, where $m = |G| = |\aleph(S)|$.

the condition iv of Lemma 32

Lemma 32. *Assume that we have the following conditions:*

i. For each L with $0 < L \leq \frac{|G|}{2}$, with probability at most $\varsigma_1(n)$, $L_{S_i} \notin [\mathrm{LB}_{S_i} - 2L, \mathrm{LB}_{S_i} + 2L]$ and $R_{S_i} \notin [\mathrm{RB}_{S_i} - 2L, \mathrm{RB}_{S_i} + 2L]$ for $i = 1, 2$, where $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ =Collision-Detection(S_1, U_1, S_2, U_2), U_1 =Point-Selection($S_1, L, [1, |S_1|]$), and U_2 =Point-Selection($S_2, L, [1, |S_2|]$).

ii. For each L with $0 < L \leq \frac{|G|}{2}$, if S_1 has $\mathrm{roughLeft}_{S_1} \notin [\mathrm{LB}_{S_1} - L, \mathrm{LB}_{S_1} + L]$ and $\mathrm{roughRight}_{S_1} \notin [\mathrm{RB}_{S_1} - L, \mathrm{RB}_{S_1} + L]$, then with probability at most $\varsigma_2(n)$, $L_{S_i''} \notin [\mathrm{LB}_{S_i''} - 2L, \mathrm{LB}_{S_i''} + 2L]$ for $i = 1, 2$, where $(L_{S_1}, R_{S_1}, L_{S_i''}, R_{S_i''})$ =Collision-Detection(S_1, U_1, S_i'', U_2), U_1 =Point-Selection($S_1, L, [\mathrm{roughLeft}_{S_1} - 2L, \mathrm{roughLeft}_{S_1} + 2L]$)$\cup$ Point-Selection($S_1, L, [\mathrm{roughRight}_{S_1} - 2L, \mathrm{roughRight}_{S_1} + 2L]$), and U_2 =Point-Selection($S_i'', L, [1, |S_i''|]$).

iii. The inequality $(P_0 + Q_0) < c_0$ holds for some constant $c_0 < 1$, where Q_0 is defined at equation (13) and $P_0 = \varsigma_1(n) + \frac{2(v+u_1)c^{v+u_1}}{(1-c)^2} + \frac{c^v}{1-c} + \frac{1}{5 \cdot 2^x n}$.

iv. The inequality $1 - 2(Q_0 + V_0 + (R + 2\epsilon)) - (\alpha + \epsilon) > 0$ holds, where $V_0 = (2(\varsigma_2(n) + (v + u_1)\frac{c^{v+u_2}}{1-c} + \frac{c^v}{1-c}) + \epsilon)$.

Then the algorithm generates a set of at most k_2 subsequences for voting and votes a sequence G' such that

(1) with probability at most $e^{-\Omega(k_1)} + e^{-\Omega(k_2)}$, $|G'| \neq |G|$, and

(2) for each $1 \leq i \leq |G|$, with probability at most $e^{-\Omega(k_1)} + e^{-\Omega(k_2)}$, $G'[i] \neq G[i]$.

Before proving Lemma 29, we note that both $\varsigma_1(n)$ and $\varsigma_2(n)$ are at most $\frac{1}{2^x n^3}$ for all of the three algorithms. They will be proved by Lemma 41 and Lemma 42 for

the case algorithm-type=RANDOMIZED-SUBLINEAR, Lemma 44 and Lemma 45 for the case algorithm-type=RANDOMIZED-SUBQUADRATIC, and Lemma 48 for the case algorithm-type=DETERMINISTIC-SUPERQUADRATIC.

Proof:

By Lemmas 22, with probability at most $P_0 = \varsigma_1(n) + \frac{2(v+u_1)c^{v+u_1}}{(1-c)^2} + \frac{c^v}{1-c} + \frac{1}{5 \cdot 2^x n}$, $\text{roughLeft}_{S'_{2i-1}} \notin [\text{LB}(S'_{2i-1}) - (v+u_1), \text{LB}(S'_{2i-1})]$ or $\text{roughRight}_{S'_{2i-1}} \notin [\text{RB}(S'_{2i-1}), \text{RB}(S'_{2i-1}) + (v+u_1)]$.

By Lemma 23, with probability at most $P_a = e^{-(0.5-P_0-\epsilon)^2 k_1/3} = e^{\Omega(k_1)}$, the approximate motif length l_{motif} is not in the range $[|G| - 2(v+u_1), |G| + 2(v+u_1)]$.

By Lemma 27, with probability at most Q_0, a $\Theta(n, G, \alpha)$ sequence does not contain a stable motif region. Therefore, with probability at most $P_1 = (P_0 + Q_0)^{k_1}$, the following statement is false.

(i) One of S'_{2i-1} for $i = 1, \cdots, k_1$ has $\text{roughLeft}_{S'_{2i-1}} \in [\text{LB}(S'_{2i-1}) - (v+u_1), \text{LB}(S'_{2i-1})]$, $\text{roughRight}_{S'_{2i-1}} \in [\text{RB}(S'_{2i-1}), \text{RB}(S'_{2i-1}) + (v+u_1)]$, and has a stable motif region.

By Lemma 26, with probability at most $P_2 = e^{-\frac{\epsilon^2 k_2}{3}}$, there are more than $(2(\varsigma_2(n) + (v+u_1)\frac{c^{v+u_2}}{1-c} + \frac{c^v}{1-c}) + \epsilon)k_2$ sequences S''_i with $\text{roughLeft}_{S''_i} \notin [\text{LB}(S''_i) - (v+u_2), \text{LB}(S''_i)]$ or $\text{roughRight}_{S''_i} \notin [\text{RB}(S''_i), \text{LB}(S''_i) + (v+u_2)]$. In other words, with probability at most P_2, the following statement is false:

(ii) There are no more than $V_0 k_2$ sequences S''_i with $\text{roughLeft}_{S''_i} \notin [\text{LB}(S''_i) - (v+u_2), \text{LB}(S''_i)]$ or $\text{roughRight}_{S''_i} \notin [\text{RB}(S''_i), \text{RB}(S''_i) + (v+u_2)]$, where $V_0 = (2(\varsigma_2(n) + (v+u_1)\frac{c^{v+u_2}}{1-c} + \frac{c^v}{1-c}) + \epsilon)$.

Assume that Statement (ii) is true. By Lemma 30, with probability at most $P_3 = c^{k_2}$, the following statement is false.

(iii) $M(G_L, G_R)$ contains at most $(Q_0 + V_0 + \epsilon)k_2$ empty sequences.

We start from the rough left boundary $\mathrm{roughLeft}_1$ of S_1 to match the other left boundaries of S_i'' for $i = 1, \cdots, k_2$. There are totally at most $2(v + u_1)$ candidates to consider.

By Lemma 30, if $M(G_l, G_r)$, which consists k_2 matched regions, has at most $(Q_0 + V_0 + \epsilon)k_2$ empty sequences, then it has more than $(R + \epsilon)k_2$ from non-motif regions with probability at most $P_4 = 2(v + u_1)e^{-\frac{\epsilon^2 k_2}{3}}$. After the pattern is fixed, those events in the matching are considered to be independent to each other. This is why we can apply the Chernoff bound to deal with them. So, the probability is at most P_4, the following statement is false.

(iv). If $M(G_l, G_r)$ contains at most $(Q_0+V_0+\epsilon)k_2$ empty sequences, then $M(G_l, G_r)$ contains at most $(Q_0 + V_0 + \epsilon + (R + \epsilon))k_2 = (Q_0 + V_0 + (R + 2\epsilon))k_2$ elements not from motif regions $\{\aleph(S_i'') : 1 \leq i \leq k_2\}$.

Therefore, with probability at most $P_1 + P_2 + P_3 + P_4 = e^{-\Omega(k_1)} + e^{-\Omega(k_2)}$, the sequences are not ready for voting in the next phase, which means the following two conditions are satisfied:

(a). There exists G_l and G_r generated by the algorithm such that $M(G_l, G_r)$ contains at most $(Q_0 + V_0 + (R + 2\epsilon))k_2$ elements not from motif regions $\{\aleph(S_i'') : 1 \leq i \leq k_2\}$.

(b). For every G_l and G_r that $M(G_l, G_r)$ contains at most $(Q_0 + V_0 + \epsilon)k_2$ empty sequences generated by the algorithm, $M(G_l, G_r)$ contains at most $(Q_0 + V_0 + \epsilon + (R + \epsilon))k_2 = (Q_0 + V_0 + (R + 2\epsilon))k_2$ elements not from motif regions $\{\aleph(S_i'') : 1 \leq i \leq k_2\}$.

Statement (1): For a $M(G_l, G_r)$ with at most $(Q_0 + V_0 + (R + 2\epsilon))k_2$ elements not from motif regions $\{\aleph(S_i'') : 1 \leq i \leq k_2\}$, we still have $k_2 - (Q_0 + V_0 + (R + 2\epsilon))k_2$ elements in $M(G_l, G_r)$ from motif regions $\{\aleph(S_i'') : 1 \leq i \leq k_2\}$. By the condition (iv) in this lemma, we have $k_2 - (Q_0 + V_0 + (R + 2\epsilon))k_2 > (Q_0 + V_0 + (R + 2\epsilon))k_2$. Therefore, $|G'|$ is selected to be the length of G in the Voting-Phase().

Statement (2): For a $M(G_l, G_r) = \{G_1'', \cdots, G_{k_2}''\}$ with at most $(Q_0+V_0+(R+2\epsilon))k_2$ elements

not from motif regions $\{\aleph(S_i'') : 1 \leq i \leq k_2\}$, we still have $k_2 - (Q_0 + V_0 + (R + 2\epsilon))k_2$ elements in $M(G_l, G_r)$ from motif regions $\{\aleph(S_i'') : 1 \leq i \leq k_2\}$. By Corollary 18, with probability at most $e^{-\frac{\epsilon^2 k_2}{3}}$ there are more than $(\alpha + \epsilon)k_2$ characters that are mutated in the same position among all k_2 the motif regions for the sequences in Z_2. We have that $k_2 - (Q_0 + V_0 + (R + 2\epsilon))k_2 - (\alpha + \epsilon)k_2 > (Q_0 + V_0 + (R + 2\epsilon))k_2$ by the condition (iv) in this lemma. We let $G'[j]$ be the most frequent character among $G_1''[j], \cdots, G_{k_2}''[j]$ in Voting-Phase. Therefore, with probability at most $e^{-\Omega(k_1)} + e^{-\Omega(k_2)}$, $G'[j] \neq G[j]$.

∎

We will use multiple variable functions to characterize the computational time for three algorithms. In order to unify the complexity analysis of three algorithms, we introduce the following notation.

Definition 33. A function $T(x, y) : N \times N \to N$ is monotonic if it is monotonic on both variables. If for arbitrary positive constants c_1 and c_2, $T(c_1 x, c_2 y) \leq cT(x, y)$ for some positive constant c, then $T(x, y)$ is *slow*.

Lemma 34. *Assume that $T(x, y)$, $s(n, L)$ and $g(n, l)$ are monotonic slow functions. Assume that Collision-Detection(S_1, U_1, S_2, U_2) returns the result in $t(n, ||U_1|| + ||U_2||)$ time and the Point-Selection(S_1, S_2, L)) selects $s(n, L)$ positions in $g(n, L)$ time. Assume that with probability at most $\varphi(n)$, the function Initial-Boundaries() does not stop when $L \leq |G|/4$, and $||U_{S'_{2i-1}}|| + ||U_{S''_j}||$ in the algorithm Recover-Motif is no more than $f(n, |G|)$.*

Then with probability at most $k_1\varphi(n)$, the entire algorithm Recover-Motif does not stop in the time complexity $(O(k_1(\sum_{i=1}^{i_0}(T(n, s(n, \frac{n}{2^i n^{2/5}})) + g(n, \frac{n}{2^{i_0} n^{2/5}}))) + k_1 h^2 \log n + k_1 k_2 t(n, f(n, |G|)) +$

$h^2 \log n) + k_1 k_2 (\log n)(\log \log n)), O(k_2))$, *where* i_0 *is the largest* j *such that* $\frac{n}{2^j n^{2/5}} \leq \min(n^{2/5}, |G|)$ *and* $h = \min(n^{2/5}, |G|)$.

Proof: The function Initial-Boundaries()is executed k_1 times. According to the condition that with probability at most $\varphi(n)$, the function Initial-Boundaries(.) does not stop when $L \leq |G|/4$, we have the fact that with probability at most $k_1\varphi(n)$, one of those executions of Initial-Boundaries(.) does not stop when $L \leq |G|/4$.

In the rest of the proof, we assume that all executions of Initial-Boundaries(.) stops when $L \leq |G|/4$.

When $L = O(h)$, we detect rough left and right motif boundaries and run Improve-Boundaries(), which takes $O(h^2 \log n)$ time. It takes $O(\sum_{i=1}^{i_0}(T(n, s(n, \frac{n}{2^i n^{2/5}})) + g(n, \frac{n}{2^i n^{2/5}}) + h^2 \log n)$ time to run Initial-Boundaries(S'_{2i-1}, S'_{2i}) one time for one pair (S'_{2i-1}, S'_{2i}) in Z_1. It takes $O(k_1(\sum_{i=1}^{i_0}(t(n, s(n, \frac{n}{2^i n^{2/5}})) + g(n, \frac{n}{2^i n^{2/5}}) + k_1 h^2 \log n)$ time to run Initial-Boundaries(S'_{2i-1}, S'_{2i}) one time for all pairs (S'_{2i-1}, S'_{2i}) in Z_1.

It takes $k_2(t(n, f(n, |G|)) + h^2 \log n)$ time to find the rough boundaries for all sequences in Z_2 with a fixed sequence S from Z_1 by executing the for loop "For each $S''_j \in Z_2$" in the algorithm Recover-Motif. It takes $k_1 k_2(t(n, f(n, |G|)) + h^2 \log n)$ time to find the rough boundaries for all sequences in Z_2 via all sequences S'_{2i-1} from Z_1 through for loop "For each $S''_j \in Z_2$" in the algorithm Recover-Motif.

Recall that parameters v and u_1 are constants, and u_2 is $O(\log \log n)$. Calling Match(G_l, G_r, S''_i) takes $O((v + u_2) \log n)$ time for each $S''_i \in Z_2$. The total times for calling Match(G_l, G_r, S''_i) is $O(k_1 k_2 (v + u_1)(v + u_2) \log n) = O(k_1 k_2 (\log n)(\log \log n))$.

The voting part takes $O(k_2)$ time for executing voting for recovering one character in motif.

Randomized Algorithms for Motif Detection

In this section, we present two randomized algorithms for motif detection. The first one is a sublinear time algorithm that can handle $\frac{1}{(\log n)^{2+\mu}}$ mutation, and the second one is a super-linear time algorithm that can handle $\Omega(1)$ mutation. They also share some common functions.

Lemma 35. *Let c be a constant in $(0,1)$. Assume m and n are two non-negative integers with $m \leq n$. Then for every integer m_1 with $0 \leq m_1 \leq \frac{\delta_c m}{\ln n}$, $\binom{n}{m_1} c^m \leq e^{(m \ln c)/2}$, where constant $\delta_c = \frac{-\ln c}{2}$ as defined in Definition 8.*

Proof: We have the inequalities

$$\begin{aligned}
\binom{n}{m_1} c^m &\leq n^{m_1} c^m && (23)\\
&= e^{m_1 \ln n} c^m && (24)\\
&\leq e^{\frac{\delta_c m}{\ln n} \ln n} c^m && (25)\\
&= e^{\delta_c m} e^{m \ln c} && (26)\\
&= e^{(m \ln c)/2} && (27)
\end{aligned}$$

Lemma 36. *Let $S = U \cup V$ be a set of n elements with $U \cap V = \emptyset$. Assume that $x_1, \cdots, x_m$ are m random elements in S. Then with probability at most $\binom{||U||}{m_1} (\frac{||V||+m_1}{n})^m$, the list $x_1, \cdots, x_m$ contains at most m_1 different elements from U (in other words, $||\{ x_1, \cdots, x_m\} \cap U|| \leq m_1$).*

Proof: For a subset $S' \subseteq S$ with $|S'| = m_1$, the probability is at most $(\frac{m_1}{n})^m$ that all elements $x_1, \cdots, x_m$ are in S'. For every subset $X \subseteq S$ with $|X| \leq m_1$, there exists another subset $S' \subseteq S$ such that $|S'| = m_1$. We have that $\Pr[||\{ x_1, \cdots, x_m\} \cap U|| \leq m_1] \leq \Pr[\{x_1, \cdots, x_m\} \cap U \subseteq U' \text{ for some } U' \subseteq U \text{ with } ||U'|| = m_1]$. There are $\binom{||U||}{m_1}$ subsets of U with size m_1. We have the probability at most $\binom{||U||}{m_1}(\frac{||V||+m_1}{n})^m$ that $x_1, \cdots, x_m$ contains at most m_1 different elements in U.

∎

Lemma 37. *Let δ be the same as that in Lemma 35. Let β be a constant in $(0, 1)$ and $c = 1 - \frac{\beta}{2}$. Let $m_1 \leq \frac{\delta_c m}{\ln \beta n}$ and $m \leq n^{1-\epsilon}$ for some fixed $\epsilon > 0$. Let S_1 and S_2 be two sets of n elements with $|S_1 \cap S_2| \geq \beta n$ and C be a set of size $|C| \leq \gamma m_1$ for some constant $\gamma \in (0, 1)$. Then for all large n, with probability at most $2e^{-\frac{(1-\gamma)m_1 m}{n}}$, we have $(A-C)\cap(B-C) = \emptyset$, where $A = \{x_1, \cdots, x_m\}$ and $B = \{y_1, \cdots, y_m\}$ are two sets, which may have multiplicities, of m random elements from S_1 and S_2, respectively.*

Proof: In the entire proof of this lemma, we always assume that n is sufficiently large. We are going to give an upper bound about the probability that B does not contain any element in $A - C$. For each element $y_i \in B$, with probability at most $1 - \frac{m_1}{n}$, y_i is not in A. Therefore, the probability is at most $(1 - \frac{||A||-||C||}{n})^m$ that B does not contain any element in $A - C$.

By Lemma 36, the probability is at most $\binom{\beta n}{m_1}(\frac{(1-\beta)n+m_1}{n})^m$ that $||A \cap (S_1 \cap S_2)|| \leq m_1$. We have the inequalities

$$\Pr[(A - C) \cap (B - C) = \emptyset] \tag{28}$$

$$= \Pr[(A - C) \cap (B - C) = \emptyset \mid ||A \cap (S_1 \cap S_2)|| \geq m_1] \cdot \Pr[||A \cap (S_1 \cap S_2)|| \geq m_1] + \tag{29}$$

$$\Pr[(A-C)\cap(B-C)=\emptyset|\ ||A\cap(S_1\cap S_2)||<m_1]\cdot\Pr[|A\cap(S_1\cap S_2)|<m_1] \quad (30)$$

$$\leq \Pr[(A-C)\cap(B-C)=\emptyset|\ ||A\cap(S_1\cap S_2)||\geq m_1]+\Pr[||A\cap(S_1\cap S_2)||<m_1] \quad (31)$$

$$\leq (1-\frac{||(A\cap S_1\cap S_2)||-||C||}{n})^m+\binom{\beta n}{m_1}(\frac{(1-\beta)n+m_1}{n})^m \quad (32)$$

$$\leq (1-\frac{(1-\gamma)m_1}{n})^m+\binom{\beta n}{m_1}(\frac{(1-\beta)n+m_1}{n})^m \quad (33)$$

$$\leq e^{-\frac{(1-\gamma)m_1 m}{n}}+\binom{\beta n}{m_1}(\frac{(1-\beta)n+m_1}{n})^m \quad (34)$$

$$\leq e^{-\frac{(1-\gamma)m_1 m}{n}}+\binom{\beta n}{m_1}(1-\frac{\beta}{2})^m \quad (35)$$

$$\leq e^{-\frac{(1-\gamma)m_1 m}{n}}+e^{(m\ln c)/2} \quad (36)$$

$$\leq 2e^{-\frac{(1-\gamma)m_1 m}{n}}. \quad (37)$$

The inequality $(1-\frac{(1-\gamma)m_1}{n})^m \leq e^{-\frac{(1-\gamma)m_1 m}{n}}$, which is used from (33) to (34), follows from the fact that $1-x\leq e^{-x}$. The transition from (34) to (35) follows from the fact $\frac{m_1}{n}\leq\frac{\beta}{2}$ since $m_1=o(n)$ according to the conditions of the lemma.

It is easy to see that $\frac{2(1-\gamma)m_1 m}{-m\ln c}=\frac{2(1-\gamma)m_1}{-\ln c}\leq n$ for all large n. Thus, $\frac{(1-\gamma)m_1 m}{n}\geq(m\ln c)/2$ (note that $\ln c<0$ as $c\in(0,1)$). Thus, by Lemma 35, $\binom{\beta n}{m_1}(1-\frac{\beta}{2})^m\leq e^{m\ln c/2}\leq e^{-\frac{(1-\gamma)m_1 m}{n}}$. This is why we have the transition from (36) to (37). Therefore, $\Pr[(A-C)\cap(B-C)=\emptyset]\leq 2e^{-\frac{(1-\gamma)m_1 m}{n}}$. ∎

Sublinear Time Algorithm for $\frac{1}{(\log n)^{2+\mu}}$ Mutation Rate

In this section, we give an algorithm for the case with at most $\frac{1}{(\log n)^{2+\mu}}$ mutation rate. The performance of the algorithm is stated in Theorem 2.

Definition 38. A position p in the motif region $\aleph(S)$ of an input sequence S is *damaged* if there exists at least one mutation in $S[p,p+d_0\log n-1]$.

Lemma 39. *Assume that* $\alpha L = (\log n)^{1+\Omega(1)}$. *With probability at most* $e^{-(\log n)^{1+\Omega(1)}}$, *there are more than* $\frac{M_1}{(\log n)^{\Omega(1)}}$ *positions that are from the* M *sampled positions in an interval of length* L *and are damaged.*

Proof: By Theorem 17, with probability at most $P_1 = 2^{-\alpha L}$ (let $\delta = 2$), there are more than $3\alpha L$ mutation in an interval of length L. Therefore, with probability at most $2^{-\alpha L} = e^{-(\log n)^{1+\Omega(1)}}$, there are more than $3\alpha L \log n$ positions that are damaged. Therefore, each random position in an interval of length L has at most probability $\frac{3\alpha L \log n}{L} = 3\alpha \log n$ to be damaged.

Since $\alpha = (\frac{1}{(\log n)^{2+\Omega(1)}})$ and M positions are sampled, by Theorem 17, with probability at most $P_2 = 2^{-(3\alpha \log n)M} = e^{-(\log n)^{1+\Omega(1)}}$ (let $\delta = 2$), the number of damaged positions sampled in an interval of length L is more than $(1+\delta)3\alpha \log n)M = (9\alpha \log n)M = \frac{M_1}{(\log n)^{\Omega(1)}}$. Thus, with total probability at most $P_1 + P_2 = e^{-(\log n)^{1+\Omega(1)}}$, there are more than $\frac{M_1}{(\log n)^{\Omega(1)}}$ damaged positions that are from the M sampled positions in an interval of length L. ∎

Definition 40. Let A be a set of positions in an input sequence S with $\aleph(S) = [i, j]$. Let $A(S, \aleph(S)) = A \cap [i, j]$.

Lemma 41. *Assume that* $|G| \geq \frac{(\log n)^{3+\tau}}{100}$ *and* $d_0 \log n \leq L \leq |G|/2$. *Let* I_1 *be a union of intervals that include* $[\mathrm{LB}(S_1) - 2L, \mathrm{LB}(S_1) + 2L]$ *and* $[\mathrm{RB}(S_1) - 2L, \mathrm{RB}(S_1) + 2L]$. *Let* U_1 =*Point-Selection*(S_1, L, I_1), U_2 =*Point-Selection*$(S_2, L, [1, |S_2|])$, *and* $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ =*Collision-Detection*$(S_1, U_1, S_2, , U_2)$. *Then*

i. With probability at most $\frac{1}{2^x n^3}$, *the left rough boundary* $Ł_{S_1}$ *has at most* $2L$ *distance from* $\mathrm{LB}(S_1)$ *and the left rough boundary* L_{S_2} *has at most* $2L$ *distance from* $\mathrm{LB}(S_2)$.

ii. *With probability at most* $\frac{1}{2^x n^3}$*, the right rough boundary* R_{S_1} *has at most* $2L$ *distance from* $\mathrm{RB}(S_1)$*; and the right boundary of* R_{S_2} *has at most* $2L$ *distance from* $\mathrm{RB}(S_2)$.

Proof: We prove the following two statements which imply the lemma.

i. With probability at most $\frac{1}{2^x n^3}$, there is no intervals A_i from S_1 and B_j from S_2 such that (1) $|A_i(S_1, \aleph(S_1)) \cap B_j(S_2, \aleph(S_2))|$ is at least $\frac{L}{2}$; (2) the left boundary of S_1 has at most $2L$ distance from A_i; (3) the left boundary of S_2 has at most $2L$ distance from B_j; and (4) there is collision between the sampled positions in A_i and B_j.

ii. With probability at most $\frac{1}{2^x n^3}$, there are no intervals A_i from S_1 and B_j from S_2 such that (1) $|A_i(S_1, \aleph(S_1)) \cap B_j(S_2, \aleph(S_2))|$ is at least $\frac{L}{2}$; (2) the right boundary of S_1 has at most $2L$ distance from A_i; (3) the right boundary of S_2 has at most $2L$ distance from B_j; and (4) there is collision between the sampled positions in A_i and B_j.

We only prove the statement i. The proof for statement ii is similar to that for statement i. Note that L goes down by half each cycle in the algorithm. Assume that L satisfies the condition of this lemma.

Select A_i from S_1 and B_j from S_2 to be the first pair of intervals with $||A_i(S_1, \aleph(S_1)) \cap B_j(S_2, \aleph(S_2))|| \geq \frac{L}{2}$. It is easy to see that such a pair exists and both have distance from the left boundary with distance at most $2L$. This is because when an leftmost interval of length L is fully inside the motif region of the first sequence, we can always find the second interval from the second sequence with intersection of length at least $\frac{L}{2}$.

Replace m by $M(L)$, m_1 by $M_1(L)$ (see Definition 10), and n by L to apply Lemma 37. We also let C be the set of damaged positions affected by the mutated positions. With probability at

most $o(\frac{1}{2^x n^3})$, C has size more than $\Omega(M_1(L))$ by Lemma 39. With probability at most $o(\frac{1}{2^x n^3})$, there is no intersection A_i from S_1 and B_j from S_2. ∎

Lemma 42. *Assume that* $|G| < \frac{(\log n)^{3+\tau}}{100}$ *and* L *is an integer with* $d_0 \log n \leq L \leq |G|/2$. *Let* I_1 *be a union of intervals that include* $[\mathrm{LB}(S_1) - 2L, \mathrm{LB}(S_1) + 2L]$ *and* $[\mathrm{RB}(S_1) - 2L, \mathrm{RB}(S_1) + 2L]$. *Let* U_1 =*Point-Selection*(S_1, L, I_1), U_2 =*Point-Selection*$(S_2, L, [1, |S_2|])$, *and* $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ =*Collision-Detection*$(S_1, U_1, S_2, , U_2)$. *Then*

i. With probability at most $\frac{1}{2^x n^3}$, *the left rough boundary* L_{S_1} *has at most* $|G|/4$ *distance from* $\mathrm{LB}(S_1)$ *and the left rough boundary* L_{S_2} *has at most* $|G|/4$ *distance from* $\mathrm{LB}(S_2)$.

ii. With probability at most $\frac{1}{2^x n^3}$, *the right rough boundary* R_{S_1} *has at most* $|G|/4$ *distance from* $\mathrm{RB}(S_1)$*; and the right boundary of* R_{S_2} *has at most* $|G|/4$ *distance from* $\mathrm{RB}(S_2)$.

Proof: For two sequences S_1 and S_2, it is easy to see that there is a common position in both motif regions of the two sequences such that there is no mutation in the next $d_0 \log n$ characters with high probability. This is because that mutation probability is small.

By Theorem 17, with probability at most $P_{l,1} = 2^{-\alpha|G|/4}$ (let $\delta = 2$), there are more than $3\alpha\frac{|G|}{4}$ mutated characters in the interval $\aleph(S_i)[1, \frac{|G|}{4}]$ for $i = 1, 2$. Therefore, with probability at most $2^{-\alpha|G|/4} = e^{-(\log n)^{1+\Omega(1)}}$, there are more than $3\alpha\frac{|G|}{4}\log n$ positions that are damaged in $\aleph(S_i)[1, \frac{|G|}{4}]$.

Since the mutation probability is $\alpha = (\frac{1}{(\log n)^{2+\Omega(1)}})$ and $M(L)$ positions are sampled, with probability at most $P_{l,2} = 2^{-(3\alpha d_0 \log n)\frac{|G|}{4}} = e^{-(\log n)^{1+\Omega(1)}}$ (with $\delta = 2$), the number of damaged positions is more than $((5\alpha d_0 \log n)\frac{|G|}{4}) = \frac{|G|}{(\log n)^{\Omega(1)}}$ by Theorem 17. The probability is $P_l = P_{l,1} + P_{l,2} = e^{-(\log n)^{1+\Omega(1)}}$ that left side has more than $((5\alpha d_0 \log n)\frac{|G|}{4}) = \frac{|G|}{(\log n)^{\Omega(1)}}$ damaged positions.

We have similar $P_r = P_{r,1} + P_{r,2} = e^{-(\log n)^{1+\Omega(1)}}$ probability for the right side for more than $((5\alpha d_0 \log n)\frac{|G|}{4}) = \frac{|G|}{(\log n)^{\Omega(1)}}$ damaged positions in $\aleph(S_i)[\frac{3|G|}{4} - 1, |G|]$.

Now we assume that left side has more than $((5\alpha d_0 \log n)\frac{|G|}{4}) = \frac{|G|}{(\log n)^{\Omega(1)}}$ damaged positions and the right side for more than $((5\alpha d_0 \log n)\frac{|G|}{4}) = \frac{|G|}{(\log n)^{\Omega(1)}}$ damaged positions in $\aleph(S_i)[\frac{3|G|}{4} - 1, |G|]$. Since each position in each interval of length L is selected in Point-Selection(S_1, S_2, L), it is easy to verify the conclusions of this lemma. ∎

Lemma 43. *For the case* algorithm-type=RANDOMIZED-SUBLINEAR*, we have*

i. CollisionDetection(S_1, U_1, S_2, U_2) takes $t(n, ||U_1|| + ||U_2||) = O((||U_1|| + ||U_2||) \log n)$ time.

ii. Point-Selection($S_1, L, [1, |S_1|]$) selects $s(n, L) = O((\frac{n}{L})M(L))$ positions in $g(n, L) = O(s(n, L))$ time if $L \geq \frac{(\log n)^{3+\tau}}{100}$.

iii. Point-Selection($S_1, L, [1, |S_1|]$) selects $s(n, L) = O(n)$ positions in $g(n, L) = O(n)$ time if $L < \frac{(\log n)^{3+\tau}}{100}$.

iv. $||U_{S'_{2i-1}}|| + ||U_{S''_j}||$ in the algorithm Recover-Motif is no more than $f(n, |G|) = O(M(|G|) + \frac{n}{|G|}M(|G|))$.

v. With probability at most $\frac{k}{2^x n^3}$, the algorithm Recover-Motif does not stop in $(O(k(\frac{n}{\sqrt{h}}(\log n)^{\frac{5}{2}} + h^2 \log n)), O(k))$ time.

Proof: Statement i. The parameter $\omega_{\text{RANDOMIZED-SUBLINEAR}}$ is set to be 0 in the Collision-Detection. It follows from the time complexity of bucket sorting, which is described in standard algorithm textbooks.

Statements ii and iii. They follows from the implementation of Point-Selection().

Statement iv. It follows from the choice of Point-Selection(.) for the sublinear time algorithm at Recover-Motif(.).

Statement v. It follows from Lemma 42, Lemma 41, Lemma 34 and Statements i, ii, and iii, and iv. ∎

We give the proof for Theorem 2.

Proof: [Theorem 2] The computational time part of this theorem follows from Lemma 43.

By Lemma 41, Lemma 42, we can let $\varsigma_1(n) = \frac{1}{2^x n^3} \leq \varsigma_0$ for the probability bound $\varsigma_1(n)$ in the condition (i) of Lemma 32.

By Lemma 41, Lemma 42, we can let $\varsigma_2(n) = \frac{1}{2^x n^3} \leq \varsigma_0$ for the probability bound $\varsigma_1(n)$ in the condition (ii) of Lemma 32.

By inequality (12), the condition (iii) of Lemma 32 is satisfied.

By inequality (11), we know that the condition (iv) of Lemma 32 can be satisfied.

The failure probability part of this theorem follows from Lemma 21, and Lemma 32 by using the fact that k_1, k_2, and k are of the same order (see equation (18)). ∎

Randomized Algorithm for $\Omega(1)$ Mutation Rate

In this section, we give an algorithm for the case with $\Omega(1)$ mutation rate. The performance of the algorithm is stated in Theorem 4.

Lemma 44. *Assume that $d_0 \log n \leq L \leq |G|/2$ and $|G| \geq \frac{(\log n)^{3+\tau}}{100}$. Let I_1 be a union of intervals that include $[\mathrm{LB}(S_1) - 2L, \mathrm{LB}(S_1) + 2L]$ and $[\mathrm{RB}(S_1) - 2L, \mathrm{RB}(S_1) + 2L]$. Let U_1 =Point-Selection(S_1, L, I_1), U_2 =Point-Selection$(S_2, L, [1, |S_2|])$, and $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ =Collision-Detection$(S_1, U_1, S_2, , U_2)$. Then*

i. With probability at most $\frac{1}{2^x n^3}$, the left rough boundary $Ł_{S_1}$ has at most $2L$ distance from $\mathrm{LB}(S_1)$ *and the left rough boundary L_{S_2} has at most $2L$ distance from* $\mathrm{LB}(S_2)$.

ii. With probability at most $\frac{1}{2^x n^3}$, the right rough boundary R_{S_1} has at most $2L$ distance from $\mathrm{RB}(S_1)$*; and the right boundary of R_{S_2} has at most $2L$ distance from* $\mathrm{RB}(S_2)$.

Proof: We prove the following two statements which imply the lemma.

i. With probability at most $\frac{1}{2^x n^3}$, there are no intervals A_i from S_1 and B_j from S_2 such that (1) $||A_i(S_1, \aleph(S_1)) \cap B_j(S_2, \aleph(S_2))||$ is at least $\frac{L}{2}$; (2) The left boundary of S_1 has at most $2L$ distance from A_i; (3) The left boundary of S_2 has at most $2L$ distance from B_j; and (4) There is collision between the sampled positions in A_i and B_j.

ii. With probability at most $\frac{1}{2^x n^3}$, there are no intervals A_i from S_1 and B_j from S_2 such that (1) $||A_i(S_1, \aleph(S_1)) \cap B_j(S_2, \aleph(S_2))||$ is at least $\frac{L}{2}$; (2) The right boundary of S_1 has at most $2L$ distance from A_i; (3) The right boundary of S_2 has at most $2L$ distance from B_j; and (4) There is collision between the sampled positions in A_i and B_j.

We only prove the statement i. The proof for statement ii is similar. Note that L goes down by half each cycle in the algorithm. Assume that L_0 satisfies the condition of this lemma, and let $L = L_0$ happen in the algorithm.

Select A_i from S_1 and B_j from S_2 to be the first pair of intervals with $||A_i(S_1, \aleph(S_1)) \cap B_j(S_2, \aleph(S_2))|| \geq \frac{L}{2}$. It is easy to see that such a pair exists and both have distance from the left boundary with distance at most $2L$. This is because when a leftmost interval of length L is fully inside the motif region of the first sequence, we can always find the second interval from the second sequence with intersection of length at least $\frac{L}{2}$.

Replace m by $M(L)$, m_1 by $M_1(L)$ (see Definition 10), and n by L to apply Lemma 37. We do not consider any damaged position in this algorithm, therefore, let C be empty. With probability at most $o(\frac{1}{2^x n^3})$, there is no intersection A_i from S_1 and B_j from S_2. ∎

Lemma 45. *Let U_1 and U_2 contain all positions of the input sequences S_1 and S_2, respectively. Assume $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ =Collision-Detection$(S_1, U_1, S_2, , U_2)$. Then*

i. With probability at most $\frac{1}{2^x n^3}$, the left rough boundary L_{S_1} has at most $d_0 \log n$ distance from $\mathrm{LB}(S_1)$ and the left rough boundary L_{S_2} has at most $d_0 \log n$ distance from $\mathrm{LB}(S_2)$.

ii. With probability at most $\frac{1}{2^x n^3}$, the right rough boundary R_{S_1} has at most $d_0 \log n$ distance from $\mathrm{RB}(S_1)$; and the right boundary of R_{S_2} has at most $d_0 \log n$ distance from $\mathrm{RB}(S_2)$.

Proof: For two sequences S_1 and S_2, let $\aleph(S_a)$ be the subsequence $S_a[i_a, j_a]$ for $a = 1, 2$. By Corollary 18, with probability at most $P_l = 2c^{d_0 \log n} \leq \frac{2}{5 \cdot 2^x n^3}$ (see inequality 8 at Definition 14), there are more than $(\alpha + \epsilon) d_0 \log n$ mutations in $S_a[i_a, i_a + d_0 \log n - 1]$ for $a = 1, 2$.

In this case, every position in the two sequences S_1 and S_2 is selected by Point-Selection(S_1, S_2). With probability at most P_l, the left boundary position is missed during the matching. We have similar P_r to miss the right boundary.

Assume that p_1 and p_2 are two positions of S_1 and S_2 respectively. If one of two positions is outside the motif region and has more than $d_0 \log n$ distance to the motif boundary, with probability at most $c^{-d_0 \log n} \leq \frac{1}{5 \cdot 2^x n^3}$ (see inequality 8 at Definition 14) for them to match that requires $\mathrm{diff}(Y_1, Y_2) \leq \beta$ by Lemma 55, where Y_a is a subsequence $S_a[p_a, p_a + d_0 \log n - 1]$ for $a = 1, 2$.

∎

Lemma 46. *Assume that* $d_0 \log n \leq L \leq |G|/2$ *and* $c_0 \log n \leq |G| < \frac{(\log n)^{3+\tau}}{100}$. *Let* I_1 *be a union of intervals that include* $[\mathrm{LB}(S_1) - 2L, \mathrm{LB}(S_1) + 2L]$ *and* $[\mathrm{RB}(S_1) - 2L, \mathrm{RB}(S_1) + 2L]$. *Let* U_1 =*Point-Selection*(S_1, L, I_1), U_2 =*Point-Selection*$(S_2, L, [1, |S_2|])$, *and* $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ =*Collision-Detection*$(S_1, U_1, S_2, , U_2)$. *Then*

i. With probability at most $\frac{1}{2^x n^3}$, *the left rough boundary* L_{S_1} *has at most* $d_0 \log n$ *distance from* $\mathrm{LB}(S_1)$ *and the left rough boundary* L_{S_2} *has at most* $d_0 \log n$ *distance from* $\mathrm{LB}(S_2)$.

ii. With probability at most $\frac{1}{2^x n^3}$, *the right rough boundary* R_{S_1} *has at most* $d_0 \log n$ *distance from* $\mathrm{RB}(S_1)$*; and the right boundary of* R_{S_2} *has at most* $d_0 \log n$ *distance from* $\mathrm{RB}(S_2)$.

Proof: In this case, every position in the two sequences S_1 and S_2 is selected by Point-Selection(S_1, S_2). It follows from Lemma 45. ∎

Lemma 47. *For the case* algorithm-type=RANDOMIZED-SUBQUADRATIC*, we have*

i. CollisionDetection(S_1, U_1, S_2, U_2) *takes* $t(n, ||U_1|| + ||U_2||) = O((||U_1|| + ||U_2||)^2 \log n)$ *time.*

ii. Point-Selection$(S_1, L, [1, |S_1|])$ *selects* $s(n, L) = O((\frac{n}{L})M(L))$ *positions in* $g(n, L) = O(s(n, L))$ *time if* $L \geq \frac{(\log n)^{3+\tau}}{100}$.

iii. Point-Selection$(S_1, L, [1, |S_1|])$ *selects* $s(n, L) = O(n)$ *positions in* $g(n, L) = O(n)$ *time if* $L < \frac{(\log n)^{3+\tau}}{100}$.

iv. $||U_{S'_{2i-1}}|| + ||U_{S''_j}||$ *in the algorithm Recover-Motif is no more than* $f(n, |G|) = O(M(|G|) + \frac{n}{|G|}M(|G|))$.

v. *With probability at most* $\frac{k}{2^x n^3}$*, the algorithm Recover-Motif does not stop in* $(O(k(\frac{n^2}{|G|}(\log n)^{O(1)}+ h^2 \log n)), O(k))$ *time.*

Proof: Statement i. The parameter $\omega_{\text{RANDOMIZED}-\text{SUBLINEAR}}$ is set to be β in the Collision-Detection. It follows from the time complexity of brute force method.

Statements ii and iii. They follow from the implementation of Point-Selection().

Statement iv. It follows from the choice of Point-Selection(.) for the sublinear time algorithm at Recover-Motif(.).

Statement iv. It follows from Lemma 45, Lemma 46, Lemma 34, and Statements i, ii, and iii. ∎

We give the proof for Theorem 6.

Proof: [Theorem 4] The computational time part of this theorem follows from Lemma 47.

By Lemma 44, Lemma 45, we can let $\varsigma_1(n) = \frac{1}{2^x n^3} \leq \varsigma_0$ for the probability bound $\varsigma_1(n)$ in the condition (i) of Lemma 32.

By Lemma 44, Lemma 45, we can let $\varsigma_2(n) = \frac{1}{2^x n^3} \leq \varsigma_0$ for the probability bound $\varsigma_2(n)$ in the condition (i) of Lemma 32.

By inequality (12), the condition (iii) of Lemma 32 is satisfied.

By inequality (11), we know that the condition (iv) of Lemma 32 can be satisfied.

The failure probability part of this theorem follows from Lemma 21, and Lemma 32 by using the fact that k_1, k_2, and k are of the same order (see equation (18)). ∎

Deterministic Algorithm for $\Omega(1)$ Mutation Rate

In this section, we give a deterministic algorithm for the case with $\Omega(1)$ mutation rate. The performance of the algorithm is stated in Theorem 6.

Lemma 48. *Assume that* $d_0 \log n \leq L \leq |G|/2$ *and* $c_0 \log n \leq |G|$. *Let* I_1 *be a union of intervals that include* $[\mathrm{LB}(S_1) - 2L, \mathrm{LB}(S_1) + 2L]$ *and* $[\mathrm{RB}(S_1) - 2L, \mathrm{RB}(S_1) + 2L]$. *Let* U_1 =*Point-Selection*(S_1, L, I_1), U_2 =*Point-Selection*$(S_2, L, [1, |S_2|])$, *and* $(L_{S_1}, R_{S_1}, L_{S_2}, R_{S_2})$ =*Collision-Detection*$(S_1, U_1, S_2, , U_2)$. *Then*

i. *With probability at most* $\frac{1}{2^x n^3}$, *the left rough boundary* L_{S_1} *has at most* $d_0 \log n$ *distance from* $\mathrm{LB}(S_1)$ *and the left rough boundary* L_{S_2} *has at most* $d_0 \log n$ *distance from* $\mathrm{LB}(S_2)$.

ii. *With probability at most* $\frac{1}{2^x n^3}$, *the right rough boundary* R_{S_1} *has at most* $d_0 \log n$ *distance from* $\mathrm{RB}(S_1)$; *and the right boundary of* R_{S_2} *has at most* $d_0 \log n$ *distance from* $\mathrm{RB}(S_2)$.

Proof: In this case, every position in the two sequences S_1 and S_2 is selected by Point-Selection(S_1, S_2). It follows from Lemma 45. ∎

Lemma 49. *For the case* algorithm-type=DETERMINISTIC-SUPERQUADRATIC, *we have*

i. *CollisionDetection*(S_1, U_1, S_2, U_2) *takes* $t(n, ||U_1|| + ||U_2||) = O((||U_1|| + ||U_2||)^2 \log n)$ *time.*

ii. *Point-Selection*$(S_1, L, [1, |S_1|])$ *selects* $s(n, L) = O(n)$ *positions in* $g(n, L) = O(n)$ *time.*

iii. $||U_{S'_{2i-1}}|| + ||U_{S''_j}||$ *in the algorithm Recover-Motif is no more than* $f(n, |G|) = O(|G| + n)$.

iv. *With probability at most* $\frac{k}{2^x n^3}$, *the algorithm Recover-Motif does not run in computational complexity* $(O(k(n^2(\log n)^{O(1)} + h^2 \log n)), O(k))$.

Proof: Statement i. The parameter $\omega_{\mathrm{DETERMINISTIC-SUPERQUADRATIC}}$ is set to be β in the Collision-Detection. It follows from the time complexity of brute force method.

Statement ii. They follows from the implementation of Point-Selection().

Statement iii. It follows from the choice of Point-Selection(.) for the sublinear time algorithm at Recover-Motif(.).

Statement iv. It follows from Lemma 48, Lemma 34 and Statements i, ii, and iii. ∎

We give the proof for Theorem 6.

Proof: [Theorem 6] The computational time part of this theorem follows from Lemma 49.

By Lemma 48, we let $\varsigma_1(n) = \frac{1}{2^x n^3} \leq \varsigma_0$ for the probability bound $\varsigma_1(n)$ in the condition (i) of Lemma 32.

By Lemma 48, we can let $\varsigma_2(n) = \frac{1}{2^x n^3} \leq \varsigma_0$ for the probability bound $\varsigma_2(n)$ in the condition (i) of Lemma 32.

By inequality (12), the condition (iii) of Lemma 32 is satisfied.

By inequality (11), we know that the condition (iv) of Lemma 32 can be satisfied.

The failure probability part of this theorem follows from Lemma 21, and Lemma 32 by using the fact that k_1, k_2, and k are of the same order (see equation (18)). ∎

Summary

In this chapter, we develop two randomized algorithms and one deterministic algorithm under the probabilistic model. One of them finds the implanted motif with high probability if the alphabet size is at least 4, the motif length is in $[(\log n)^{7+\mu}, \frac{n}{(\log n)^{1+\mu}}]$ and each character in motif region has probability at most $\frac{1}{(\log n)^{2+\mu}}$ of mutation. The motif region can be detected and each motif character can be recovered in sublinear time. A sub-quadratic randomized algorithm is developed to recover the motif with $\Omega(1)$ mutation rate. A quadratic deterministic algorithm is developed to recover the motif with $\Omega(1)$ mutation rate.

CHAPTER IV

ENUMERATIVE ALGORITHMS FOR MOTIF DETECTION

An Overview of Enumerative Algorithm

In the following, Definition 50 characterizes the motifs which are difficult for our algorithm to discover. Lemma 51 shows that such difficult motifs are relatively few. Lemma 51 is based on the fact that for a random sequence, two equal length subsequences are not similar to each other if they are long enough.

Definition 50. Let Σ to be alphabet with at least 2 characters. Let h and m be integers with $x \leq m$.

- Define $\Phi_{m,x,d}(\Sigma)$ be the set of all sequences S in Σ^m such that $\mathrm{diff}(S[i, i+x-1], S[j, j+x-1]) \leq d$ for some two $i \neq j$ with $1 \leq i < j \leq m-x+1$.

- Define $\Psi_{m,x,d}(\Sigma) = \cup_{u=x}^{m} \Phi_{m,u,d}(\Sigma)$.

We have the following lemma that slightly improves the analysis at [5] for $|\Sigma| > 4$.

Lemma 51. *For every constant* $\epsilon \in (0,1)$,
$\frac{|\Psi_{\rho,x,\frac{2}{3}((1-\frac{1}{|\Sigma|})^2-\epsilon)}(\Sigma)|}{|\Sigma^\rho|} \leq 2\rho^2 \cdot \frac{c^{x/12}}{1-c}$ *for all large* x, *where* $c = e^{-\frac{\epsilon^2}{3}} < 1$.

Proof: Assume that S is a random sequence of Σ^ρ. Let $u \geq x$. We consider $S[i, i+u-1]$ and $S[j, j+u-1]$. The probability that the a-th characters of these two subsequences are not the same is $1 - \frac{1}{|\Sigma|}$.

For two triples $(S[i+r], S[j+r], S[j+(j-i)+r])$ and $(S[i+r'], S[j+r'], S[j+(j-i)+r'])$ of characters in S, they are independent if $\{i+r, j+r, j+(j-i)+r\} \cap \{i+r', j+r', j+(j-i)+r'\} = \emptyset$. We can pick up $\left\lfloor \frac{u}{3} \right\rfloor$ independent pairs $(S[i+r], S[j+r], S[j+(j-i)+r])$ for $i = 1, \cdots, \lfloor u/3 \rfloor$. For each triple, we can get an two different positions with probability at least $(1 - \frac{1}{|\Sigma|})^2$.

Let H be the set of those independent pairs. Set H contains at least $\frac{u}{3}$ independent pairs. By Corollary 18, with probability at most

$$\begin{aligned} e^{-\frac{(\frac{\epsilon}{2})^2 \left\lfloor \frac{u}{3} \right\rfloor}{3}} &\leq e^{-\frac{(\frac{\epsilon}{2})^2 (\frac{u}{3}-1)}{3}} \\ &\leq e^{\frac{\epsilon^2}{12}} \cdot e^{-\frac{\epsilon^2 u}{36}} \\ &\leq 2e^{-\frac{\epsilon^2 u}{36}}, \end{aligned}$$

H contains at most $((1 - \frac{1}{|\Sigma|})^2 - \epsilon/2)|H|$ triples $(S[i+r], S[j+r], S[j+(j-i)+r])$ with $S[i+r] \neq S[j+r]$ and $S[j+r] = S[i+(j-i)+r] \neq S[j+(j-i)+r]$. For all large u, $((1 - \frac{1}{|\Sigma|})^2 - \epsilon/2)|H| \geq ((1 - \frac{1}{|\Sigma|})^2 - \epsilon/2) \left\lfloor \frac{u}{3} \right\rfloor \geq \frac{1}{3}((1 - \frac{1}{|\Sigma|})^2 - \epsilon)u$. For all large u, with probability at most $2e^{-\frac{\epsilon^2 u}{36}}$, $\mathrm{diff}(S[i, i+u-1], S[j, j+u-1]) \leq \frac{2}{3}((1 - \frac{1}{|\Sigma|})^2 - \epsilon)$. For all large u, with probability at most $2\rho^2 e^{-\frac{\epsilon^2 u}{36}}$, there exist i and j with $1 \leq i < j \leq \rho - u + 1$ such that $\mathrm{diff}(S[i, i+u-1], S[j, j+u-1]) \leq \frac{2}{3}((1 - \frac{1}{|\Sigma|})^2 - \epsilon)$.

For all large x, with probability at most $\sum_{u=x}^{\rho} 2\rho^2 e^{-\frac{\epsilon^2 x}{36}} \leq 2\rho^2 \cdot \frac{c^{x/12}}{1-c}$, there are integers i, j, and u with $1 \leq i < j \leq \rho - u + 1$ and $x \leq u \leq \rho$ such that $\mathrm{diff}(S[i, i+u-1], S[j, j+u-1]) \leq \frac{2}{3}((1 - \frac{1}{|\Sigma|})^2 - \epsilon)$. ∎

For an intuitive understanding of Lemma 51, note that $\Psi_{\rho,x,\epsilon}(\Sigma)$ is a subset of Σ^ρ. If y and ϵ are constants, we can select constant c such that $2\rho^2 \cdot \frac{c^{x/12}}{1-c} < \frac{1}{2^y}$ with $x \geq c \log \rho$. Therefore, $\Psi_{\rho,x,\epsilon}(\Sigma)$ is a small portion of sequences in Σ^ρ when, for example, $y \geq 10$.

Enumerative Algorithm

Parameters Setting

In this section, we set up some parameters that are used in the algorithm.

We first set up some constants and parameters as follows.

We have a signal Detect-Non-$\Psi_{m,x,d}$-Motif to indicate that a motif is in $\Sigma^m - \Psi_{m,x,d}$ if it is set to be true.

Let

$$t = |\Sigma|. \tag{38}$$

Select constants (as large as possible) $\sigma_2 > 0$, and $\sigma_1 > 0$ such that

$$2(\alpha + \sigma_2) \quad < \quad \frac{t-1}{t} - \sigma_1, \text{and} \tag{39}$$

$$\tag{40}$$

If Detect-Non-$\Psi_{m,x,d}$-Motif, and $\alpha < \frac{1}{2}((1 - \frac{1}{|\Sigma|}) - d)$ for positive constants α and d, then select constant $\epsilon > 0$ such that

$$2\alpha + 3\epsilon < ((1 - \frac{1}{|\Sigma|}) - d). \tag{41}$$

It used to control the boundary minimal aligned length.

We define

$$c_1 \quad = \quad e^{\frac{-\sigma_1^2}{3}}, \tag{42}$$

$$c_2 = e^{\frac{-\sigma_2^2}{3}}, \quad \text{and} \tag{43}$$

$$c_3 = e^{\frac{-\epsilon^2}{3}}. \tag{44}$$

Select a positive constant δ_1 such that

$$1 - \alpha - 5\delta_1 - 2\epsilon > \frac{1}{2}. \tag{45}$$

Select a parameter δ_3 and such that

$$(1 - (\delta_3 + \epsilon + \alpha + \epsilon)) > \frac{1}{2} \tag{46}$$

Let l_0 be a parameter and is set to be $\max(\frac{c_1^{l_0}}{1-c_1}, \frac{c_2^{l_0}}{1-c_2}, nc_1^h) \leq \delta_1$.

Let $h = O(\log n)$ such that

$$\max(\frac{2l_0c_3^h}{1-c_3}, \frac{nc_3^h}{1-c_3}) \leq \delta_3. \tag{47}$$

Initial Motif Region

In this section, we detect the initial motif region via comparing two sequences.

Initial-Motif(S_a, S_b, p_a, p_b, r)

Input: two sequences S_a and S_b, a position p_a in S_a, a position p_b in S_b, and an integer r for predicting the length of the motif;

Steps:

Let $p'_a = p_a + r - 1$ and $p'_b = p_b + r - 1$;

(L1) $\mathrm{diff}(S_a[p_a, p_a + i], S_b[p_b, p_b + i]) \leq 2\alpha + 2\sigma_2$ for all $l_0 \leq i \leq h$;

(R1) $\mathrm{diff}(S_a[p'_a, p'_a - i], S_b[p'_b, p'_b - i]) \leq 2\alpha + 2\sigma_2$ for all $l_0 \leq i \leq h$; and

If both conditions are satified, then return $true$; else return false

End of function Initial-Motif

Recover-Motif(A) recovers the motif by voting on the character in each column of A in the motif region, which is between the boundaries returned from the function Detect-Motif-Boundary.

Exact Boundaries and Voting

Let $\mathrm{Maj}(A, j)$ denote the character that appears the largest number of times in the column j of A. For a character $a \in \Sigma$, $\mathrm{Occur}(a, j, A)$ denotes the number of times that a appears in the column j of A.

We will show that if the alignment has a suitably small error, then Detect-Motif-Boundary returns the boundaries of the motif region with high probability if the number of the input sequences is reasonably large.

Detect-Motif-Boundary(A)

Input: a $k \times 3n$ matrix A that holds k aligned sequences $S_1, \cdots, S_k$; and two rough boundary positions (p_L, p_R).

Steps:

Select the leftmost column $j_{\mathrm{L}} \in [p_L - l_0, p_L + l_0]$ in A such that $\mathrm{Occur}(\mathrm{Maj}(A, j_{\mathrm{L}} + 1), j_{\mathrm{L}} + 1, A) \geq \mu_0 k$.

Select the rightmost column $j_{\mathrm{R}} \in [p_R - l_0, p_R + l_0]$ in A such that $\mathrm{Occur}(\mathrm{Maj}(A, j_{\mathrm{R}} - 1), j_{\mathrm{R}} - 1, A) \geq \mu_0 k$.

return $(j_{\mathrm{L}}, j_{\mathrm{R}})$;

End of function Detect-Motif-Boundary

Recover-Motif will detect the columns that contain the motif from a σ_1-error alignment. Each character of the motif is recovered by voting on the characters in the same column. Each column that does not contain motif character does not have a character that appears as frequently as in the motif region. The subroutine Recover-Motif uses a subroutine called Detect-Motif-Boundary(A) that finds a pair of column indices j_{L} and j_{R} such that most copies of the motif in the input sequences will be located from column $j_{\mathrm{L}}+1$ to column $j_{\mathrm{R}}-1$.

Recover-Motif(A) recovers the motif by voting on the character in each column of A in the motif region, which is between the boundaries returned from the function Detect-Motif-Boundary.

Recover-Motif(A)

Input: $k \times 3n$ matrix A that holds k aligned sequences $S_1, \cdots, S_k$;

let $(j_{\mathrm{L}}, j_{\mathrm{R}}) =$ Detect-Motif-Boundary(A);

let g'_s be the character that appears the largest number of times in column $j_{\mathrm{L}}+s$ of A for $0 < s < j_{\mathrm{R}} - j_{\mathrm{L}}$;

return $g'_1 \cdots g'_h$ as the motif G;

End of function Recover-Motif

Combining Them Together

Motif-Detection($S_1, \cdots, S_k$)

i. Set up the parameters according to previous Section.

ii. Find the least p_1^*, some p_2 and the largest r such that Initial-Motif(S_1, S_2, p_1^*, p_2, r) returns nonempty (p_2, p'_2);

iii. For each $p_1 \in [p_1^* - l_0, p_1^* + l_0]$

iv. For each $2 \leq i \leq k$, Initial-Motif(S_1, S_i, p_1, p_i, r)

for all possible position p_i in S_i;

v. Remove those sequences S_i that have more than

$4l_0$ matched positions with S_1;

vi. If there are more than $(1 - \epsilon - 4\delta_1)k$ sequences left

then

vii. Put the sequences $S_1, S_2, \cdots, S_k$ into an $k \times$

$3n$ array A.

viii. Recover-Motif(A);

End of Motif-Detection

Analysis of Algorithm

Definition 52. Assume $\sigma_1 > 0$. Given k $\Theta_\alpha(n, G)$-sequences $S_1, \cdots, S_k$, an *alignment* puts them into k rows such that each sequence S_i is arranged in $|S_i|$ consecutive positions at row i for the $|S_i|$ characters of S_i. An alignment for $S_1, \cdots, S_k$ is a σ_1-*error alignment* for $S_1, \cdots, S_k$ if at least $(1 - \sigma_1)k$ sequences have their $\aleph(S_i)$ in the same columns. We often use an array A of k rows to hold an alignment of k sequences.

Our main algorithm Algorithm-One consists two sub-routines. The first sub-routine Align-Sequences aligns the input sequences so that most copies of the motif are in the same column

regions. The second sub-routine Recover-Motif recovers the motif based on the output of the first sub-routine. The performance of Discover-Motif, Align-Sequences, and Recover-Motif is described in the following theorem.

Theorem 53. *Assume that α with $\alpha < \frac{1}{2}((1 - \frac{1}{|\Sigma|}) - d)$ and x are positive constants. Then, there exist constants $\sigma_1, \delta_1, \delta_2, \epsilon > 0$ such that given an input of $\Theta_\alpha(n, G)$-sequences $S_1, \cdots, S_k$, the algorithm satisfies the following two conditions for all sufficiently large n (the maximum length of the input sequences).*

i. If $G \in \Sigma^\rho - \Psi_{\rho,h,d}(\Sigma)$ and $\rho \geq h + l_0 = O(\log n)$, then with probability at most $e^{-\Omega(h)} + e^{-\Omega(k)}$, it fails to return a unique G' with $|G| = |G'|$, and G and G' are very similar.

ii. For every G, with probability at most $2c_3^k + \delta_1$, it fails to return at most $2^{O(k)}$ sequences $G'_1, \cdots, G'_m$ such that at least one G'_i is similar to G.

iii. The algorithm takes $O(kn^2(\log n)^2)$ time.

Lemma 54 shows that each input sequence has the property that with small probability, its subsequence in its motif region is not similar to the original motif G. The similarity is measured at every subsequence of parameter length s in the motif region.

Lemma 54. *Assume that S is a random $\Theta_\alpha(n, G)$-sequence. Then we have*

i. With probability at most $\frac{e^{\frac{-\epsilon^2 z}{3}}}{1-e^{\frac{-\epsilon^2}{3}}}$, $\mathrm{diff}(\aleph(S)[i, i+s-1], G[i, i+s-1]) > (\alpha + \epsilon)$ for fixed i and s with $z \leq s \leq |G|$ and $1 \leq i \leq |\aleph(S)| - s + 1$.

ii. With probability at most $\frac{ne^{\frac{-\epsilon^2 z}{3}}}{1-e^{\frac{-\epsilon^2}{3}}}$, $\mathrm{diff}(\aleph(S)[i, i+s-1], G[i, i+s-1]) > (\alpha + \epsilon)$ for some i and s with $z \leq s \leq |G|$ and $1 \leq i \leq |\aleph(S)| - s + 1$.

Proof: Each character in the motif region has probability at most α to mutate. By Corollary 18, for each $s \geq z$ and a fixed i, the probability is at most $e^{\frac{-\epsilon^2 s}{3}}$ that $\mathrm{diff}(\aleph(S)[i, i+s-1], G[i, i+s-1]) > (\alpha+\epsilon)$. For a fixed S, the probability for some $s \geq z$ and $i \leq |S| - s$ with $\mathrm{diff}(\aleph(S)[i, i+s-1], G[i, i+s-1]) > (\alpha+\epsilon)$ is at most $n \sum_{s=z}^{\infty} e^{\frac{-\epsilon^2 s}{3}} \leq \frac{n e^{\frac{-\epsilon^2 z}{3}}}{1-e^{\frac{-\epsilon^2}{3}}}$. ∎

Lemma 55. *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 probability for $\mathrm{diff}(X_1, X_2) \leq \frac{t-1}{t} - \epsilon$ is at most $e^{-\frac{\epsilon^2 |X_1|}{3}}$.*

Proof: 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 $\frac{t-1}{t}|X_1|$. The probability for $\mathrm{diff}(X_1, X_2) \leq \frac{t-1}{t} - \epsilon$ is at most $e^{-\frac{\epsilon^2}{3}|X_1|}$ by Corollary 18. ∎

Definition 56. For each sequence S, let $L(S)$ be the left boundary position of the motif in S.

Definition 57. For two sequences S_a and S_b, let p_a and p_b be two positions in them respectively. If the two sequences can be aligned via Initial-Motif(S_a, S_b, p_a, p_b, r) at p_a and p_b with $p_b \notin [L(S_b) - l, L(S_b) + l]$ for some $r \geq m$, then such an alignment is called l-shift alignment.

Lemma 58. *Let p_a be a position in S_a.*

i. If $r \geq m$, with probability at most $2(\frac{c_1^{l_0}}{1-c_1} + nc_1^h) \leq 4\delta_1$, there is a l_0-shift via Initial-Motif(S_a, S_b, p_a, p_b, r).

ii. For all $i > 1$, statements i are independent.

Proof: Statement i. We prove this statement by considering two cases. Each of them needs the condition $r \geq m$.

According to our setting in inequality (39), we have $2(\alpha + \sigma_2) < \frac{t-1}{t} - \sigma_1$.

Assume that $p_b = L(S_b) - x$ and $h \geq x \geq l_0$. By Lemma 55, we have

$$\begin{aligned} \text{Probability}(x \geq l_0) &\leq \sum_{x=l_0}^{+\infty} e^{-\frac{\sigma_1^2 x}{3}} && (48) \\ &\leq \frac{c_1^{l_0}}{1-c_1} && (49) \\ &\leq \delta_1. && (50) \end{aligned}$$

Consider the case that $x > h$, the probability is at most nc_1^h by Lemma 55.

Similaryly, assume that $p_b = L(S_b) + y$ and $y \geq l_0$. As $r \geq m$, there are at least x random characters on the right side of S_b. By Lemma 55, we have

$$\begin{aligned} \text{Probability}(y \geq l_0) &\leq \sum_{y=l_0}^{+\infty} e^{-\frac{\sigma_1^2 y}{3}} && (51) \\ &\leq \frac{c_1^{l_0}}{1-c_1} && (52) \\ &\leq \delta_1. && (53) \end{aligned}$$

Consider the case that $y > h$, the probability is at most nc_1^h by Lemma 55.

Statement ii. When S_1 is fixed, S_i's background is random, and independent each other.

∎

Lemma 59. *Let p_a be a position in S_a. If $r \geq m + 2l_0$, then with probability at most $\frac{2c_1^{l_0}}{1-c_1} \leq \delta_1$, Initial-Motif$(S_a, S_b, p_a, p_b, r)$ returns true.*

Proof: If $r \geq m + 2l_0$, then one of the two sides of have at least l_0 random characters in S_b.

According to our setting in inequality (39), we have $2(\alpha + \sigma_2) < \frac{t-1}{t} - \sigma_1$.

$$\begin{aligned} \text{Probability}(r \geq m + 2l_0) &\leq 2\sum_{x=l_0}^{+\infty} e^{-\frac{\sigma_1^2 x}{3}} && (54) \\ &\leq \frac{2c_1^{l_0}}{1-c_1} && (55) \\ &\leq \delta_1. && (56) \end{aligned}$$

∎

Lemma 60.

i. For each sequence S_a, with probability at most $\frac{2c_2^{l_0}}{1-c_2} \leq \delta_1$, $\text{diff}[G[1,i], \aleph(S_a)[1,i]) \geq \alpha + \sigma_1$ for some $l_0 \leq i \leq m$.

ii. Assume that $p_a = L(S_a)$ and $p_b = L(S_b)$. Then with probability at most $\frac{c_2^{l_0}}{1-c_2}$, p_b does not satisfies the condition in Initial-Motif(S_a, S_b, p_a, m), where m is the length of the motif as it is defined before.

iii. Assume that $\text{diff}[G[1,i], \aleph(S_1)[1,i]) \leq \alpha + \sigma_1$ for some $l_0 \leq i \leq m$. Then with probability at most $e^{\frac{-\epsilon^2 k}{3}} = c_3^k$, there are more than $(\delta_1 + \epsilon)k$ sequences S_i do not satisfy the condition in Initial-Motif(S_1, S_i, p_a, p_b, m).

Proof: By Lemma 54, with probability at most $\frac{c_2^{l_0}}{1-c_2}$, $\text{diff}(\aleph(S)[i, i+s-1], G[i, i+s-1]) > (\sigma_2 + \epsilon)$ for some i and s with $l_0 \leq s \leq |G|$ and $1 \leq i \leq |\aleph(S)| - s + 1$. Thus, we have proved Statements i and ii.

Statement iii follows from Statement i and Corollary 18. ∎

Lemma 61. *With probability at most c_3^k, there are more than $k(4\delta_1+\epsilon)$ sequences that are removed in line v in Motif-Detection(.).*

Proof: It follows from Lemma 58 and Corollary 18. ∎

Definition 62. Two sequences are *well aligned* if their motif region are in the same column region.

Lemma 63. *The output of the algorithm has the following two properties:*

i. The algorithm outputs at most $(2l_0)^k$ possible candidates $G_1, \cdots, G_y$ for the motif.

ii. With probability at most $2c_3^k+\delta_1$, there is no case such that at least $(1-5\delta_1-2\epsilon)k$ sequences are well aligned with S_1.

iii. At least one of them has length equal to m, and for each i, with probability at most c_3^k, $G_j[i] \neq G[i]$.

Proof: Statement i. By Lemma 58, each S_i has at most $2l_0$ possible matched positions.

Statement ii. By Lemma 60, S_1 and S_i align well in the motif region. We have that S_1 and most S_is align well in the motif region.

Let $\tau_1 = (4\delta_1 + \epsilon)$, and $\tau_2 = \delta_1 + \epsilon$. By Lemma 61 and (iii) of Lemma 60, with a small probability at most $2c_3^k + \delta_1$, S_1 does not align well in the motif region with at least $(1 - \tau_1 - \tau_2)k$.

Statement iii. It follows from the voting method. We consider the voting at each column. By Corollary 18, with probability at most $e^{-\frac{\epsilon^2 k}{3}} = c_3^k$, there are fewer than $(1 - \alpha - \epsilon)k$ characters unequal to $G[j']$. Thus, with probability at most $e^{-\frac{\epsilon^2 k}{3}} = c_3^k$, there are fewer than $(1 - \alpha - \epsilon -$

$\tau_1 - \tau_2)k = (1 - \alpha - 5\delta_1 - 2\epsilon)k > \frac{k}{2}$ (by inequality (45)) characters unequal to $G[j']$ in the same column. It is easy to see that with probability at most $e^{-\frac{\epsilon^2 k}{3}}$, $G_[j']' \neq G[j']$.

∎

We put an additional restriction for checking the alignment between S_1 and S_i in Initial-Motif(S_1, S_i, p_1, p_i, r). It requires than $\text{diff}(S_1[p_1, p_1 + h - 1], S_i[p_1, p_1 + h - 1]) \leq P_0 - 2\alpha - 3\epsilon$, and $l_0 \leq \epsilon h$, where $P_0 = d$.

Definition 64. Define $P_3 = 2l_0 \times \frac{e^{\frac{-\epsilon^2 h}{3}}}{1-e^{\frac{-\epsilon^2}{3}}}$.

Lemma 65. *Assume that G is in $\Sigma^m - \Phi_{m,h,d}(\Sigma)$. Then with probability at most $e^{\frac{-\epsilon^2 k}{3}}$, there are more than $(1 - (P_3 + \epsilon))k$ sequences S_i that are not well aligned with S_1.*

Proof: By Lemma 54, we have that with probability at most $\frac{e^{\frac{-\epsilon^2 h}{3}}}{1-e^{\frac{-\epsilon^2}{3}}}$, $\text{diff}(\aleph(S)[i, i + h - 1], G[i, i + h - 1]) > (\alpha + \epsilon)$.

Thus, with probability at most $2l_0 \times \frac{e^{\frac{-\epsilon^2 h}{3}}}{1-e^{\frac{-\epsilon^2}{3}}}$, $\text{diff}(\aleph(S_1)[i, i+h-1], G[i, i+h-1]) > (\alpha+\epsilon)$ for some $i \leq l_0$.

Thus, with probability at most $2l_0 \times \frac{e^{\frac{-\epsilon^2 h}{3}}}{1-e^{\frac{-\epsilon^2}{3}}}$, $\text{diff}(\aleph(S_j)[i, i+h-1], G[i, i+h-1]) > (\alpha+\epsilon)$ for some $i \leq l_0$.

Assume that S_1 and S_i are aligned at the positions p_1 and p_i, respectively, i.e., $\text{diff}(S_1[p_1, p_1 + h-1], S_i[p_i, p_i+h-1]) \leq P_0-2\alpha-3\epsilon$. Let $S_1[p_1, p_1+h-1] = S_1[p_1, p_1+l-1]S_1[p_1+l, p_1+h-1]$ such that $S_1[p_1 + l, p_1 + h - 1]$ is in the motif region of S_1. Let $S_i[p_i, p_i + h - 1] = S_i[p_i, p_i + l - 1]S_i[p_i + l, p_i + h - 1]$ such that $S_i[p_i + l, p_i + h - 1]$ is in the motif region of S_1. We note that $0 \leq l \leq l_0$. Let $S_1[p_1+l, p_1+h-1] = \aleph(S_1)[j_1, j_1+j]$, and $S_i[p_i+l, p_i+h-1] = \aleph(S_i)[j_i, j_i+j]$, where $j = h - l - 2$.

We have $\text{diff}(S_1[p_1, p_1 + h - 1], S_i[p_i, p_i + h - 1]) = \text{diff}(S_1[p_1, p_1 + l - 1], S_i[p_i, p_i + l - 1]) + \text{diff}(\aleph(S_1)[j_1, j_1 + j], \aleph(S_i)[j_i, j_i + j]) \geq \text{diff}(\aleph(S_1)[j_1, j_1 + j], \aleph(S_i)[j_i, j_i + j])$. Thus, $\text{diff}(\aleph(S_1)[j_1, j_1 + j], \aleph(S_i)[j_i, j_i + j]) \leq P_0 - 2(\alpha + \epsilon)$. The $\text{diff}(.)$ function follows the triangle inequality

$$
\begin{aligned}
& \text{diff}(G[j_1, j_1 + h - 1], G[j_i, j_i + h - 1]) && (57)\\
\leq\ & \text{diff}(G[j_1, j_1 + j], G[j_i, j_i + j]) + l_0/h && (58)\\
\leq\ & \text{diff}(G[j_1, j_1 + j], \aleph(S_1)[j_1, j_1 + j]) + && (59)\\
& \text{diff}(\aleph(S_1)[j_1, j_1 + j], \aleph(S_i)[j_i, j_i + j]) + && (60)\\
& \text{diff}(\aleph(S_i)[j_i, j_i + j], G[j_i, j_i + j],) + l_0/h && (61)\\
\leq\ & \alpha + \epsilon + P_0 - 2(\alpha + \epsilon) + \alpha + \epsilon + l_0/h && (62)\\
<\ & P_0. && (63)
\end{aligned}
$$

As those events are independent when S_1 is fixed. By Corollary 18, with probability at most $e^{\frac{-\epsilon^2 k}{3}}$, there are more than $(P_3+\epsilon)k \leq (\delta_3+\epsilon)k$ (by inequality (47)) sequences are not well aligned with S_1.

With probability at most c_3^k, there are at most $(\alpha + \epsilon)k$ mutations in each motif column. By inequality 46, we have $(1 - (\delta_3 + \epsilon + \alpha + \epsilon)) > \frac{1}{2}$. This makes the voting at the corresponding column to fail with a small probability. ∎

Lemma 66. *Assume that A is a σ_1-error alignment for $S_1, \cdots, S_k$. Then with probability at most $2l_0e^{-\Omega(k)}$ and there are at most l_0 shift to the real boundary, the algorithm* Detect-Motif-

Boundary*(A) fails to return $(b_{\rm L}, b_{\rm R})$, where $b_{\rm L}$ and $b_{\rm R}$ are the left and the right boundaries of the motif, respectively.*

Proof: Let $b_{\rm L}$ be the left motif boundary of A and $b_{\rm R}$ be the right motif boundary of A. We only prove in detail below that with probability at most $e^{-\Omega(k)}$, index $j_{\rm L}$ computed by the algorithm Detect-Motif-Boundary(A) is not equal to $b_{\rm L}$. One can also provide a symmetric analogous proof that with probability at most $e^{-\Omega(k)}$, $j_{\rm R} \neq b_{\rm R}$.

Each character in the motif region of each sequence has probability at most α to mutate, and A is a σ_1-error alignment for $S_1, \cdots, S_k$. By Corollary 18, we have the following two facts:

Fact 1. For j with $b_{\rm L} < j < b_{\rm R}$, if column j of A contains $\aleph(S_1)[j']$ with $j' = j - b_{\rm L}$, then with probability at most $e^{-\frac{1}{3}\sigma_2^2 k}$, the character $G[j']$ appears fewer than $(1 - (\alpha + \sigma_1 + \sigma_2))k = \mu_0 k$ times in the column j of A.

Fact 2. For j with $j \leq b_{\rm L}$, with probability at most $e^{-\frac{1}{3}\sigma_2^2 k}$, each character of Σ appears more than $(\frac{1}{|\Sigma|} + (\sigma_1 + \sigma_2))k < \mu_0 k$ times in column j of A (by inequality (39)).

There are l_0 possible shift positions. Thus, with probability at most $2l_0 e^{-\Omega(k)}$, the algorithm Detect-Motif-Boundary(A) fails to return $(b_{\rm L}, b_{\rm R})$, where $b_{\rm L}$ and $b_{\rm R}$ are the left and the right boundaries of the motif, respectively. ∎

Lemma 67. *The he algorithm takes $O(kn^2(\log n)^2)$ time.*

Proof: It takes $O(n^2(\log n)^2)$ time to compare two sequences. It takes $O(kn)$ time for voting. Therefore, the total time is still $O(kn^2(\log n)^2)$. ∎

Proof: [Theorem 53] Part i of Theorem 53 follows from Lemma 65 and Lemma 66. The voting part is similar to Statement iii of Lemma 63.

Part ii of Theorem 53 follows from Lemma 63.

Part iii of Theorem 53 follows from Lemma 67. ∎

Summary

In this chapter, we develop one enumerative algorithm under the same probabilistic model. which can discover a hidden motif from a set of sequences for any alphabet Σ with $|\Sigma| \geq 2$ in $O(kn^2(\log n)^2)$ time, where k is the number of sequences and n is the maximal sequence length. Since the alphabet size is at least 2, thus our algorithm is applicable to both DNA motif discovery and other potential applications.

CHAPTER V

EXPERIMENTS AND RESULTS

Aiming at solving the motif discovery problem, we implemented our randomized and enumerative algorithms in Java separately. Our tests were all done on a laptop with a 1.5 Ghz CPU and 3.0 GB Memory. In the following parts, we will test our algorithms on simulated data and biological data separately. In the first experiment, we tested our algorithms on simulated data sets, which are generated by using our probability model with a small mutation rate. Each input set contains 15 or 20 sequences, and each sequence contains 500 or 600 base pairs. Each base pair of the simulated gene sequences was generated independently with the same occurrence probability. A motif with a fixed length was randomly planted to each input sequence. The minimum Hamming distances between the outputs and original consensus are counted.

Experiments on Simulated Data Sets

Table 1 shows the results of randomized algorithm on simulated data sets. In the table, N is the number of sequences in a set, M is the length of sequences, L is the length of planted motif and R is the number of test repetitions. From the table, we could find that the results of our algorithm for finding motif on simulated data sets are satisfied. Our algorithm could find all the motifs from each sequence and get the consensus with the accuracy rate of 100%. If the data set has a high mutation rate, we could increase the number of repetitions so that the result on the data sets will be improved.

Table 1: Results of Randomized Algorithm on Simulated data

	N	M	L	R	Accuracy Rate
Set 1	20	600	15	60	100
Set 2	15	600	15	10	100
Set 3	20	600	12	15	100
Set 4	20	500	15	40	100

For the enumerative algorithm, we designed two programs by using JAVA and C# separately, the main difference between the two programs is that the program written in C# was added a vector function to calculate the score of each consensus, so that making it better for dealing with high mutation rates.

Table 2: Results of Enumerative Algorithm on Simulated data

	N	M	L	R	Program 1	Program 2
Set 1	20	600	15	60	100	87
Set 2	15	600	15	10	100	80
Set 3	20	600	12	15	100	74
Set 4	20	500	15	40	100	78

Table 2 shows the experimental results of the two programs on simulated data sets. From the table, we could find that the results of our two programs on simulated data sets are encouraging. Program 1 could find all the motifs and consensus from each sequence sets, so the accuracy rates are 100%. Program 2(vector) missed motifs in some specific sequences, therefore its accuracy rates are around 80%.

Experiments on Biological Data Sets

In the second experiment, we tested our algorithms on real sequence sets, which are downloaded from SCPD. SCPD contains a large number of gene data and transcription factors of yeast. Sequences in the same set are all regulated by a common motif. We chose 1000bp as the length of input sequences. In order to show the advantages of our algorithm, we also compared the result of

our algorithm with the results of several other existing motif finding methods on the same data set, such as Gibbs, MEME, Info-Gibbs and Consensus. Table 3 shows the details of the data sets we used in the following experiments.

Table 3: Number of sequences and Motif length

	Bas1	GCN4	GCR1	Rap1Ebf1	HSE-HTSF
N	6	9	6	15	5
L	10	10	10	15	10

Table 4 and 5 show the results of our algorithms on all biological data sets. We also choose some well-known motif-detecting softwares to make comparisons. From the two tables, we see that the average mismatch numbers of program 1 on bas1, GCN4 and GCR1 are greatly lower than other four existed methods. While only on the datasets Rap1Ebf1 and HSE-HTSF, the average mismatch numbers of program 1 is a little higher than MEME. Program 2(vector) also shows good performance in some tests. The accuracy rates of program 2(vector) on some specified data sets are higher than program 1's results, and the accuracy rates of program 2 on all data sets are higher than the average accuracy rates of all other the algorithms. The performance of randomized algorithm is similar to Program 1, but Program 1 has better performance on some specific data sets.

In addition, our algorithms also show their high speeds in computations compared to other four motif finding methods. Because the starting pattern of algorithms are represented by a string, so our algorithm could avoid some extra time consuming computations unlike Gibbs sampling and EM methods, such as computations of likelihoods. According to this feature, we use the consensus string of the voting operation obtained from the last iteration as a new starting pattern in program, and continue doing voting operations repeatedly until there is no further improvement. Experimental results show that if we set the number of iterations to be a large integer, the programs

could give more accurate results within a reasonable time.

Table 4: Total number of Mismatch Positions

	Bas1	GCN4	GCR1	Rap1Ebf1	HSE-HTSF
Randomized Algorithm	10	8	4	45	5
Program 1	6	9	4	42	4
Program 2	38	6	18	37	6
Gibbs	8	51	5	202	7
MEME	8	15	10	32	3
InfoGibbs	9	21	5	46	9
Consensus	8	9	5	42	7

Table 5: Average Mismatch Numbers per Sequence

	Bas1	GCN4	GCR1	Rap1Ebf1	HSE-HTSF
Randomized Algorithm	1.67	0.89	0.67	3	1
Program 1	1	1	0.67	2.8	0.8
Program 2	6.3	0.6	3	2.46	1.2
Gibbs	1.33	5.6	0.83	13.46	1.4
MEME	1.33	1.67	1.67	2.13	0.6
InfoGibbs	1.5	2.33	0.83	3.06	1.8
Consensus	1.33	1	0.83	2.8	1.4

CHAPTER VI

CONCLUSIONS AND FUTURE WORKS

Our algorithms have advantages in some aspects compared to other popular motif finding methods, the experimental results are significant. However, there are still some improvements could be done on this algorithm. As we know, though a set of sequences have the consensus, each sequence's motif may have mutations, and the length of each motif could also be different. So the two factors increase the difficulties in finding unknown motifs. In addition, an interesting open problem is whether there exists an efficient algorithm to recover all the motifs for an alphabet with four characters when the parameter α has a similar bound. In the future, we plan to give some approximation measures to solve above problems and improve the efficiency of our algorithms, so that to make our algorithms working better on discovering unknown motifs.

It is an interesting problem if there is an algorithm to handle the case for the alphabet of size 3. A more interesting problem is to extend the algorithm to handle larger mutation probability.

REFERENCES

1. F. Chin and H. Leung. Voting algorithms for discovering long motifs. In Proceedings of the 3rd Asia-Pacific Bioinformatics Conference, pages 261–272, 2005.

2. J. Dopazo, A. Rodríguez, J. C. Sáiz, and F. Sobrino. Design of primers for PCR amplification of highly variable genomes. Computer Applications in the Biosciences, 9:123–125, 1993.

3. M. Frances and A. Litman. On covering problems of codes. Theoretical Computer Science, 30:113–119, 1997.

4. B. Fu, M.-Y. Kao, and L. Wang. Probabilistic analysis of a motif discovery algorithm for multiple sequences. SIAM Journal Discrete Mathematics, 23(4):1715–173, 2009.

5. 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. ACM Transactions on Algorithms, 7(2):26, 2011.

6. L. Gąsieniec, J. Jansson, and A. Lingas. Efficient approximation algorithms for the Hamming center problem. In Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms, pages S905–S906, 1999.

7. D. Gusfield. Algorithms on Strings, Trees, and Sequences. Cambridge University Press, 1997.

8. 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 3rd International Conference on Bioinformatics and Genome Research, pages 201–216, 1995.

9. U. Keich and P. Pevzner. Finding motifs in the twilight zone. Bioinformatics, 18:1374–1381, 2002.

10. U. Keich and P. Pevzner. Subtle motifs: defining the limits of motif finding algorithms. Bioinformatics, 18:1382–1390, 2002.

11. J. K. Lanctot, M. Li, B. Ma, L. Wang, and L. Zhang. Distinguishing string selection problems. Information and Computation, 185:41–55, 2003.

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

13. 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, pages 473–482, 1999.

14. M. Li, B. Ma, and L. Wang. On the closest string and substring problems. Journal of the ACM, 49(2):157–171, 2002.

15. X. Liu, B. Ma, and L. Wang. Voting algorithms for the motif problem. In Proceedings of Computational Systems Bioinformatics Conference (CSB'08), pages 37–47, 2008.

16. 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. Computer Applications in the Biosciences, 7:525–529, 1991.

17. R. Motwani and P. Raghavan. Randomized Algorithms. Cambridge University Press, 2000.

18. P. Pevzner and S. Sze. Combinatorial approaches to finding subtle signals in DNA sequences. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology*, pages 269–278, 2000.

19. V. Proutski and E. C. Holme. Primer master: a new program for the design and analysis of PCR primers. Computer Applications in the Biosciences, 12:253–255, 1996.

20. G. Stormo. Consensus patterns in DNA, in R. F. Doolitle (ed.), Molecular evolution: computer analysis of protein and nucleic acid sequences. Methods in Enzymolog, 183:211–221, 1990.

21. G. Stormo and G. Hartzell III. 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.

22. L. Wang and L. Dong. Randomized algorithms for motif detection. Journal of Bioinformatics and Computational Biology, 3(5):1039–1052, 2005.

BIOGRAPHICAL SKETCH

Mr.Yuan Xue is currently a graduate student in the department of Computer Science at University of Texas-Pan American, and will obtain his master of science in Computer Science in July 2013. Before that, he got his B.E. in Computer Science and Technology from Wuhan University, P.R.of China.

Permanent mailing address:

Yuan Xue

984 Mailbox, No.64

Beijing, China, 10091