RiboFSM: Frequent Subgraph Mining for the Discovery of RNA Structures and Interactions

by

Alexander Gawronski

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the Master degree in
Computer Science, Bioinformatics Option

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

Frequent subgraph mining is a useful method for extracting biologically relevant patterns from a set of graphs or a single large graph. Here, the graph represents all possible RNA structures and interactions. Patterns that are significantly more frequent in this graph over a random graph are extracted. We hypothesize that these patterns are most likely to represent a biological mechanisms. The graph representation used is a directed dual graph, extended to handle intermolecular interactions. The graph is sampled for subgraphs, which are labeled using a canonical labeling method and counted. The resulting patterns are compared to those created from a randomized dataset and scored. The algorithm was applied to the mitochondrial genome of the kinetoplastid species *Trypanosoma brucei*. This species has a unique RNA editing mechanism that has been well studied, making it a good model organism to test RiboFSM. The most significant patterns contain two stem-loops, indicative of gRNA, and represent interactions of these structures with target mRNA.

Acknowledgements

The foremost person I would like to thank is my supervisor Marcel Turcotte. You always had the time to share your wisdom and expertise and your input was always valuable. I feel very fortunate to have you as a supervisor and am grateful for you introducing me to the interesting field of RNA research. This work could not have been possible without your guidance.

I also want to thank Gertraud Burger for sharing her data with us and introducing us to the interesting problem of trans-splicing in Diplonema. As well as the Natural Sciences and Engineering Research Council (NSERC) for funding this research.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Only about 1.2% of the human genome is made up of protein-coding genes. A significantly larger portion of the genome holds the potential to code for a different kind of gene. These genes are non-coding ribonucleic acids (ncRNA) which are RNA fragments that are never translated into proteins. Many of these genes play important regulatory roles and are often referred to as RNA genes. To gain a better understanding of these genes, sensitive and accurate RNA gene prediction methods are needed. Unfortunately the identification of new RNA genes remains a challenge. Currently many RNA gene prediction methods use the idea of moving a window of fixed-length along the genomic sequence and the lowest free energy structure for this window is calculated. If the window yields a stable secondary structure, it is seen as evidence of the presence of an RNA gene. This method has its limitations, such as lack of scalability for genome-wide analysis and only being able to identify intramolecular interactions. These methods have been used with limited success. Methods involving finding consensus sequences have been more effective. Graph mining to find frequent patterns is one such approach that has not received much attention in the RNA community. A new, high-throughput method using frequent subgraph mining can capture both RNA secondary structure as well as interactions between multiple RNA strands.

1.2 Biological Question

In most organisms the process of protein synthesis is well understood. Deoxyribonucleic acid (DNA) is transcribed into messenger ribonucleic acids (mRNA), which are then translated into polypeptides that fold to create proteins. However there are a few families of organisms where the process deviates from the norm. One such family is the *Kinetoplastida* in the kingdom *Excavata*. The mRNA produced from the mitochondrial DNA of this family cannot be directly translated into protein but must be prepared by a process called RNA editing. This process is mediated by short RNA molecules called guide RNA (gRNA) [3]. Another closely related family, Diplonema, also has a unique editing system in its mitochondria. In this case the genes are fragmented into "modules" which are transcribed separately and then assembled by an unknown mechanism [27].

Little work has been done in the development of computational methods for discovering gRNA. The existing methods suffer from poor precision and dependence on experimental transcript data [40, 47]. Even less has been done for Diplonema. The current research has only shown that known cis-splicing mechanisms are not present and suggests that RNA guides or proteins mediate the process [27]. A computational approach was used in one study but again suffered from a lack of precision, generating millions of candidate structures [26].

The objective of this new methodology is to discover RNA interactions occurring in known or novel RNA-mediated mechanisms. These mechanisms involve RNA with a specific secondary structure formed by complementary sequences within the molecule. Depending on the location of these stems different substructures can be created. There are four basic substructures: stem-loops, interior loops, bulges and pseudoknots (See Appendix A). These RNAs also contain complementary sequences to other RNAs, such as target mRNA, to allow them to form a quaternary structure. If all possible stems between a set of molecules was known, any RNA mechanism would be a subset of those stems. The problem then becomes finding that correct subset of stems. However the number of possible stems is very large, and the number of combinations of these stems is enormous.

1.2.1 Kinetoplastids

The *Kinetoplastid* family has been well studied due to its disease causing species, such as *Trypanosoma brucei* which causes sleeping sickness and *Trypanosoma cruzi* which causes Chagas disease. The former is the most studied and therefore was used as a model in this work. What is of interest in these organisms is their unique mitochondrial genome, called a "kinetoplast". The genome is made up of two chromosome types; maxicircles and minicircles. Maxicircles are 23-36kbp long and contain 18 protein-coding and 2 rRNA. However 12 of the genes are "cryptogenes" that produce transcripts that have too many or too few uridines. Minicircles are 465bp to 10,000bp (∼1kbp average) long and code for guide RNA (gRNA) which are part of the editosomes that corrects the pre-mRNA created from cryptogenes [43].

The size range of a gRNA is 50-70nt and has three components. Starting at the 5' end, the first is an anchor sequence, 5-21nt long, which binds to the target mRNA. The second is a conserved secondary structure of one or two stem-loops depending on species. This structure contains the "guiding domain" which is complementary to the mRNA but with additional adenines. The third is a 3' poly(U) tail, 5-25nt long, which is added post-transcription (Figure 1.1) [51]. There are 3-5 gRNA per minicircle separated by 110nt and are flanked by 18nt imperfect repeats [37].

There are 1-10 sites on mRNA where an anchor can bind. Often the sequence does not match any anchor until the sequence has been repaired by the gRNA binding to the upstream anchor. A gRNA binds to the complementary sequence on the 3' end of the pre-mRNA and is stabilized by the poly(U) tail. An endonuclease is recruited at the first mismatch between the two sequences and the pre-mRNA is cleaved. If an insertion is required, a terminal uridyl transferase (TuTase) inserts a uridine at the cleavage site. If a deletion is required an exonuclease removes a uridine. When insertion/deletion is completed, the strand is ligated by an RNA ligase (Figure 1.2). As this process continues, the stem loop gradually separates until a full RNA duplex is created. At this time the next gRNA binds or the process is complete [16].

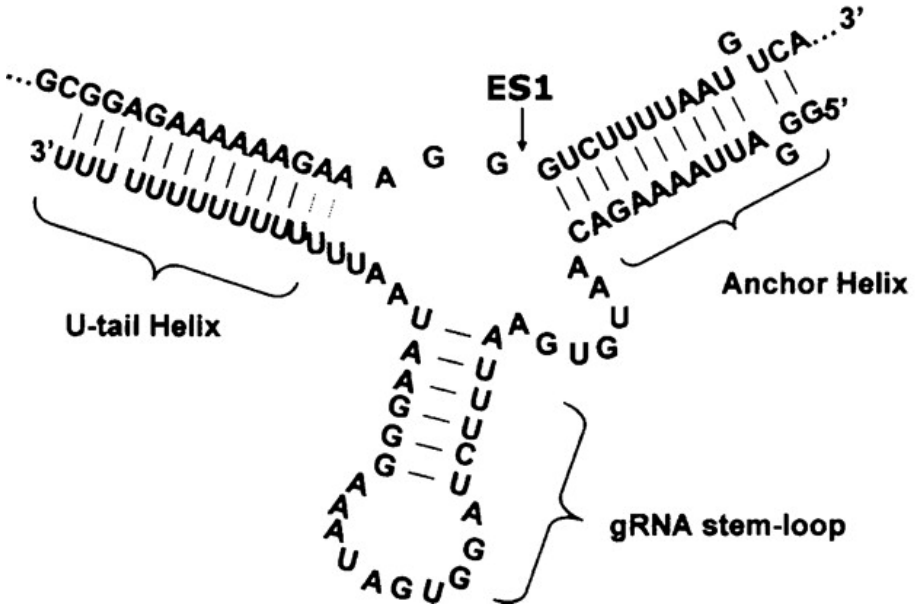

Figure 1.1: CYbU-NgCYb-558 gRNA/mRNA pairing with first editing site marked (ES1) [51].

1.2.2 Diplonema

Diplonemids are a closely related family to *Kinetoplastids*. The species *Diplonema papillatum* is the most studied because, like *Kinetoplastids*, it has a very unique editing process. Unlike *Kinetoplastids*, little is known about the mechanism involved. The genome of this species is made up of hundreds of 6 and 7 kbp long circular chromosomes, A-class and B-class respectively. Genes are broken into 70-350nt long "modules" located on separate chromosomes. Most of the chromosome sequence is identical within the class or between all the chromosomes. The only unique region is where the module is. The module and its flanking regions are called a "cassette" (Figure 1.3). The first gene to be closely examined is cytochrome oxidase 1 (cox1) which is made up of 9 separate modules [26].

It was first hypothesized that each module/cassette was assembled through cis-splicing. More specifically that each module/cassette contains information for self-assembly of the modules. However no evidence was found to support this hypothesis. This lead to the conclusion that the process may be mediated by a third party molecule, in a trans-splicing

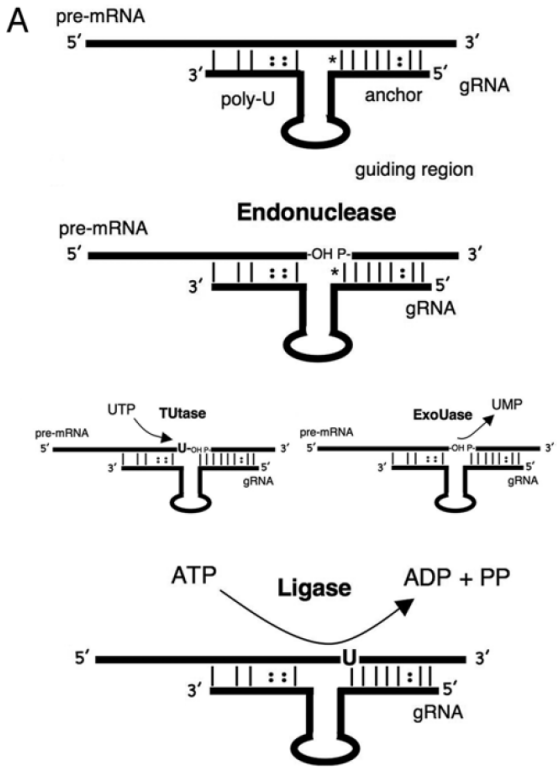

Figure 1.2: The RNA editing mechanism in Kinetoplastids [16].

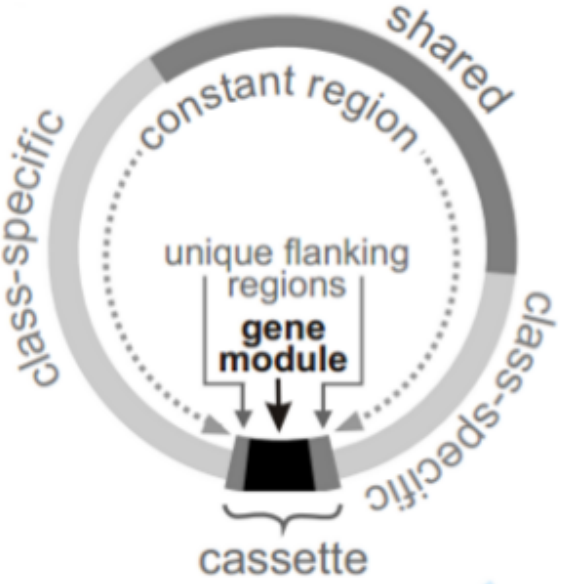

Figure 1.3: *Diplonema papilatum* general chromosome structure [26].

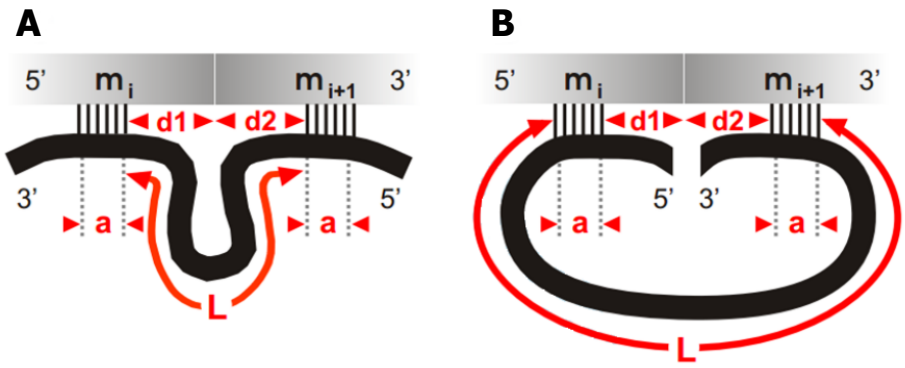

Figure 1.4: Hypothetical ppRNA structure for trans splicing in *Diplonema papilatum*. Two conformations one with (A) reverse complement anchors and the other (B) with complement anchors. M_i and M_{i+1} are adjacent modules. L is the bridge length and $a, d1, d2$ are variables for the anchor length and location [26].

fashion. Non-coding RNA was suspected to be involved in this assembly. Using computational methods, many potential ncRNA candidates were found that could mediate this process. These potential guides have been called post-transcriptional processes-guiding RNAs (ppRNA). They are hypothesized to be approximately 25nt long and bind to approximately 6nt long ends of gene module transcripts. They assemble the modules into a complete transcript in no particular order, since clear intermediates were found for all combinations of adjacent modules [26].

Four hypothetical structures for the ppRNA were proposed. All four include two 6nt long "anchor" regions which bind to adjacent modules with a max 50% GU pairing. The first is a simple, short RNA with no nucleotides between the anchors. The second is a circularized RNA where one contiguous region binds to both modules. The third and forth are two anchors with a "bridge" between them, one with reverse compliment anchors and the other with complement anchors (Figure 1.4). A genome-wide search for these structures yielded millions of possible candidates, however it would be infeasible to verify each of these candidates manually. Also structures with more complex bridge regions have not yet been investigated.

1.3 Graph Mining

1.3.1 Graph Representation

The first challenge is determining how to represent the data. The information of interest is the position of the complementary sequences, the length of these sequences, and the relative locations of these sequences. Since this data is many single units of information, interconnected by there relative locations, it is well suited for a graph representation. Furthermore, graphs are ideal for representing complex topologies, which in this context allows for representation of complex RNA structures and interactions. Graph representations for RNA structure have been used in other studies, usually using a planar tree graph representation. In tree graphs, loops are collapsed into nodes and the stems forming the loop become edges. The main drawback with tree graphs is that pseudoknots cannot be represented. Background information on RNA secondary structure can be found in Appendix A.

An alternative solution is to use the reverse representation, that is, stems become nodes and loops become two edges. With this modification pseudoknots can be represented by three edges between two nodes. This type of graph is called a dual graph [13]. However some ambiguities may occur with this representation. The order of stems while going along the RNA strand can be modified but would still produce the same dual graph. This can be resolved by creating directed edges, creating the directed dual graph representation, which was used in this work [13]. This representation is more robust than tree graphs with no disadvantages, and its application to modeling the interactions between many RNA molecules has not been explored. An example of each graph representation is shown below (Figure 1.5).

1.3.2 Frequent Subgraph Mining

A graph created from all complementary sequences will be largely noise. An accepted method for extracting useful data from graphs is called Frequent Subgraph Mining (FSM). FSM is the process of finding subgraphs in a graph with a frequency no lower than a specified threshold. In the single graph setting, the number of "embeddings" of

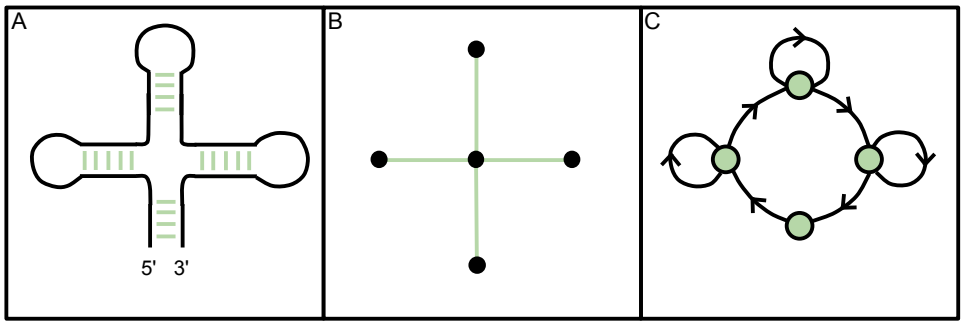

Figure 1.5: Example RNA structure (A) with corresponding tree graph (B) and directed dual graph (C).

a subgraph within the graph are counted to determine frequency. This presents unique challenges since many embeddings can be overlapping which could lead to the frequency not being downward closed. More specifically, the anti-monotonicity property would not hold, which declares that a subgraph can only be frequent if all of its subgraphs are frequent [6].

In the graph representation used, any connected subgraph where the nodes are regions that are non-overlapping represent a viable secondary structure and potential relations between these structures. Subgraphs that occur in the graph more frequently than expected by chance are expected to be structures and interactions of biological interest. The problem of finding these subgraphs in this context is called the frequent subgraph mining problem for a single large graph.

1.4 Contribution

The main contributions are listed below:

- Generalization of the directed dual graph for both intramolecular and intermolecular complementary regions, thus capturing secondary structure as well as interactions between RNA structures.
- First application of frequent subgraph mining on dual graphs.

- First application of frequent subgraph mining for the discovery of RNA motifs and interactions in a single graph (including intermolecular) setting.
- Efficient and compact algorithm and data structures that can scale for graphs with millions of nodes and edges.
- An algorithm that can discover *de novo* motifs and interactions rather than simply querying the graph.
- Successfully applied the algorithm to both synthetic and real data sets.

The work was presented at the International Symposium on Bioinformatics Research and Applications (ISBRA)[15] as an oral presentation and is expected to be published in BMC Bioinformatics.

Chapter 2

Literature Review

2.1 Literature Review

This work encompasses two domains. The first is graph mining for the discovery of frequent subgraphs, including graph representation, and the second is RNA motif discovery. Graph representation is an established concept, however graph mining has only become popular within the last decade. This is largely due to the availability of massive graph data sets from Internet applications and other computer systems. RNA motif discovery is also a fairly new field, which gained popularity with the discovery of the extended functionality of RNA after the 1980s. This chapter will be a review and evaluation of the existing work in these two fields.

2.1.1 Graph Representation

Early work on RNA graph representation produced a representation called "tree graphs" [34, 2, 13]. In tree graphs, loops are collapsed into nodes and the stems forming the loop become edges. This representation has limitations, such as the inability to represent pseudoknots. Veksler et. al. [46] proposed a solution to this problem. This was to add additional edges between nodes that form pseudoknots. However this breaks down the tree representation and a special case must be made for matching these structures.

This representation was developed initially to help classify and visualize existing RNA structures [55]. Tree graphs were later adapted to help guide RNA secondary structure prediction and for motif discovery described later in this chapter. All of these applications deal with representing RNA structures derived from crystal structure or from RNA secondary structure prediction methods. There are no examples of tree representations used in a data mining context.

Later the reverse representation, where stems become nodes and loops become two edges, was developed called the "dual graph" [13]. This representation allows for pseudoknots to be modeled and removes other ambiguities. Recently this representation was used on a moderately large scale to produce the RNA as Graph (RAG) database [23]. This database stores dual graph representations of all known RNA motifs as well as their tree graph versions. The algorithm produced approximately 53,000 dual graphs up to a size of 9 nodes from individual RNA, which they classified as RNA-like and non-RNA-like. This is dwarfed by the search space of all possible RNA structures across multiple molecules, which is in the tens of millions of subgraphs. Soon after a paper was published stating that dual graphs without labels are not good enough to identify RNA-like structures (up to 5 nodes) [29]. With the use of labels, and higher order structures, dual graphs are still useful.

Another graph representation using the reverse representation is a "stem graph" developed by Hamada et. al. [17]. This is a directed, labeled graph where the nodes represent candidate stems. The edges represent the relative locations of the stems and are labeled P (Parallel), N (Nested) or K (Pseudoknotted) to capture how stems are orientated. Like dual graphs, this representation can support pseudoknots because of this labeling technique. However this approach requires edges from each node to all the downstream nodes. This could cause scalability issues for complex or large structures.

An important quality required for the graph representation used by RiboFSM is to be able to model both intermolecular and intramolecular interactions. The tree graph representation by its definition cannot handle this case. An edge is a connection between exactly two nodes, and an intermolecular stem has four adjacent unpaired nucleotides which would produce four nodes. Creating one edge connecting four nodes cannot be done. However with the reverse, dual graph representation this would only require the addition of an edge. There is no limit on the number of edges between two nodes so the

central graph concepts are not broken. No studies have explored extending dual graph representation to include intermolecular RNA interactions.

2.1.2 Frequent Subgraph Mining

Frequent subgraph mining is the process of discovering frequent subgraphs in a collection of graphs (transactions) or in a single large graph. The former has received the most attention due to its large variety of applications. If the focus of this work was only on intramolecular patterns then this would be the approach. The frequency in this case would be the number of graphs that contain the subgraph. However since the goal is to discover both intramolecular and intermolecular interactions, the search space becomes a single large graph. In this case the frequency is the number of "embeddings" of a subgraph in the single graph. The difficulty here is how to handle overlapping embeddings to ensure anti-monotonicity and an appropriate support measure must be determined.

Many approaches solve this problem by creating a overlap graph [33, 32], where a subgraph instance is a node and edge exists between nodes if the instances overlap. The size of the maximum independent set (MIS) is determined and is used as the support value. This is costly since finding the MIS is a NP-complete problem [14] so generally it is approximated or alternative approaches are used. Fiedler and Borgelt [11] suggested a definition called harmful overlap support. This method still calculates the MIS, however it divides overlaps into two kinds, harmful and simple. The process is sped up by ignoring the simple overlaps and still maintains anti-monotonicity. Bringmann and Nijssen [4] defined a new support measure that does not rely on the calculation of MIS but instead on the number of unique nodes in the graph to which a node of the pattern is mapped. Therefore it is less expensive computationally than the other two methods, but still relatively costly.

Another challenge is determining whether two subgraphs have the same topology. This is called the subgraph isomorphism problem and it is also NP-complete [12]. Various approximations have been purposed including the use of canonical labeling. Canonical labeling allows for a "code" to be assigned to a subgraph that will be consistent even if the order of vertices and edges changes [33]. This is accepted as the fastest method for determining subgraph isomorphism [33, 49, 25]. There was no need to expand this

method or develop a new method.

A difficulty of single-graph FSM is the often enormous size of the input graph and consequently the search space. Few algorithms attempt to search the entire search space and do not scale well, such as hSiGraM/vSiGraM [33]. Most approaches use heuristics or stochastic methods to find approximate solutions. Some examples are compression-based methods (SUBDUE [25]), pruning methods (GREW [32]) and sampling methods [54] in order of how well they scale, worst to best.

Due to the potential size of the input graphs for RiboFSM, a sampling approach was chosen. The random sampling methods described by Zou and Holder [54] were ineffective for this application. However they found what they call "Random Areas Selection Sampling" to be the most effective. This approach is similar to the K-nearest neighbor approach, but selecting all adjacent nodes rather than those closest based on edge label. The K-nearest neighbor algorithm has been used in other applications, such as clustering. An adaptation of this approach for graph mining in this context may be useful since structural elements tend to be close to each other. Zou and Holder [54] also implemented "Random Walk Sampling", where from a random starting point the subgraph is extended by randomly selecting edges. The results were worse than the random area selection sampling so this method was not pursued, however it may be more effective when using a dual graph representation.

Applications in Bioinformatics

FSM has been applied to problems in the domains of biology and chemistry [24]. For biological applications, the most common is mining biological networks [30, 21, 44], specifically protein-protein interaction (PPI) networks. The use of FSM for RNA structure discovery has not been thoroughly explored. FSM was applied by Horvath et. al. [20] and Hamada et. al. [17] but only in a graph transaction and intramolecular-only setting. Another group, Wang et. al. [48] created Fast Frequent Subgraph Mining (FFSM) for tertiary motif discovery, but again this was in a graph transaction context and did not consider intermolecular interactions.

The most similar approach is an algorithm called Linear Graph Miner [45] which uses

a similar graph model and discusses potential applications to RNA secondary structure. However, they did not test their algorithm with any RNA data and did not discuss the potential for modeling intermolecular interactions. There is also no implementation provided by the author to allow for comparison of performance. They did discuss how using linear graphs allow for fast solutions for graph isomorphism and support determination. The parallels of linear graphs to directed dual graphs indicate that they may also be promising for RNA structure modeling and mining.

2.1.3 RNA Motif Discovery

The RNA Motif Discovery problem is to find common structural motifs given an input of functionally related RNA sequences. This is not equivalent to the problem discussed in this work but is closely related. These approaches could be applied to the discovery of motifs related to RNA editing. However they do not consider intermolecular interactions and are limited in the number of sequences and length of sequences that can be used as input.

RNA Motif discovery methods fall into one of four categories: dynamic programming based, data structure based, evolutionary algorithm or expectation maximization approaches. Dynamic programming is a method of solving complex problems by reducing them to simpler subproblems where the solutions to the subproblems are cached rather than recomputed. The first algorithm to apply this approach for RNA Motif discovery was the the Sankoff algorithm [42] and following algorithms were extensions of this work, such as FOLDALIGN [19].

Data structure methods use complex data structures to speed up access and retrieval of words. Two examples are affix trees and suffix arrays. The affix tree data structure stores information about a string in forward and reverse. Using this structure the algorithm can find motifs that appear in a specified number of sequences [39]. Suffix arrays are a space efficient variation of a suffix tree and are used to store all suffixes of a string. This data structure was exploited for the discovery of RNA motifs with the creation of a tool called Seed [1]. This is done by using a generalized suffix array which captures the sequence and its reverse complement. The algorithm then selects a "seed" sequence and builds a specific motif and uses that motif to search all the other input sequences.

By using the range minimum query, all possible stems can be efficiently enumerated. A breadth-first search algorithm is then used to enumerate all combinations of these stems. Combinations that do not have the minimum support are pruned. The support is defined as the fraction of the input sequences having an instance of the motif. These approaches focus on secondary structure prediction with a small number of input sequences.

Heuristic methods generally fall into two categories, evolutionary algorithms or expectation maximization. Evolutionary algorithms build a set of solutions and selects the best solutions based on a fitness function. The best solutions are then modified by "crossover" and "mutation" and the process is repeated until an approximately optimal solution is found. Genetic algorithms [5] and genetic programming [22] have been applied to RNA motif discovery. The former uses a free energy based fitness function while the latter uses a fitness function that favors motifs that exist in a given positive set.

Methods that use expectation maximization (EM) often use covariance models. The covariance model starts with a tree representation of an RNA sequence where the nodes in the tree are nucleotide pairs and branches are different stems. The nucleotide pairs are then replaced with one of 6 "states" which are matches, insertions or deletions. Going from one state to another is determined using "state transition probabilities". The resulting probabilistic model is the covariance model [10]. Tools that fall into this category are CMFinder [50] and COVE [9]. These methods are efficient and accurate, however since they use context-free grammars they are unable to handle intermolecular interactions or pseudoknots. Alternatively context-sensitive grammars could be used [38], but it would make this approach even more computationally expensive than it already is.

Frequent subgraph mining has also been applied to RNA motif discovery. The algorithm is called RNAmine [17] which mines "stem graphs" to find frequent motifs. The approach was shown to run faster and perform better than covariance model methods. However the authors note that performance would suffer for longer sequences or larger data sets. The same authors also developed a bi-clustering approach [18]. This method is able to handle larger datasets, similar is size to the *Trypanosma brucei* data, but with considerably long runtime (21 days with one CPU).

2.1.4 MicroRNA Target Identification

A possible application of RiboFSM would be microRNA, or miRNA, target identification. MicroRNA are short RNA molecules that bind to mRNA to regulate gene expression. This can occur by marking the mRNA for degradation or blocking translation. These non-coding RNAs have a conserved secondary structure and an anchor sequence similar to gRNA. The miRNA forms a stem-loop with conserved internal structures, such as bulges and interior loops, as well as conserved distances between features. For this reason current algorithms in this field will be briefly reviewed.

In animals, the anchor sequence has a "seed" region that can be as short as 6nt long, which forms a complete duplex with the mRNA. The rest of the anchor is an imperfect match with bulges. Nearly all methods for miRNA target prediction start with finding the seed region by using the region's conserved characteristics [52]. Many tools also incorporate thermodynamic stability of the binding sites to improve the prediction, such as RNA-Hybrid [31] and TargetScan [35]. Other algorithms deviate from this approach, such as those that use gene expression profiles (InMiR [41]) and machine learning, specifically support vector machines (miTarget [28]).

These approaches focus heavily on the seed region, while other algorithms focus on the structure of miRNA. It seems that current methods lack a holistic approach considering both the secondary structure and the seed. The RiboFSM dual graph representation can capture both these elements and find candidates that may be miRNA. This could be a fruitful, future direction for the algorithm. These methods are also very specific to miRNA so they cannot be adapted directly for gRNA. However some of the techniques used could possibly be integrated into RiboFSM, such as determining thermodynamic stability of stems.

Chapter 3

Method Description

3.1 Problem Description

The problem of this work is split into two subproblems. The first is how to model all possible RNA secondary structures and the interactions between them. More specifically, if given a set of input sequences, how can a search space be created that contains all possible structures and interactions that is space-efficient but also robust enough to capture the major RNA motifs? The second sub-problem is how to search this enormous search space in a reasonable amount of time. A graph theoretic approach was used for both of these problems. In the graph theoretic context these problems are graph representation and graph mining problems.

3.1.1 Graph Representation

A graph G is defined as $G = (V, E)$ where V is the set of vertices and E is the set of edges. A "vertex", also called a "node", is a single unit of information and an "edge" is the abstract entity that relates two vertices together. Both vertices and edges may be labeled or unlabeled and edges can be directed or undirected. The problem here is to decide what information should be contained in vertices and edges which will determine what the topology of the graph will look like and how dense or sparse the graph will

be. The ideal representation would capture all the necessary information about what the graph is modeling while keeping the number of vertices and edges to a minimum. Since this graph is mined for frequent subgraphs the representation must support fast traversal and construction. The final challenge is how to build and store the graph to use the least amount of memory and runtime.

3.1.2 Frequent Subgraph Mining

A graph $G_s = (V_s, E_s)$ given a graph $G = (V, E)$ is a **subgraph** if V_s subset V and E_s subset E. The frequent subgraph mining problem for a single large graph is to determine all subgraphs that have a frequency above a given threshold. This presents two more subproblems, determining subgraph isomorphism and determining a support measure. Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are **isomorphic** if they are topologically identical to each other. Formally, there must be a one-to-one mapping for every vertex in V_1 to V_2 such that each edge in E_1 has a one-to-one mapping to each edge in E_2. If the vertices and edges are labeled, the labels in the mapping must be equal. The two graphs are **automorphic** if $G_1 = G_2$.

The second subproblem is determining a support measure, or more specifically, how to count the number of embeddings of a given subgraph in a graph. This involves deciding whether embeddings should be counted only if they do not share any edges/vertices or whether overlaps are allowed, and if they are allowed how much overlap is allowed. This choice has a large effect on the tractability of the algorithm and the resulting frequency values.

3.2 Approach

This section describes the approaches used to solve both problems.

3.2.1 Finding Complementary Regions

The first step of the algorithm is to find all possible complementary regions between the input sequences. A complementary region of a sequence is another sequence where the nucleotides complement each other (A-U/T, C-G, G-U/T) and are in reverse order. All sequences are appended together and treated as one long sequence to simplify the process. The start and end points of each molecule are annotated so matches do not occur over the junction points. The input sequences are derived from parsed files in FASTA format. There are two input files, one containing expected functional RNA and another containing mRNA or the target of the functional RNA. Intermolecular complementary sequences will only be considered between molecules in different sets.

The raw nucleotide data is stored as a Java BitSet object. This object allows for storage and manipulation of an arbitrarily long bit string. Since there are four possible nucleotides, only 2 bits are needed to represent each nucleotide. The FASTA parser can also support and alternative 4-bit mode to represent all IUPAC nucleic acid codes and will accept RNA or DNA characters. This method minimizes the memory usage for storage of the input sequences. This also allows for 16-bit character matching to be replaced with bit operations on 2-bit or 4-bit BitSets, which should have better performance.

The algorithm used to match the sequences is a naive method involving matching each position i in a sequence of size n with every other position n-j, with some heuristics (Algorithm 1). Firstly, since i and j are indexing the same sequence, comparing $i = 1$ to $j = 2$ is equivalent to $i = 2$ to $j = 1$. Therefore the heuristic of enforcing $j > i$ is used to avoid finding duplicate matches. Secondly the maximal match is always used and the match positions are stored from the last iteration so $i + 1$ is not matched to $j - 1$. This avoids the creation of nested matches which would greatly increase the number of nodes without adding any more information. The matches must be of specified minimum size and can only contain a certain percent of G-U matches. These parameters can be used to control the size of the graph.

The naive approach without heuristics would compare each i with each j which would give a runtime of $O(n^2k)$, where n is the size of the input sequence and k is the match length. The $j > i$ heuristic reduces the number of (i, j) pairs by half and the maximality heuristic allows for a large number of pairs to be skipped proportional to the number of

matches. These heuristics significantly reduced the runtime, but it is still proportional to the input sequence length squared.

3.2.2 Graph Representation

The graph representation used is the directed dual graph [13]. In this representation every complementary region is represented as a node. Unpaired nucleotides are represented as edges which are directed in the 5' to 3' direction. The graph is also not necessarily fully connected since edges are only created between nodes on the same molecule. This representation was adapted to handle intermolecular interactions by adding the notion of intermolecular nodes. Edges are only created between nodes on the same molecule. This allows all possible interactions to be captured without the creation of excess edges for each node.

Since each node is made of two sequences, each with a 5' and 3' end, the degree of any node can be at most 4. In a basic case with one molecule, there must be at least one 5' and 3' end that is not connected to any other node and one self loop, therefore $|E| = 2|V| + 2 - 3$ or $|E| = 2|V| - 1$. This makes the graph (2,1)-sparse. The sparseness of this representation justifies the use of an adjacency list rather than a matrix. However since the edges are always between adjacent nodes the data structure can be simplified to a list of nodes. Additionally every node has a reference to a "sister node" elsewhere in the list. This allows for both sequences in a match to be treated as one node while still maintaining the simplicity of the list data structure. In other words, if you traversed the sequence from 5' to 3', every node will be visited exactly twice, and that path would create the graph (Figure 3.1).

When a complementary region is found, two nodes are created. One node is for the sequence at the *ith* position and the other is the sister node with the sequence at the *jth* position. The first node is added to the master list of nodes and the sister node is added to a priority queue. The priority queue ensures the nodes are sorted based on position. If the priority queue is not empty it is checked for nodes that have an index matching the current position. If such a node exists it is popped out of the queue and added to the master list (Algorithm 2). The procedure builds the graph in $O(nlogn)$ where n is the number of complementary regions, due to sorting occurring in the queue.

Algorithm 1 Pseudo-code for finding complementary regions.

```
1: for all i, j do
2:     percentGU = 0
3:     gu = 0
4:     bestIndex = 0
5:     kMax = value ensuring i + k < j − k
6:     for k = 0; k ≤ kMax; k++ do
7:         nuc1 = genome.getNucleotideAt(i + k)
8:         nuc2 = genome.getNucleotideAt(j − k)
9:         if nuc1 NOT complement of nuc2 then
10:            if k == 0 then
11:                return 0
12:            end if
13:            break
14:        else
15:            if GU pairing then
16:                gu++
17:            end if
18:        end if
19:        percentGU = gu/(k + 1)
20:        if percentGU ≤ GUMAX then
21:            bestIndex = k
22:        end if
23:    end for
24:    bestIndex++
25:    if bestIndex < MINLEN then
26:        bestIndex = −bestIndex
27:    end if
28:    return bestIndex
29: end for
```

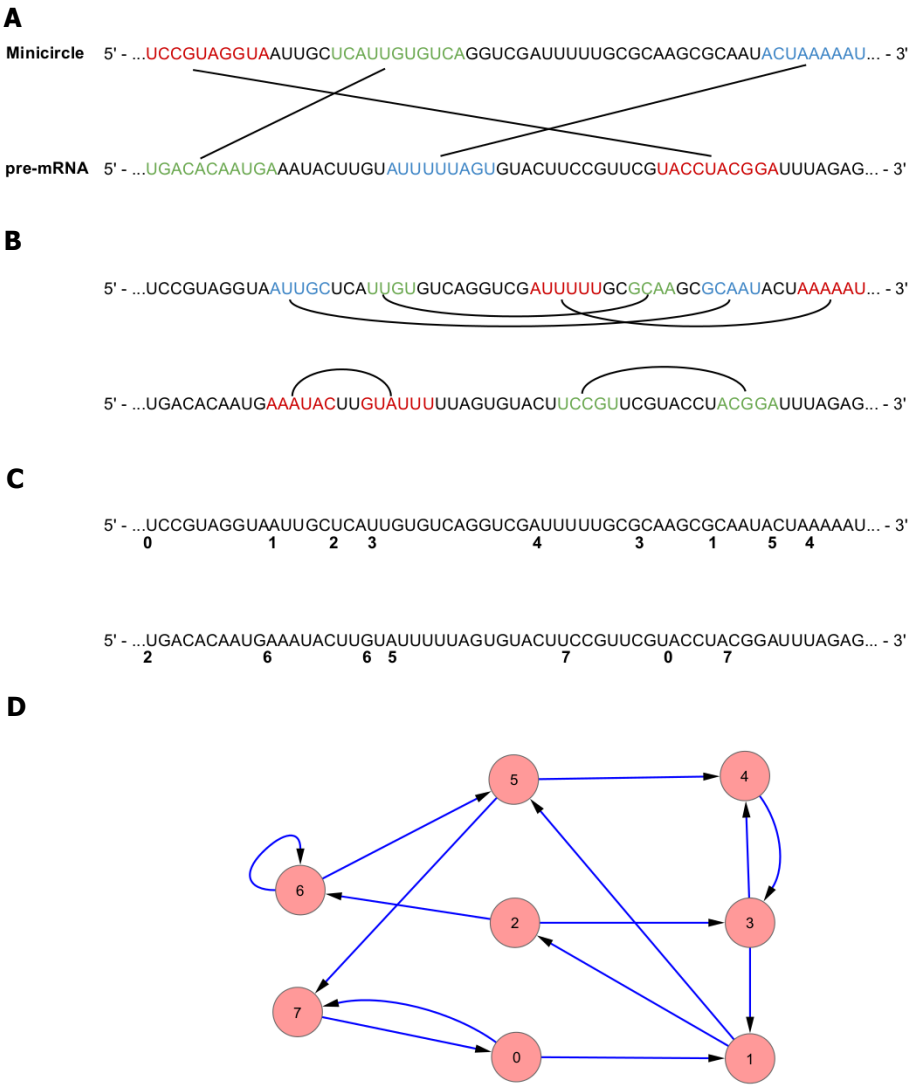

Figure 3.1: Example of graph building given two input sequences. All intramolecular (A) and intermolecular (B) complementary regions are discovered and enumerated (C). Traversing the sequence using the priority queue, the nodes corresponding to each region can be created to form the final graph (D).

Algorithm 2 Pseudo-code for building the graph.

```
1: max = last position
2: queue = new PriorityQueue
3: graph = new Graph
4: for all Sequences : S do
5:     for i = S.start; i ≤ S.end; i++ do
6:         if queue not empty then
7:             sisNode = queue.peek()
8:             while sisNode.location = i do
9:                 graph.addNode(sisNode)
10:                sisNode = queue.peek()
11:            end while
12:        end if
13:        for j = max; j ≥ i; j−− do
14:            if Not nested of last match then
15:                if match(i,j) then
16:                    create node
17:                    create sisNode
18:                    graph.addNode(node)
19:                    queue.add(sisNode)
20:                end if
21:            end if
22:        end for
23:    end for
24: end for
```

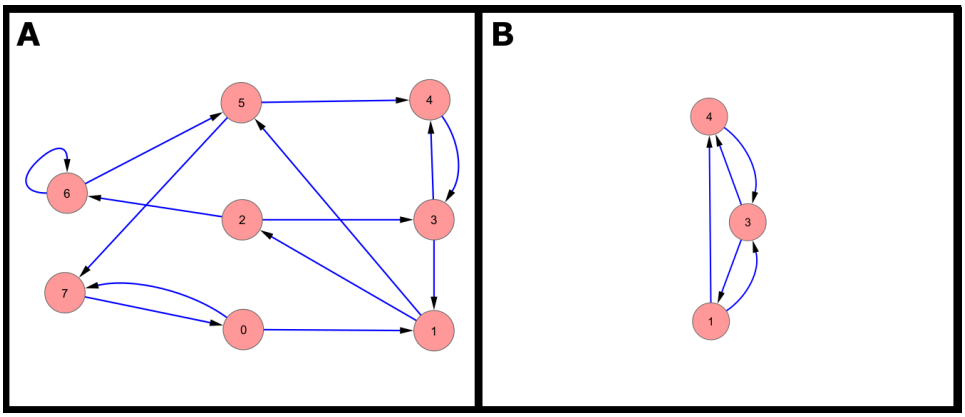

Figure 3.2: An example of a three node subgraph being sampled form a dual graph. **A** is the complete dual graph and **B** is the sampled subgraph.

3.2.3 Sampling of Subgraphs

The created graph now becomes the search space for frequent subgraph mining. Since edges are the unpaired nucleotides between two adjacent nodes, two nodes that are not adjacent would not have an edge connecting them but still form a valid subgraph. This makes the use of a pattern-growth method for searching the graph insufficient since many potential graphs will be missed. Therefore it is necessary to sample sets of nodes, which changes the edges between the nodes. For example, if the nodes 1, 3 and 4 are selected from the graph in Figure 3.2 A, it would produce the subgraph in Figure 3.2 B. Note that the edges (1, 2) (2, 3) and (1, 5) (5, 4) become edges (1, 3) and (1, 4) respectively.

This causes a unique challenge that cannot be solved with common FSM algorithms. It could be more appropriate to view the graph as a set of nodes, and a subgraph as a subset. The consequence is that the search space is the power set of the set of nodes. Fortunately many of the subsets produced are invalid subgraphs. An invalid subgraph is one where nodes are overlapping or contain nodes that are disconnected. Regardless the search space is large enough that a deterministic search of all subsets is infeasible. For this reason it was decided that a sampling approach would be used with the assumption that if the sample size is large enough the proportions of frequent subgraphs in the sample would be equal to that of the entire graph.

Two sampling methods were used, a random sampling method and a k-nearest neigh-

bor sampling method. The parameters for the random sampling method are the number of iterations K and the sample size S. For each iteration, up to S nodes are selected one at a time. When a node is selected all nodes overlapping the node are flagged as "unavailable". All subsequent nodes are checked for this flag and if they are flagged a different node is selected. This function is also parameterized to allow for partial overlaps. The sample is then checked for distinct valid subgraphs. The first required property of no overlaps was checked during the sampling. The second property is checked by creating subgraphs of size one and subsequently adding nodes if they contain at least one common molecule. This creates a set of disjoint subgraphs which are added to a list of subgraphs.

If a large sample size is selected it may be difficult to find such a node due to the large number of overlaps. To mediate this problem a set threshold value for "misses" is set and if it is reached the sample is returned as is. Therefore the specified sample size is a maximum and there is no guarantee that it will be reached. Generally the number of nodes needed to get complete coverage of the graph, or in other words the case where no more nodes can be added to the sample, is much smaller than the total size of the graph.

The second sampling method, k-nearest neighbor, uses a k parameter rather than a sample size parameter. A random node is selected in the graph and again all overlapping nodes are flagged as "unavailable". The algorithm then adds $(k-1)/2$ nodes in both the 5' and 3' direction until a subgraph of size k is created, if enough nodes are available (Algorithm 3). This method is preferred over the random method since it has a bias toward creating structures with elements close to each other which are more likely to occur in nature. This is because elements closer together form smaller loops, which have lower free energy than large loops and are therefore more favorable. Furthermore it draws out more intramolecular nodes, which are vastly outnumbered by intermolecular nodes. This method has an additional option to allow unrestricted overlap of intermolecular and intramolecular nodes, but not two nodes of the same type. This allows subgraphs to represent both states of a mechanism, which are when the molecules are separate and when they are binded.

Another parameter of the sampling process is a restriction on the number of molecules contained in a subgraph. Due to the large number of intermolecular nodes, subgraphs representing large chains of molecules readily appear in the results. Currently, known

Algorithm 3 Pseudo-code for K-nearest neighbor sampling.

```
1: sample = new List(Node)
2: randomIndex = generateRandomIndex()
3: node = nodes.get(randomIndex)
4: index = randomIndex
5: K = sampleSize − 1
6: sample.add(node)
7: flagNodeAsUnavailable(node)
8: for i = 0; i < K/2; i++ do
9:     while NOT node.isAvailable() do
10:        index++
11:        if index ≥ nodes.size() then
12:            break
13:        end if
14:        node = nodes.get(index)
15:    end while
16:    sample.add(node)
17:    flagNodeAsUnavailable(node)
18: end for
19: index = randomIndex
20: for i = 0; i < K − K/2; i++ do
21:    while NOT node.isAvailable() do
22:        index−−
23:        if ( thenindex < 0)
24:            break
25:        end if
26:        node = nodes.get(index)
27:    end while
28:    sample.add(node);
29:    flagNodeAsUnavailable(node)
30: end for
31: for node in sample do
32:    flagNodeAvailable(node)
33: end for
34: return sample
```

RNA complexes are comprised of a small number of molecules. To enforce this, an additional parameter was created to limit the number of molecules. An additional approach to reducing the number of intermolecular nodes and irrelevant structures was the creation of a ncRNA/target mode. This mode separates the output into two groups, one with sequences containing possible ncRNA and one with sequences containing the targets (generally mRNA). Intermolecular nodes are only created between the two groups reducing the number of intermolecular nodes. This has some disadvantages since some *a priori* information is needed to separate the sequences and more complex interactions may be missed. However this is realistic requirement for a variety of domains, such as guides and microRNA.

3.2.4 Canonical Labeling

To determine the frequency of each subgraph, isomorphic graphs need to be grouped and counted. This is accomplished by labeling each subgraph based on topology, node labels and edge labels. The combination of these labels allows for the canonicalization of the subgraphs and greatly increases the speed of comparing graphs. In the general case for simple undirected graphs, this process is as difficult as the subgraph isomorphism problem which is NP-Complete. This is because for a given subgraph, all possible permutations of node/edge labels must be compared and the lexicographically largest or smallest label must be selected [33]. Due to the ordered nature of the directed dual graph representation, the nodes can only be arranged in one way and therefore only one label can exist. This makes it possible for the labeling process to be completed in linear time with respect to the size of the subgraph.

Each subgraph is labeled and compared to other already labeled subgraphs. At this point a unique label based on node IDs is checked for determining automorphism which ensure no duplicates are counted. Every time a unique subgraph is found a "pattern" is created which encapsulates the label information and stores instances of the pattern. Subsequent subgraphs are compared to existing patterns and either added or form new patterns. The combined labels are hashed for fast access and matching. Once all subgraphs are labeled the final output is the set of patterns.

All labels are sequences of digits or boolean values separated by pipe characters. The

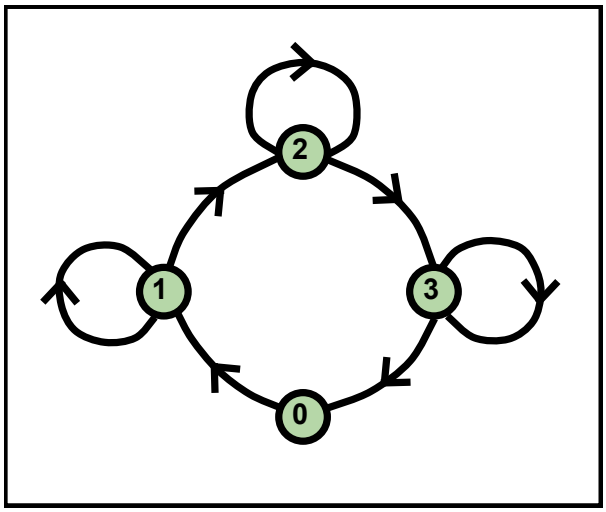

Figure 3.3: Simple dual graph with node IDs.

Table 3.1: Labels corresponding to the graph in Figure 3.3

Property	Label
Topology	0|1|1|2|2|3|3|0
Match Length	4|5|4|5
Inter/Intra	Intra|Intra|Intra|Intra
Distance	3|6|3|6|3|6|3

topology label is created by assigning indices to nodes as they are visited in the subgraph. If a sister node is reached the index of first instance is used. This method can capture any dual graph topology and will have a one-to-one mapping of labels to topologies. Match length and Intermolecular/Intramolecular labels are the values assigned to the nodes, in the order that the nodes are visited. The former describes how many nucleotides are in the complementary sequence and the latter whether the match is between sequences on the same molecule or different molecules. The distance label is similar to the topology label but the values reflect the differences between the positions of the sequences represented by the nodes. The length and distance values are normalized to a discrete scale, 0 to 5 by default (Table 3.1).

The use of a distance label is only applicable to random sampling. This is because the

k-nearest neighbor method will always produce small distance edges. However some mechanism may have very specific spacing between element so the use of the distance label may still be relevant, however without the discrete scale.

3.2.5 Determining Statistically Significant Patterns

Frequencies of patterns on their own are not very useful for determining biologically relevant patterns. Some patterns are far more likely to occur over others by chance. For example a pattern containing one node with the minimal match length will always be most frequent, but is not very useful for further analysis. To solve this problem the algorithm runs twice, once with the real data and again with the data randomize. The size and number of molecules and nucleotide frequency is maintained. This produces two sets of patterns which can then be compared to determine patterns that are significantly more frequent in the biological data. Using a greater number of background graphs was investigated (Table 3.2) however it was discovered to make little difference in number of patterns produced and the metrics. P-values do fluctuate, but only for the very small p-values of the top patterns (shown in Table 3.2) which have almost no impact the results. Due to this observation, and the large computational cost, one background graph is used.

Each pattern in each set has a number of unique embeddings. The first step is to determine the proportion of embeddings (p) associated with each pattern by dividing the frequency (f) over all embeddings (N) (Equation 3.1). The total number of embeddings is equal to the number of valid subgraphs sampled from the entire graph. For the proportions of the sample to reflect the real proportions, the number of subgraphs selected in the sample must be large. As the number of subgraphs sampled approaches the total number of possible subgraphs, the hypothetical proportions approach the real proportions of the graph. A benefit of using proportions over raw frequency values is that they are relative to the total number of subgraphs, which will vary between the two sets.

The next step is to find all the differences between the proportions for each pattern. Due to the large number of embeddings and patterns, the proportions are very small. Taking the differences of these numbers would yield another set of very small numbers. Furthermore there would be a bias towards patterns with large numbers of embeddings which can have large numerical differences even if they are relatively close. For this

Table 3.2: Comparison of results for one graph using ten different randomized graphs with standard deviation for each metric.

Run	Precision	Recall	MinP	MaxP	AvgP	Patterns	Total
1	41.01%	54.23%	3.14E-12	2.39E-06	7.96E-07	3	102
2	44.33%	54.23%	1.37E-13	2.30E-12	1.22E-12	2	102
3	44.33%	54.23%	5.17E-10	6.63E-09	3.57E-09	2	102
4	44.33%	54.23%	3.20E-09	2.02E-08	1.17E-08	2	102
5	44.33%	54.23%	3.33E-10	2.65E-09	1.49E-09	2	102
6	44.33%	54.23%	2.28E-08	1.20E-07	7.13E-08	2	102
7	44.33%	54.23%	1.44E-12	1.86E-11	1.00E-11	2	102
8	44.33%	54.23%	1.53E-09	1.03E-08	5.92E-09	2	102
9	44.33%	54.23%	1.68E-08	1.42E-07	7.96E-08	2	102
10	44.33%	54.23%	1.53E-08	7.96E-08	4.74E-08	2	102
Std. Dev.	**0.009965**	**1.11E-16**	**8.26E-09**	**7.05E-07**	**2.33E-07**	**0.3**	**0**

reason the ratio between proportions are used. This ratio is calculated by dividing the observed proportion (p_{obs}) by the expected proportion (p_{exp}) (Equation 3.1).

To avoid divide-by-zero errors, if the expected frequency is zero then the frequency is set to one. Since the expected frequency cannot be predicted and assuming the value is one skews the distribution, these values are not used in further calculations. However they are retained to estimate the p-value for rare patterns after the distribution is created. Conversely if the observed frequency is zero, then the pattern is ignored also because any assumption would skew the distribution. Therefore the ratios used for the final step are only those that correspond to patterns that exist in both sets.

$$\begin{aligned} p_{obs} &= \tfrac{f_{obs}}{N_{obs}} \\ p_{exp} &= \tfrac{f_{exp}}{N_{exp}} \\ lpr &= log(\tfrac{p_{obs}}{p_{exp}}) \end{aligned} \tag{3.1}$$

The final step is to examine the resulting set of ratios. The distribution of the ratios themselves lay between one and zero where most observations are close to one and drop exponentially when moving towards zero. However taking the log of the ratio produces an

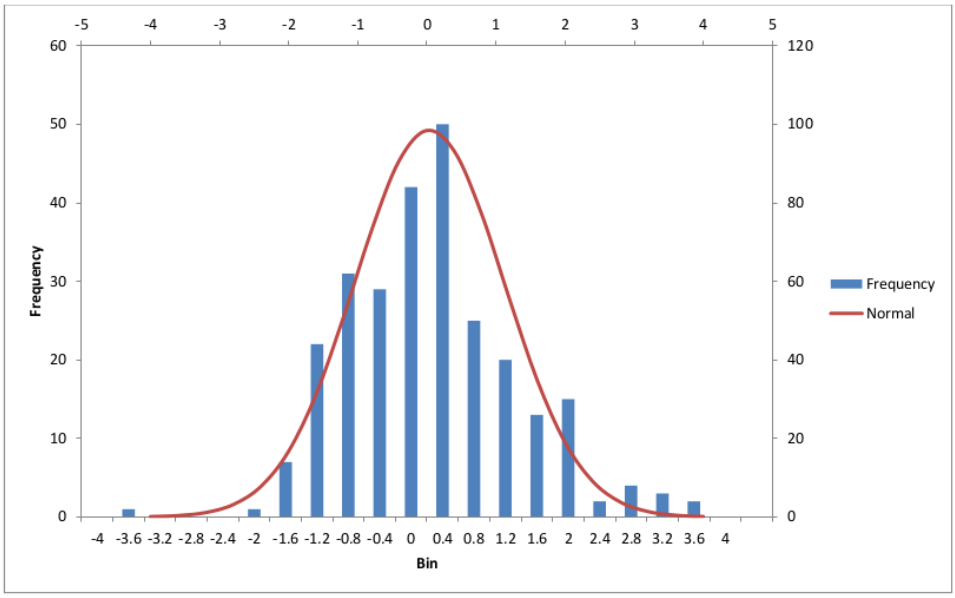

Figure 3.4: Histogram of log ratios of the proportions of subgraphs for patterns produced from the synthetic data set.

almost normal distribution (Figure 3.4). As would be expected there is a skew towards the positive end since they correspond to patterns found in higher frequency in the observed results.

How close the distribution is to the normal distribution was quantified using an Anderson-Darling test for normality. This test is carried out by determining the Anderson-Darling test statistic (Equation 3.2, 3.3). Φ is the cumulative distribution function of the standard normal distribution, and $\overline{x}$ and s are mean and standard deviation of the data values [7]. The p-value is calculated by using equation 3.4 and represents the confidence in the rejection of the null hypothesis, in this case, that the two distributions are equal. The test yielded a p-value between 0.03 and 0.05, which means the difference between the distributions is just significant enough to reject the null hypothesis. However the Anderson-Darling test is very sensitive and therefore it was decided that the distribution is sufficiently close to normal to make the assumption that it is normal. The consequence of this assumption is that some p-values may be smaller than those produced from the actual distribution and subsequently overestimate the significance of certain patterns.

$$A = -n - \frac{1}{n}\sum_{i=1}^{n}[2i-1][\ln(p_{(i)}) + \ln(1 - p_{(n-i+1)})] \tag{3.2}$$

$$p_{(i)} = \Phi([x_{(i)} - \overline{x}]/s) \tag{3.3}$$

$$Z = A(1.0 + 0.75/n + 2.25/n^2) \tag{3.4}$$

The mean and standard deviation of these log ratios are calculated and used to create a normal distribution. From this distribution the p-value is calculated for each pattern's ratio by determining one minus the cumulative probability. Patterns are then ranked by p-value and those with the best p-value represent potential biological mechanisms. The construction of the normal distribution and operations on the distribution were implemented using Apache Math Commons version 3.2.

3.3 Implementation

RiboFSM was written entirely in Java, specifically Java 7. Java was selected mostly due to existing experience with the language. However other factors also influenced the decision. Java's platform, Java virtual machine, is very portable due to its ability to run on most operating systems and hardware types. Also Java is commonly used for many bioinformatics tools and has very good support for integration with other languages. The only external library used was Apache Math Commons version 3.2.

The application is made of 5 packages: core algorithm, data management, graph management, IO and utilities. The high level diagram of the flow of data through the system is shown in Figure 3.5. The algorithm starts with parsing the input files which are in FASTA format. Each input file is treated as a separate "chromosome" object, contained in a "genome" object. As each entry in each file is parsed, "gene" objects are created. These objects are the abstract representation of input with meaningful information such as IDs and locations. These objects may be extended in the future to create a more complex model.

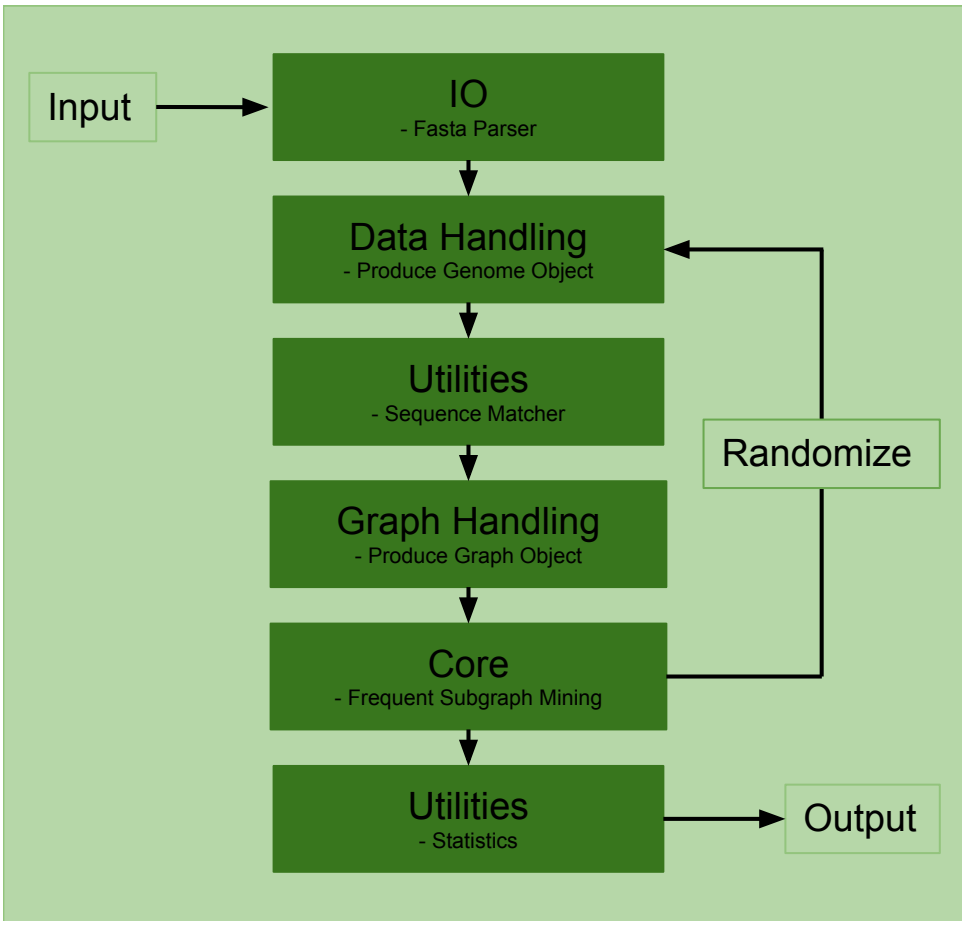

Figure 3.5: High level overview of algorithm execution.

Next the sequence matcher utility finds all possible complementary regions. This is in a separate module so it can be replaced with a more sophisticated matching algorithm in the future. This module produces node objects that are encapsulated in a graph object. The implementation of the graph was created very carefully to ensure that the minimal amount of memory is used. The creation of the graph is the limiting factor in terms of memory usage.

After the graph is created it is passed into the core frequent subgraph mining component. The frequent subgraph mining algorithm is executed in three steps: sampling, subgraph building, and subgraph labeling. Sampling is either random or K-nearest neighbor sampling and this generates a set of nodes. The second step takes the nodes and produces a set of valid subgraph objects, which is an extension of the graph object. Only zero or one subgraphs are created for the K-nearest neighbor method. Finally the subgraphs are labeled and added to their appropriate pattern object. At this point the subgraphs are also checked for duplicates.

Once the frequent subgraph mining is completed, the genome is randomize using a Markov chain of order 1. The process is then repeated and the result is two sets of patterns with varying frequencies. These patterns are then passed into the statistical module which builds the distribution and determines the p-value of each pattern in the observed set of patterns. These patterns are then sorted and the top results are printed to file, along with the metrics if available.

Chapter 4

Experimental Setup

To successfully evaluate a knowledge discovery algorithm, the type of test data and evaluation methods must be carefully chosen. This chapter explains the type of data sets used to test RiboFSM. It also outlines the metrics used to evaluate the performance of the algorithm.

4.1 Experimental Data

Two kinds of data sets were used for evaluation. The first is a controlled synthetic data set and the second is a real data set. Synthetic data is produced by generating random sequences and inserting positive examples. Other examples can be inserted to create various levels of noise. The real data set is from biological data with known examples that can be verified. If the algorithm can perform well for these two cases, where the performance can be empirically determined, then the algorithm can be used with confidence on an unknown data set. A third data set, the mitochondrial genome of *Diplonema papillatum*, was also used as a data set with an unknown mechanism.

Table 4.1: Nucleotide frequencies and percent composition for the *Trypanosoma brucei* mitochondrial genome.

Type	T	A	C	G
minicircle	123703	124685	39013	59515
appx. ratio	36%	36%	11%	17%
mRNA	4966	1913	427	1565
appx. ratio	55%	22%	5%	18%

4.1.1 Synthetic Data

The synthetic data is based on the gRNA mechanism of *Trypanosoma brucei*. The input is split into two sequence types, minicircles and mRNA. 200 sequences are synthetic minicircles and 5 sequences are synthetic mRNA, each sequence being 800nt. The ratio of minicircle to mRNA as well as the length of each sequence was derived from real data. The true ratio of minicircle to mRNA is approximately 38:1 which is close to the used ratio of 40:1. Maxicircle mRNA transcripts are an average length of 737nt [40], which was rounded to 800nt for this data set. Minicircle lengths for *Trypanosoma brucei* are, on average, approximately 1Kb long [37], however 800nt was selected to have a balanced data set. The nucleotide composition was determined by counting nucleotide frequencies in the real data. The frequencies were determined by chromosome type and the approximate percent composition of each nucleotide was used to generate the synthetic sequences. The nucleotide frequencies are summarized in table 4.1.

The model gRNA were made of three complementary sequences. Going from 5' to 3' these are an anchor sequence followed by two stem-loops. The anchor overlaps with the first stem-loop as it does in nature [53]. The anchor is 12nt long, based on the maximum size provided in the work of Hajduk and Ochsenreiter [16]. Stem-loop stem sizes vary so a conservative size of 10bp was chosen. The spacing between these elements was chosen to be 5nt. This produces a pattern with a label "0|1|1|2|2-inter|intra|intra", however the pattern label "0|1|0|2|2-intra|inter|intra" would also be acceptable if the stems were extended by chance.

An algorithm was then used to insert model gRNA into the sequences by modifying regions of the sequences. This algorithm identifies three regions of appropriate length

and location. The first is stored and the reverse complement is inserted into one of the mRNA, creating the anchor. The second and third are also converted into reverse complements and inserted downstream to created the stem-loops. 4 gRNA were inserted into each minicircle which is the median number of gRNA genes per minicircle, with the range being between 3 and 5 gRNA [37].

4.1.2 *Trypanosoma brucei* Data

The second data set used to test the algorithm was a real biological dataset from the public KISS database [40]. The complete data has minicircle sequences and mRNA transcripts both edited and unedited. Since the unedited mRNA is included in the edited mRNA, only the edited mRNA was used. This also reduces the number of false positives due to anchors being found in unedited regions. The data also includes mRNA from normal genes that do not undergo editing, which were removed. In the context of RNA editing, it is often simple to determine which mRNA undergoes editing by comparing the sequence to experimentally determined mature mRNA or polypeptide sequences. The final data set includes 455 minicircles and 12 mRNA sequences. The approximate total length was 358,000nt for the minicircle sequences and 8,800nt for the mRNA sequences.

4.1.3 *Diplonema papillatum* Data

This data set was used to test the algorithm in a case where the mechanism is unknown. The data was taken from the work of Kiethega et. al. [26]. It includes the complete mitochondrial genome as well as expressed sequence tags (EST) which were used as the first input where ppRNA are expected to be found. This portion of the data contains approximately 6,000 sequences. The second input are the 9 modules of the cytochrome oxidase subunit 1 gene, since this is the only gene the authors have sequenced so far. The total size of the data set is 3,450,000nt for the first input and 1600nt for the second. All ESTs with unknown sequences (only 'N' characters) were removed and areas with low complexity were masked with 'N' characters. This brought down the size of the data set to 2,816,000nt. This is still significantly larger than the other data sets so it was also useful for testing the scalability of the algorithm.

4.2 Data Randomization

In order to create the background graph for determining biological significance, random data was created. This was done using a Markov chain of order 0 and 1 to randomize the input sequences while maintaining nucleotide and dinucleotide frequencies respectively. The randomized sequences undergo the same procedure as the input data to produce a background graph and a set of frequent patterns from this graph.

4.2.1 Markov Chain Order 0

A DNA sequence in the Markov chain model is a sequence of random characters over the alphabet {A, C, G, T}. The probability of selecting a character depends on the m characters that appear before it. If the Markov chain is of order 0 ($m = 0$), then the probability of each character appearing in the sequence is independent. Therefore the random sequences generated using this method use probabilities based solely on the nucleotide frequencies of the real sequences.

4.2.2 Markov Chain Order 1

With Markov chain order 1 ($m = 1$), each character in the sequence depends on one character before it. Therefore a 4x4 transition matrix needs to be created which holds the probability of each nucleotide occurring after each other nucleotide. The probabilities are derived from dinucleotide frequencies of the real sequence. The first character is determined from single nucleotide frequencies and each following character is determined from the transition matrix. This is the most widely used model in the statistical analysis of biological sequences, and is the preferred method in this work.

Table 4.2: Contingency table.

	Predicted	
Actual	T	F
T	TP	FN
F	FP	TN

4.3 Metrics

Metrics allow for the validation of the method with a data set containing known examples of RNA structures and interactions. This allows for choosing good parameters for analyzing new unknown data sets and to assess the performance of the algorithm. It also allows for finding weaknesses in the algorithm and allows for comparison to other methods.

4.3.1 Contingency Table

The four common metrics are: accuracy, precision (positive predictive value), sensitivity (recall) and specificity. All four of these metrics can be calculated using the four values presented in a "contingency table" (Table 4.2). The rows in this table correspond to the number of examples that are positive and negative in the actuality and the columns are those predicted by the algorithm. This creates four cells corresponding to the true positives (TP), false negatives (FN), false positives (FP) and true negatives (TN). How these values are used to calculate the metrics are shown in equations 4.1 to 4.4.

$$Accuracy = \frac{TP + TN}{N} \tag{4.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{4.3}$$

$$Specificity = \frac{TN}{TN + FP} \tag{4.4}$$

In the context of the experiments the examples are subgraph embeddings. The actual positive examples are the inserted or real gRNA. When the synthetic data set was used, the subgraphs had to have a label matching the correct pattern label and have node locations that are within a specified distance from the actual positions. This distance was set to 5nt for all test since any extension beyond this would be unlikely to occur by chance. The *Trypanosoma brucei* data uses somewhat different approach outlined in the next section.

The predicted positive examples are the embeddings corresponding to the patterns which meet p-value and frequency thresholds. Since a sampling technique was used the total number of real positive examples only includes those that were sampled and all others remain unclassified. This is unlike conventional data mining techniques where the complete set of positive and negative examples can be enumerated. This is an important factor to consider when interpreting the results.

Like with other biological data sets, the ones used in these experiments have much fewer positive examples than negative. For the synthetic data, the sample produced contains a 1:150 positive to negative example ratio. With imbalanced data like this, precision and recall are more effective as a metric of performance [8]. Precision is the proportion of true positives in the set of predicted positive examples and recall is the proportion of true positives in the set of actual positive examples. Often one can be increased at the cost of decreasing the other, but not necessarily. The ideal score for both of these measures is 100%, which would be the case when the predicted positive set contains only and all of the the true positives.

Another problem with imbalanced data is that the precision/recall values have low numerical value and may seem to be very poor when in actuality they are not. For example, if the precision is only 10% one would think the result is very poor considering the maximum score of 100%. However since only 0.66% of the total number of examples is positive, finding 10% is a fairly good result. To ensure that this difference is apparent, a "straw man" is generated for each result, which acts as a baseline. The baseline is calculated using the same number of positive and negative examples as the actual result but they are chosen at random.

The objective score used to compare the final results is the F score (Equation 4.5). This score is based on the harmonic mean of the precision and recall. The harmonic

mean is different from the arithmetic mean because it is less sensitive to outliers. This is appropriate in this context since values closer to each other are preferred. For example, 1% precision and 100% recall would have a higher arithmetic mean than 50% precision and 20% recall, even though it is less meaningful. However the harmonic mean is higher in the latter than the former. The F score is also weighted (β), which allows for favoring precision over recall or vice versa. In these experiments the value is fixed to 1, since neither is preferred.

$$F_\beta score = (1 + \beta^2) \times \frac{precision \times recall}{(\beta^2 \times precision) + recall} \tag{4.5}$$

The F-scores for the baseline are also calculated. This has two benefits; the first is that it can be used to objectively determine whether the algorithm is more effective than a random classifier and the second is the ability to correct the F-score of the observed data. The former is done by running a paired T-test. The result of the one-tailed test shows whether the null hypothesis, the classifier being the same as the random classifier, can be rejected. The test was carried out using Microsoft Excel 2010. The latter is calculated by subtracting the observed F-score by the baseline F-score. This process allows different runs to be compared based on how much more effective they are at classifying the data over a random classifier.

4.3.2 Validation of *Trypanosoma brucei* Data

The *Trypanosoma brucei* data set is from the KISS database which also stores all experimentally verified gRNA as well as gRNA predicted using a modified BLAST search called WUBLAST [36]. These matches can be saved to file in General Feature Format (GFF). This a standard format for annotating sequences with feature information. The first two columns of this file contain the sequence ID and the source of the sequence. The next three columns describe what the feature is, followed by the start and end locations in the sequence. This is followed by additional information: score, strand (+ or -) and phase (0,1,2 or no phase indicated by "."). The final column is for feature attributes which is any information about the feature with its own separator.

For the purposes of validating the results in this work, only the sequence ID, feature

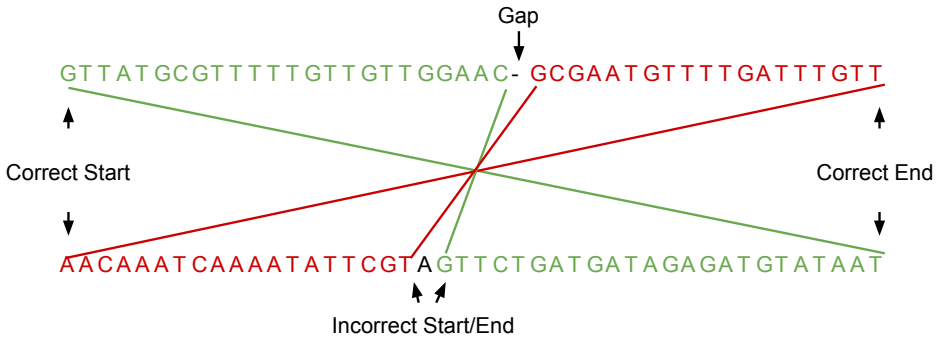

Figure 4.1: Example of a complementary sequence with a gap. If the algorithm were to look for the correct start and end points, it would fail and mark the subgraph as a negative example.

name, start, end and feature attributes are used. This file is parsed when the input sequences are parsed and annotate the input sequences. Only "match" features are extracted, which are the gRNA anchor regions on the mRNA. The start and end locations of the anchor site is recorded. Next, the attribute column is parsed, which contains the corresponding gRNA ID as well as the start and end locations on the minicircle sequence. Whenever a positive subgraph is identified, it is checked for an intermolecular node. The molecule information is extracted from the node and the node location is matched to annotated locations. If a match is found, the subgraph is considered a positive example. This method is not as stringent as the method used for synthetic data.

One problem that occurred was that many real anchor matches have a gap. A gap is where one strand in a stem has an extra nucleotide, that if it were removed, the sequences would match on both strands (Figure 4.1). The naive method used for finding complementary regions does not create stems with gaps. If these were ignored, many potential positive examples would be missed reducing the precision and recall. To handle this case, the start location on the mRNA is matched to the end location on the gRNA, and vice versa. This way, if the stem is split into two nodes in the graph, they will still be categorized as positive examples.

Only the positive examples that occur in the search space (the graph) are considered. This was done because of the difficulty involved in determining all possible positive examples in the *Trypanosoma brucei* data, which would be any combination of stems on the molecules containing an anchor. The consequence of this is that the recall is

Table 4.3: Summary of RiboFSM parameters.

Parameter	Type	Default	Min	Max	Description
Length	int	8	1	100	Minimum length of complementary sequence.
%GU	double	0	0	1	Maximum proportion of base pairs that are GU.
Iterations	int	2000000	1	∞	Number of iterations (apprx. number of samples).
K	int	3	1	100	Maximum number of nodes per subgraph.
Genes	int	2	1	200	Maximum number of genes (molecules) in subgraph.
Norm	int	3	1	100	Discrete scale for length, 0 - Norm.
P-Value	double	0.001	0	1	P-value maximum.
Support	int	5	1	1000	Minimum number of embeddings.

specific to a set of parameters and makes some comparisons between runs misleading. To mitigate this, the number of true positives for each run are also presented.

4.4 Parameters

RiboFSM has 8 parameters as described in Table 4.3. The default parameters were used as a starting point specific to the synthetic data. The parameters K and number of genes remain fixed for all experiments. The percent GU parameter was fixed for the synthetic data experiments since the inserted structures do not contain any GU base pairs. This value was set to a default 25% for the real data. All other parameters were checked individually and the optimal value were selected. A combined run of all the best parameters was then carried out to obtain the final performance values. Each set of parameters was tested with at least six replicates.

Chapter 5

Results and Discussion

This chapter describes the results obtained from running RiboFSM as outlined in chapter 4, as well as a discussion of the significance of each result. Accompanying data can be found in Appendix C.

5.1 Synthetic Data

RiboFSM was tested by modifying one parameter at a time and recording the precision and recall as well as the amalgamated score. Each subsection will discuss the effect of the parameter on the quality of the result.

5.1.1 Complementary Region Length

Complementary region length is the minimum number of nucleotides needed in a complementary region to create a node in the graph. The smaller this value is, the more nodes will be created. The testing range chosen here is between 6 and 10 nucleotides. Values below 6 create and unnecessarily large graph and has a large impact on memory usage. The maximum value was chosen because the smallest stem in the inserted structures is of size 10. Choosing a value greater than 10nt would lead to many of these stems

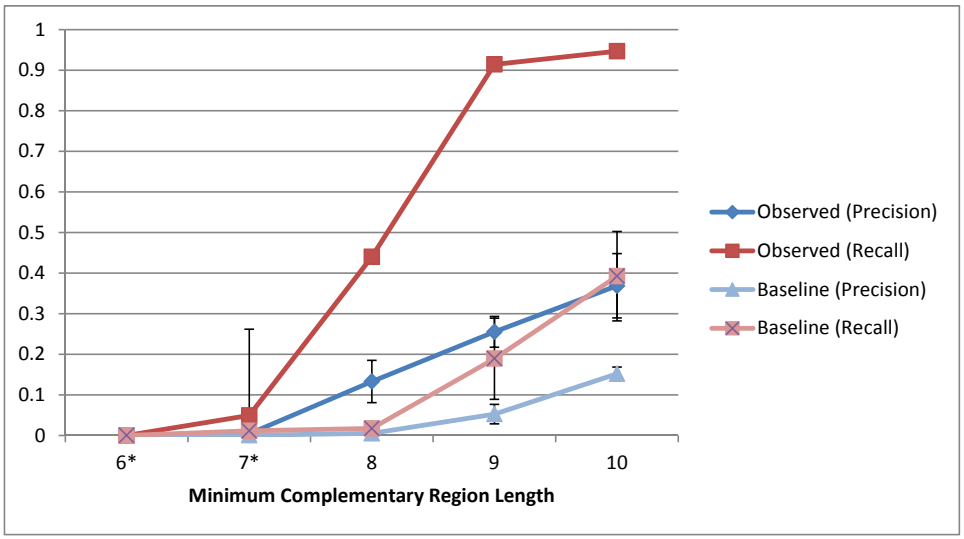

Figure 5.1: Precision and recall for the observed data and baseline with different minimum complementary region lengths. Values with an asterisk have a higher p-value cutoff to produce meaningful results. Error bars represent the 95% confidence interval.

being missed in the graph.

The results for this test are presented in Figure 5.1. When short lengths of 6nt and 7nt were used, nearly no positive examples were found. This may have occurred due to the k-nearest neighbor method used. Lower values create many more nodes increasing the likelihood of nodes occurring between the nodes in the subgraph corresponding the correct structure. This could be mitigated by skipping a random number of nodes when growing the subgraph. This may be required for other applications where stems are further apart than in the synthetic data.

The scores increase after 7nt, and only increase in precision for 10nt. The precision value is approximately 0.2 higher over the expected baseline, and recall value is up to 0.7 better over the baseline. The number of patterns produced ranges from 27 patterns for 6nt to 7 patterns for 10nt. The number of embeddings in the positive set was reduced from 8500 at 7nt, to 1600 embeddings for 10nt (See Appendix C). Therefore as the minimum increases, the results become more refined without a loss of recall. The value with the highest score (10nt) and the value with the largest difference from the baseline (9nt) were chosen for further testing.

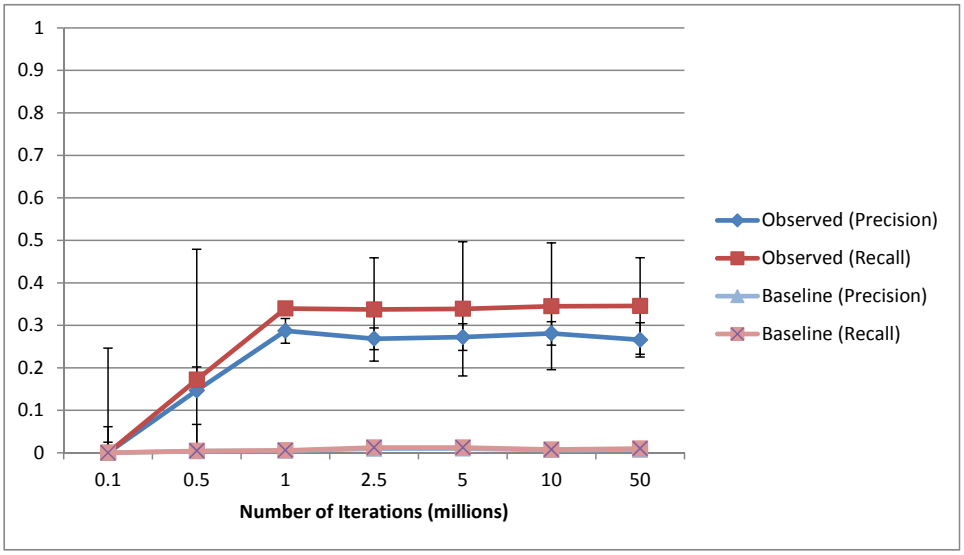

Figure 5.2: Precision and recall for the observed data and baseline using different numbers of iterations. Error bars represent the 95% confidence interval.

5.1.2 Iterations

The number of iterations is the number of size K subgraphs sampled from the graph. Many of the sampled subgraphs are invalid and are discarded so the true sample size is usually much smaller than this value. The algorithm scales well in terms of the number of iterations so large numbers can be selected. The values selected were 0.1, 0.5, 1, 2.5, 5, 10, 100 million iterations.

For the chosen value of K, all possible subgraphs that can be found using K nearest neighbor are found within the first million iterations. This was determined by investigating the total number of embeddings which does not increase even at 100 million iterations (data not shown). The subgraphs sampled after a million iterations are duplicates and are discarded. Many more subgraphs of size K exist where the nodes are not joined to their nearest neighbor, but they are unlikely to represent biological structures due to large distances between stems. Even 100,000 iterations captured 88% of the possible subgraphs. For other, larger data sets, or with different parameters that increase the number of nodes, a larger number of iterations may be required. As a reference point, the graph produced in this experiment contained 48,000 nodes and produced

25,600 subgraphs. The previous experiment with a smaller length minimum produced a 535,000 node graph and after 2 million iterations, 485,000 subgraphs were found. Since 100 million iterations can still be used in a relatively short time, this suggests that the algorithm can handle very large graphs.

5.1.3 Length Normalization

The lengths of the complementary regions can vary from the minimum size to hundreds of nucleotides. If these raw values were used there would be a large number of different combinations of lengths yielding many different patterns. This can be a problem since two patterns can be different in an insignificant way, such as one stem being one nucleotide shorter, and be classified as a different pattern. To solve this the lengths are normalized to a discrete scale with a size specified by this parameter. The minimum is 1 which would lead to all lengths labels being equal, effectively removing the effect of the length label on mining. The maximum is 10 which gives labels similar to those that would be produced without normalization.

Due to the effect of this parameter on the number of patterns, a more stringent p-value was used. This allowed for a more consistent number of subgraphs in the positive set to better isolate the effect of this parameter.

The results for normalization are erratic (Figure 5.3). There was a significant improvement in all metrics, as well as stability, between no normalization and a two bin (short/long) or three bin normalization. Increasing the number of bins beyond that drastically reduced the performance of the algorithm. However using 7 or 8 bins created a dramatic rise in recall. The baseline precision remains nearly zero for all values, which would be expected. There was a slight increase in baseline recall which can be attributed to an increase in the size of the positive set. The number of embeddings in the positive set ranges between 200 and 3600, which likely contributes to the irregularity in the graphs. These values fluctuate because the number of patterns above the p-value threshold increases since more patterns unlikely to occur by chance are produced. At the same time the patterns become more specific, reducing the number of embeddings per pattern.

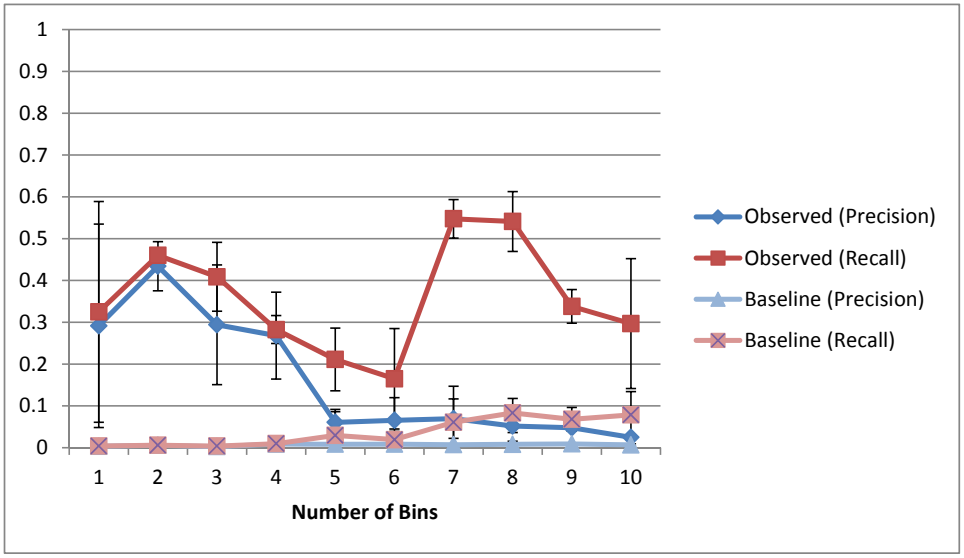

Figure 5.3: Precision and recall for the observed data and baseline with different levels of length normalization. Error bars represent the 95% confidence interval.

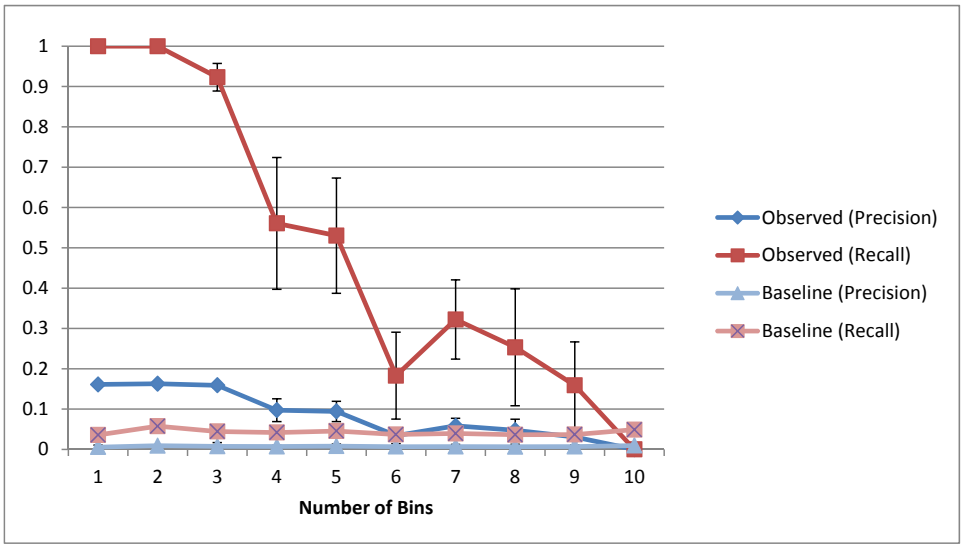

Figure 5.4: Precision and recall for the observed data and baseline with different levels of length normalization with a fixed number of positive examples (1000). Error bars represent the 95% confidence interval.

To mitigate the irregularity, another experiment was carried out where the number of examples in the positive set was kept constant. In this experiment only the most significant patterns, with a total of no more than 1000 embeddings, were counted as the positive set, instead of using a p-value cutoff. As shown in figure 5.4, the trend is more regular with a clear decrease in all metrics as the the number of bins increases. This result would be a strong indicator that this label is not useful for discovering biologically relevant patterns. However this could also be a consequence of the relatively short stems which are all of the same size. The real data contains larger stems and a greater variety of stems. Length normalization may be more useful in that data set.

The average p-value of the positive set ranges from 6×10^{-2} for 1 bin to 1.27×10^{-7} for 10 bins. However when the lower number of bins was used, the true positives were associated with the patterns with higher p-values. When a larger number of bins was used the true positives were more dispersed. Therefore even though the p-values were lower on average for fewer bins, they are more useful for discerning good patterns. An alternative explanation for the poor performance of length normalization may be due to shortcomings in the statistical model.

Although these results show that length normalization leads to poorer precision and recall, they do produce more specific patterns. A more specific pattern provides more information than a general pattern, which may be of more interest to biologists. If a pattern is too general it may be difficult to design an experiment to biologically verify the existence of a mechanism. It may be more beneficial to produce a larger set of candidates that are easier to verify than fewer general patterns that are difficult to verify.

5.1.4 P-Value Cutoff

This parameter is the maximum p-value allowed for a pattern to be included in the positive set. Varying this value has a large effect on precision and recall. Increasing this value increases recall but lowers precision and decreasing this value produces the opposite effect. This value also represents the confidence in the difference between the observed and expected frequencies for each pattern. Generally a p-value of at least 0.05 (95% confidence) is selected, but in this experiment a value of 0.1 will also be tested. A minimum value of 1×10^{-7} was used since the best results generally do not have markedly

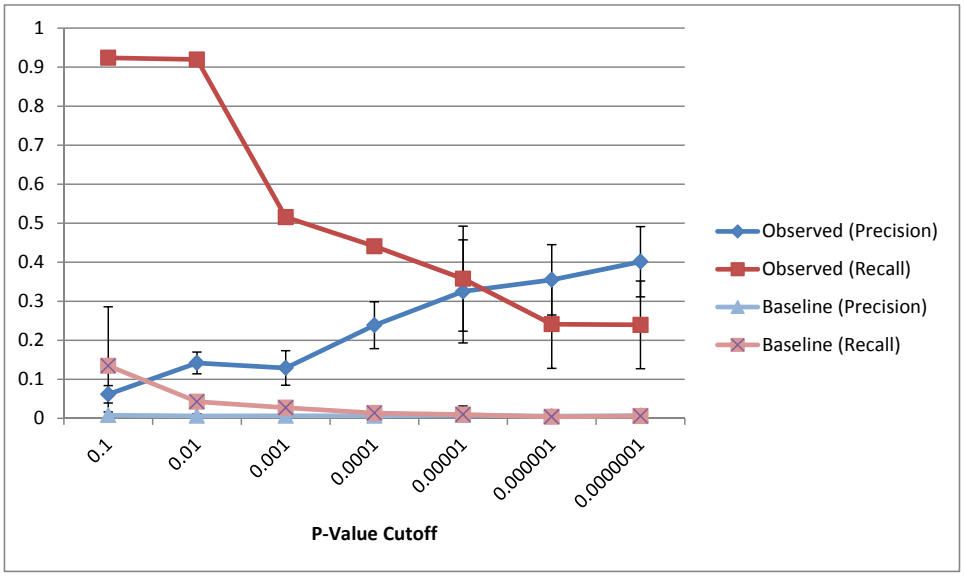

Figure 5.5: Precision and recall for the observed data and baseline with different P-value cutoffs. Error bars represent the 95% confidence interval.

smaller p-values.

As expected, as the p-value cutoff decreased the recall decreased while the precision increased (Figure 5.5). The result also became more variable as the p-value cutoff decreased. This is because the positive set has fewer patterns as the cutoff is decreased, and becomes more easily influenced by variations in pattern frequencies between runs. The highest score was obtained with a p-value cutoff of 0.01, however this leads to a large recall/precision imbalance. Using a cutoff of 1×10^{-5} has a much better balance with only a moderate reduction in score. This balance point changes based on other parameters so additional tests were needed to determine the optimal p-value for further tests.

5.1.5 Support Cutoff

Support is the minimum frequency a pattern must have to be considered a positive result. Similar to p-value, this value has an effect on precision and recall. Often patterns with small p-values have high frequency but not always. Due to this effect, the p-value

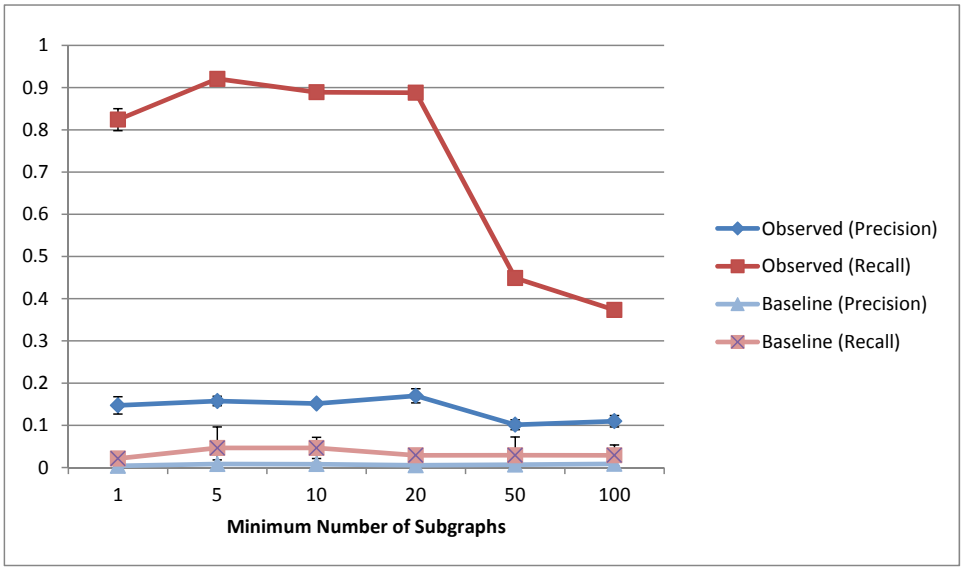

Figure 5.6: Precision and recall for the observed data and baseline with different support cutoffs. Error bars represent the 95% confidence interval.

cutoff was raised for this set of experiments. The minimum value tested was 1 which is equivalent to no restriction on frequency. The maximum value was set to 100 since the best results rarely have higher frequencies.

Figure 5.6 summarizes the effect of changing the support cutoff. Increasing the minimum support up to 20 embeddings had little effect on either precision or recall. This indicates that the pruned results included an equal number of positive and negative examples. As the support minimum increases above 20, all metrics begin to drop. This implies that more positive examples are being removed than negative examples. This result supports the hypothesis that frequency values alone are not a good indicator of biologically relevant patterns. What is surprising is that this data is suggesting that the most frequent patterns are even less likely to be biologically relevant the those with lower frequency.

The increase in recall between support cutoffs of 1 and 5 decreased with an increased number of replicates. However a near 10% increase still remained which may indicate that there is some positive effect. For this reason values of 10 and 20 were selected for the final test as well.

Table 5.1: Parameters selected for the final experiments.

Run	Length	% GU	Iterations	K	Genes	Norm	P-Value	Support
1	9	0	1000000	3	2	2	0.00001	10
2	10	0	1000000	3	2	2	0.00001	10
3	9	0	1000000	3	2	2	0.00001	20
4	10	0	1000000	3	2	2	0.00001	20
5	9	0	1000000	3	2	8	0.00001	10
6	10	0	1000000	3	2	8	0.00001	10
7	9	0	1000000	3	2	8	0.00001	20
8	10	0	1000000	3	2	8	0.00001	20

5.1.6 Best Parameter Combinations

Using the results of the analyses of each parameter, a test case of the best parameters was created (Table 5.1). This test case has 8 different combinations with 6 replicates per parameter combination. The average of the 6 replicates was used for comparison.

The results of this test case are summarized in Table 5.2. The initial results showed precision values significantly lower than the recall. For this reason the test case was run again with a more stringent p-value. However this lead to a reduction in both recall and precision. The best score was for run number 4, with a F score of 63.18%. The best single run was the third replicate of run number 2 which yielded a precision of 98.31% and recall of 79.48%. This run also had the maximum precision of all runs. The highest recall recorded was 97.57% from the third replicate of run 4. All test performed much better than the random classifier ($P = 6.29 \times 10^{-}5$) as determined by a paired t-test.

The baseline performance varies between runs with some runs having higher precision and recall than others. This may be an indicator that a specific run is only better than others because it should be better simply by chance. To objectively compare the various runs, the results should be corrected depending on the corresponding baseline. To accomplish this the F score of the baseline results are subtracted from the F score of the observed results. Considering this, the best result is run number 1, with a 5% higher corrected F score than run number 4.

The best result produces exactly one significant pattern which contains nearly all of

Table 5.2: Results of final set of runs with F score. Bold results are the best for the given metric.

	Observed			**Baseline**			
Run	**Precision**	**Recall**	**F-Score**	**Precision**	**Recall**	**F-Score**	**F-Score Diff**
1	46.81%	67.77%	55.38%	5.13%	6.52%	5.74%	**49.63%**
2	42.72%	52.95%	47.29%	16.65%	16.03%	16.33%	30.95%
3	38.80%	57.36%	46.29%	4.70%	7.79%	5.86%	40.43%
4	**51.21%**	**82.46%**	**63.18%**	15.74%	23.76%	18.94%	44.24%
5	19.71%	63.73%	30.11%	5.44%	16.39%	8.17%	21.94%
6	25.38%	82.33%	38.79%	15.33%	53.32%	23.82%	14.98%
7	17.84%	45.99%	25.71%	5.70%	14.14%	8.12%	17.59%
8	27.64%	76.46%	40.60%	16.39%	45.38%	24.08%	16.52%
Avg.	33.76%	66.13%	43.42%	10.64%	22.92%	13.88%	29.53%

Table 5.3: The pattern result of the best run (run 2.3).

P-value	**Frequency**	**Label**
9.40E-7	532	0|0|0|-|0|1|1|2|2|-|inter|intra|intra|

the inserted structures (Table 5.3). This is the pattern containing an anchor and two stem-loops where the anchor starts just before the first stem-loop. The other runs for this set of parameters also included this pattern, as well as a similar pattern that have the anchor incorrectly downstream of the two stem-loops. The average p-value for all of the runs was 8.04×10^{-7}. The false negatives are most likely the other correct pattern of |0|1|0|2|2|, where the anchor is entirely within the hairpin. The only way for this result to be better would be for the second correct pattern to be the second result. The other two results for run 2 were similar, with one or two patterns with high frequency of correct embeddings.

The best result had a very good average p-value, however the worst result had a very similar average p-value (5.22×10^{-7}). Furthermore the most significant patterns in this result set have p-values in the 1×10^{-12} range. This is because these patterns contain the stem-loop structures but are missing an anchor, or have a misplaced anchor. Good

p-values are always indicators of exceptional structures and interactions, however they do not guarantee that these structures exist in nature. All that can be said is that the set of patterns with a p-value less than a specified threshold have a high likelihood of containing biological structures.

5.2 *Trypanosoma brucei* Data

The second data set used to test the algorithm was a real *Trypanosoma brucei* data set from the KISS database. The default parameters used as a starting point were different than those used for the synthetic data. Minimum complementary length was raised to 15nt since the real anchors are, on average, longer than the inserted anchors. The maximum GU% value was set to 25% rather than zero since the real anchors and stems contain GU matches. The number of iterations was increased to 10 million due to the larger data set size, and consequently larger search space. Normalization was set to 10 bins due to the increased variety of stem lengths and finally the support value was set to one based on the results from the synthetic data.

In this data set maximum GU% is not fixed, so the parameter was investigated. Also minimum complementary region length and normalization were re-visited due to the different composition of anchors in the real data set.

5.2.1 Maximum GU%

The maximum GU% parameter is the maximum proportion of GU base pairs allowed within a complementary region. This is required in the real dataset because this pairing often occurs in nature. It increases the average length of complementary regions and increases the number of nodes. The minimum value tested was 0% and increased in 5% intervals up to 50%. The stems in the KISS database rarely exceed 50% GU base pairing.

The results of this test are shown in Figure 5.7. Recall values below 20% GU are irregular and much worse than random. This is because with the minimum length parameter set to 15nt and low GU%, few of the stems would be expected to occur by

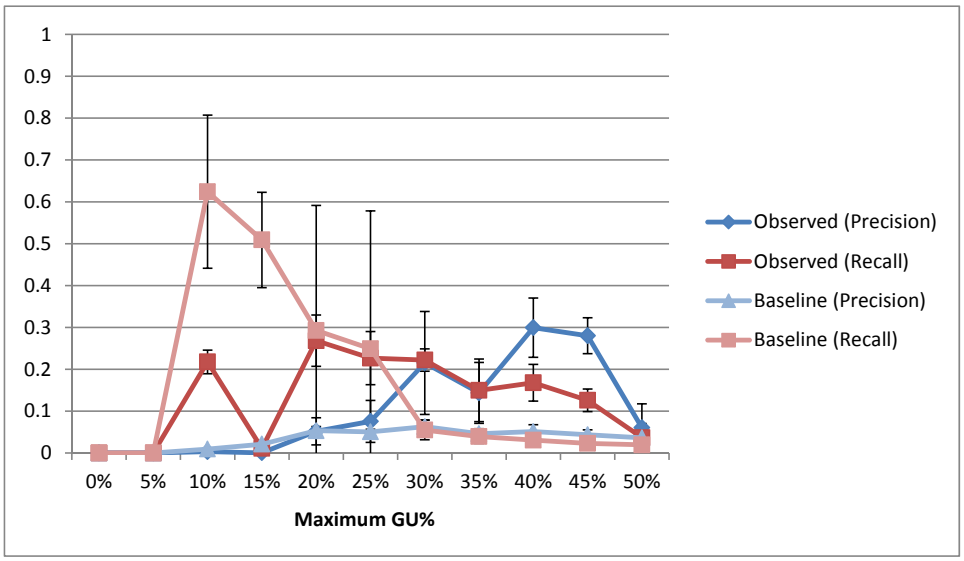

Figure 5.7: Precision and recall for the observed data and baseline, when maximum GU% is varied. Error bars represent one standard deviation.

chance. As a consequence many patterns have high p-values which in turn produces a large positive set. Since the baseline is based on the size of the observed positive set, it would be expected to have high recall simply by chance. Furthermore the baseline values are highly variable, as shown by the error bars. Under these circumstances, the results are unreliable and should be disregarded.

Above 15% GU, the trend was more regular. Precision increases dramatically and recall decreases. They start off being fairly close to the baseline, but the gap increases over 25% GU. The increase in precision is due to the increase in positive examples that contain high numbers of GU pairings. Furthermore the stability improves after this point, as can be seen with the reduced length of the error bars. Above 45% GU the precision drops, which is likely due to an increase in false positives from low complexity regions and poly-A/U tracks. The drop in recall was likely a consequence of the reduced number of significant patterns due to an increased number of subgraph instances in the background graph. To control for this effect, a similar experiment to length normalization was carried out which only selects the top 1000 positive examples.

The results for the test with a fixed number of positive examples is shown in Figure 5.8. This made the results more regular however the recall spike at 10% GU became more

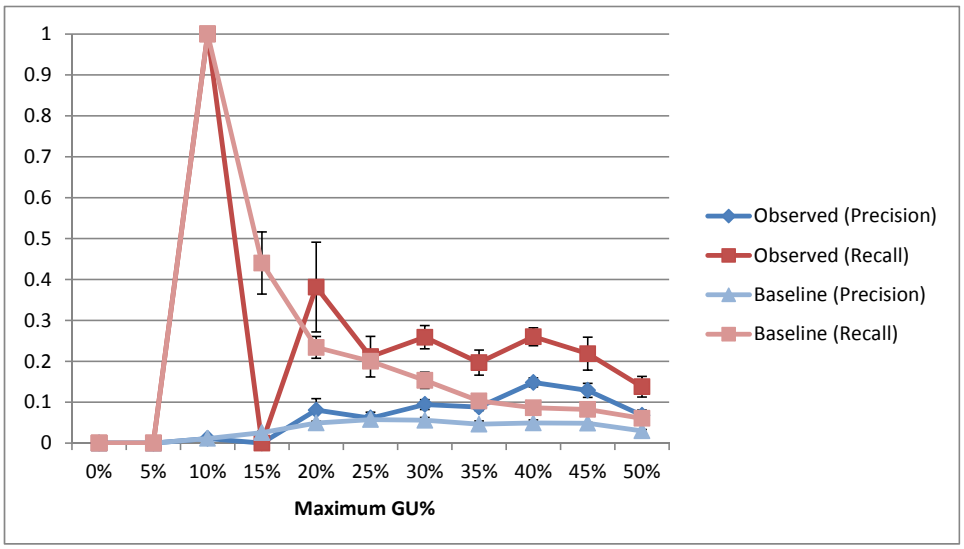

Figure 5.8: Precision and recall for the observed data and baseline, when maximum GU% is varied. A fixed number of positive examples were used (1000). Error bars represent the 95% confidence interval.

pronounced. Again, this is due to low probability that any stem would occur under these parameters. The baseline also retains the recall spike, confirming that this result is insignificant. After 15% GU the precision rises gradually up to 40% GU while the recall remains steady, with a maximum at 20% GU. This is indicative of the number of positive examples increasing while the number of false positives decreasing.

The recall is based on the graph produced with the given parameters. This can generate the very high recall rates seen in Figure 5.8 when the number of positive examples is low and all the patterns have high p-values. To better illustrate the effectiveness of the algorithm the number of true positives was plotted (Figure 5.9). Here the peak at 10% GU is nearly non-existent and there is a clear peak at 40% GU. The precision also peaked at this value so 40% was selected for further analysis.

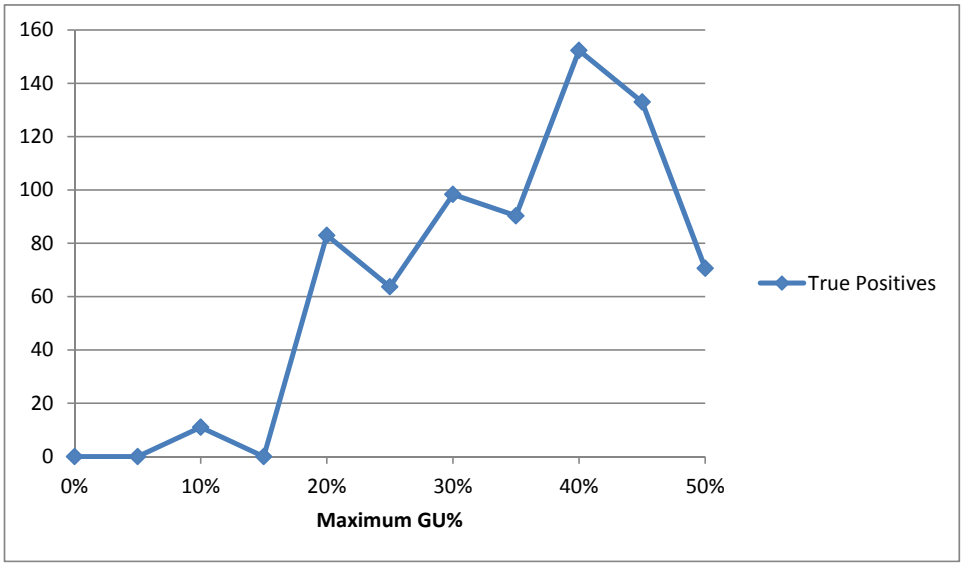

Figure 5.9: True positives for the observed data when maximum GU% is varied. A fixed number of positive examples were used (1000).

5.2.2 Complementary Region Length

Unlike the synthetic data that has a controlled set of stem lengths, the *Trypanosoma brucei* data has a wide range of stem lengths. The lengths are also generally much longer. For this reason this parameter was tested with a wider range for this dataset. The smallest value tested was 8nt minimum length which produced a 3 million node graph. Reducing the number further would have been possible but would require significantly more runtime and resources. The largest value tested was 25nt which only produced 45 subgraphs, indicating that larger minimum would not be useful. P-values were modified slightly for the extreme cases to provide meaningful results.

As seen in the synthetic data, all metrics increase as the minimum increases up to the minimum anchor length of 20nt (Figure 5.10). However in this case the lower minimums have a non-zero precision and recall. Although the increase in metrics is dramatic, the baseline increase is as high or higher. Any value above 14nt has poor performance compared to the baseline. This phenomenon is similar to the low GU%, few stems of length greater than 14nt can occur by chance so all patterns become significant. Below this value however, the metrics are consistently higher than the baseline, between 0.1

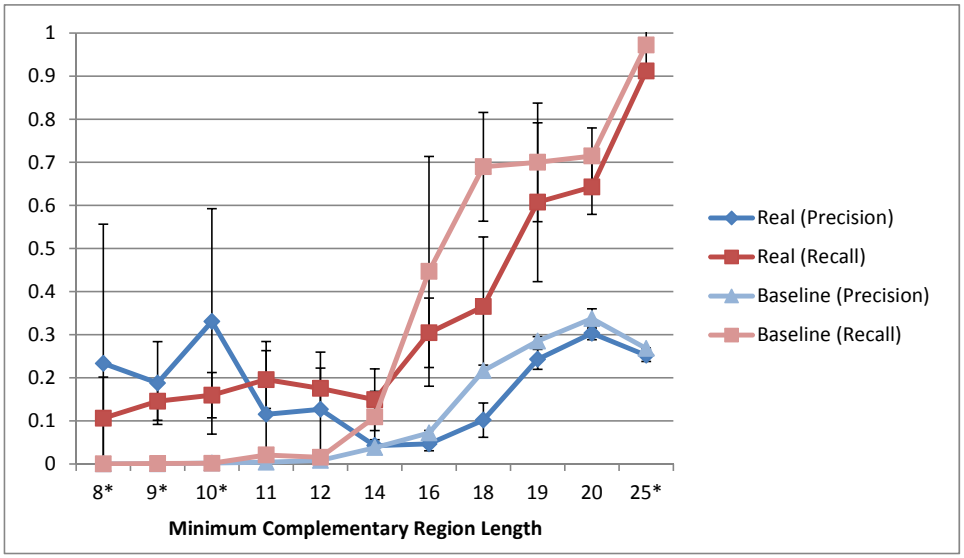

Figure 5.10: Precision and recall for the observed data and baseline with different minimum complementary region lengths. Values with an asterisk have a higher or lower p-value cutoff in tenfold increments to produce meaningful results. Error bars represent one standard deviation.

and 0.4 higher.

Like with GU%, the true positives were plotted separately (Figure 5.11). This illustrates that using higher minimums only reduces the number of true positives and that using a smaller value is more favorable. As the minimum increases over 9nt, the number of nodes decreases. This has two effects: the first is that the number negative examples is reduced making it more difficult to separate out positive examples and the second is that many anchors will not be able to form valid subgraphs because of the lack of nodes on the same molecule.

After 14nt, the number of true positives rises since the positive set grows due to low probability of stems of this length occurring. However these patterns are less useful to biologist since structures with only long stems occur infrequently in nature. This corresponds to the 14nt parameter used by KISS [40] to determine stem loops encompassing anchors. As the value increases over 14nt, the number of these stems is reduced within the graph. Over 18nt, anchors start to get missed by the algorithm reducing the number of positive examples.

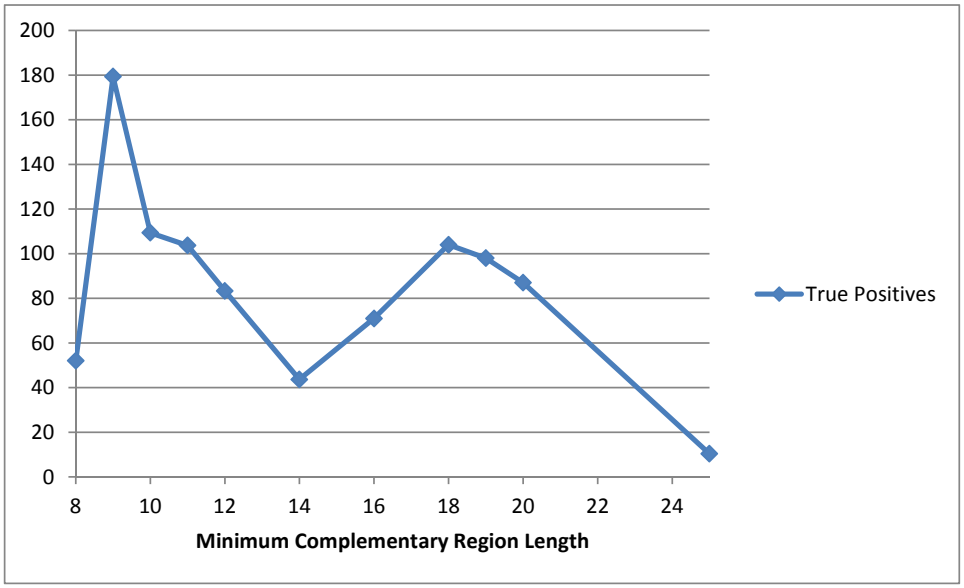

Figure 5.11: True positives for the observed data with different minimum complementary region lengths.

The minimum length of 10nt is a good choice for further testing due to the high precision and fairly high number of true positives. However the ideal GU% was shown to be 40% GU and this test was run with 25% GU. To compensate for the increased average stem length, higher minimum lengths were also tested.

5.2.3 Length Normalization

Length normalization was expected to be more effective for the real data set due to the variety of stem lengths. The variety allowed for a wider range of bin numbers to be chosen. The minimum was the same as the synthetic data, being 1 bin negating the effect of the label. The maximum selected was 22 bins which shows a clear drop off in scores. Increments of 1 bin was used for 1 to 10 bins, then an increment of 3 bins was used for above 10 bins. Again this test was run by selecting the top 1000 examples.

Like the synthetic data, there is a general negative trend as the number of bins increases. Unlike the synthetic data, using one bin is much less effective than more than one bin (Figure 5.12). Also the negative trend is less pronounced, even increasing during cer-

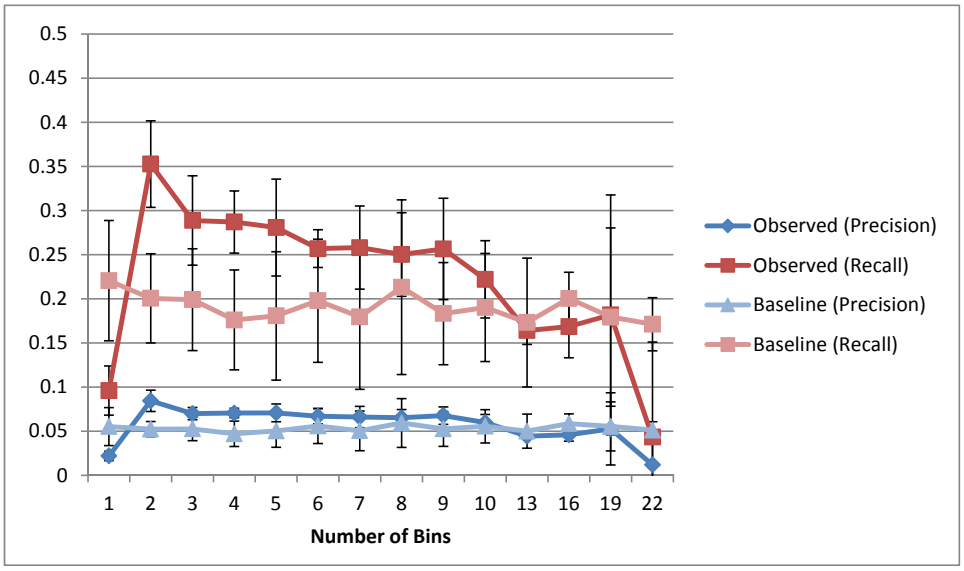

Figure 5.12: Precision and recall for the observed data and baseline, with different levels of length normalization with a fixed number of positive examples (1000).

tain intervals. When looking at the most significant results with the highest proportion of true positives, the labels usually contain the bins corresponding to longer complementary regions. This indicates that this label is drawing out good results. However these results and the results from the synthetic data, show that the most effective method is a boolean label separating long and short complimentary regions. This was chosen for the final test, however 5 bins was also tested since it is the largest number of bins before the recall drops. Also as shown by the error bars, using any value from 3 to 5 bins can lead to a higher recall than 2 bins.

5.2.4 Best Parameter Combinations

Using the results of the analyses of each parameter, a test case of the best parameters was created (Table 5.4). This test case has 8 different combinations with 6 replicates per parameter combination. The average of the replicates is used for comparison.

The results for this test case is summarized in Table 5.5. The best F score was significantly lower than that of the synthetic data set, however still significantly better

Table 5.4: Parameters selected for the final experiments.

Run	Length	% GU	Iterations	K	Genes	Norm.	P-Value	Support
1	10	0.4	1000000	3	2	2	0.01	1
2	10	0.4	1000000	3	2	5	0.01	1
3	12	0.4	1000000	3	2	2	0.01	1
4	12	0.4	1000000	3	2	5	0.01	1
5	14	0.4	1000000	3	2	2	0.01	1
6	14	0.4	1000000	3	2	5	0.01	1

Table 5.5: Final results for the *Trypanosoma brucei* data set. Bold values are the best value for the category.

	Observed			**Baseline**			
Run	**Precision**	**Recall**	**F-Score**	**Precision**	**Recall**	**F-Score**	**F-Score Diff**
1	6.85%	0.33%	0.63%	0.18%	0.01%	0.01%	0.61%
2	46.59%	11.36%	18.26%	0.21%	0.07%	0.11%	18.15%
3	80.52%	15.82%	26.45%	1.15%	0.20%	0.33%	26.11%
4	**84.80%**	**26.29%**	**40.13%**	1.26%	0.38%	0.59%	**39.55%**
5	2.63%	5.84%	3.63%	2.78%	6.06%	3.82%	-0.19%
6	7.48%	21.13%	11.05%	2.89%	8.70%	4.34%	6.71%
Avg.	38.14%	13.46%	16.69%	1.41%	2.57%	1.53%	15.16%

than the random classifier (P = 0.033) as determined by a paired t-test. The best run was determined to be run number 4 with an F score of 40.13%. However run number 2 achieved the highest precision, with one replicate having 100% precision. The other replicates of this run were in the the 40% range while run number 4 consistently had >80% precision. The best recall for a single replicate was 33.63% from run number 3. Again this was only one replicate, while run number 4 was more consistent with >20% recall. Even after correction for the baseline, run number 4 retained the best score.

The best result of run number 4 produced 11 patterns shown in Table 5.6. The two most significant patterns are anchors flanked by stem-loops both downstream and upstream. There is a fair amount of overlap in the downstream stem-loop with the

Table 5.6: Patterns produced by best replicate of run number 4.

P-value	**Freq.**	**Pattern**	**TP%**
9.25E-07	57	0|4|0|-|0|0|1|2|2|-|intra|inter|intra|	100%
2.92E-05	59	0|3|0|-|0|0|1|2|2|-|intra|inter|intra|	91.53%
2.18E-04	19	0|0|2|-|0|1|2|2|1|0|-|inter|inter|inter|	100%
4.47E-04	16	0|4|0|-|0|1|2|0|1|2|-|inter|inter|inter|	75%
7.62E-04	14	0|0|4|-|0|1|2|0|1|2|-|intra|inter|inter|	71.43%
0.001375	60	4|0|-|0|1|1|-|inter|intra|	96.67%
0.001895	11	0|3|0|-|0|1|2|2|0|1|-|inter|inter|intra|	54.55%
0.00224	21	0|0|3|-|0|1|0|2|1|-|intra|intra|inter|	80.95%
0.003987	169	0|3|-|0|1|0|-|intra|inter|	80.47%
0.005346	57	0|2|0|-|0|0|1|2|2|-|intra|inter|intra|	75.44%
0.005664	8	0|4|-|0|1|0|1|-|inter|inter|	37.50%

anchor. The elements that were expected are present but not in the order that was expected. The next two patterns are anchors broken into separate nodes either by gaps or areas of high GU%. The fifth pattern is the most interesting. It appears to not be the correct pattern, but if it is separated into two subgraphs it would produce two |0|1|0| patterns which is a correct anchor with an encompassing stem-loop. The gRNA seems to be able to edit at two different sites on the ATPase subunit 6 gene.

All of the embeddings for the first pattern correspond to the same anchor on the gene cytochrome oxidase subunit 3 and second pattern to NADH dehydrogenase subunit 7. Each embedding links the gene to gRNA gene on different minicircles. These are likely duplicate minicircles, since *Trypanosoma brucei* has many copies of each minicircle. As would be expected, the probability that these minicircle duplicates would occur by chance in the background data is very unlikely so this pattern would be drawn out. These minicircles are amongst those with the largest number of duplicates.

Table 5.7: Top 5 results for *Diplonema papillatum* data.

P-Value	**Freq.**	**Pattern**
1.73E-06	12	0|0|0|0|-|0|1|2|2|1|3|-|inter|intra|intra|inter|
7.76E-06	20	0|0|0|0|-|0|1|2|3|2|0|1|3|-|inter|inter|intra|intra|
1.77E-05	9	0|0|0|0|-|0|1|2|1|3|3|-|inter|intra|inter|intra|
1.77E-05	9	0|0|0|-|0|1|2|0|1|2|-|intra|intra|intra|
5.81E-05	23	0|0|0|0|-|0|1|2|3|1|2|3|0|-|inter|inter|intra|intra|

5.3 *Diplonema papillatum* Data

Some preliminary experiments were run on a third dataset with the mitochondrial genome from *Diplonema papillatum*. In this case the mechanism in unknown so precision and recall cannot be calculated. The hypothesized mechanism involves a RNA guide joining two mRNA modules by binding to 6nt regions at the ends of each module. Since the hypothesized anchors are of 6nt, the minimum complementary length was set to this value with a low 10% GU maximum to ensure only good anchors are created. The maximum number of molecules was increased to 3, since 3 are involved in the hypothetical mechanism. Normalization was set to 4 bins, and p-value and support were set to 0.1 and 1 respectively. The value of K was initially set to 3 to return the simplest possible structure, but the resulting set was empty. Increasing the K value to 4 returned many more results.

The top 5 results of this experiment are presented in Table 5.7. As can be seen in the pattern column of the table, 4 of the 5 patterns have two intermolecular nodes so they have potential to be a ppRNA. However, the patterns with two intermolecular nodes adjacent each other are always on the same module, which leaves patterns 1 and 3 as potential ppRNA. The first pattern has embeddings that include anchors on module 3/4 as well as anchors on module 7/8. The embeddings of the third pattern also cover module 3/4 and additionally module 4/5 and module 6/7.

From the third pattern, a hypothetical ppRNA can be constructed (Figure 5.13). This structure has two stems, one followed by the other, and two anchors near the first stem. The first anchor is just upstream of the stem, while the other one is within the stem.

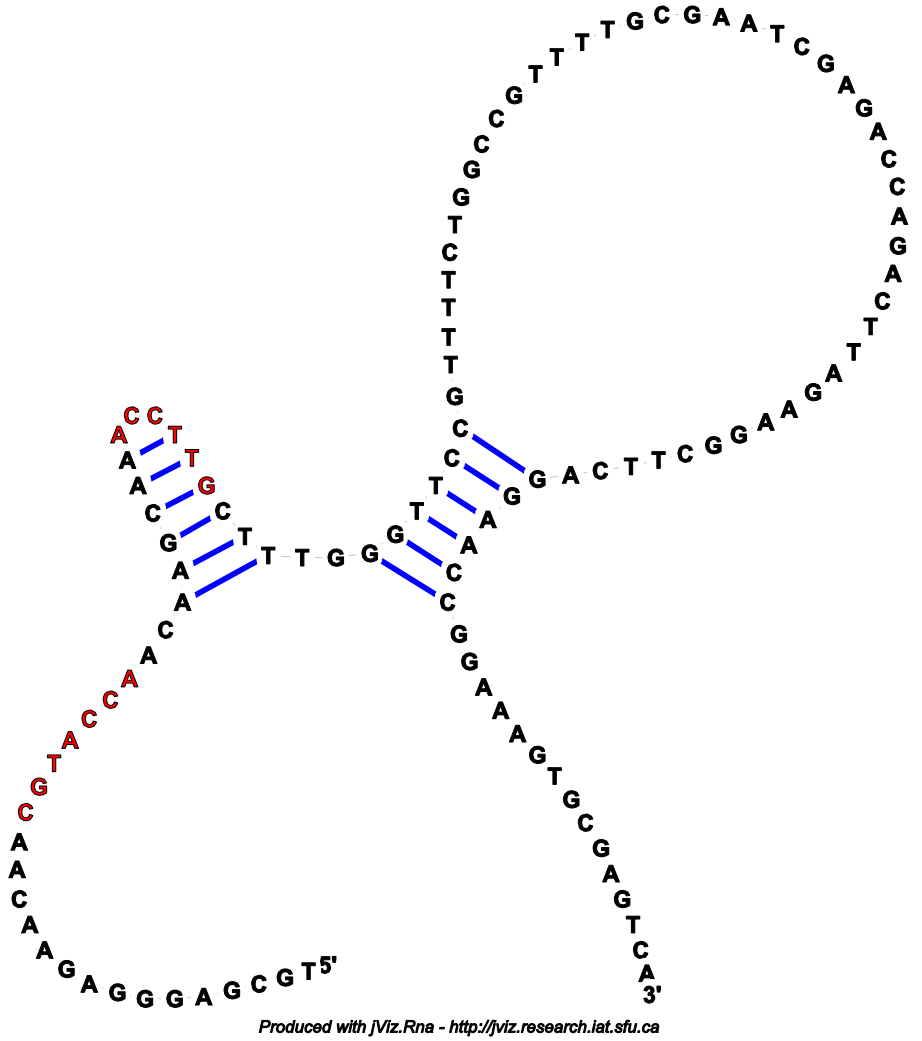

Figure 5.13: Predicted ppRNA based on the second pattern of the result set. The red regions mark the anchor which bind to module 4 and module 5.

The double stem-loop with an anchor within the stem is similar to the gRNA structure. However the locations of the anchors on the modules are not near the end, so this may make this structure not viable.

Chapter 6

Conclusion

We presented a novel framework for the discovery of RNA elements. It is an adaptation of frequent subgraph mining techniques on a large directed dual graph, representing all possible complementary sequences. This is the first instance of dual graphs being used in this way and the first application of frequent subgraph mining for this problem. Furthermore the dual graph representation was extended to not only represent secondary structures but also interactions between RNA.

It was applied to a synthetic and real-world data set, the *Trypanosoma brucei* mitochondrial genome, containing known gRNA and their target mRNA. The results are promising with the application being able to find patterns representing gRNA structures with high precision and recall. Furthermore the patterns linked these structures to their target mRNA. The algorithm was also applied to a *Diplonema papillatum* data set where the mechanism is unknown. RiboFSM was able to find patterns that could hypothetically guide editing in the organism. These results indicate that the algorithm will be able to predict novel gRNA and could later be applied to other RNA-related applications.

6.1 Limitations

One limitation of RiboFSM is scalability in terms of graph building. Since the graphs being produced contain every possible complementary sequence, as the sequence length increases the number of nodes increases dramatically. Although the execution time remains reasonable the memory usage becomes very high, beyond the capabilities of an average desktop computer. This can be mitigated by increasing the minimum length of complementary regions, but this reduces recall and may yield the algorithm ineffective for finding structures with short stems. However the frequent subgraph mining component scales well as the graph size increases and is capable of running billions of iterations if necessary.

Another limitation is that the quality of the results is very sensitive to the parameters chosen. The best parameters are very specific to the input sequences used and the nature of the structures within them, which is often unknown. It would be difficult to select default parameters that could consistently provide good results. The best that can be done is to provide guidelines as to what may be a good result.

One such guideline would be that the observed graph should be fairly similar in size to the background graph. Similarly sized graphs produce a distribution of p-values which clearly separates a small set of patterns with good p-values. In this case p-values are indicators of good results, as seen in the various runs with high precision and recall. Alternatively, when the sizes are too different, p-values cannot be used to reliably determine good patterns, even if the p-values are much better. This property is affected largely by the minimum complementary length and the maximum GU%. Of course if the data set contains nothing of biological interest, all parameters chosen will produce patterns with insignificant p-values. Some parameters, such as number of molecules and K, cannot be predicted using an unknown data set. A prior knowledge is needed or the user would need to experiment to find the best result.

The use of background data for the determination of p-value has drawbacks. Depending on the parameters used, the graph produced from background data will be either too similar or to different from the graph produced from the real data. The consequence of this is that the p-values will either all be very poor, in the case when the data is similar, or all be very good, in the case when they are very different. This in turn hurts precision

since deciding on a cut off for p-value has no effect. This could be resolved by using a more complex method for generating background data.

The final structures and interactions produced are fairly general. The stems are produced by finding complementary regions and do not consider free energy. RiboFSM is not meant to be used as a secondary structure prediction tool and does not guarantee that the produced structures represent the actual folding of the RNA. The results should be refined using more sophisticated method for structural prediction before making claims that it is the true structure. A possible extension of the algorithm would be to incorporate the produced patterns into secondary structure prediction by using them as a skeleton.

6.2 Future Work

The algorithm can be improved or extended in a variety of ways. Currently the graph building component is the bottleneck in terms of runtime due to the naive approach used. Sophisticated methods exist for finding all complementary sequences which run more efficiently than the naive algorithm. One such method is the use of a generalized suffix tree or array as used in SEED [1]. This would make it possible to find all complimentary regions in linear time, improving on the current $O(n^2k)$ runtime. Furthermore, more heuristics could be developed to produce nodes which are most likely to be part of a real structure or mechanism using biological knowledge. For example, masking of low complexity regions.

Another way the algorithm may be extended is by investigating other sampling approaches. Many such approaches exist and have been used in other applications. One example would be Random Walks, where a root node is chosen and then the subgraph is extended by randomly choosing an edge. Probabilistic models could also be integrated at this point, where the probability of adding a node to the subgraph depends on properties of the node or how the node would influence the structure. This may be based on free energy or other properties of the complementary regions.

The labeling approach used can also be revisited. Many different combinations of labels containing a variety of information could be used. Currently the nodes are only labeled based on graphical properties. However labels can contain biological informa-

tion as well, such as nucleotide composition or free energy. This process could also be more sophisticated, with labeling based on more than one node. This would be the case if a certain protein binding motif was known, which are very important since proteins are often involved with RNA mechanisms. The motif would always produce a specific subgraph, so if this subgraph was detected it could be labeled based on this information. This would allow for the creation of complex structures and systems. Also the annotations would be useful for interpreting the results.

The final area that can be explored is the calculation of statistical significance. Currently the distributions of ratios is assumed to be normal even though it is not quite normal. Different distributions can be explore to see if they are a better fit for the data, such as an extreme value distribution. The entire approach is fairly simple and could be replaced with a more complex statistical model. It would be especially effective if a model could be created that would not rely on background data at all. This would greatly reduce the runtime and memory usage of the algorithm. This could be done by predicting the properties of a randomized graph rather than producing the graph itself.

Besides extensions to the algorithm, improvements can be made to usability. Most popular bioinformatics tools have a web interface or a standalone graphical user interface (GUI) for use by researchers. Currently RiboFSM is a command line tool. Creating a web interface would allow for many more people to use the algorithm. It would also allow for user to easily change parameters. In addition, the server used to run RiboFSM can have the computational power and memory to run any input. Visualization of the results would also be an improvement. The algorithm produces only flat files with text results. It would be much easier to analyze the results if the user could see a graphical representation or image of a folded structure.

Appendix A

Primer on RNA Secondary Structure

Coding sequences of DNA are evolutionarily conserved because of the correspondence between codons and the amino acids they encode. The sequence must be maintained to create viable proteins. Non-coding RNA differs from coding DNA because the structure is tolerant of changes in sequence as long as paired (or unpaired) regions persist. For this reason secondary structure of RNA is very important. The most basic secondary structure is the **stem**, or stacked pair. This structure is formed when two RNA sequences have complementary bases, A/U, G/C as well as the less favorable G/U, and the bases are in reverse order. They pair together to form a duplex similar to double stranded DNA.

If one of the two strands had a series of nucleotides that do not complement the other strand, they will form a single stranded region. This region will protrude from one side of the stem and is called a **bulge**. If the other strand also had a series of nucleotides that are not complementing the other strand within the same region, a single stranded protrusion will form on that strand as well. The two opposing single stranded regions within the stem are called an **interior loop**.

Coming back to the stem, if you joined the two strands at either end a loop would be formed. This structure is called a **stem-loop**. If you had a series of stem-loops made

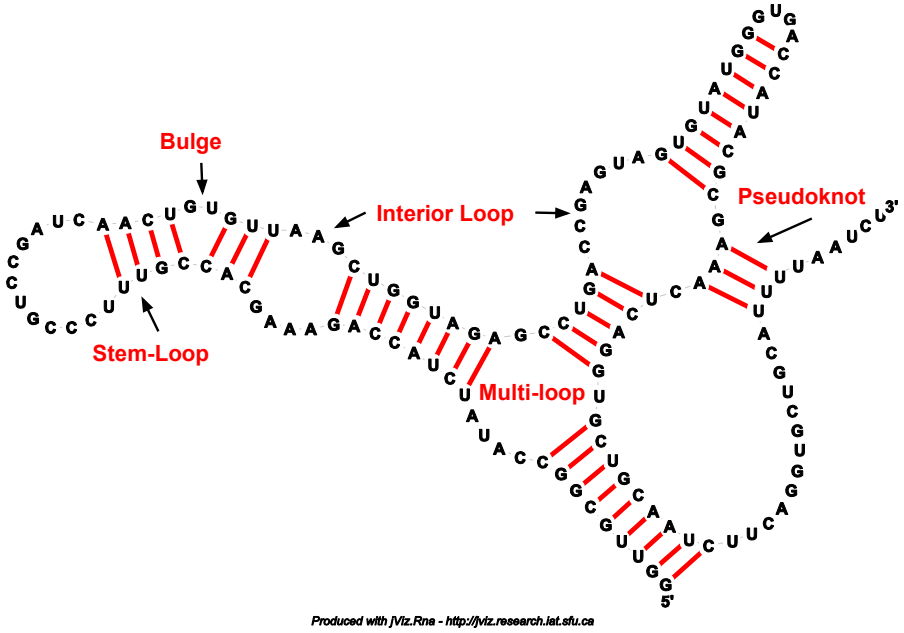

Figure A.1: Example RNA secondary structure with all basic RNA substructures.

of one strand and then you joined the ends of this strand, these loops would become nested in the larger loop. This structure is called a **multi-loop**. The final type of structure is called a **pseudoknot**. This is a special case where a strand forms a stem within another loop created by the same strand. These structures are often ignored because they increase the complexity of structure prediction. All of these structures are illustrated in Figure A.1.

Appendix B

Ambiguities in Dual Graphs

There are circumstances where directionality is very important. For example, the two secondary structures in Figure B.1 are quite different. If you convert these structures into dual graphs, the two dual graphs are the same. This occurs since switching the direction of a strand in a complementary region is a negligible change in an undirected dual graph, but is a major change for secondary structure. Adding directionality easily solves this problem as can be seen in the the two different directed dual graphs.

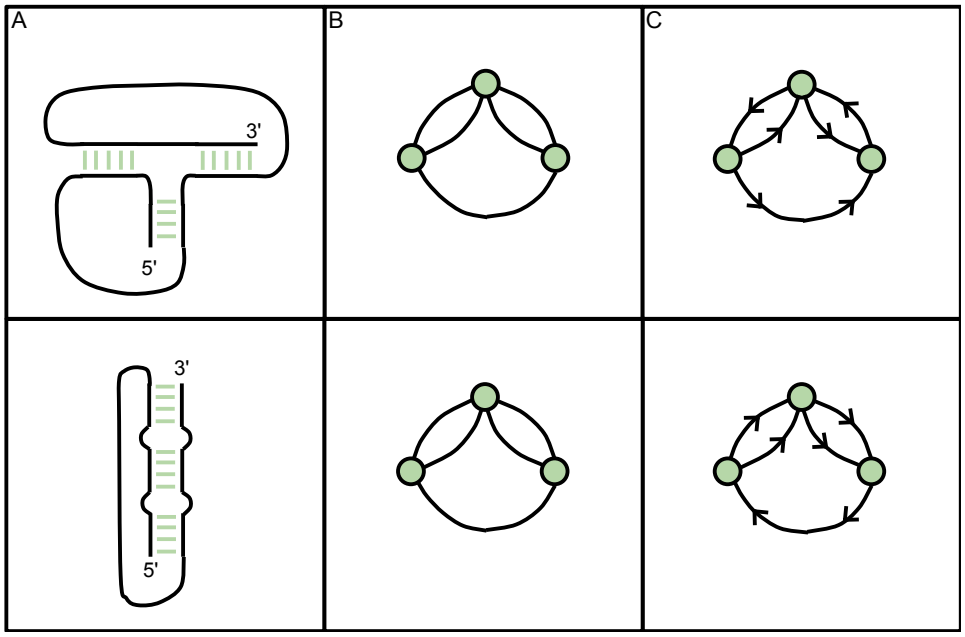

Figure B.1: Example RNA secondary structures (A) with equal dual graph (B) and different directed dual graph (C) representations.

Appendix C

Raw Data

Only 3 replicates for each set of parameters are shown for brevity. Column descriptions are shown in Table C.1.

Table C.1: Descriptions of columns used in raw data tables.

Column Name	Description
<Param Name >	The first column is the parameter being investigated, or run number
Obs.	Observed graph size, in terms of number of nodes
Back.	Background graph size, in terms of number of nodes
P	Positive set (TP + FN), number of embeddings
N	Negative set (FP + TN), number of embeddings
TP	True positives, number of embeddings
FP	False positives, number of embeddings
TN	True negatives,, number of embeddings
FN	False negatives, number of embeddings
MinP	Minimum p-value of all patterns in the positive set
MaxP	Maximum p-value of all patterns in the positive set
AvgP	Average p-value of all patterns in the positive set
Pat.	Number of patterns in the positive set
Total	Total number of patterns produced

Table C.2: Synthetic minimum complementary region length results.

NT	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
6	534996	527006	0	485325	0	8504	476821	0	0	0.086657	0.015042	27	151
6	534996	533354	0	485353	0	8554	476799	0	0	0.064356	0.024472	26	150
6	534996	530252	0	485314	0	4459	480855	0	0	0.098686	0.046477	16	149
7	157426	148826	27	120780	0	355	120425	27	0	0.00719	0.002758	12	176
7	157426	145774	29	120778	0	648	120130	29	0	0.00859	0.002181	12	176
7	157426	144566	27	120780	4	315	120465	23	0	0.002849	6.68E-04	12	176
8	48022	44204	174	25503	76	599	24904	98	0	8.50E-04	2.85E-04	8	169
8	48022	44784	172	25505	76	501	25004	96	0	3.65E-04	1.36E-04	6	169
8	48022	42644	172	25505	76	416	25089	96	0	1.40E-04	3.86E-05	10	169
9	17298	13404	455	8035	415	1148	6887	40	0	9.20E-04	3.08E-04	13	187
9	17298	12270	454	8036	416	1334	6702	38	0	8.70E-04	2.09E-04	13	187
9	17298	12610	455	8035	416	1168	6867	39	0	7.83E-04	1.76E-04	14	187
10	8442	3510	666	3623	630	997	2626	36	0	1.99E-04	4.87E-05	7	124
10	8442	3680	666	3623	630	997	2626	36	0	7.97E-04	2.46E-04	7	124
10	8442	3310	663	3626	629	1266	2360	34	0	7.47E-04	1.49E-04	16	124

Table C.3: Synthetic number of iterations results.

Iter.	Obs.	Back.	P	N	TP	FP	TN	FN	MinP	MaxP	AvgP	Pat.	Total
0.1	48022	43842	216	28296	0	0	28296	216	0	0	0	0	182
0.1	48022	44038	208	28376	0	0	28376	208	0	0	0	0	181
0.1	48022	43542	216	28210	0	0	28210	216	0	0	0	0	182
0.1	48022	44220	215	28355	0	0	28355	215	0	0	0	0	175
0.1	48022	42842	205	28493	0	0	28493	205	0	0	0	0	176
0.1	48022	44276	227	28448	0	0	28448	227	0	0	0	0	175
0.1	48022	41404	219	28423	0	0	28423	219	0	0	0	0	181
0.1	48022	43902	231	28347	0	0	28347	231	0	0	0	0	174
0.1	48022	43288	216	28332	0	0	28332	216	0	0	0	0	175
0.1	48022	43594	209	28443	0	0	28443	209	0	0	0	0	175
0.5	48022	42770	294	36434	105	260	36174	189	7.21E-08	6.78E-04	1.73E-04	8	193
0.5	48022	39614	294	36409	0	0	36409	294	0	0	0	0	193
0.5	48022	46222	294	36419	105	255	36164	189	1.09E-07	3.81E-04	1.13E-04	8	193
0.5	48022	44070	294	36410	96	168	36242	198	1.01E-08	3.24E-04	7.59E-05	6	193
0.5	48022	44148	291	36412	0	0	36412	291	0	0	0	0	193
0.5	48022	41212	292	36396	0	0	36396	292	0	0	0	0	193
0.5	48022	43034	291	36411	0	0	36411	291	0	0	0	0	193
0.5	48022	43164	293	36409	104	307	36102	189	1.05E-09	7.31E-04	1.87E-04	10	193
0.5	48022	42172	291	36426	0	0	36426	291	0	0	0	0	193
0.5	48022	43114	289	36389	94	251	36138	195	9.90E-10	8.02E-04	1.86E-04	8	193
1	48022	44902	294	36523	96	219	36304	198	3.68E-07	7.78E-04	1.70E-04	6	193
1	48022	43250	294	36523	105	256	36267	189	1.61E-08	8.61E-04	2.02E-04	8	193
1	48022	43812	294	36522	105	229	36293	189	4.33E-08	4.35E-04	1.54E-04	7	193
1	48022	41966	294	36523	105	325	36198	189	9.98E-09	8.54E-04	1.96E-04	10	193
1	48022	43666	294	36523	105	255	36268	189	1.49E-08	9.56E-04	2.08E-04	8	193
1	48022	43414	294	36523	90	209	36314	204	6.87E-07	5.46E-04	1.78E-04	5	193
1	48022	42432	294	36523	99	272	36251	195	1.81E-06	9.43E-04	2.64E-04	8	193
1	48022	41446	294	36523	92	260	36263	202	1.01E-07	8.91E-04	3.29E-04	7	193

Continued on the next page

Iter.	Obs.	Back.	P	N	TP	FP	TN	FN	MinP	MaxP	AvgP	Pat.	Total
1	48022	44476	294	36522	96	245	36277	198	4.87E-09	6.35E-04	1.33E-04	7	193
1	48022	44696	294	36522	105	229	36293	189	2.95E-07	7.30E-04	1.85E-04	7	193
2.5	48022	44236	294	36523	92	257	36266	202	1.19E-08	3.27E-04	5.70E-05	7	193
2.5	48022	43682	294	36523	105	330	36193	189	1.17E-09	2.82E-04	6.17E-05	10	193
2.5	48022	42836	294	36523	101	252	36271	193	2.02E-07	5.88E-04	1.56E-04	7	193
2.5	48022	43106	294	36523	105	262	36261	189	1.77E-07	6.17E-04	2.62E-04	8	193
2.5	48022	42380	294	36523	92	201	36322	202	1.29E-07	4.95E-04	1.01E-04	5	193
2.5	48022	42584	294	36523	96	446	36077	198	2.20E-08	6.05E-04	1.35E-04	8	193
2.5	48022	45964	294	36523	86	250	36273	208	1.98E-06	7.52E-04	3.11E-04	7	193
2.5	48022	41340	294	36523	105	278	36245	189	8.87E-10	3.98E-04	7.08E-05	9	193
2.5	48022	42450	294	36523	105	252	36271	189	1.94E-07	6.74E-04	2.34E-04	8	193
2.5	48022	41494	294	36523	105	256	36267	189	5.97E-08	7.51E-04	2.31E-04	8	193
5	48022	44364	294	36523	86	224	36299	208	1.20E-06	1.69E-04	5.91E-05	5	193
5	48022	43212	294	36523	105	274	36249	189	1.89E-10	5.76E-04	1.55E-04	9	193
5	48022	44036	294	36523	96	219	36304	198	5.75E-09	2.66E-04	5.41E-05	6	193
5	48022	42780	294	36523	101	334	36189	193	2.70E-10	3.69E-04	7.44E-05	10	193
5	48022	41686	294	36523	96	305	36218	198	5.31E-08	7.09E-04	1.21E-04	9	193
5	48022	41774	294	36523	105	315	36208	189	1.13E-09	8.53E-04	1.41E-04	10	193
5	48022	43812	294	36523	96	252	36271	198	1.89E-07	5.84E-04	1.79E-04	7	193
5	48022	42422	294	36523	105	262	36261	189	6.54E-06	9.64E-04	3.94E-04	8	193
5	48022	42716	294	36523	105	290	36233	189	2.15E-09	8.65E-04	2.46E-04	10	193
5	48022	44350	294	36523	101	220	36303	193	1.76E-08	4.09E-04	1.34E-04	8	193
10	48022	41754	294	36523	105	256	36267	189	2.57E-08	9.94E-04	2.29E-04	8	193
10	48022	41506	294	36523	105	278	36245	189	1.04E-08	3.50E-04	1.10E-04	9	193
10	48022	44918	294	36523	105	303	36220	189	3.74E-09	7.93E-04	1.85E-04	10	193
10	48022	43836	294	36523	96	216	36307	198	6.03E-09	9.60E-04	2.77E-04	8	193
10	48022	44630	294	36523	96	219	36304	198	8.55E-09	3.38E-04	9.74E-05	6	193
10	48022	45460	294	36523	96	278	36245	198	9.31E-10	7.80E-04	1.28E-04	8	193
10	48022	42490	294	36523	96	246	36277	198	8.25E-09	6.53E-04	1.47E-04	7	193
10	48022	43032	294	36523	105	256	36267	189	9.55E-08	7.85E-04	1.88E-04	8	193
10	48022	42814	294	36523	105	306	36217	189	1.24E-08	4.08E-04	1.38E-04	10	193

Continued on the next page

Iter.	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
10	48022	42888	294	36523	105	255	36268	189	8.04E-09	7.67E-04	2.13E-04	8	193
50	48022	43704	294	36523	105	521	36002	189	1.97E-09	9.86E-04	1.58E-04	12	193
50	48022	42362	294	36523	101	211	36312	193	2.39E-08	4.97E-04	1.24E-04	6	193
50	48022	43020	294	36523	92	230	36293	202	1.42E-08	4.15E-04	7.49E-05	6	193
50	48022	44788	294	36523	105	241	36282	189	3.32E-09	8.43E-04	1.51E-04	8	193
50	48022	45422	294	36523	105	335	36188	189	2.15E-10	7.47E-05	1.24E-05	11	193
50	48022	45056	294	36523	101	278	36245	193	8.08E-09	8.15E-04	2.47E-04	8	193
50	48022	42042	294	36523	105	415	36108	189	4.88E-09	8.95E-04	2.77E-04	8	193
50	48022	42464	294	36523	105	155	36368	189	3.51E-09	2.31E-04	6.30E-05	6	193
50	48022	43708	294	36523	96	331	36192	198	1.04E-08	7.07E-04	2.46E-04	10	193
50	48022	42232	294	36523	101	315	36208	193	9.47E-09	5.45E-04	1.94E-04	10	193

Table C.4: Synthetic length normalization results.

Bins	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
1	48022	43880	166	25403	0	0	25403	166	0	0.00E+00	NaN	0	61
3	48022	46704	173	25504	43	230	25274	130	0	9.30E-06	2.25E-06	5	169
3	48022	42850	171	25506	76	202	25304	95	0	9.26E-06	2.71E-06	5	169
3	48022	42216	172	25505	76	79	25426	96	0	1.31E-06	6.99E-07	3	169
4	48022	43136	175	25576	57	148	25428	118	0	2.29E-06	8.01E-07	5	243
4	48022	44832	175	25576	46	150	25426	129	0	9.87E-06	2.47E-06	4	243
4	48022	43316	175	25576	46	112	25464	129	0	8.10E-06	2.78E-06	4	243
5	48022	41992	182	25681	58	1086	24595	124	0	6.12E-06	1.57E-06	12	355
5	48022	42364	180	25683	40	322	25361	140	0	2.95E-06	1.40E-06	5	355
5	48022	42344	181	25682	17	385	25297	164	0	5.27E-06	1.41E-06	6	355
6	48022	43810	190	25758	24	581	25177	166	0	7.79E-06	2.96E-06	16	440
6	48022	42228	191	25757	20	396	25361	171	0	8.44E-06	3.18E-06	9	440
6	48022	41412	188	25760	32	237	25523	156	0	5.44E-06	1.85E-06	8	440
7	48022	43044	186	25921	114	623	25298	72	0	5.11E-06	1.02E-06	15	599
7	48022	44636	187	25920	107	2583	23337	80	0	9.77E-06	1.65E-06	27	599
7	48022	44506	186	25921	106	1327	24594	80	0	5.92E-06	1.17E-06	16	599
8	48022	45376	193	25997	82	2359	23638	111	0	8.81E-06	1.53E-06	30	682
8	48022	43302	192	25998	100	1548	24450	92	0	7.52E-06	1.36E-06	19	682
8	48022	42066	193	25997	109	1981	24016	84	0	7.43E-06	1.10E-06	39	682
9	48022	43258	201	26147	71	1641	24506	130	0	8.33E-06	2.09E-06	37	840
9	48022	45286	202	26146	72	1618	24528	130	0	8.49E-06	2.34E-06	36	840
9	48022	42696	201	26147	71	1156	24991	130	0	5.78E-06	1.55E-06	19	840
10	48022	40828	204	26206	81	3593	22613	123	0	5.27E-06	6.42E-07	76	902
10	48022	42398	204	26206	50	1926	24280	154	0	7.23E-06	1.28E-06	41	902
10	48022	45974	204	26206	55	1604	24602	149	0	8.20E-06	1.49E-06	41	902

Table C.5: Synthetic length normalization results (top 1000).

Bins	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
1	48022	43848	166	25403	166	847	24556	0	0	0.429147	0.181698	22	61
1	48022	43082	167	25402	167	895	24507	0	0	0.399402	0.107331	24	61
1	48022	43252	166	25403	166	887	24516	0	0	0.441247	0.117221	22	61
2	48022	46138	168	25436	168	877	24559	0	0	0.280245	0.083786	37	96
2	48022	43096	167	25437	167	856	24581	0	0	0.241187	0.098782	33	96
2	48022	45210	168	25436	168	895	24541	0	0	0.174693	0.062454	25	96
3	48022	44954	173	25504	160	842	24662	13	0	0.031335	0.007445	22	169
3	48022	44492	174	25503	166	839	24664	8	0	0.014888	0.002562	20	169
3	48022	44490	172	25505	160	844	24661	12	0	0.006045	0.001951	18	169
4	48022	42340	175	25576	95	919	24657	80	0	0.001846	3.69E-04	21	243
4	48022	43644	175	25576	84	938	24638	91	0	0.002674	9.18E-04	19	243
4	48022	42190	175	25576	72	935	24641	103	0	0.002042	7.83E-04	15	243
5	48022	42274	183	25680	62	948	24732	121	0	1.40E-06	3.52E-07	12	355
5	48022	42126	183	25680	84	936	24744	99	0	5.85E-05	1.62E-05	12	355
5	48022	44152	182	25681	124	899	24782	58	0	0.002288	3.47E-04	30	355
6	48022	43168	189	25759	13	995	24764	176	0	6.84E-05	2.40E-05	20	440
6	48022	42598	188	25760	56	977	24783	132	0	1.04E-05	2.10E-06	12	440
6	48022	42912	191	25757	13	993	24764	178	0	1.12E-05	4.18E-06	17	440
7	48022	41902	184	25923	52	949	24974	132	0	1.43E-08	5.01E-09	6	599
7	48022	43418	186	25921	53	1021	24900	133	0	2.48E-09	5.23E-10	7	599
7	48022	43640	185	25922	95	906	25016	90	0	8.48E-09	1.36E-09	8	599
8	48022	43542	193	25997	82	933	25064	111	0	1.55E-07	3.49E-08	10	682
8	48022	44194	190	26000	52	1000	25000	138	0	1.02E-11	3.58E-12	5	682
8	48022	44344	193	25997	53	955	25042	140	0	6.23E-10	1.91E-10	6	682
9	48022	42004	199	26149	17	986	25163	182	0	3.70E-09	1.13E-09	13	840
9	48022	39840	200	26148	29	978	25170	171	0	5.79E-07	9.19E-08	19	840
9	48022	42148	200	26148	52	1027	25121	148	0	3.95E-07	8.76E-08	14	840

Continued on the next page

Bins	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
10	48022	43318	203	26207	0	1009	25198	203	0	7.13E-08	1.50E-08	14	902
10	48022	43430	204	26206	0	1076	25130	204	0	6.01E-07	1.66E-07	16	902
10	48022	43986	203	26207	0	1038	25169	203	0	0	0	5	902

Table C.6: Synthetic P-value results.

PVal	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
0.1	48022	42186	173	25504	160	2748	22756	13	0	0.069049	0.015269	31	169
0.1	48022	43550	174	25503	162	2969	22534	12	0	0.098585	0.024147	32	169
0.1	48022	42814	174	25503	162	2404	23099	12	0	0.041887	0.007566	20	169
0.01	48022	45358	173	25504	156	757	24747	17	0	0.006536	0.002198	18	169
0.01	48022	42876	171	25506	159	1575	23931	12	0	0.007066	0.002561	32	169
0.01	48022	42896	171	25506	155	935	24571	16	0	0.005441	0.001185	21	169
0.001	48022	41086	172	25505	76	709	24796	96	0	9.74E-04	2.42E-04	12	169
0.001	48022	43990	174	25503	76	604	24899	98	0	8.61E-04	3.06E-04	10	169
0.001	48022	42062	171	25506	76	661	24845	95	0	9.62E-04	2.07E-04	9	169
1.00E-04	48022	45362	171	25506	76	242	25264	95	0	1.18E-05	3.69E-06	5	169
1.00E-04	48022	42994	173	25504	76	517	24987	97	0	6.16E-05	1.21E-05	8	169
1.00E-04	48022	44090	173	25504	76	241	25263	97	0	4.65E-05	6.85E-06	8	169
1.00E-05	48022	44440	173	25504	76	109	25395	97	0	3.82E-07	1.45E-07	4	169
1.00E-05	48022	43512	171	25506	33	133	25373	138	0	6.88E-06	4.55E-06	3	169
1.00E-05	48022	44386	172	25505	76	130	25375	96	0	9.53E-06	2.21E-06	5	169
1.00E-06	48022	43662	171	25506	76	79	25427	95	0	6.65E-07	2.32E-07	3	169
1.00E-06	48022	41866	173	25504	43	79	25425	130	0	7.79E-08	4.12E-08	2	169
1.00E-06	48022	45154	171	25506	43	79	25427	128	0	6.22E-07	3.39E-07	2	169
1.00E-07	48022	45058	172	25505	76	79	25426	96	0	1.13E-08	3.83E-09	3	169
1.00E-07	48022	43674	173	25504	43	79	25425	130	0	2.51E-08	1.33E-08	2	169
1.00E-07	48022	42430	173	25504	0	0	25504	173	0	0	NaN	0	169

Table C.7: Synthetic minimum support results.

Sup	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
1	48022	44126	172	25505	153	793	24712	19	0	0.005208	0.001262	12	169
1	48022	41752	174	25503	81	652	24851	93	0	0.008392	0.002416	14	169
1	48022	42712	172	25505	152	823	24682	20	0	0.008638	0.002031	12	169
5	48022	41604	172	25505	160	756	24749	12	0	0.008693	0.002825	20	169
5	48022	45176	172	25505	159	930	24575	13	0	0.009651	0.003473	21	169
5	48022	42102	173	25504	160	831	24673	13	0	0.006208	0.001444	19	169
10	48022	44662	173	25504	153	845	24659	20	0	0.007065	0.001029	12	169
10	48022	41112	172	25505	154	879	24626	18	0	0.005931	0.00131	14	169
10	48022	44650	173	25504	153	859	24645	20	0	0.00685	0.001073	13	169
20	48022	45416	173	25504	154	762	24742	19	0	0.007163	0.001095	10	169
20	48022	41414	172	25505	153	763	24742	19	0	0.006441	8.59E-04	10	169
20	48022	43822	174	25503	153	856	24647	21	0	0.00656	0.001361	12	169
50	48022	44248	172	25505	78	565	24940	94	0	0.008452	0.002845	4	169
50	48022	46234	171	25506	77	635	24871	94	0	0.008231	0.001822	5	169
50	48022	42688	172	25505	77	737	24768	95	0	0.004363	0.001455	6	169
100	48022	43388	173	25504	78	534	24970	95	0	0.00178	0.001026	3	169
100	48022	42528	173	25504	77	433	25071	96	0	0.00273	0.001562	2	169
100	48022	42128	174	25503	77	535	24968	97	0	0.009907	0.004312	3	169

Table C.8: Synthetic best parameter results.

Run	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
1	17298	11716	449	7953	281	291	7662	168	0	1.90E-08	9.60E-09	2	99
1	17298	12222	450	7952	281	291	7661	169	0	9.32E-06	5.49E-06	2	99
1	17298	12588	449	7953	281	626	7327	168	0	8.54E-08	2.85E-08	3	99
2	8442	3466	659	3581	0	0	3581	659	0	0	NaN	0	75
2	8442	3372	659	3581	523	470	3111	136	0	5.84E-07	4.49E-07	2	75
2	8442	3594	659	3581	0	0	3581	659	0	0	NaN	0	75
3	17298	12500	450	7952	281	10	7942	169	0	9.25E-06	9.25E-06	1	99
3	17298	12820	450	7952	423	460	7492	27	0	3.55E-06	1.37E-06	5	99
3	17298	12556	450	7952	423	739	7213	27	0	6.43E-06	2.24E-06	4	99
4	8442	3746	659	3581	523	470	3111	136	0	1.43E-07	1.08E-07	2	75
4	8442	3684	659	3581	523	470	3111	136	0	1.10E-07	8.30E-08	2	75
4	8442	3730	659	3581	523	633	2948	136	0	1.62E-06	5.45E-07	3	75
5	17298	12388	495	8429	253	1031	7398	242	0	8.39E-06	1.38E-06	29	621
5	17298	12118	492	8432	300	1115	7317	192	0	3.52E-06	7.37E-07	39	621
5	17298	12010	498	8426	329	1023	7403	169	0	7.57E-06	7.99E-07	40	621
6	8442	3504	696	3869	562	1573	2296	134	0	8.48E-06	1.42E-06	40	400
6	8442	3346	698	3867	624	2141	1726	74	0	8.86E-06	1.32E-06	73	400
6	8442	3618	698	3867	606	2142	1725	92	0	7.19E-06	6.93E-07	64	400
7	17298	12008	496	8428	301	1472	6956	195	0	6.66E-06	8.34E-07	47	621
7	17298	12594	494	8430	344	1166	7264	150	0	8.44E-06	6.56E-07	41	621
7	17298	12780	495	8429	329	1476	6953	166	0	7.73E-06	1.04E-06	51	621
8	8442	3656	701	3864	598	1937	1927	103	0	9.29E-06	1.62E-06	49	400
8	8442	3262	698	3867	563	1388	2479	135	0	9.08E-06	1.66E-06	33	400
8	8442	3398	697	3868	605	2261	1607	92	0	6.13E-06	7.64E-07	63	400

Table C.9: *Trypanosoma brucei* minimum complementary length results.

NT	Obs.	Back.	P	N	TP	FP	TN	FN	MinP	MaxP	AvgP	Pat.	Total
8	3552418	3386332	1936	2778352	0	335	2778017	1936	0	0.007892	0.003838	5	356
8	3552418	3332132	1942	2778532	39	605	2777927	1903	0	0.008892	0.003633	7	353
8	3552418	3412850	1941	2778491	117	700	2777791	1824	0	0.007138	0.00426	8	355
9	1407254	1291090	1104	987523	121	428	987095	983	0	1.69E-04	8.52E-05	4	359
9	1407254	1320280	1099	987516	171	397	987119	928	0	4.59E-05	2.25E-05	3	359
9	1407254	1284298	1106	987544	246	5461	982083	860	0	7.93E-04	3.07E-04	13	359
11	280918	243296	567	122122	103	3216	118906	464	0	5.10E-05	1.20E-05	14	320
11	280918	232800	567	122122	155	1108	121014	412	0	5.67E-05	1.92E-05	7	320
11	280918	229032	569	122120	53	3526	118594	516	0	8.65E-05	3.74E-05	7	320
19	2702	598	159	394	119	338	56	40	0	9.68E-05	1.89E-05	27	68
19	2702	514	163	390	65	235	155	98	0	4.86E-05	1.38E-05	6	68
19	2702	478	163	390	110	327	63	53	0	7.44E-05	1.20E-05	22	68
10	581160	512820	671	325598	109	2021	323577	562	0	7.76E-04	1.16E-04	8	331
10	581160	536550	661	325608	109	263	325345	552	0	7.45E-04	1.91E-04	6	331
10	581160	527104	670	325599	110	315	325284	560	0	7.42E-04	1.39E-04	6	331
11	280918	236482	564	122125	173	5243	116882	391	0	2.47E-04	9.18E-05	13	320
11	280918	236548	572	122117	160	5207	116910	412	0	6.76E-04	1.00E-04	14	320
11	280918	241756	571	122118	190	5932	116186	381	0	6.39E-04	1.73E-04	21	320
11	280918	237074	570	122119	102	124	121995	468	0	1.39E-05	3.93E-06	4	320
11	280918	235602	566	122123	102	3557	118566	464	0	2.76E-05	4.82E-06	9	320
11	280918	238226	578	122111	155	3421	118690	423	0	4.49E-05	9.81E-06	8	320
12	173594	143200	505	61698	102	1141	60557	403	0	5.74E-05	1.47E-05	8	246
12	173594	148486	506	61697	46	52	61645	460	0	7.72E-07	5.65E-07	2	246
12	173594	149100	502	61701	102	1336	60365	400	0	6.84E-05	1.77E-05	9	246
14	34760	23148	403	9838	43	821	9017	360	0	3.80E-05	8.02E-06	9	214
14	34760	22650	391	9850	44	717	9133	347	0	8.79E-05	2.91E-05	8	214
14	34760	23178	395	9846	44	2078	7768	351	0	8.81E-05	2.54E-05	15	214
16	12432	6866	283	3429	95	2657	772	188	0	5.13E-05	6.00E-06	34	165

Continued on the next page

NT	Obs.	Back.	P	N	TP	FP	TN	FN	MinP	MaxP	AvgP	Pat.	Total
16	12432	7222	280	3432	57	764	2668	223	0	7.46E-05	1.61E-05	14	165
16	12432	6804	283	3429	61	1508	1921	222	0	8.07E-05	2.35E-05	11	165
18	4064	994	259	923	47	646	277	212	0	5.92E-05	1.29E-05	10	89
18	4064	1162	261	921	123	732	189	138	0	9.47E-05	1.88E-05	13	89
18	4064	1178	256	926	142	772	154	114	0	5.97E-05	9.91E-06	17	89
20	2038	354	134	261	88	201	60	46	0	4.01E-04	8.38E-05	16	54
20	2038	336	136	259	78	193	66	58	0	1.73E-04	4.07E-05	13	54
20	2038	326	136	259	95	204	55	41	0	5.91E-04	1.19E-04	17	54
25	1016	6	11	32	10	31	1	1	0	0	0	13	14
25	1016	12	11	32	10	31	1	1	0	0	0	13	14
25	1016	10	12	31	11	30	1	1	0	0	0	13	14

Table C.10: *Trypanosoma brucei* maximum GU% results.

GU%	Obs.	Back.	P	N	TP	FP	TN	FN	MinP	MaxP	AvgP	Pat.	Total
0	540	112	0	54	0	16	38	0	0	4.78E-11	2.39E-11	4	22
0	540	142	0	54	0	49	5	0	0	2.25E-06	1.13E-07	20	22
0	540	106	0	54	0	50	4	0	0	8.28E-07	4.14E-08	20	22
0.05	558	124	0	62	0	51	11	0	0	0	0	20	24
0.05	558	124	0	62	0	15	47	0	0	7.95E-07	2.98E-07	3	24
0.05	558	132	0	62	0	15	47	0	0	6.78E-06	2.71E-06	3	24
0.1	3074	1026	11	970	2	698	272	9	0	7.14E-06	1.95E-06	12	91
0.1	3074	880	10	971	2	698	273	8	0	4.99E-06	4.26E-07	12	91
0.1	3074	964	11	970	2	746	224	9	0	2.94E-06	4.67E-07	17	91
0.15	8256	3810	62	2910	1	1171	1739	61	0	3.93E-06	5.82E-07	17	188
0.15	8256	3138	61	2911	2	1508	1403	59	0	7.81E-06	7.61E-07	15	188
0.15	8256	3460	60	2912	3	1785	1127	57	0	9.80E-06	2.19E-06	25	188
0.2	15778	8910	213	4324	43	337	3987	170	0	1.12E-06	5.76E-07	5	178
0.2	15778	9090	213	4324	43	501	3823	170	0	1.40E-06	4.33E-07	8	178
0.2	15778	8674	216	4321	51	2641	1680	165	0	7.83E-06	1.61E-06	20	178
0.25	18766	11196	296	5165	52	375	4790	244	0	8.65E-06	3.01E-06	4	197
0.25	18766	11126	302	5159	78	3251	1908	224	0	5.81E-06	1.17E-06	28	197
0.25	18766	10300	299	5162	53	260	4902	246	0	8.38E-07	6.53E-07	2	197
0.3	28026	15172	382	7087	73	66	7021	309	0	5.60E-07	3.37E-07	2	183
0.3	28026	18084	383	7086	73	434	6652	310	0	4.38E-06	1.45E-06	7	183
0.3	28026	16138	382	7087	73	373	6714	309	0	9.16E-06	2.03E-06	6	183
0.35	39784	25516	460	9406	85	321	9085	375	0	6.79E-06	1.82E-06	4	201
0.35	39784	25410	460	9406	69	365	9041	391	0	3.96E-06	8.39E-07	5	201
0.35	39784	25024	458	9408	0	297	9111	458	0	4.86E-06	1.69E-06	3	201
0.4	50770	32182	593	11667	121	284	11383	472	0	5.11E-06	1.71E-06	7	202
0.4	50770	32906	594	11666	79	206	11460	515	0	4.84E-06	1.65E-06	3	202
0.4	50770	31732	592	11668	80	325	11343	512	0	9.91E-06	2.75E-06	5	202
0.45	53380	34532	608	12271	92	309	11962	516	0	7.20E-06	2.42E-06	7	216

Continued on the next page

GU%	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
0.45	53380	32296	610	12269	69	340	11929	541	0	9.48E-06	4.07E-06	5	216
0.45	53380	33344	609	12270	69	87	12183	540	0	2.38E-06	1.82E-06	2	216
0.5	60722	36652	510	13872	24	327	13545	486	0	2.32E-06	1.29E-06	3	216
0.5	60722	37022	513	13869	33	259	13610	480	0	4.91E-06	1.70E-06	4	216
0.5	60722	38230	514	13868	0	191	13677	514	0	1.29E-06	1.29E-06	1	216

Table C.11: *Trypanosoma brucei* maximum GU% results (top 1000).

GU%	Obs.	Back.	P	N	TP	FP	TN	FN	MinP	MaxP	AvgP	Pat.	Total
0	540	160	0	54	0	54	0	0	0	0.629837	0.058213	22	22
0	540	118	0	54	0	54	0	0	0	5.84E-04	5.31E-05	22	22
0	540	104	0	54	0	54	0	0	0	7.64E-04	6.95E-05	22	22
0.05	558	120	0	62	0	62	0	0	0	1	0.120061	24	24
0.05	558	154	0	62	0	62	0	0	0	1	0.083333	24	24
0.05	558	154	0	62	0	62	0	0	0	1	0.125	24	24
0.1	3074	1090	11	970	11	970	0	0	0	1	0.130778	91	91
0.1	3074	896	11	970	11	970	0	0	0	0.554122	0.128821	91	91
0.1	3074	946	11	970	11	970	0	0	0	1	0.141101	91	91
0.15	8256	3506	61	2911	0	1025	1886	61	0	8.57E-07	7.72E-08	14	188
0.15	8256	3750	61	2911	0	1070	1841	61	0	2.08E-05	1.51E-06	15	188
0.15	8256	3256	62	2910	0	1046	1864	62	0	9.75E-07	1.11E-07	14	188
0.2	15778	8888	218	4319	101	946	3373	117	0	5.11E-04	1.18E-04	14	178
0.2	15778	8674	219	4318	105	902	3416	114	0	4.71E-04	1.18E-04	20	178
0.2	15778	7772	213	4324	43	993	3331	170	0	7.31E-07	1.56E-07	15	178
0.25	18766	10100	300	5161	53	1020	4141	247	0	5.42E-05	1.11E-05	13	197
0.25	18766	10480	303	5158	53	1012	4146	250	0	4.20E-04	1.23E-04	10	197
0.25	18766	10818	301	5160	85	924	4236	216	0	0.001349	3.13E-04	25	197
0.3	28026	16522	383	7086	109	955	6131	274	0	8.88E-04	1.21E-04	24	183
0.3	28026	17408	381	7088	86	946	6142	295	0	6.75E-04	1.32E-04	23	183
0.3	28026	15712	375	7094	100	931	6163	275	0	6.81E-05	1.27E-05	10	183
0.35	39784	25122	461	9405	85	918	8487	376	0	5.75E-05	2.43E-05	8	201
0.35	39784	26738	458	9408	85	962	8446	373	0	7.81E-05	1.57E-05	10	201
0.35	39784	25080	458	9408	101	929	8479	357	0	0.002777	5.43E-04	15	201
0.4	50770	33386	586	11674	154	912	10762	432	0	0.001307	3.64E-04	23	202
0.4	50770	30542	586	11674	144	868	10806	442	0	0.001233	3.26E-04	24	202
0.4	50770	33426	586	11674	159	844	10830	427	0	0.002037	4.79E-04	17	202
0.45	53380	30570	616	12263	116	885	11378	500	0	0.002314	6.75E-04	20	216

Continued on the next page

GU%	Obs.	Back.	P	N	TP	FP	TN	FN	MinP	MaxP	AvgP	Pat.	Total
0.45	53380	33382	604	12275	143	888	11387	461	0	0.001023	2.16E-04	23	216
0.45	53380	31594	606	12273	140	920	11353	466	0	0.001946	6.31E-04	22	216
0.5	60722	39502	508	13874	59	986	12888	449	0	0.001487	4.55E-04	19	216
0.5	60722	38840	513	13869	86	930	12939	427	0	0.00559	0.001189	21	216
0.5	60722	36558	517	13865	67	1000	12865	450	0	0.001117	3.63E-04	18	216

Table C.12: *Trypanosoma brucei* length normalization results.

Bins	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
1	18766	10580	259	5050	27	1022	4028	232	0	0.160397	0.067184	16	45
1	18766	10614	262	5047	20	992	4055	242	0	0.177122	0.079396	17	45
1	18766	11340	263	5046	20	1060	3986	243	0	0.118086	0.045997	17	45
2	18766	11258	268	5053	113	953	4100	155	0	0.06677	0.021454	18	57
2	18766	11240	268	5053	86	915	4138	182	0	0.060366	0.029831	13	57
2	18766	11160	267	5054	77	927	4127	190	0	0.074268	0.03226	15	57
3	18766	11234	270	5064	66	940	4124	204	0	0.015402	0.005044	14	70
3	18766	11274	272	5062	68	946	4116	204	0	0.018792	0.003212	12	70
3	18766	10592	272	5062	67	994	4068	205	0	0.024567	0.007808	15	70
4	18766	11310	273	5073	77	948	4125	196	0	0.028619	0.008919	18	82
4	18766	10900	272	5074	67	957	4117	205	0	0.023109	0.00723	12	82
4	18766	10942	273	5073	70	937	4136	203	0	0.041334	0.015928	17	82
5	18766	10502	282	5078	65	942	4136	217	0	0.00386	0.001266	12	96
5	18766	10682	286	5074	76	930	4144	210	0	7.13E-04	2.12E-04	15	96
5	18766	12064	280	5080	64	964	4116	216	0	0.008508	0.002957	13	96
6	18766	11980	283	5081	83	920	4161	200	0	0.011177	0.00283	15	100
6	18766	11734	287	5077	67	948	4129	220	0	0.007442	0.00181	15	100
6	18766	9706	284	5080	69	936	4144	215	0	0.004918	0.001243	14	100
7	18766	10074	286	5099	64	963	4136	222	0	0.001126	2.46E-04	16	121
7	18766	11408	286	5099	88	933	4166	198	0	0.006742	0.001214	17	121
7	18766	10796	286	5099	56	949	4150	230	0	5.84E-05	1.49E-05	15	121
8	18766	10230	293	5133	60	985	4148	233	0	4.66E-04	1.19E-04	10	162
8	18766	10612	292	5134	60	1000	4134	232	0	0.002348	8.38E-04	11	162
8	18766	11094	292	5134	70	978	4156	222	0	0.001487	5.90E-04	18	162
9	18766	11468	291	5140	58	957	4183	233	0	0.001007	3.54E-04	12	167
9	18766	10590	291	5140	67	976	4164	224	0	0.001912	6.83E-04	22	167
9	18766	11060	296	5135	58	951	4184	238	0	1.10E-04	3.09E-05	13	167
10	18766	10308	298	5163	53	958	4205	245	0	4.24E-06	1.13E-06	17	197

Continued on the next page

Bins	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
10	18766	11430	299	5162	53	987	4175	246	0	1.80E-05	3.21E-06	9	197
10	18766	10986	297	5164	69	941	4223	228	0	0.001174	3.51E-04	18	197
13	18766	10798	305	5214	46	1028	4186	259	0	3.16E-06	7.80E-07	12	255
13	18766	10804	306	5213	46	958	4255	260	0	4.14E-06	7.67E-07	16	255
13	18766	10386	307	5212	46	1052	4160	261	0	7.41E-06	1.24E-06	11	255
16	18766	10856	309	5239	45	973	4266	264	0	1.61E-07	5.35E-08	13	284
16	18766	10054	311	5237	45	1031	4206	266	0	2.89E-07	6.92E-08	12	284
16	18766	10396	308	5240	45	1040	4200	263	0	1.26E-08	2.31E-09	10	284
19	18766	10436	319	5321	50	956	4365	269	0	6.51E-07	1.38E-07	15	376
19	18766	11170	317	5323	0	1066	4257	317	0	0	0	2	376
19	18766	9876	318	5322	110	894	4428	208	0	5.55E-16	9.25E-17	12	376
22	18766	9974	318	5331	0	1066	4265	318	0	0	0	2	385
22	18766	10842	318	5331	0	1066	4265	318	0	0	0	2	385
22	18766	11870	322	5327	0	1066	4261	322	0	0	0	2	385

Table C.13: *Trypanosoma brucei* best parameter results.

Run	Obs.	Back.	P	N	TP	FP	TN	FN	MinP	MaxP	AvgP	Pat.	Total
1	1376570	1189596	2330	863359	0	108	863251	2330	0	0.008311	0.003558	5	79
1	1376570	1183912	2330	863400	0	114	863286	2330	0	0.007901	0.002404	5	79
1	1376570	1186020	2331	863386	0	132	863254	2331	0	0.006157	0.002987	5	79
1	1376570	1237572	2330	863381	19	74	863307	2311	0	0.005637	0.003021	5	79
1	1376570	1156850	2334	863376	10	78	863298	2324	0	0.008405	0.004313	4	79
1	1376570	1188152	2336	863384	17	166	863218	2319	0	0.008713	0.002705	8	79
2	1376570	1145934	2369	863420	155	408	863012	2214	0	0.004201	0.001761	6	174
2	1376570	1196248	2361	863455	57	0	863455	2304	0	7.28E-04	7.28E-04	1	174
2	1376570	1169686	2363	863459	151	410	863049	2212	0	0.007008	0.004054	6	174
2	1376570	1148986	2362	863462	501	681	862781	1861	0	0.006435	0.001242	14	174
2	1376570	1193036	2365	863451	510	684	862767	1855	0	0.009697	0.002016	15	174
2	1376570	1171110	2357	863461	236	354	863107	2121	0	0.006749	0.003132	7	174
3	306214	247514	1363	112623	152	49	112574	1211	0	0.006092	0.003047	2	72
3	306214	231204	1362	112624	163	24	112600	1199	0	9.60E-04	4.84E-04	2	72
3	306214	239032	1361	112625	187	63	112562	1174	0	0.008899	0.003454	5	72
3	306214	242064	1362	112624	458	87	112537	904	0	0.006099	0.001121	7	72
3	306214	244292	1365	112621	171	27	112594	1194	0	0.003015	0.001197	3	72
3	306214	244928	1367	112619	163	54	112565	1204	0	0.00167	6.09E-04	3	72
4	306214	259294	1388	112663	357	56	112607	1031	0	0.006777	0.001938	9	137
4	306214	228668	1395	112656	375	59	112597	1020	0	0.008054	0.00211	10	137
4	306214	247828	1391	112660	415	76	112584	976	0	0.005664	0.001997	11	137
4	306214	232080	1395	112656	374	60	112596	1021	0	0.009308	0.003094	10	137
4	306214	239478	1396	112655	264	67	112588	1132	0	0.009563	0.002723	10	137
4	306214	233454	1397	112654	413	70	112584	984	0	0.007982	0.002303	10	137
5	88466	63422	670	23313	5	1631	21682	665	0	0.004716	0.003805	2	59
5	88466	58968	673	23310	2	1041	22269	671	0	0.002044	0.002044	1	59
5	88466	65246	670	23313	17	1489	21824	653	0	0.00915	0.006474	4	59
5	88466	59988	674	23309	48	1061	22248	626	0	0.009111	0.005931	5	59

Continued on the next page

Run	**Obs.**	**Back.**	**P**	**N**	**TP**	**FP**	**TN**	**FN**	**MinP**	**MaxP**	**AvgP**	**Pat.**	**Total**
5	88466	61356	666	23317	155	1482	21835	511	0	0.006621	0.003459	6	59
5	88466	59044	673	23310	7	1901	21409	666	0	0.004089	0.002914	3	59
6	88466	60894	703	23328	177	1258	22070	526	0	0.006561	0.002587	12	107
6	88466	59214	704	23327	148	2121	21206	556	0	0.004297	0.002623	10	107
6	88466	61494	703	23328	147	2134	21194	556	0	0.009275	0.003586	11	107
6	88466	60418	699	23332	115	1746	21586	584	0	0.009972	0.004349	11	107
6	88466	63806	694	23337	151	1687	21650	543	0	0.008284	0.004019	11	107
6	88466	56182	694	23337	149	2725	20612	545	0	0.009222	0.00447	13	107

Appendix D

Glossary of Terms and Abbreviations

3' - Three prime end, or tail end, of an DNA/RNA marking the hydroxyl group of the third carbon in the deoxyribose/ribose sugar.

5' - Five prime end, or start, of an DNA/RNA marking the phosphate group of the fifth carbon in the deoxyribose/ribose sugar.

A, C, G, T, U - Abbreviations of the five bases of the DNA/RNA, which correspond to: A -adenine ; C - cytosine; G - guanine; T - thymine (DNA-only); U-uridine (RNA-only).

bp (Kbp, Mbp) - Base pairs, unit of measurement of RNA/DNA duplex sequences base on number of matching nucleotides.

nt - Nucleotides, unit of measurement for single stranded sequences.

RNA - Ribonucleic acid

mRNA - Messenger ribonucleic acid, transcribed from DNA to be translated into protein.

DNA - Deoxyribonucleic acid

Graph - A graph $G = (V, E)$ where V is the set of vertices and E is the set of edges.

Subgraph - A graph $G_s = (V_s, E_s)$ given a graph $G = (V, E)$ is a **subgraph** if V_s subset V and E_s subset E.

Vertex - A single unit of information, also called a "node".

Edge - The abstract entity that relates two vertices together.

Pattern - A generalization of a subgraph. There is a many to one relationship from subgraphs to patterns.

Anchor - A sequence where its complement is on another RNA molecule and acts to bind the two molecules.

Stem - Secondary structure element where a duplex is formed between two complementary sequences.

Stem-loop - A stem where two ends on either side are joined by unpaired strand.

Interior loop - A region between two stems where each strand is unpaired to the other.

Bulge - A region between two stems where one strand is unpaired to the other.

Pseudoknot - Secondary structure that occurs when a strand with a stem-loop forms a duplex within its own loop.

Bibliography

[1] Mohammad Anwar, Truong Nguyen, and Marcel Turcotte. Identification of consensus RNA secondary structures using suffix arrays. *BMC Bioinformatics*, 7:244:1471–2105, 2006.

[2] G. Benedetti and S. Morosetti. A graph-topological approach to recognition of pattern and similarity in RNA secondary structures. *Biophys. Chem.*, 59(1-2):179–184, Mar 1996.

[3] B. Blum, N. Bakalara, and L. Simpson. A model for RNA editing in kinetoplastid mitochondria: "guide" RNA molecules transcribed from maxicircle DNA provide the edited information. *Cell*, 60(2):189–198, Jan 1990.

[4] Bjrn Bringmann and Siegfried Nijssen. What is frequent in a single graph? In Takashi Washio, Einoshin Suzuki, KaiMing Ting, and Akihiro Inokuchi, editors, *Advances in Knowledge Discovery and Data Mining*, volume 5012 of *Lecture Notes in Computer Science*, pages 858–863. Springer Berlin Heidelberg, 2008.

[5] J. H. Chen, S. Y. Le, and J. V. Maizel. Prediction of common secondary structures of RNAs: a genetic algorithm approach. *Nucleic Acids Res.*, 28(4):991–999, Feb 2000.

[6] H. Cheng, X. Yan, and J. Han. Mining Graph Patterns. In Charu C. Aggarwal and Haixun Wang, editors, *Managing and Mining Graph Data*, volume 40 of *Advances in Database Systems*, pages 365–392. Springer US, 2010.

[7] R.B. D'Agostino and M.A. Stephens. *Goodness-of-Fit Techniques*, chapter Tests Based on EDF Statistics, pages 97–194. Marcel Dekker, 1986.

[8] Jesse Davis and Mark Goadrich. The relationship between Precision-Recall and ROC curves. *Proceedings of the 23rd international conference on Machine learning - ICML '06*, pages 233–240, 2006.

[9] S. R. Eddy and R. Durbin. RNA sequence analysis using covariance models. *Nucleic Acids Res.*, 22(11):2079–2088, Jun 1994.

[10] Sean R Eddy and Richard Durbin. RNA analysis using covariance models. *Nucleic Acids Research*, 22(11):2079–2088, 1994.

[11] Mathias Fiedler and Christian Borgelt. Support computation for mining frequent subgraphs in a single graph. *Proc. 5th Int. Workshop on Mining and Learning*, 2007.

[12] S. Fortin. The Graph Isomorphism Problem (tech. rep. no. tr96-20). Technical report, University of Alberta, Department of Computing Science, 1996.

[13] H. H. Gan, S. Pasquali, and T. Schlick. Exploring the repertoire of RNA secondary motifs using graph theory; implications for RNA design. *Nucleic Acids Res.*, 31(11):2926–2943, Jun 2003.

[14] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of np-completeness.* W. H. Freeman and Company., 1979.

[15] Alexander Gawronski and Marcel Turcotte. Novel framework for the discovery of rna elements and its application to euglenozoa. In *In International Symposium on Bioinformatics Research and Applications (ISBRA)*, 2013.

[16] S. Hajduk and T. Ochsenreiter. RNA editing in kinetoplastids. *RNA Biol*, 7(2):229–236, 2010.

[17] M. Hamada, K. Tsuda, T. Kudo, T. Kin, and K. Asai. Mining frequent stem patterns from unaligned RNA sequences. *Bioinformatics*, 22(20):2480–2487, Oct 2006.

[18] Michiaki Hamada, Toutai Mituyama, and Kiyoshi Asai. Large scale similarity search for locally stable secondary structures among rna sequences. *IPSJ Transactions on Bioinformatics*, 2:36–46, 2009.

[19] J. H. Havgaard, E. Torarinsson, and J. Gorodkin. Fast pairwise structural RNA alignments by pruning of the dynamical programming matrix. *PLoS Comput. Biol.*, 3(10):1896–1908, Oct 2007.

[20] Tams Horvth, Jan Ramon, and Stefan Wrobel. Frequent subgraph mining in outerplanar graphs. In *In Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 197–206, 2006.

[21] Haiyan Hu, Xifeng Yan, Yu Huang, Jiawei Han, and Xianghong Jasmine Zhou. Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics (Oxford, England)*, 21 Suppl 1:i213–21, June 2005.

[22] Yuh-Jyh Hu. Prediction of consensus structural motifs in a family of coregulated RNA sequences. *Nucleic acids research*, 30(17):3886–93, September 2002.

[23] Joseph a Izzo, Namhee Kim, Shereef Elmetwaly, and Tamar Schlick. RAG: an update to the RNA-As-Graphs resource. *BMC Bioinformatics*, 12(1):219, January 2011.

[24] Chuntao Jiang, Frans Coenen, and Michele Zito. A Survey of Frequent Subgraph Mining Algorithms. *To appear: Knowledge Engineering Review*, 00:1–31, 2012.

[25] N. S. Ketkar, L. B. Holder, and D. J. Cook. Subdue: compression-based frequent pattern discovery in graph data. In *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations*, OSDM '05, pages 71–76, New York, NY, USA, 2005. ACM.

[26] G. Kiethega, Y. Yan, M. Turcotte, and G. Burger. RNA-level unscrambling of fragmented genes in Diplonema mitochondria. *RNA Biol*, 10(2), Jan 2013.

[27] G. N. Kiethega, M. Turcotte, and G. Burger. Evolutionarily conserved cox1 transsplicing without cis-motifs. *Mol. Biol. Evol.*, 28(9):2425–2428, Sep 2011.

[28] S. K. Kim, J. W. Nam, J. K. Rhee, W. J. Lee, and B. T. Zhang. miTarget: microRNA target gene prediction using a support vector machine. *BMC Bioinformatics*, 7:411, 2006.

[29] Debra Knisley, Jeff Knisley, Chelsea Ross, and Alissa Rockney. Classifying Multigraph Models of Secondary RNA Structure Using Graph-Theoretic Descriptors. *ISRN Bioinformatics*, 2012:1–11, 2012.

[30] Mehmet Koyutürk, Ananth Grama, and Wojciech Szpankowski. An efficient algorithm for detecting frequent subgraphs in biological networks. *Bioinformatics (Oxford, England)*, 20 Suppl 1:i200–7, August 2004.

[31] J. Kruger and M. Rehmsmeier. RNAhybrid: microRNA target prediction easy, fast and flexible. *Nucleic Acids Res.*, 34(Web Server issue):W451–454, Jul 2006.

[32] M. Kuramochi and G. Karypis. GREW - a scalable frequent subgraph discovery algorithm. In *Data Mining, 2004. ICDM '04. Fourth IEEE International Conference on*, pages 439–442, 2004.

[33] M. Kuramochi and G. Karypis. Finding frequent patterns in a large sparse graph. *Data Min. Knowl. Discov.*, 11(3):243–271, November 2005.

[34] S. Y. Le, R. Nussinov, and J. V. Maizel. Tree graphs of RNA secondary structures and their comparisons. *Comput. Biomed. Res.*, 22(5):461–473, Oct 1989.

[35] B. P. Lewis, C. B. Burge, and D. P. Bartel. Conserved seed pairing, often flanked by adenosines, indicates that thousands of human genes are microRNA targets. *Cell*, 120(1):15–20, Jan 2005.

[36] R. Lopez, V. Silventoinen, S. Robinson, A. Kibria, and W. Gish. WU-Blast2 server at the European Bioinformatics Institute. *Nucleic Acids Res.*, 31(13):3795–3798, Jul 2003.

[37] M. J. Madej, M. Niemann, A. Huttenhofer, and H. U. Goringer. Identification of novel guide RNAs from the mitochondria of Trypanosoma brucei. *RNA Biol*, 5(2):84–91, 2008.

[38] H. Matsui, K. Sato, and Y. Sakakibara. Pair stochastic tree adjoining grammars for aligning and predicting pseudoknot RNA structures. *Bioinformatics*, 21(11):2611–2617, Jun 2005.

[39] Giancarlo Mauri and Giulio Pavesi. Algorithms for pattern matching and discovery in RNA secondary structure. *Theoretical Computer Science*, 335(1):29 – 51, 2005. Pattern Discovery in the Post Genome.

[40] T. Ochsenreiter, M. Cipriano, and S. L. Hajduk. KISS: the kinetoplastid RNA editing sequence search tool. *RNA*, 13(1)(1355-8382):1–4, 2007.

[41] M. H. Radfar, W. Wong, and Q. Morris. Computational prediction of intronic microRNA targets using host gene expression reveals novel regulatory mechanisms. *PLoS ONE*, 6(6):e19312, 2011.

[42] David Sankoff. Simultaneous solution of the rna folding, alignment and protosequence problems. *SIAM Journal on Applied Mathematics*, 45(5):810–825, October 1985.

[43] S. D. Seiwert. The ins and outs of editing RNA in kinetoplastids. *Parasitol. Today (Regul. Ed.)*, 11(10):362–368, Oct 1995.

[44] R. Shen, N. C. Goonesekere, and C. Guda. Mining functional subgraphs from cancer protein-protein interaction networks. *BMC Syst Biol*, 6 Suppl 3:S2, 2012.

[45] Yasuo Tabei, Daisuke Okanohara, Shuichi Hirose, and Koji Tsuda. LGM: mining frequent subgraphs from linear graphs. *15th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2011.

[46] I. Veksler-Lublinsky, M. Ziv-Ukelson, D. Barash, and K. Kedem. A structure-based flexible search method for motifs in RNA. *J. Comput. Biol.*, 14(7):908–926, Sep 2007.

[47] A. von Haeseler, B. Blum, L. Simpson, N. Sturm, and M. S. Waterman. Computer methods for locating kinetoplastid cryptogenes. *Nucleic Acids Research*, 20(11)(0305-1048):2717–24, 1992.

[48] X Wang, JS Snoeyink, and Wei Wang. Mining RNA tertiary motifs with structure graphs. *19th International Conference on Scientific and Statistical Database Management*, (Ssdbm), 2007.

[49] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. *2002 IEEE International Conference on Data Mining*, pages 721 – 724, 2002.

[50] Z. Yao, Z. Weinberg, and W. L. Ruzzo. CMfinder–a covariance model based RNA motif finding algorithm. *Bioinformatics*, 22(4):445–452, Feb 2006.

[51] Laura E Yu and Donna J Koslowsky. Interactions of mRNAs and gRNAs involved in trypanosome mitochondrial RNA editing: structure probing of a gRNA bound to its cognate mRNA. *RNA (New York, N.Y.)*, 12(6):1050–60, June 2006.

[52] Hao Zheng, Rongguo Fu, Jin-Tao Wang, Qinyou Liu, Haibin Chen, and Shi-Wen Jiang. Advances in the Techniques for the Prediction of microRNA Targets. *International journal of molecular sciences*, 14(4):8179–87, January 2013.

[53] Alena Zíková, Jana Kopecná, Maria a Schumacher, Kenneth Stuart, Lukás Trantírek, and Julius Lukes. Structure and function of the native and recombinant mitochondrial mrp1/mrp2 complex from trypanosoma brucei. *International journal for parasitology*, 38(8-9):901–12, July 2008.

[54] R. Zou and L. B. Holder. Frequent subgraph mining on a single large graph using sampling techniques. In *Proceedings of the Eighth Workshop on Mining and Learning with Graphs*, MLG '10, pages 171–178, New York, NY, USA, 2010. ACM.

[55] Michael Zuker and David Sankoff. RNA Secondary Structures and their Prediction. *Bulletin of Mathematical Biology*, 46(4):591–621, 1984.