Improving An Exact Solution to the (l, d)-Planted Motif Problem

A Thesis

Presented to the

Faculty of the Graduate School

Ateneo de Manila University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Maria Clara Isabel D. Sia

2015

The thesis entitled:

Improving An Exact Solution to the (l, d)-Planted Motif Problem

submitted by **Maria Clara Isabel D. Sia** has been examined and is recommended for oral defense.

MARLENE M. DE LEON, Ph.D.
Chair

PROCESO L. FERNANDEZ, JR., Ph.D.
Adviser

EVANGELINE P. BAUTISTA, Ph.D.
Dean
School of Science and Engineering

The Faculty of the Graduate School of Ateneo de Manila University accepts the THESIS entitled:

Improving An Exact Solution to the (l, d)-Planted Motif Problem

submitted by **Maria Clara Isabel D. Sia** in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science.

ANDREI D. CORONEL, Ph.D. Member	JULIETA Q. NABOS Member
JOSE ALFREDO A. DE VERA, Ph.D. Member	PROCESO L. FERNANDEZ, JR., Ph.D. Adviser

EVANGELINE P. BAUTISTA, Ph.D.
Dean
School of Science and Engineering

Grade: **Very Good**

Date: 23 October 2015

Abstract

DNA motif finding is widely recognized as a difficult problem in computational biology and computer science. Because of the usual large search space involved, exact solutions typically require a significant amount of execution time before discovering a motif of length l that occurs in an input set $\{S_1, ..., S_n\}$ of sequences, allowing for at most d mismatches due to mutation.

This study implements a novel speedup technique for EMS-GT, an exact motif search algorithm which operates on a compact bit-based representation of the search space. Our novel technique takes advantage of distance-related patterns in this representation, in order to speed up the bulk bit-setting operations performed by the algorithm. A Java implementation shows the improved EMS-GT to be highly competitive against PMS8 and qPMS9, two current state-of-the-art exact algorithms. With the speedup technique, EMS-GT outperforms both competitors for challenging (l, d) instances (9,2), (11,3), (13,4) and (15,5) showing runtime reductions from qPMS9 of at least 76%, 81%, 77% and 37% respectively for these instances, while ranking second to qPMS9 for challenge instance (17,6).

TABLE OF CONTENTS

LIST OF FIGURES ii

LIST OF TABLES iii

CHAPTER

I INTRODUCTION 1

II REVIEW OF RELATED LITERATURE 2

III METHODOLOGY 3

IV RESULTS AND ANALYSIS 4
4.1 Block patterns in l-mer neighborhoods 4
4.2 Derivation of patterns based on Hamming distances 6
4.3 Pattern-based EMS-GT speedup technique 9
4.4 Performance improvement in EMS-GT 9
4.5 Performance comparison to PMS8 and qPMS9 9

V CONCLUSIONS 10

APPENDIX

BIBLIOGRAPHY 12

A Source code for EMS-GT, with speedup technique 13

LIST OF FIGURES

4.1 Bit patterns followed by blocks in the bit-array representation of $N(\texttt{acgtacgtacgt}, 5)$. Black signifies a bit set to 1. 5
4.2 Distance distribution from tacgt to all $4^5 = 32 \times 32$ k-suffixes, k=5. 7
4.3 Correspondence between the value of $d_H(y, y')$ and bit patterns for $N(\texttt{acgtacgtacgt}, 5)$. Black signifies a bit set to 1. 8

LIST OF TABLES

4.1 Allowable suffix mismatches, for a fixed number of prefix mismatches, between d-neighbors. 6

CHAPTER I

INTRODUCTION

CHAPTER II

REVIEW OF RELATED LITERATURE

CHAPTER III

METHODOLOGY

CHAPTER IV

RESULTS AND ANALYSIS

This section derives the distance-related patterns observed in an l-mer neighborhood (represented with a 4^l-bit array) in EMS-GT. It then describes how a speedup technique for EMS-GT was developed based on these patterns. Finally, it compares EMS-GT performance with and without the speedup technique, and compares the performance of improved EMS-GT against state-of-the-art algorithms PMS8 and qPMS9.

4.1 Block patterns in l-mer neighborhoods

We can represent the neighborhood of l-mer x as an array N_x of 4^l bit flags, set to 1 if the corresponding l-mer is a neighbor and 0 otherwise.

$$N_x[\, x' \,] = \begin{cases} 1 & \text{if } d_H(x, x') \le d, \\ 0 & \text{otherwise.} \end{cases} \quad \text{for any } l\text{-mer } x'. \tag{4.1}$$

We can divide our l-mer x into its **prefix** $\boldsymbol{y}$ (the first $l - k$ characters) and its $\boldsymbol{k}$**-suffix** $\boldsymbol{z}$ (the last k characters). We use the notation $x = yz$.

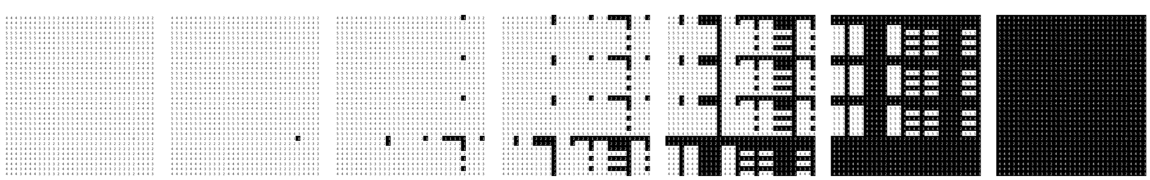
Ex. For k = 5, x = `acgtacgtacgt` $\rightarrow$ y = `acgtacg` and z = `tacgt`.

If we partition N_x into blocks of 4^k bits each, for some $k < l$, the 4^k l-mers represented in each block will all start with the same **block prefix** and all have different k-suffixes. This is because N_x represents l-mers in alphabetical order.

Ex. Blocks in N_x for x = `acgtacgtacgt`, k = 5:

Block 0:	bit flags for `aaaaaaaaaaaa` – `aaaaaaattttt`
Block 1:	bit flags for `aaaaaacaaaaa` – `aaaaaacttttt`
	...
Block 1,734:	bit flags for `acgtacgaaaaa` – `acgtacgttttt`
	...
Block 16,833:	bit flags for `ttttttgaaaaa` – `ttttttgttttt`
Block 16,834:	bit flags for `tttttttaaaaa` – `tttttttttttt`

Each such block in N_x will also conform to one of $(k+2)$ bit patterns.



Pattern -1 Pattern 0 Pattern 1 Pattern 2 Pattern 3 Pattern 4 Pattern 5

Figure 4.1. Bit patterns followed by blocks in the bit-array representation of $N(\texttt{acgtacgtacgt}, 5)$. Black signifies a bit set to 1.

If we can derive these pattterns, we will be able to build N_x in blocks, instead of setting bits one by one as EMS-GT currently does. The next section uses the additive property of Hamming distances to derive the bit patterns in N_x.

4.2 Derivation of patterns based on Hamming distances

Since Hamming distances count mismatches in corresponding characters, the distance between $x = yz$ and another l-mer $x' = y'z'$ is the sum of the mismatches between their prefixes and the mismatches between their k-suffixes, or:

$$d_H(x, x') = d_H(y, y') + d_H(z, z') \tag{4.2}$$

Given Equations (4.1) and (4.2), we can redefine N_x as:

$$N_x[\ x'\ ] = \begin{cases} 1 & \text{if } d_H(y, y') + d_H(z, z') \leq d, \\ 0 & \text{otherwise.} \end{cases} \quad \text{for } x' = y'z'. \tag{4.3}$$

Intuitively, if x and x' are neighbors, and there are $\boldsymbol{d_H(y, y')}$ **prefix mismatches** between them, we can allow $\boldsymbol{d_H(z, z') \leq d - d_H(y, y')}$ ***k*-suffix mismatches** for x and x' to have d or fewer total mismatches. Table 4.1 shows the $(k+2)$ cases for distributing d allowable mismatches between prefix and k-suffix.

	prefix mismatches	***k*-suffix mismatches**
Case -1	more than d	–
Case 0	d	0
Case 1	d - 1	0, 1
Case 2	d - 2	0, 1, 2
...	...	...
Case k-1	d - (k-1)	0, 1, 2, ..., (k-1)
Case k	$d - k$ or less	0, 1, 2, ..., (k-1), k

Table 4.1. Allowable suffix mismatches, for a fixed number of prefix mismatches, between d-neighbors.

The $(k+2)$ cases shown in Table 4.1 correspond to the $(k+2)$ bit patterns followed by the blocks in N_x.

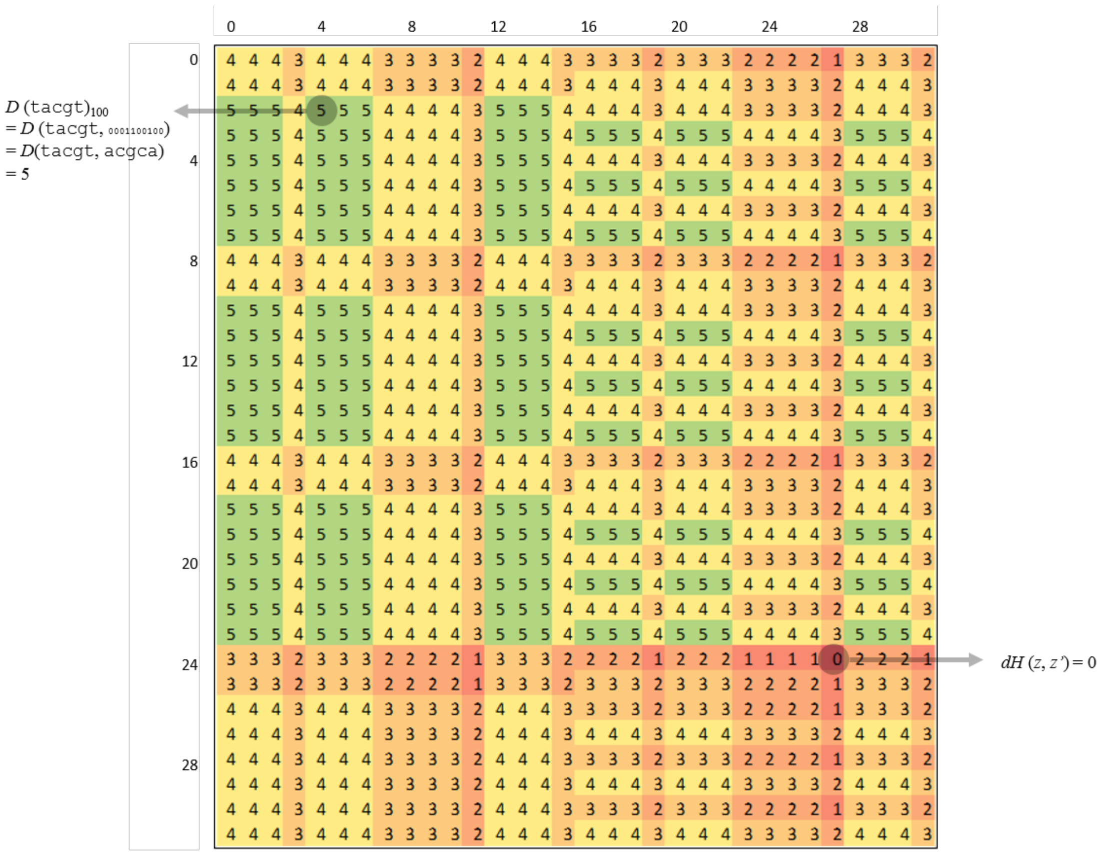
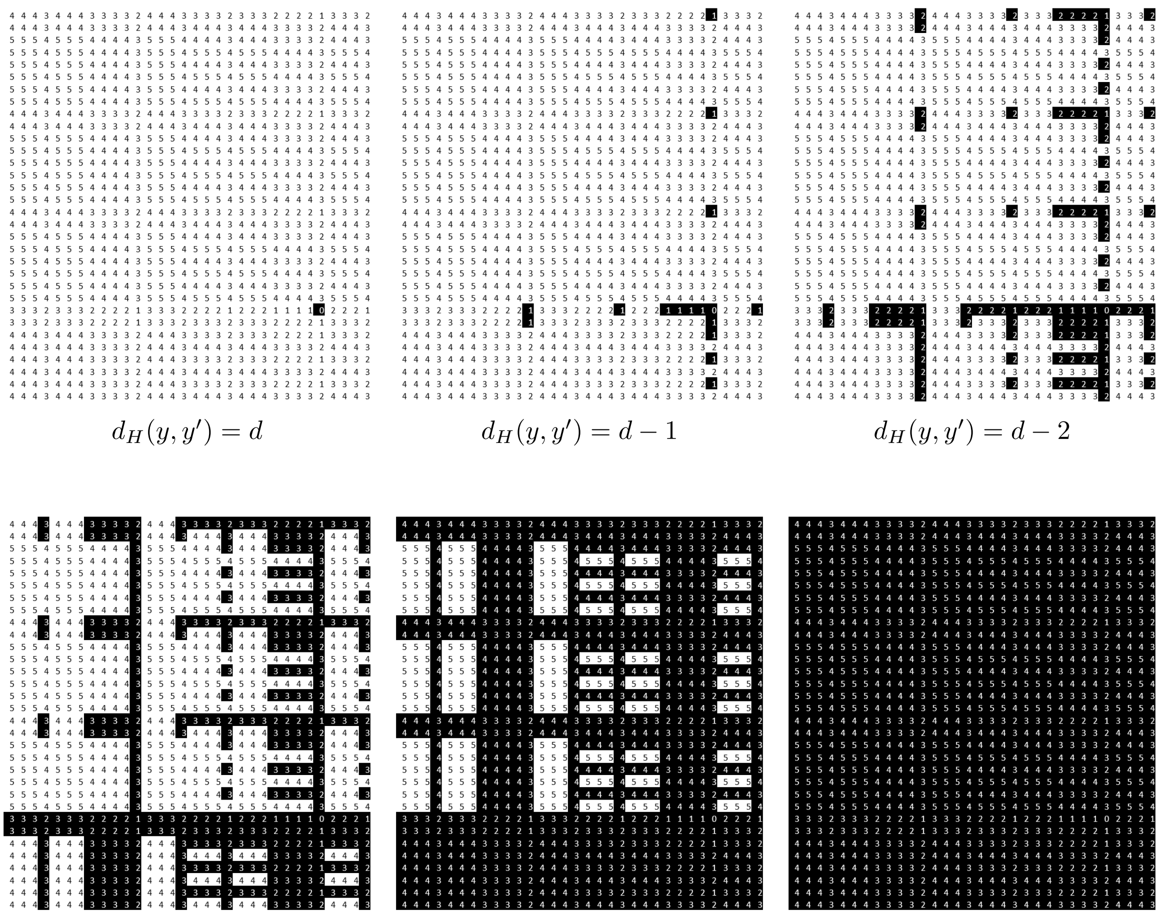


Figure 4.2. Distance distribution from `tacgt` to all $4^5 = 32 \times 32$ k-suffixes, k=5.

The value of $d_H(y, y')$ determines which pattern applies to which block:



$d_H(y, y') = d$ $\quad$ $d_H(y, y') = d - 1$ $\quad$ $d_H(y, y') = d - 2$

$d_H(y, y') = d - 3$ $\quad$ $d_H(y, y') = d - 4$ $\quad$ $d_H(y, y') \leq d - 5$

Figure 4.3. Correspondence between the value of $d_H(y, y')$ and bit patterns for $N(\texttt{acgtacgtacgt}, 5)$. Black signifies a bit set to 1.

Note that when $d_H(y, y') > d$, the number of prefix mismatches already exceeds the limit for neighbors, hence no bits in the block are set (see Pattern -1, Fig 4.1).

4.3 Pattern-based EMS-GT speedup technique

1. Step 1

2. Step 2

4.4 Performance improvement in EMS-GT

4.5 Performance comparison to PMS8 and qPMS9

CHAPTER V

CONCLUSIONS

In line with our research objectives, we make the following conclusions:

1. Our novel speedup technique takes advantage of the distance-related block patterns observed in the search space. Initially EMS-GT generates, and sets the bit for, each individual neighbor of an l-mer x. However, our speedup technique allows EMS-GT to set these bits in blocks of 4^k bits each, using pre-generated bit patterns; we find the ideal value of k to be 5.

2. The speedup technique improves EMS-GT's performance on challenging (l,d) instances (11,3), (13,4), (15,5) and (17,6), with runtime reductions of at least 6.7%, 47.5%, 38.1% and 43.0% respectively; however, on challenge instance (9,2), overhead increases EMS-GT's runtime from 0.06 s initially to 0.11 s with the speedup technique.

3. The speedup technique allows EMS-GT to outperform the current best algorithm, qPMS9, on challenging (l,d) instances (9,2), (11,3), (13,4) and (15,5) with runtime reductions of at least 76%, 81%, 77% and 37% respectively for these instances, while ranking second to qPMS9's runtime on challenge instance (17,6).

Directions for further research on improving EMS-GT include:

1. Refining the bit-based search space representation (i.e. with compression techniques) to be able to represent the motif search space for $l > 17$;

2. Creating a multiprocessor version of EMS-GT to solve the planted motif problem faster, in parallel, for larger values of (l, d); and

3. Delegating the bit-masking speedup technique and other bulk bit operations to the graphics card, as explored in [1], for faster performance.

BIBLIOGRAPHY

[1] Naga Shailaja Dasari, Ranjan Desh, and Mohammad Zubair. An efficient multicore implementation of planted motif problem. In *High Performance Computing and Simulation (HPCS), 2010 International Conference on*, pages 9–15. IEEE, 2010.

APPENDIX A

Source code for EMS-GT, with speedup technique