HIDDEN MARKOV MODEL-BASED HOMOLOGY SEARCH AND GENE PREDICTION IN NGS ERA

By

Prapaporn Techa-angkoon

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science – Doctor of Philosophy

2017

ABSTRACT

HIDDEN MARKOV MODEL-BASED HOMOLOGY SEARCH AND GENE PREDICTION IN NGS ERA

By

Prapaporn Techa-angkoon

The exponential cost reduction of next-generation sequencing (NGS) enabled researchers to sequence a large number of organisms in order to answer various questions in biology, ecology, health, etc. For newly sequenced genomes, gene prediction and homology search against characterized protein sequence databases are two fundamental tasks for annotating functional elements in the genomes. The main goal of gene prediction is to identify the gene locus and their structures. As there is accumulating evidence showing important functions of RNAs (ncRNAs), comprehensive gene prediction should include both protein-coding genes and ncRNAs. Homology search against protein sequences can aid identification of functional elements in genomes. Although there are intensive research in the fields of gene prediction, ncRNA search, and homology search, there are still unaddressed challenges. In this dissertation, I made contributions in these three areas. For gene prediction, I designed an HMM-based *ab initio* gene prediction tool that considers G+C gradient in grass genomes. For homology search, I designed a method that can align short reads against protein families using profile HMMs. For ncRNA search, I designed an ncRNA alignment tool that can align highly structured ncRNAs using only sequence similarity. Below I summarize my contributions.

Despite decades of research about gene prediction, existing gene prediction tools are not carefully designed to deal with variant G+C content and $5'$-$3'$ changing patterns inside coding regions. Thus, these tools can miss genes with positive or negative G+C gradient in grass genomes such as rice, maize, sorghum, etc. I implemented a tool named AUGUSTUS-GC that accounts for $5'$-$3'$

G+C gradient. Our tool can accurately predict protein-coding genes in plant genomes especially grass genomes.

A large number of sequencing projects produced short reads from the whole genomes or transcriptomic data. I designed a short reads homology search tool that employs paired-end reads to improve homology search sensitivity. The experimental results show that our tool can achieve significantly better sensitivity and accuracy in aligning short reads that are part of remote homologs.

Despite the extensive studies of ncRNA search, the existing tools that heavily depend on the secondary structure in homology search cannot efficiently handle RNA-seq data that is accumulating rapidly. It will be ideal if we can have a faster ncRNA homology search tool with similar accuracy as those adopting secondary structure. I implemented an accurate ncRNA alignment tool called glu-RNA that can achieve similar accuracy to structural alignment tools while keeping the same running time complexity as sequence alignment tools. The experimental results demonstrate that our tool can achieve more accurate alignments than the popular sequence alignment tools and a well-known structural alignment program.

To my beloved family, my tireless teachers, my encouraging friends, and the inspired scientists

ACKNOWLEDGMENTS

The completion of this dissertation would not have been possible without the support, encouragement, and guidance of several people. I would like to express my sincere gratitude to all of them. First of all, I would like to express the deepest appreciation and thanks to my advisor, Dr. Yanni Sun, for accepting me into her research group and all her excellent support, valuable guidance, understanding, patience, and encouragement that she gave me throughout my research and dissertation work in the fields of Computational Biology and Bioinformatics. It was a pleasure working with her and learning from her research expertise.

I am also grateful my Ph.D. guidance committee members: Dr. Shin-Han Shiu, Dr. Eric Torng, and Dr. Jin Chen, for their interest in my work and their precious time. They gave me a lot of important suggestions, helpful discussions, and information throughout my Ph.D. program. I also gratefully acknowledge Dr. C. Titus Brown, Dr. Kevin Childs, and Dr. Tiffany Liu for their valuable time, useful suggestions, and all research support. I would also like to thank all the faculty and staff members in the Department of Computer Science and Engineering who have taught me or assisted me. Without their support, my work would have undoubtedly been much more challenging.

I also want to thank my lab mates, Dr. Yuan Zhang, Dr. Rujira Achawanantakun, Dr. Jikai Lei, Dr. Cheng Yuan, Jiao Chen, and Nan Du for providing a friendly working environment, discussing and cooperating on various research topics. I would also like to thank Yuanchun Zhao, Xiaoqing Zhu, Ke Shen, Dr. Aditya K. Sood, Dr. Surangkana Rawangyot, Worawut Srisukkham, Chotirat Rattanasinchai, Suttipun Sungsuwan, other graduate students in the department, and my friends in Thailand for their support and help in my daily life and useful discussions throughout these years.

I would also like to acknowledge Panjasawatwong family and Techa-angkoon family especially

my beloved parents for their love, care, and support. I thank you for taking care of my little son and daughter. I thank you from the bottom of my heart. I would also like to give my special thanks to my husband, Krit Panjasawatwong, for patient love, hard work, endless help, support and encouragement throughout my life and my Ph.D. program. I thank you for working hard and raising our little son and daughter. I also thank my little son and daughter, Danuphat Panjasawatwong and Chanyaphach Panjasawatwong, for making my life more complete and meaningful and being my great motivation. I love you with all my heart.

Last but not least. I gratefully acknowledge the Royal Thai Government, Faculty of Science at Chiang Mai University, NSF CAREER Grant DBI-0953738, NSF IOS-1126998, Dr. Yanni Sun, and my husband for financial support throughout these years. I would like to thank the staff at the Office Of Education Affairs (OEADC), all faculty and staff members at Faculty of Science, Chiang Mai University for all help and support during my graduate study.

TABLE OF CONTENTS

LIST OF TABLES . . . . . **x**

LIST OF FIGURES . . . . . **xii**

Chapter 1 Introduction **1**
1.1 Gene prediction 1
1.1.1 Prokaryotic gene prediction 3
1.1.2 Eukaryotic gene prediction 3
1.2 Homology search 4
1.2.1 The challenges of homology search in large-scale genomic data 5
1.2.1.1 Next-generation sequencing technology 5
1.2.1.2 The challenges of homology search for next-generation sequencing data 7
1.2.1.3 Non-coding RNA homology search in large-scale genomic data . 8
1.3 Contributions 10

Chapter 2 a Review of Markov Chain and Hidden Markov Models **12**
2.1 Hidden Markov Models (HMMs) 12
2.1.1 Markov Chains 12
2.1.2 Hidden Markov models 15
2.1.2.1 The most probable state path: Viterbi Algorithm 15
2.1.2.2 The Forward Algorithm 17
2.1.2.3 The Backward Algorithm 18

Chapter 3 AUGUSTUS-GC: a Gene Prediction Model accounting for 5′-3′ G+C Gradient **19**
3.1 Introduction 19
3.2 Method 21
3.2.1 The Hidden Markov Model of AUGUSTUS-GC 21
3.3 Experimental Results 27
3.3.1 Gene Prediction in *Arabidopsis Thaliana* 29
3.3.1.1 Training the HMM model 29
3.3.1.2 Performance comparison between different gene prediction tools 30
3.3.2 Gene Prediction in *Oryza Sativa* 33
3.3.2.1 Gene Identification in *Oryza Sativa* obtained from Childs Lab, Department of Plant Biology, Michigan State University 34
3.3.2.2 Finding genes in *Oryza Sativa* retrieved from Stanke et al 38

Chapter 4 Sensitive Short Read Homology Search for Paired-End Read Sequencing Data **44**
4.1 Introduction 44

4.1.0.3 Is HMMER good enough for short-read homology search? 46
4.2 Methods 48
4.2.1 Constructing fragment length distribution 50
4.2.2 Probabilistic model 51
4.3 Experimental results 53
4.3.1 Profile-based short read homology search in *Arabidopsis Thaliana* RNA-Seq dataset 54
4.3.1.1 Determination for true membership of paired-end reads 54
4.3.1.2 Performance of fragment length distribution 55
4.3.1.3 Our method can align significantly more reads 55
4.3.1.4 Sensitivity and accuracy of short read homology search 56
4.3.1.4.1 Performance of case 1: 57
4.3.1.4.2 Performance of case 2: 57
4.3.1.5 Performance evaluation on domain-level 59
4.3.1.6 Running time analysis 61
4.3.2 Profile homology search for short reads in a metagenomic dataset from synthetic communities 62
4.3.2.1 Dataset 62
4.3.2.2 Determination of true membership of paired-end reads 62
4.3.2.3 Performance of fragment length distribution 63
4.3.2.4 Our method can align more reads 63
4.3.2.5 Sensitivity and accuracy of short read homology search 63
4.3.2.5.1 Case 1: one end is aligned by HMMER 63
4.3.2.5.2 Case 2: both ends are aligned by HMMER 65
4.3.2.6 Domain-level performance evaluation 65
4.3.2.7 Running time on a metagenomic dataset of synthetic communities 67
4.4 Discussion and conclusion 68

Chapter 5 glu-RNA: aliGn highLy strUctured ncRNAs using only sequence similarity 70
5.1 Introduction 70
5.2 Related Work 72
5.3 Method 74
5.3.1 Alignment path and similarity between two alignments 78
5.3.2 Choosing the base alignment 80
5.3.3 Improving alignment accuracy using machine learning methods 82
5.3.3.1 Constructing a legal alignment 84
5.3.4 Running time analysis 85
5.4 Experimental Results 85
5.4.1 Data 85
5.4.1.1 Training data 89
5.4.1.2 Testing data 90
5.4.2 Performance comparison between different classification and regression models 90
5.4.3 Performance comparison of different alignment programs 92

5.4.4 Discussion 96
5.5 Conclusion and future work 97

Chapter 6 Conclusion and Future Work 99

BIBLIOGRAPHY 101

LIST OF TABLES

Table 3.1: Performance comparison of gene prediction tools on *Arabidopsis Thaliana*. The transition probabilities were divided into three equal portions. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS. 31

Table 3.2: Performance comparison of gene prediction tools on *Arabidopsis Thaliana*. The transition probabilities was trained by computing maximum likelihood estimation. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS. 31

Table 3.3: Performance comparison of gene prediction approaches on *Oryza Sativa* obtained from Childs Lab. The transition probabilities were divided into three equal parts. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS. 36

Table 3.4: Performance comparison of gene prediction approaches on *Oryza Sativa* obtained from Childs Lab. The transition probabilities was calculated by using maximum likelihood estimation. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS. 36

Table 3.5: Performance comparison of gene prediction tools on *Oryza Sativa* provided by Stanke et al. The transition probabilities were divided into three equal parts. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS. 39

Table 3.6: Performance comparison of gene prediction tools on *Oryza Sativa* provided by Stanke et al. The transition probabilities was trained using maximum likelihood estimation. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS. 39

Table 4.1: The performance of HMMER under different cutoffs on classifying reads in the *Arabidopsis Thaliana* RNA-Seq dataset. There are 2,972,809 paired-end reads being uniquely mapped to 3,577 annotated protein domain families in the reference genome. However, HMMER under E-value cutoff 10 missed at least half of the reads when aligning these reads to input domains. The alignments by HMMER can be divided into three cases. Case 1: only one end can be aligned to one or multiple domains. Case 2: both ends can be aligned to one or multiple domains. Case 3: no end is aligned to any domain. In the table, the percentage of a case can be represented by each number. "HMMER w/o filtration" represents shutting off all filtration steps and running full Forward/Backward. "HMMER GA cutoff" represents applying HMMER with gathering thresholds. . 46

Table 4.2: The percentages of all three cases of paired-end read alignments by HMMER and our tool for the RNA-Seq data. Case 1 represents one end. Case 2 represents both ends. Case 3 represents no end. 56

Table 4.3: The running time of HMMER under different cutoffs and our tool on the *Arabidopsis Thaliana* RNA-Seq dataset. *Note:* The running time is the total running time of homology search tool to align 9,559,784 paired-end reads with 3962 domains. 61

Table 4.4: The percentages of all three cases of paired-end read alignments by HMMER and our tool for the synthetic metagenomic data. Case 1 represents one end. Case 2 represents both ends. Case 3 represents no end. 64

Table 4.5: The running time of HMMER under different cutoffs and our tool on a metagenomic dataset of synthetic communities. *Note:* The running time is the overall running time of homology search tool to align 52,486,341 reads with 111 domains. 68

Table 5.1: Number of best alignments generated by four sequence alignment tools for 662 pairs of tRNAs and Riboswitches. Note that N-W is Needleman-Wunsch. ClustalW, a multiple sequence alignment tool, can also be applied to a pair of sequences. 81

Table 5.2: Information of the Training Data Set 87

Table 5.3: Information of the Testing Data Set. Note that there is no overlap between the training and testing data. 91

Table 5.4: The accuracy of different models. 92

Table 5.5: Average distances of different alignment programs on 360 pairs of ncRNAs in the testing data set. **Bold number** represents the lowest average distance for an ncRNA family. Underlined numbershows the second lowest average distance for an ncRNA family. 95

Table 5.6: Average distances of different alignment programs on 58 pairs of sequences in five families with sequence similarity below 40%. **Bold number** represents the lowest average distance. Underlined numbershows the second lowest average distance. 95

LIST OF FIGURES

Figure 1.1: Prokaryotic gene structure. 2

Figure 1.2: Eukaryotic gene structure. 2

Figure 1.3: Simplified representation of next generation sequencing technology that produce single-end reads or paired-end reads. First, the reference genome sequence is fragmented. For single-end sequencing, one end of DNA fragment is sequenced. For paired-end sequencing, both ends of fragmented DNA are sequenced. Insert size is the length of sequence Read 1 and Read 2 as well as unknown sequence between read pair. 7

Figure 1.4: A tRNA sequence and its secondary structure 9

Figure 2.1: An example of Markov Chain 13

Figure 2.2: An example of Markov Chain. Begin and end states are added to a Markov chain model for modelling the beginning and the ending of a sequence. 14

Figure 3.1: The state diagram of *ab initio* AUGUSTUS [101].The states beginning with r represents the reverse strand. E_{single}: a single exon. E_{init} : the initial coding exon of a multi-exon gene. *DSS*: a donor splice site. I_{short}: an intron emitting at most d nucleotides. I_{fixed}: a longer intron with the first d nucleotides. I_{geo}: a longer intron emitting one nucleotide at a time after the first d nucleotides. *ASS*: an acceptor splice site with branch point. E: an internal coding exon of a multi-exon gene. E_{term}: the last coding exon of a multi-exon gene. *IR*: intergenic region. Diamonds represent the states that emit fixed length strings. Ovals represent the states including explicit length distribution. The numbers at the arrows show the transition probabilities. The exponents 0, 1, and 2 represent the reading frame phase. For an exon state, this is the position of the last base of the exon in its codon. For the other states, the exponent are the preceding-exon phase. The small circles represent silent states. 22

Figure 3.2: The state diagram of AUGUSTUS-GC. The states beginning with r represents the reverse strand. $E^{\mathrm{H}}_{\mathrm{single}}$: a single exon of high G+C content. $E^{\mathrm{M}}_{\mathrm{single}}$: a single exon of medium G+C content. $E^{\mathrm{L}}_{\mathrm{single}}$: a single exon of low G+C content. $E_{\mathrm{init\,H}}$: the initial coding exon of a multi-exon gene with high G+C content. $E_{\mathrm{init\,M}}$: the initial exon of a multi-exon gene with medium G+C content. $E_{\mathrm{init\,L}}$: the initial exon of a multi-exon gene with low G+C content. *DSS*: a donor splice site. I_{short}: an intron emitting at most d nucleotides. I_{fixed}: a longer intron with the first d nucleotides. I_{geo}: a longer intron emitting one nucleotide at a time after the first d nucleotides. *ASS*: an acceptor splice site with branch point. E_{H}: an internal coding exon of a multi-exon gene with high G+C content. E_{M}: the internal exon of a multi-exon gene with medium G+C content. E_{L}: the internal exon of a multi-exon gene with low G+C content. $E^{\mathrm{H}}_{\mathrm{term}}$: the last coding exon of a multi-exon gene with high G+C content. $E^{\mathrm{M}}_{\mathrm{term}}$: the terminal exon of a multi-exon gene with medium G+C content. $E^{\mathrm{L}}_{\mathrm{term}}$: the terminal exon of a multi-exon gene with low G+C content. *IR*: intergenic region. Diamonds represent the states that emit fixed length strings. Ovals represent the states including explicit length distribution. The numbers at the arrows show the transition probabilities. The exponents 0, 1, and 2 represent the reading frame phase. For an exon state, this is the position of the last base of the exon in its codon. For the other states, the exponent are the preceding-exon phase. The small circles represent silent states. 23

Figure 3.3: The example of a gene with G+C content gradient in *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content. 27

Figure 3.4: G+C Content of *Arabidopsis Thaliana* data set 30

Figure 3.5: An example of a predicted gene with G+C content gradient of *Arabidopsis Thaliana* data set. This gene was predicted by AUGUSTUS-GC. X-axis represents exon index. Y-axis represents G+C content. 32

Figure 3.6: G+C Content of the first data set of *Oryza Sativa* 34

Figure 3.7: G+C Content of the second *Oryza Sativa* data set 35

Figure 3.8: A predicted gene on forward strand with G+C content gradient of the first *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content. 37

Figure 3.9: Three predicted genes on reverse strand with G+C content gradient of the first *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content. 37

Figure 3.10: An example of a uniquely predicted gene on reverse strand with G+C content gradient of second *Oryza Sativa* data set. However, regular AUGUSTUS cannot detect it. X-axis represents exon index. Y-axis represents G+C content. 41

Figure 3.11: First example of a predicted gene on forward strand with G+C content gradient of second *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content. 41

Figure 3.12: Second example of a predicted gene on forward strand with G+C content gradient of second *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content. 42

Figure 3.13: First example of a predicted gene on reverse strand with G+C content gradient of second *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content. 42

Figure 3.14: Second example of a predicted gene on reverse strand with G+C content gradient of second *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content. 43

Figure 3.15: Third example of a predicted gene on reverse strand with G+C content gradient of second *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content. 43

Figure 4.1: An example of a protein family, its alignment with a gene, and read mapping positions of a read pair against the gene. The Pkinase model had annotation line of consensus structure. The line beginning with Pkinase is the consensus of the query model. Capital letters show positions of the most conservation. Dots (.) in this line represent insertions in the target gene sequence with respect to the model. The midline represents matches between the Pkinase model and the AT2G28930.1 gene sequence. A + represents positive score. The line beginning with AT2G28930.1 is the target gene sequence. Dashes (-) in this line represents deletions in the gene sequence with respect to the model. The bottom line indicates the posterior probability of each aligned residue. A 0 represents 0-5%, 1 represents 5-15%, ..., 9 represents 85-95%, and * represents 95-100% posterior probability. The line starting with r_1 and ending with r_2 is read mapping regions on the gene sequence. A - indicates where the position of the read can be mapped to the gene sequence. 48

Figure 4.2: HMM alignments of a read pair. Paired-end reads r_1 and r_2 represented by two greyscale lines are aligned against models M_1, M_2, and M_3 with different scores of alignments. The darker lines represent bigger scores. The fragment size distribution is provided above each model. The distance between the two alignments is computed and is used to compute the likelihood of the corresponding fragment size. In this example, M_1 is most likely to be the native family. 50

Figure 4.3: An example of fragment length calculation. The alignment positions along the profile HMM can be converted into positions in each seed sequences. The fragment size is computed as the average size of those mapped regions. 52

Figure 4.4: Comparing fragment length distribution of our method (blue) to fragment length distribution constructed from read mapping results (red). X-axis represents the length of fragment in **amino acids**. Y-axis represents probability of the corresponding fragment size. 56

Figure 4.5: ROC curves of profile-based short read homology search for *Arabidopsis* RNA-Seq data. We compared HMMER and our tool on case 1, where one end can be aligned by HMMER with default E-value. Note that HMMER with GA cutoff has one data point. 58

Figure 4.6: ROC curves of profile-based short read homology search for *Arabidopsis* RNA-Seq data. We compared HMMER and our tool on case 2, where both ends are aligned by HMMER with default E-value. Note that HMMER with GA cutoff has one data point. Using posterior probability helps remove false aligned domains and thus leads to better tradeoff between sensitivity and FP rate. . . . 58

Figure 4.7: The distance comparison between our method and HMMER on case 1 of the RNA-Seq dataset of *Arabidopsis*. 377 domains with the largest distance values are listed in the four subplots. X-axis shows the indices of the domains. Smaller value indicates closer domain abundance to the ground truth. The average distances of HMMER, HMMER w/o filtration, HMMER GA cutoff, and our tool are 704.92, 781.80, 1,054.77, and 522.12, respectively. 60

Figure 4.8: The distance comparison between our method and HMMER on case 2 of the RNA-Seq dataset of *Arabidopsis*. 358 domains with the largest distances are listed in the four subplots. X-axis shows the indices of the domains. Smaller value indicates closer domain abundance to the ground truth. The average distances of HMMER, HMMER w/o filtration, HMMER GA cutoff, and our method are 818.09, 704.65, 1084.50, and 558.60, respectively. 61

Figure 4.9: Comparing fragment length distribution of our tool (blue) to fragment length distribution constructed from read mapping results (red). X-axis represents fragment length in **amino acids**. Y-axis represents the probability of the corresponding fragment size. 64

Figure 4.10: ROC curves of profile-based short read homology search for the synthetic metagenomic data set. We compared HMMER and our tool on case 1, where one end can be aligned by HMMER with default E-value. Note that HMMER under GA cutoff has one data point. 65

Figure 4.11: ROC curves of profile-based short read homology search for the synthetic metagenomic data. We compared HMMER and our tool on case 2, where both ends are aligned by HMMER under default E-value. Note that HMMER under GA cutoff has one data point. Using posterior probability helps remove false aligned domains and thus leads to better tradeoff between sensitivity and FP rate. 66

Figure 4.12: The distance comparison between our method and HMMER on case 1 of the metagenomic data set. X-axis shows the indices of the domains. Smaller value indicates closer domain abundance to the ground truth. Domains are sorted based on the distance of our tool. Due to scaling issues, domains with the largest distances are plotted in the embedded window. 67

Figure 4.13: The distance comparison between our method and HMMER on case 2 of the metagenomic data set. X-axis shows the indices of domains. Smaller value indicates closer domain abundance to the ground truth. 68

Figure 5.1: Alignments generated by four alignment programs for a pair of tRNAs with sequence similarity 80%. X-axis represents sequence 2 and Y-axis represents sequence 1. The alignment starts with (0,0). 74

Figure 5.2: Alignments generated by four alignment programs for a pair of tRNAs with sequence similarity 40%. X-axis represents sequence 2 and Y-axis represents sequence 1. The alignment starts with (0,0). 75

Figure 5.3: The reference alignment and alignments generated by four alignment programs for a pair of tRNAs with sequence similarity 40%. The predicted secondary structures are shown when each alignment is used as input to Pfold, a structure prediction program. 77

Figure 5.4: Construct a legal alignment using local search. $\times$ represent nodes in the base alignment. $\bigcirc$ represent new positions of nodes in base alignment after extension. (A). node (3,3) is moved to (3,4), which is represented by $\square$. (B). $\square$ represent inserted nodes. 84

Figure 5.5: Types of ncRNAs and their consensus secondary structures (tRNA and riboswitches). 86

Figure 5.6: Sequence similarity histogram of training data set. X-axis represents the sequence similarity. Y-axis represents the number of ncRNA pairs. 88

Figure 5.7: Consensus secondary structures of U2 and gcvT. 90

Figure 5.8: Comparing the alignments generated by glu-RNA, dpswalign, and Dynalign to the reference alignment of a THI pair with sequence similarity 45%. glu-RNA can produce more accurate alignment than dpswalign and Dynalign. 97

Chapter 1

Introduction

The exponential cost reduction of NGS enables us to sequence many more organisms including non-model species or any species that are of interest to biologists. Genome annotation, which labels functional elements inside genomes, is a fundamental step for understanding the functions of newly sequenced genomes. In particular, gene annotation is a key step in whole genome annotation. With increased knowledge about the importance of noncoding RNAs, gene annotation should include both protein-coding genes and non-coding RNA genes. Homology search is also an essential task for annotating functional elements in newly sequenced genomes. There has been extensive research in areas of gene prediction, homology search, and ncRNA search. However, there are still unaddressed challenges. During my Ph.D. study, I conducted research in these three areas. I will first describe the background of the three problems. Then I will highlight the unaddressed challenges. Finally, I summarize my work.

1.1 Gene prediction

With advanced sequencing technology, the number of new genomes has rapidly increased. The abundance of newly sequenced genomes needs better and efficient computational sequence analysis methods and tools to understand more about a wide variety of organisms. One of the most important steps in genome analysis is the identification of all genes in a genome. The objective of gene prediction is to identify the gene locus and their structure.

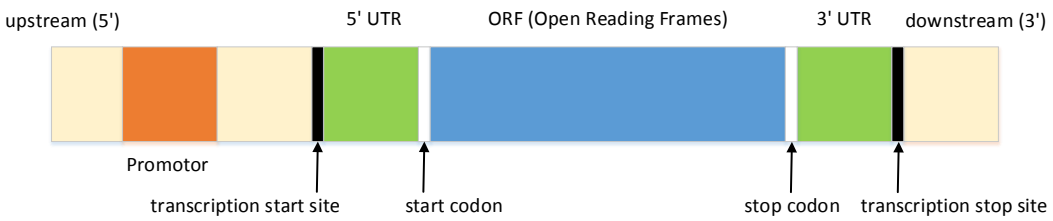


Figure 1.1: Prokaryotic gene structure.

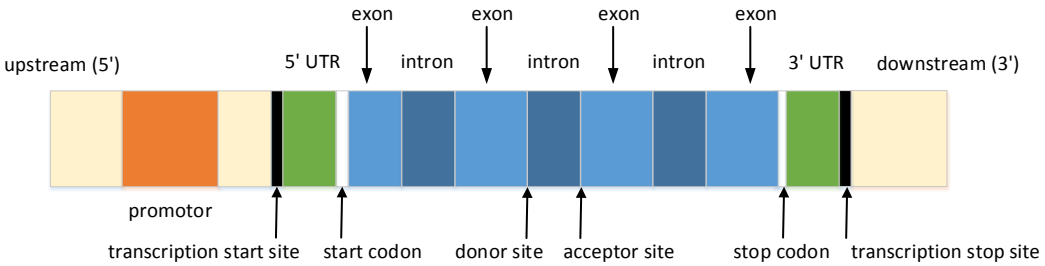


Figure 1.2: Eukaryotic gene structure.

There are two types of gene prediction distinguished by the organisms: prokaryotes and eukaryotes. Prokaryotes like bacteria and archaea are organisms whose cells have no nucleus or any other membrane-bound compartments. Eukaryotes are organisms whose cells contain nucleus and membrane-bound organelles [56]. Eukaryotes can be single-celled or multi-celled, and include animals, plants, fungi, microsporidia, etc. Identifying genes in prokaryotic genomes is less complicated owing to the high gene density and no introns in the protein coding regions. Figure 1.1 shows the structure of the prokaryotic gene. Complete prokaryotic gene structure begins at promoter region. Transcription start site (TSS) indicates the start location of transcription of the gene. Transcription stop site indicates the stop of transcription. Transcription comprises of 5′ Untranslated region (5′UTR), Open reading frames (ORFs), and 3′UTR. For translation in prokaryotes, a continuous coding segment from start codon to stop codon will be translated.

Gene prediction in eukaryotes is more challenging due to the exon-intron structure. Figure 1.2 illustrates eukaryotic gene structure. In eukaryotes, the coding genes are not continuous. Most of the genes are divided into several pieces called exons (initial exons, internal exons, terminal

exons) that are interspersed with introns. Some genes have only single exon without introns. Unlike prokaryotic gene structure, eukaryotic gene structure does not have ORF with continuous segment. Because, significant amounts of non-coding sequences that do not translate into proteins but the proportion of protein coding genes in the entire genome is relative small [44]. The genome in eukaryotes is enormous and more complicated than the genome in prokaryotes.

1.1.1 Prokaryotic gene prediction

Identifying genes in a sequence of prokaryotes is less difficult than predicting genes in a genome of eukaryotes. The gene structure of prokaryotes is less complex than that of eukaryotes because it does not contain introns. One protein-coding gene in a genome corresponds to one ORF starting with a start codon and ending with a stop codon. Because the start codon can occur inside a sequence of a gene or appear in upstream of the translation start site. The goal of prokaryotic gene prediction tools is to distinguish between the coding and non-coding regions in the genome and determine the correct start codon of each gene. In order to identify genes, there are many computational methods. These methods consist of probabilistic models such as Hidden Markov Models (HMMs) [61, 92, 23, 11, 55], and machine learning methods such as Support Vector Machines (SVMs) [49] and Self-Organizing Maps (SOMs) [64].

1.1.2 Eukaryotic gene prediction

Comparing with prokaryotic gene prediction, the gene identification in eukaryotic organisms is a more challenging problem. Because the genomes are very large, but they contain only a small proportion of protein-coding genes surrounded by the large amounts of non-coding sequence. Moreover, the complicated of gene structure in eukaryotic makes identification of the correct position

of gene structure in a genome more challenging. Several methods have been implemented. One of the most widely used approaches for gene finding is based on Hidden Markov Models (HMMs). Existing gene prediction tools roughly fall into two groups. The first category comprises of *ab initio* tools which utilize statistical patterns and an intrinsic information extracting from training data set to accurately locate the boundaries of coding regions in a query DNA sequence. Many *ab initio* tools based on probabilistic models. The success of HMM-based gene prediction has been applied in HMMgene [50, 51] and VEIL [39]. A very successful extension of HMMs for gene identification is Generalized HMM (GHMM). The generalization of standard HMM is to allow variable state length. GHMM was first implemented for gene prediction tool in GENIE [53] followed by one of the most famous gene identification GENSCAN [14, 15]. Then, there are many GHMM-based gene finders including Phat [18], AUGUSTUS [98], Exonomy [65], SNAP [47], TIGRscan [66], GlimmerHMM [66], etc. The second category consists of comparative tools which employ information other than a query sequence. Some extrinsic methods incorporate one or more sequences from other species and compare a query sequence with the genomic sequences of closely related species to identify genes. The tools in this group include SLAM [1], SGP2 [81], TWINSCAN [48], N-SCAN [34], DOUBLESCAN [73], AGenDA [105, 75], etc.

1.2 Homology search

In computational biology, homology search is one of the important steps in biological sequence analysis. The step of homology search compares a query sequence to another sequence or a sequence family database to find statistically significant similarity between sequences. Homologous sequences share a common ancestry and have the same or similar functions. The state-of-the-art method for homology search is based on comparative sequence analysis. There are two main

comparative methods for homology search. One approach for homology search is based on pairwise sequence comparison. This approach compares a query sequence against each sequence in a database and generates pairwise sequence alignment with significant similarity. One of widely used pairwise homology search tools is BLAST [3]. However, conventional pairwise sequence comparison tools such as BLAST yield low sensitivity for remotely related sequences [80]. Thus, more sensitivity homology search tools are required [63, 13, 102, 20, 2]. Pairwise sequence comparison has been extended to multiple sequences and probabilistic models. The second approach for homology search is profile-based homology search. Profile-based homology search is more sensitive than pairwise homology search. Profile-based homology search compares a query sequence to protein domain databases such as Pfam [31], InterProScan [116], FIGfams [72], TIGRFAMs [35], etc. The majority of profile-based homology search tools make use of Profile Hidden Markov Models (pHMMs). Profile Hidden Markov Model is a probabilistic model of multiple sequence alignment. pHMM represents position-specific information embedded in a multiple sequence alignment. In the protein family database such as Pfam, a pHMM [52] is used to represent sequence conservation of each column of a multiple sequence alignment. The example of a widely used package for profile-based homology search is HMMER [29].

1.2.1 The challenges of homology search in large-scale genomic data

1.2.1.1 Next-generation sequencing technology

Sequencing is the method of determining the order of base pairs within a strand of DNA/RNA. Prior to Next-Generation Sequencing (NGS) technologies, Sanger's method dominated the sequencing market for over 30 years [93, 94]. Sanger's method is DNA sequencing method based on chain-terminating inhibitors. However, Sanger sequencing has several limitations. First, Sanger's

method requires a large amount of time and resources to sequence and analyze. Using Sanger's sequencing, the first human genome required 10 years to sequence and three more years to complete the analysis [5, 109]. Due to the high cost of Sanger sequencing, this drawback makes many scientists in small laboratories inaccessible with the budget constraints [60]. Thus, faster, cheaper, and higher throughput technologies are required. To overcome these limitations of Sanger sequencing, next-generation sequencing technologies are developed. NGS technologies are massively parallel. Thus, NGS technologies can produce a large amount of data in a short period of time with low costs [121, 67, 71, 17]. The NGS technologies are dominated by the following platforms commercialized around 2005: Roche 454 Genome Sequencer, the Illumina Genome Analyzer, Solexa and the Life Technologies SOLiD System [121]. Currently, Illumina sequencing is the leader of NGS platform which offers the highest throughput per run and the lowest cost per base [60, 109].

Many NGS technologies can produce single-end and paired-end reads. Single-end sequencing reads a DNA fragment only one end generating a sequence of base pairs. For paired-end reads, the paired-end sequencing enables both ends of a DNA fragment to be sequenced to generate a pair of reads. The main advantage of paired-end sequencing is that the paired-end read contains positional information which is called insert size. Insert size is the length of sequence Read 1 and Read 2 as well as the unknown gap between both ends. Many alignment algorithms can use this information to map the reads to a reference genome leading to more accurate alignment algorithms for short reads [62, 97]. A simplified representation of single-end sequencing and paired-end sequencing is illustrated in Figure 1.3. With advances in next generation sequencing (NGS) technologies, using reads sequenced by NGS technologies enable us to study, analyze, and understand in many aspects of Biological research.

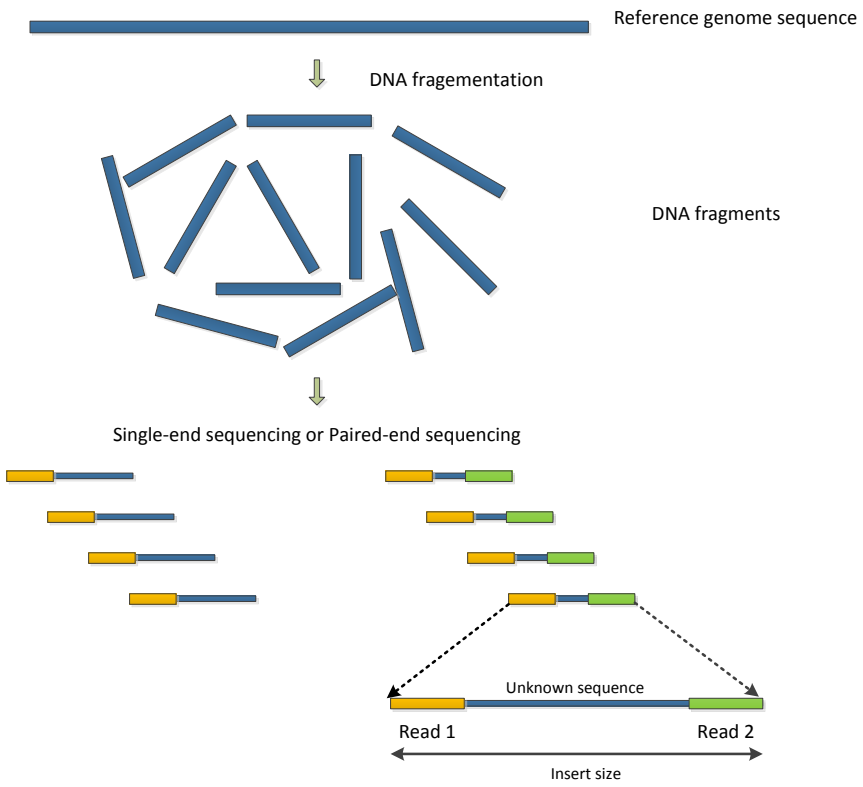


Figure 1.3: Simplified representation of next generation sequencing technology that produce single-end reads or paired-end reads. First, the reference genome sequence is fragmented. For single-end sequencing, one end of DNA fragment is sequenced. For paired-end sequencing, both ends of fragmented DNA are sequenced. Insert size is the length of sequence Read 1 and Read 2 as well as unknown sequence between read pair.

1.2.1.2 The challenges of homology search for next-generation sequencing data

With an increasing number of genomic data (DNA, RNA, and protein sequences), homology search has become an essential task for sequence analysis. One example of large-scale genomic data is next-generation sequencing (NGS) data. Applications of NGS in sequencing the transcriptomes of various non-model species enable us to identify genes and their isoforms that are lowly transcribed or only transcribed in specific tissues or conditions. New homologs of existing gene families and even new gene families are expected by analyzing the data produced by NGS. However, high-throughput data from sequencing machines present new challenges for homology search. Homology search is still the key step to annotate various NGS data. Homology search can be conducted

in two ways. The *de facto* approach is to conduct *de novo* assembly and then apply existing homology search tools to the assembled sequences. However, *de novo* assembly from billions of reads, which are very short and high as well as low-quality reads, poses a significant challenge. Furthermore, datasets of NGS are very large. These large data sets require large memory usage and/or long computational time. These challenges need to be tackled [4, 117, 120]. The second method is to conduct homology search directly to short reads. This method avoids the complicated and error-prone assembly step. However, short reads pose great challenges for homology search. With the increase of read length by the second and third-generation sequencing technologies, new opportunities appear.

However, the short reads sequenced from poorly conserved regions pose a great challenge for existing homology search tools. Therefore, we need to develop the profile-based homology search method to improve sensitivity for short reads.

1.2.1.3 Non-coding RNA homology search in large-scale genomic data

Non-coding RNAs (ncRNAs) have played important roles in many biological processes. ncRNAs are functional molecules that are not translated into protein but function directly as RNAs. There are many different types of ncRNAs such as tRNA, rRNA, miRNA, etc. As ncRNAs perform their functions through both their sequences and secondary structures, these provide useful information for homology search of ncRNAs. An example of a tRNA sequence and its secondary structure are shown in Figure 1.4. The development of NGS technologies enables researchers to sequence genomes and transcriptomes of various model and non-model species, providing unique opportunities for ncRNA mining. Conducting homology search is usually the first step for ncRNA gene finding. There are two types of homology search tools for ncRNA gene finding. The first type of methods focuses on identifying ”known” ncRNAs that are homologous to characterized ncRNA

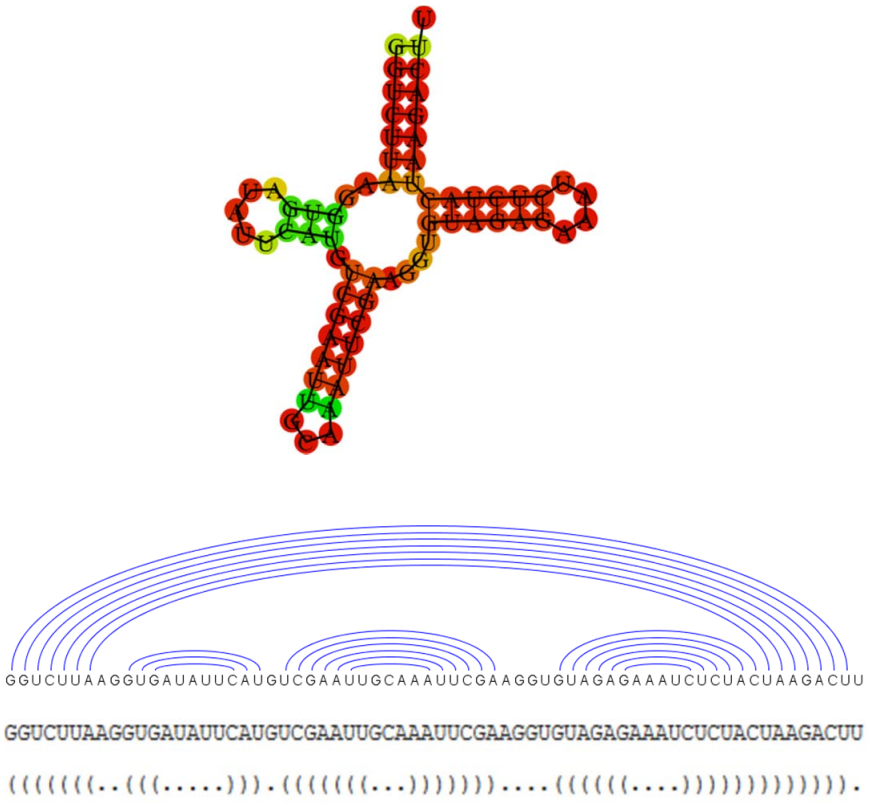


Figure 1.4: A tRNA sequence and its secondary structure

sequences or families. The representative tool that can effectively take advantage of both sequence and structural similarity is based on context-free grammar (CFG) [77]. This method uses a probabilistic model to represent a ncRNA family and aligns a query sequence with the model. The alignment score indicates how likely the query is a member of the ncRNA family. The second type of method is not restricted to known ncRNAs. By comparing two sequences, such as intergenic sequences of two genomes, or unannotated transcripts of two related species, novel ncRNAs might be detected. Usually, secondary structure-aware pairwise comparison tools are applied to ncRNA secondary structure prediction and ncRNA gene finding. Despite the extensive studies of ncRNA gene finding, the tools that heavily depend on the secondary structure in homology search cannot efficiently handle RNA-Seq data that is accumulating rapidly. It will be ideal if we can have a faster ncRNA homology search tool with similar accuracy as those adopting secondary structures.

1.3 Contributions

In this work, we mainly focus on gene prediction and homology search. Gene finding is one of the essential steps for genome annotation. Gene prediction can be applied in the annotation of genomic data produced by genome sequencing methods. For gene prediction tools in grass genomes, existing gene identification tools are not carefully modeled for $5'$-$3'$ G+C gradient. As a consequence, the existing tools have limited sensitivity and accuracy for identifying genes with G+C gradient. Homology search is also essential for protein domain analysis. One of the important steps of functional analysis is conducting homology search or classifying short reads into different types of protein families or protein domain families. The existing tools of profile-based homology search have been successfully used for genome-wide domain analysis. However, reads sequenced from poorly conserved regions may still be missed by existing homology search tools. Thus, there is a need to ”rescue” missing reads and to improve the sensitivity of profile-based homology search for short reads lacking reference genomes.

In the following sections of the dissertation, We present contributions in three areas: gene prediction, homology search, and ncNRA search. First, the gene identification method that accounts for $5'$-$3'$ G+C gradient is presented in Chapter 3. Our method benefits for grass genomes which have been investigated that there is G+C gradient inside exons of a gene. Our tool can increase the accuracy of predicting genes in plant genomes especially grass genomes such as rice, maize, etc. Second, short read homology search using paired-end read information is proposed in Chapter 4. We use an approximate Bayesian approach applying alignment scores and the fragment length distribution to rescue missing end and indicate the true domains. We can improve the accuracy for short reads that are part of remote homologs. Third, Chapter 5 introduces glu-RNA: aliGn highLy strUctured ncRNAs using only sequence similarity which can be applied to ncRNA gene finding.

We describe an accurate ncRNA alignment method to align highly structured ncRNAs using only sequence similarity. We show that we can generate accurate alignments of highly structured ncRNAs without using structural information by incorporating posterior probability and a machine learning approach. We also test our approach on over three hundreds of pairs of highly structured ncRNAs from BRAliBase 2.1.

Chapter 2

a Review of Markov Chain and Hidden Markov Models

2.1 Hidden Markov Models (HMMs)

A Hidden Markov Model (HMM) is a stochastic model which is widely used for many areas in computational biology. In the late 1960s, HMMs were first presented in statistical papers by Baum et al. [10, 9, 22]. HMMs have been applied in speech recognition since the early 1970s [88, 87]. Since the late 1980s, HMMs have been developed in the analysis of biological sequences. HMMs have been applied to various areas of computational biology. They can be used to model a sequence or a family of sequences [27] for protein-profile analysis, prediction of protein coding genes in genome sequences, etc. This section is adapted from Durbin et al. [27]. Markov chains and HMMS are described.

2.1.1 Markov Chains

A Markov model is a finite set of states connected by transitions which can transit from one state to other states. Each transition has a probability. A Markov chain for genomic sequence $X = (x_1, x_2, ..., x_L) \in \Sigma^*$ where $\Sigma = \{A, C, G, T\}$ can be represented as 1) a weighted directed graph, 2) a node represents a state, 3) an edge represents transition between states associated with transition

probability $a_{st} = Pr(x_i = t|x_{i-1} = s)$ with $\sum_{t=\{A,C,G,T\}} a_{ct} = 1$. Figure 2.1 shows an example of a Markov Chain Model. Nodes are states with names A, C, G, and T. Edges are possible transitions associated with transition probabilities. Given a sequence x of length L, we can ask the probability

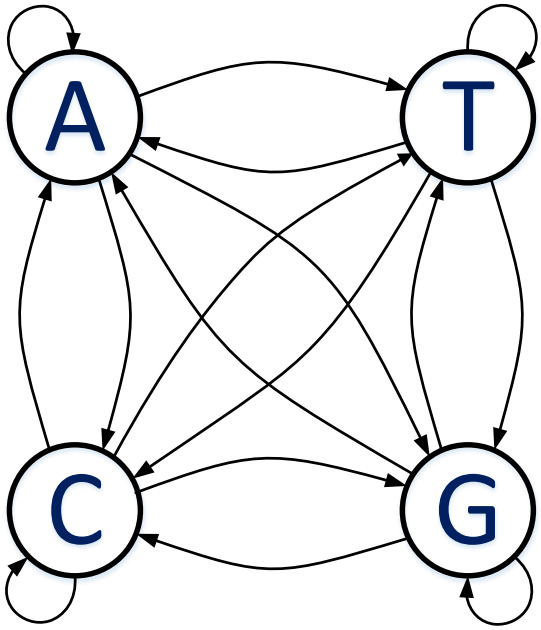


Figure 2.1: An example of Markov Chain

of generating a sequence x, $|x| = L$ given a Markov Chain Model.

$$\begin{aligned} Pr(x) &= Pr(x_L, x_{L-1}, ..., x_1) \\ &= Pr(x_L|x_{L-1}, ..., x_1) Pr(x_{L-1}|x_{L-2}, ..., x_1) ... Pr(x_1) \end{aligned}$$

Key property of a Markov chain (first order): the probability of each symbol x_i depends only

on x_{i-1}.

$$\begin{aligned} Pr(x) &= Pr(x_L|x_{L-1})Pr(x_{L-1}|x_{L-2})...Pr(x_2|x_1)Pr(x_1) \\ &= Pr(x_1)\prod_{i=2}^{L} a_{x_{i-1}x_i} \end{aligned}$$

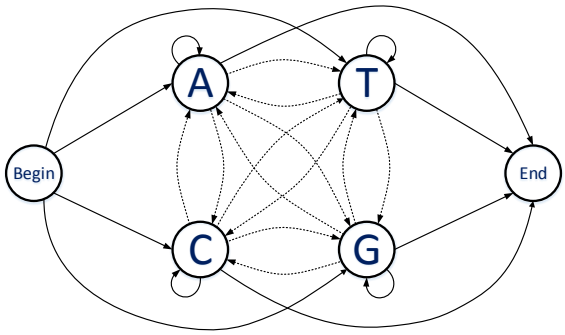


Figure 2.2: An example of Markov Chain. Begin and end states are added to a Markov chain model for modelling the beginning and the ending of a sequence.

Figure 2.2 shows the full Markov chain model modified from Figure 2.1. a *Begin* state and an *End* state are added to the Markov chain model to describe a probability distribution over all possible sequences of any sequence length. The objective of adding $End(E)$ state is that we would like to model sequence length distribution. Let B be the *Begin* state. Assume that $x_0 = B$. Thus, the probability of the first character in a sequence is

$$Pr(x_1 = s) = a_{Bs} = Pr(s) \tag{2.1}$$

where $Pr(s)$ is the background probability of a symbol s. We also model the end state. Let E be the *End* state. Thus, the probability of the ending with state t is

$$Pr(x_L = t) = a_{x_L E} \tag{2.2}$$

2.1.2 Hidden Markov models

Hidden Markov models are the extension models of the classical Markov Chains which output symbols are not the same as the states.

Definition: A Hidden Markov Model (HMM) is defined as a triplet $M = (\Sigma, Q, \Theta)$.

- Σ is an alphabet of symbols such as $\Sigma = \{A, C, G, T\}$
- Q is a finite set of states. The states are capable of emitting symbols from Σ
- Θ is the set of probabilities, including

  State transition probabilities = a_{kl} for each $k, l \in Q$

  Emission probabilities = $e_k(b)$ for each $k \in Q$ and $b \in \Sigma$

Let a path $\Pi = (\pi_1, \pi_2, ..., \pi_L)$ be a sequence of states in the model M. Given a sequence $X = (x_1, x_2, ..., x_L)$ and a path $\Pi = (\pi_1, \pi_2, ..., \pi_L)$. The probability of sequence X and a sequence of states Π is

$$Pr(X, \Pi) = a_{0\pi_1} \prod_{i=1}^{L} e_{\pi_i}(x_i) a_{\pi_i \pi_{i+1}} \tag{2.3}$$

2.1.2.1 The most probable state path: Viterbi Algorithm

As we defined the probability $Pr(X, \Pi)$, we do not know what the state sequence that emitted the given sequence. Thus, the generating path of a given sequence X is hidden.

The decoding problem can be defined as given a HMM $M = (\Sigma, Q, \Theta)$ and a sequence $X \in \Sigma^*$, we want to find the most probable path Π^* between states of a model for X that maximizes $Pr(X, \Pi^*)$

$$\Pi^* = argmax_\pi \{Pr(X, \Pi)\} \tag{2.4}$$

To find the optimal path, we can compute the most probable path in HMM employing an algorithm of dynamic programming called *Viterbi Algorithm*. Suppose $\upsilon_k(i)$ is the probability of the most probable path ending in state k, for sequence $x_1x_2...x_i$ and x_i is generated by state k. Then,

$$\upsilon_l(i+1) = e_l(x_{i+1}) max_{k\in Q} \upsilon_k(i) a_{kl} \tag{2.5}$$

Algorithm: Viterbi algorithm

Input: a HMM $M = (\Sigma, Q, \Theta)$ and a sequence $X \in \Sigma^*$

Output: the most probable path Π^*

Initialization:

$$\upsilon_0(0) = 1 \tag{2.6}$$

$$\upsilon_k(0) = 0, fork > 0 \tag{2.7}$$

Recursion: for $i = 1, ..., L$ and for $l \in Q$

$$\upsilon_l(i) = e_l(x_i) max_{k\in Q} \upsilon_k(i-1) a_{kl} \tag{2.8}$$

$$ptr_i(l) = argmax_{k\in Q} \upsilon_k(i-1) a_{kl} \tag{2.9}$$

Termination:

$$Pr(X, \Pi^*) = max_{k\in Q} \upsilon_k(L) a_{k0} \tag{2.10}$$

$$\pi_L^* = argmax_{k\in Q} \upsilon_k(L) a_{k0} \tag{2.11}$$

Traceback: for $i = L, ..., 1$

$$\pi_{i-1}^* = ptr_i(\pi_i^*) \tag{2.12}$$

2.1.2.2 The Forward Algorithm

The evaluation problem can be defined as given a HMM $M = (\Sigma, Q, \Theta)$ and a sequence $X \in \Sigma^*$, we want to compute the probability $Pr(X|M)$.

This probability can be efficiently calculated using *Forward Algorithm*. This algorithm is similar to Viterbi algorithm by replacing maximization steps by sums. The forward variable $f_k(i)$ can be defined as

$$f_k(i) = Pr(x_1 x_2 ... x_i, \pi_i = k) \tag{2.13}$$

Here, $f_k(i)$ is the probability of the observed sequence $x_1 x_2 ... x_i$, requiring that $\pi_i = k$ (x_i is generated by π_i). The recursion is as follow.

$$f_l(i+1) = e_l(x_{i+1}) \Sigma_{k \in Q} f_k(i) a_{kl} \tag{2.14}$$

Algorithm: Forward algorithm

Input: a HMM $M = (\Sigma, Q, \Theta)$ and a sequence $X \in \Sigma^*$

Output: the probability $Pr(X|M)$

Initialization ($i = 0$):

$$f_0(0) = 1 \tag{2.15}$$

$$f_k(0) = 0, fork > 0 \tag{2.16}$$

Recursion: for $i = 1, ..., L$ and for $l \in Q$

$$f_l(i) = e_l(x_i) \Sigma_{k \in Q} f_k(i-1) a_{kl} \tag{2.17}$$

Termination:

$$Pr(X) = \Sigma_{k \in Q} f_k(L) a_{k0} \tag{2.18}$$

2.1.2.3 The Backward Algorithm

For the backward algorithm, the variable of backward is similar to the forward variable. However, the backward calculation is different from the calculation of forward. Instead of calculating from the beginning of a sequence, the backward recursion is started at the end of the sequence. The backward variable $b_k(i)$ can be defined as

$$b_k(i) = Pr(x_{i+1} x_{i+2} ... x_L, \pi_i = k) \tag{2.19}$$

where $b_k(i)$ is the probability of the observed sequence $x_{i+1} x_{i+2} ... x_L$, given that $\pi_i = k$

Algorithm: Backward algorithm

Input: a HMM $M = (\Sigma, Q, \Theta)$ and a sequence $X \in \Sigma^*$

Output: the probability $Pr(X|M)$

Initialization ($i = L$): for all k

$$b_k(L) = a_{k0} \tag{2.20}$$

Recursion: for $i = L-1, ..., 1$ and for $k \in Q$

$$b_k(i) = \Sigma_{l \in Q} a_{kl} e_l(x_{i+1}) b_l(i+1) \tag{2.21}$$

Termination:

$$Pr(X) = \Sigma_{l \in Q} a_{0l} e_l(x_1) b_l(1) \tag{2.22}$$

Chapter 3

AUGUSTUS-GC: a Gene Prediction Model accounting for 5′-3′ G+C Gradient

3.1 Introduction

Computational gene prediction plays an important role in the annotation of newly sequenced genomes. The goal of gene prediction is to identify the location and structure of protein-coding genes in genomic sequences. The identification and annotation of genes in genomic DNA sequences will help us detect and understand the essential functional elements in a sequenced genome.

Computational gene prediction methods can be broadly divided into two main categories: *ab initio* methods and homology-based methods. The first category contains *ab initio* gene prediction tools that can predict genes using the query sequence as the only input sequence. Major *ab initio* gene prediction tools use Hidden Markov Models (HMMs), which are the probabilistic models that can represent the gene structure. HMMs can label different segments in an unknown sequence as gene structure elements such as intergenic region, exons, and introns. Given a sequence, we can use HMMs to infer the most probable path corresponding to an annotation of gene structure. The examples of *ab initio* gene prediction tools include GENSCAN [14, 15], GENEID [82], HMMGene [50], GeneMark.hmm [61], GlimmerHMM [66], FGENESH [91], SNAP [47], and AUGUSTUS [98] et al. The second category contains comparative gene prediction tools, which

compare a query sequence with homologous sequences of related species and employ their sequence similarity for gene annotation. The examples of the second group include GENEWISE [12], GENOMESCAN [113], AGenDA [105, 75], TWINSCAN [48], SGP2 [81], DOUBLESCAN [73], CEM [8], SLAM [1], etc.

Using more information such as homologous sequences has potential to produce better results but requires a closely-related species. Similarity-based methods perform not very well when the target sequence has no homology. Thus, *ab initio* gene prediction tools play a significant role to find novel genes in the sequences without a priori known homologies.

As the base composition and the exon length distributions can differ significantly for genes with different G+C contents, most gene prediction tools employ G+C content-dependent training [98, 101, 100, 14, 15]. Specifically, training genes are divided into several subsets according to their G+C content (e.g. low, medium, and high). Then different sets of model parameters are derived from different subsets of training data. For an input sequence, the gene prediction tool chooses the parameter set with mean G+C content closest to the input sequence [98]. Both theoretical analysis and empirical results have shown that G+C content-dependent training greatly improves the gene prediction accuracy and sensitivity.

However, these tools have two limitations for gene prediction in grass genomes such as rice, barley, maize, sorghum, and so on. First, many grass genes exhibit a sharp $5'$-$3'$ decreasing G+C content gradient [43, 90], which is not carefully modeled by existing gene prediction tools. As a result, these tools have limited sensitivity and accuracy for predicting genes with G+C gradients. Second, unlike many animal genomes, G+C content of genes in plant genomes are not well correlated with the context G+C content (i.e. lacking isochore structures) [28]; it is thus not trivial to determine which parameter set is optimal for predicting genes in an input sequence. To address these limitations, we propose a new gene prediction model with two advantages: 1) our model

can predict genes with G+C gradient with higher sensitivity and accuracy; 2) our unified model is optimized for genes of variant G+C content and $5'$-$3'$ changing patterns.

3.2 Method

In this section, we describe AUGUSTUS-GC, a tool that predicts genes with $5'$-$3'$ G+C gradient. The main novelty of our method is a modified hidden Markov model (HMM) that distinguishes exons of different G+C content. The HMM has a similar topology to the one used in AUGUSTUS [101, 100] and many other de novo gene prediction tools [14, 50]. But the major difference is that our model is designed to handle various G+C content and $5'$-$3'$ changing patterns inside coding regions.

3.2.1 The Hidden Markov Model of AUGUSTUS-GC

We modify the generalized Hidden Markov Model (GHMM) from AUGUSTUS [101], an *ab initio* gene prediction tool. The AUGUSTUS-GC is similar to AUGUSTUS *ab initio* gene prediction model. However, instead of using one state to represent an exon, we have three states to model exons of high, medium, and low G+C contents. Figure 3.1 and Figure 3.2 illustrate the difference of states between the existing AUGUSTUS HMM and HMM of AUGUSTUS-GC for gene prediction.

Here, we make a reasonable assumption that the G+C content change inside exons of multi-exon genes is relatively small. This is the tradeoff between model complexity and the resolution of the model. For single exon genes, three states ($E^{\mathrm{H}}_{\mathrm{single}}$, $E^{\mathrm{M}}_{\mathrm{single}}$, $E^{\mathrm{L}}_{\mathrm{single}}$) are created. For initial exon, three states ($E^{0}_{\mathrm{init\ H}}$, $E^{0}_{\mathrm{init\ M}}$, $E^{0}_{\mathrm{init\ L}}$) are used to model exons of high, medium, and low G+C content. Moreover, the initial exons of other phases, the internal exons of all phases, and the terminal exon

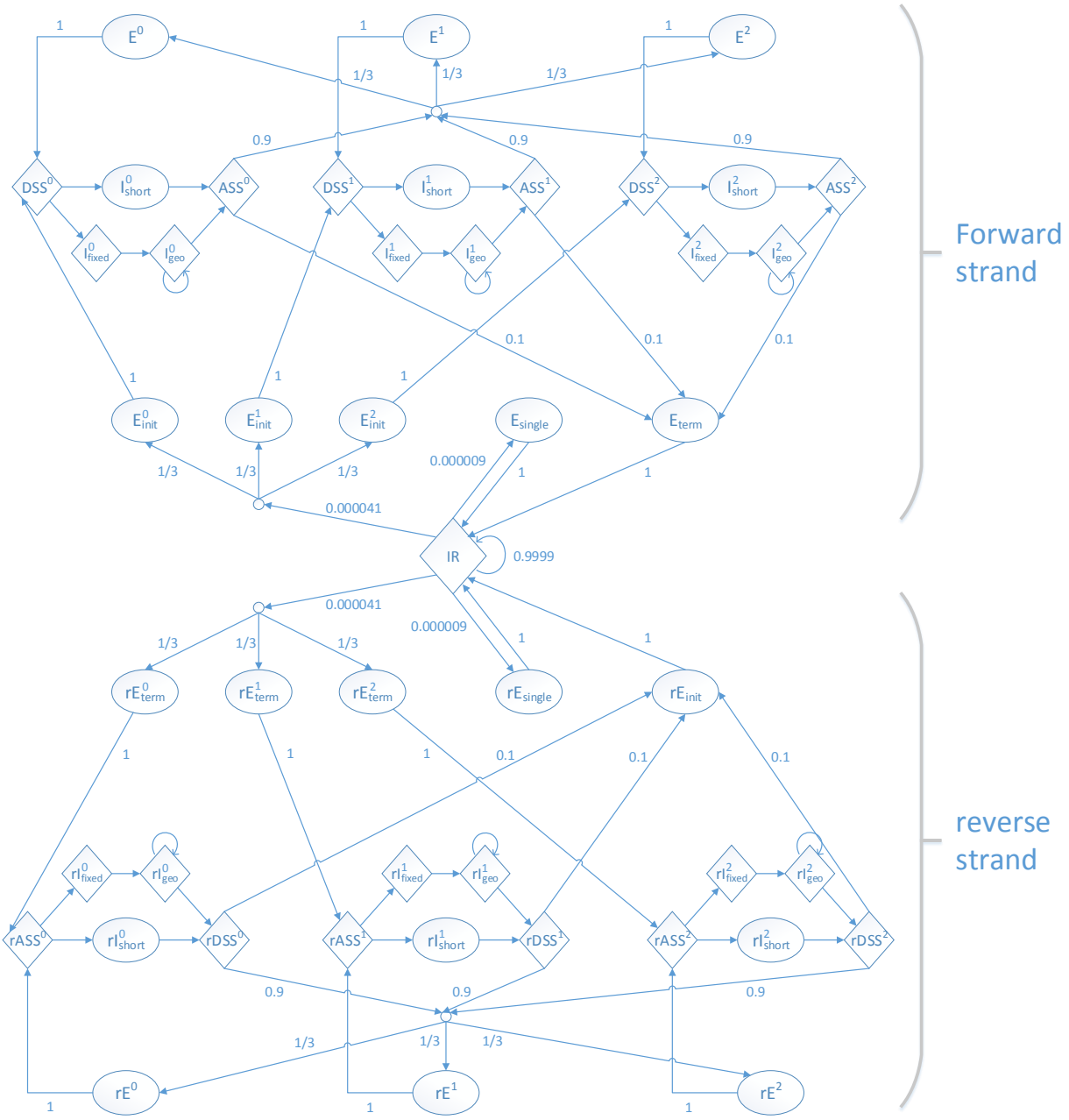


Figure 3.1: The state diagram of *ab initio* AUGUSTUS [101].The states beginning with r represents the reverse strand. E_{single}: a single exon. E_{init} : the initial coding exon of a multi-exon gene. *DSS*: a donor splice site. I_{short}: an intron emitting at most d nucleotides. I_{fixed}: a longer intron with the first d nucleotides. I_{geo}: a longer intron emitting one nucleotide at a time after the first d nucleotides. *ASS*: an acceptor splice site with branch point. E: an internal coding exon of a multi-exon gene. E_{term}: the last coding exon of a multi-exon gene. *IR*: intergenic region. Diamonds represent the states that emit fixed length strings. Ovals represent the states including explicit length distribution. The numbers at the arrows show the transition probabilities. The exponents 0, 1, and 2 represent the reading frame phase. For an exon state, this is the position of the last base of the exon in its codon. For the other states, the exponent are the preceding-exon phase. The small circles represent silent states.

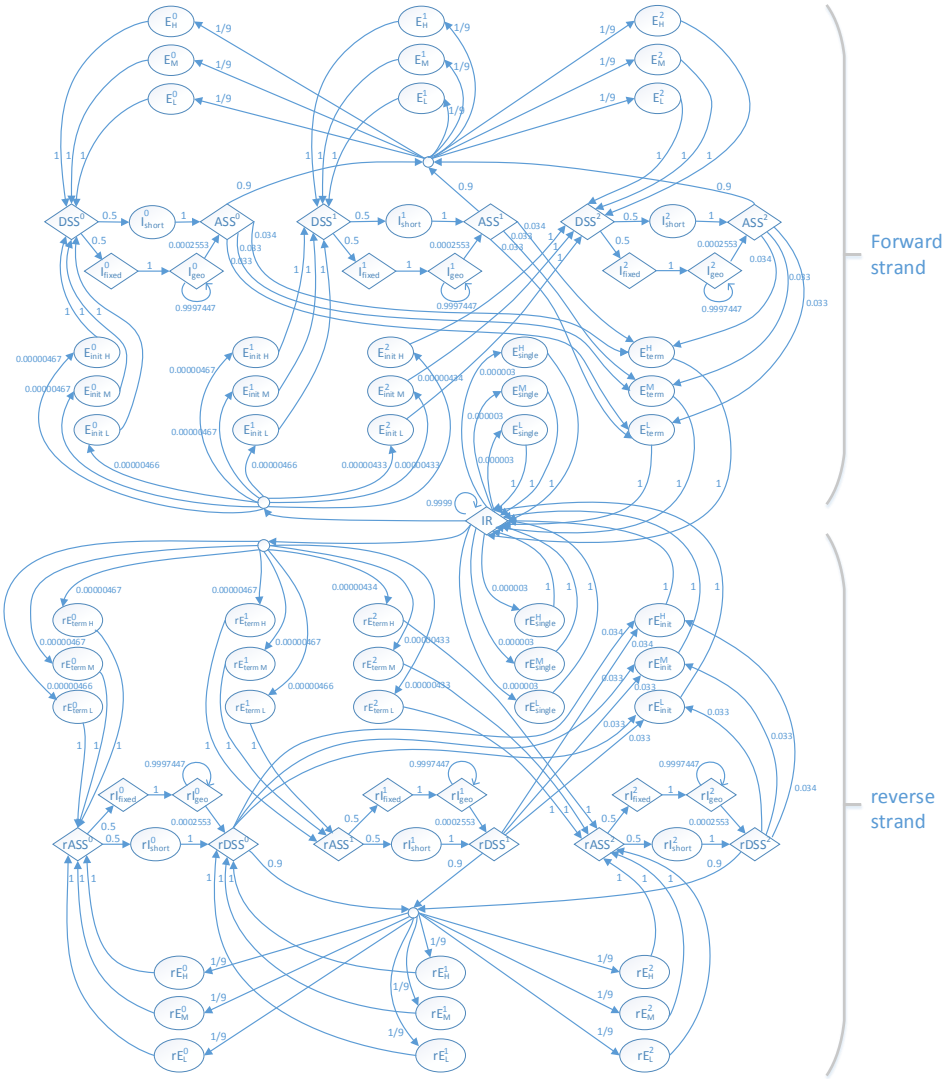


Figure 3.2: The state diagram of AUGUSTUS-GC. The states beginning with r represents the reverse strand. $E^{\mathrm{H}}_{\mathrm{single}}$: a single exon of high G+C content. $E^{\mathrm{M}}_{\mathrm{single}}$: a single exon of medium G+C content. $E^{\mathrm{L}}_{\mathrm{single}}$: a single exon of low G+C content. $E_{\mathrm{init\,H}}$: the initial coding exon of a multi-exon gene with high G+C content. $E_{\mathrm{init\,M}}$: the initial exon of a multi-exon gene with medium G+C content. $E_{\mathrm{init\,L}}$: the initial exon of a multi-exon gene with low G+C content. *DSS*: a donor splice site. I_{short}: an intron emitting at most d nucleotides. I_{fixed}: a longer intron with the first d nucleotides. I_{geo}: a longer intron emitting one nucleotide at a time after the first d nucleotides. *ASS*: an acceptor splice site with branch point. E_{H}: an internal coding exon of a multi-exon gene with high G+C content. E_{M}: the internal exon of a multi-exon gene with medium G+C content. E_{L}: the internal exon of a multi-exon gene with low G+C content. $E^{\mathrm{H}}_{\mathrm{term}}$: the last coding exon of a multi-exon gene with high G+C content. $E^{\mathrm{M}}_{\mathrm{term}}$: the terminal exon of a multi-exon gene with medium G+C content. $E^{\mathrm{L}}_{\mathrm{term}}$: the terminal exon of a multi-exon gene with low G+C content. *IR*: intergenic region. Diamonds represent the states that emit fixed length strings. Ovals represent the states including explicit length distribution. The numbers at the arrows show the transition probabilities. The exponents 0, 1, and 2 represent the reading frame phase. For an exon state, this is the position of the last base of the exon in its codon. For the other states, the exponent are the preceding-exon phase. The small circles represent silent states.

all have three states for high, medium, and low G+C content. Genes of variant G+C changing patterns can be represented by the new exon states.

The added exon states allow the HMM to predict genes of various G+C content and changing patterns with higher accuracy. For example, genes of negative G+C gradient tend to be represented by a path starting with E_{H} and ending with E_{L}. Genes with high G+C content and moderate gradient tend to be produced by a path mainly consisting of E_{H}.

For purpose of gene identification, the states of AUGUSTUS-GC in the state set Q are consistent with meaning of biology such as intron, exon, and splice site, etc. We allow a state transiting to other states in biologically meaningful ways. For example, an intron must follow an donor splice site. the model of AUGUSTUS-GC has the following 79 states:

$$\begin{aligned} Q = &\{IR, E^{\mathrm{H}}_{\mathrm{single}}, E^{\mathrm{M}}_{\mathrm{single}}, E^{\mathrm{L}}_{\mathrm{single}}, E^{\mathrm{H}}_{\mathrm{term}}, E^{\mathrm{M}}_{\mathrm{term}}, E^{\mathrm{L}}_{\mathrm{term}}, rE^{\mathrm{H}}_{\mathrm{single}}, rE^{\mathrm{M}}_{\mathrm{single}}, rE^{\mathrm{L}}_{\mathrm{single}}, rE^{\mathrm{H}}_{\mathrm{init}}, rE^{\mathrm{M}}_{\mathrm{init}}, rE^{\mathrm{L}}_{\mathrm{init}}\} \bigcup \\ &\bigcup_{i=0}^{2} \{E^{\mathrm{i}}_{\mathrm{init\ H}}, E^{\mathrm{i}}_{\mathrm{init\ M}}, E^{\mathrm{i}}_{\mathrm{init\ L}}, DSS^{\mathrm{i}}, I^{\mathrm{i}}_{\mathrm{short}}, I^{\mathrm{i}}_{\mathrm{fixed}}, I^{\mathrm{i}}_{\mathrm{geo}}, ASS^{\mathrm{i}}, E^{\mathrm{i}}_{\mathrm{H}}, E^{\mathrm{i}}_{\mathrm{M}}, E^{\mathrm{i}}_{\mathrm{L}}\} \bigcup \\ &\bigcup_{i=0}^{2} \{rE^{\mathrm{i}}_{\mathrm{term\ H}}, rE^{\mathrm{i}}_{\mathrm{term\ M}}, rE^{\mathrm{i}}_{\mathrm{term\ L}}, rDSS^{\mathrm{i}}, rI^{\mathrm{i}}_{\mathrm{short}}, rI^{\mathrm{i}}_{\mathrm{fixed}}, rI^{\mathrm{i}}_{\mathrm{geo}}, rASS^{\mathrm{i}}, rE^{\mathrm{i}}_{\mathrm{H}}, rE^{\mathrm{i}}_{\mathrm{M}}, rE^{\mathrm{i}}_{\mathrm{L}}\} \end{aligned} \tag{3.1}$$

Figure 3.2 shows the states in the state set Q. In the upper half of Figure 3.2, the states represent protein-coding genes on the forward strand. The state IR stands for the intergenic region. The diamond-shaped states are the states emitting fixed-length strings at a time. The oval-shaped states are the states emitting variable-length strings. The state $E^{\mathrm{H}}_{\mathrm{single}}$, $E^{\mathrm{M}}_{\mathrm{single}}$, and $E^{\mathrm{L}}_{\mathrm{single}}$ represent a gene containing just only one exon of high, medium, and low G+C contents. Other states in the upper half associate with genes with several exons. Regarding the reading frame position, three states for each type of exon of high, medium, and low G+C contents are shown except the terminal exon $E^{\mathrm{H}}_{\mathrm{term}}$, $E^{\mathrm{M}}_{\mathrm{term}}$, and $E^{\mathrm{L}}_{\mathrm{term}}$. The initial exon states $E^{\mathrm{i}}_{\mathrm{init\ H}}$, $E^{\mathrm{i}}_{\mathrm{init\ M}}$, $E^{\mathrm{i}}_{\mathrm{init\ L}}$ for $i = 0, 1$, and 2 are related to

the first coding exon of a gene. The internal exon states $E_{\mathrm{H}}^{\mathrm{i}}$, $E_{\mathrm{M}}^{\mathrm{i}}$, $E_{\mathrm{L}}^{\mathrm{i}}$ for $i = 0, 1$, and 2 are related to the interior coding exons of a gene. The superscript i for $i = 0, 1$, and 2 denotes the phase of reading frame where the exon ends. The superscript 0 means that the end of the exon is at the boundary of a complete codon. The superscript 1 represents that the boundary of codon is 1 position prior to the exon end. The superscript 2 means the boundary of codon is 2 positions prior to the exon end. DSS^{k} $(0 \leq k \leq 2)$ is the state of donor splice site. ASS^{k} $(0 \leq k \leq 2)$ represent the state of acceptor splice site. $I_{\mathrm{short}}^{\mathrm{k}}$ $(0 \leq k \leq 2)$ is the state of intron with the length at most d bases. $I_{\mathrm{fixed}}^{\mathrm{k}}$ is the state of a longer intron with the first d bases. $I_{\mathrm{geo}}^{\mathrm{k}}$ is the state of a longer intron with emitting single nucleotides at a time after the first d bases. The superscript k denotes the phase of reading frame of previous exon. In the lower half of Figure 3.2, the states represent protein-coding genes on the reverse strand. For each state of reverse strand, the name begins with 'r'. They have the consistent biological meaning with the states on the forward strand. The superscript on the reverse strand represents the phase of the reading frame at the rightmost position of exon. Thus, they have nine states for a terminal exon and three states for an initial exon considering high, medium, and low G+C contents.

In Figure 3.2, the arrows represent the transitions between states in the state set Q having non-zero probabilities. The transitions from and to the intron states are the same as those of states described in the AUGUSTUS model [101]. Figure 3.1 shows the state model of original AUGUSTUS [101]. The transition probabilities are fixed values given in Figure 3.1 rounded up to 6 decimal places. For AUGUSTUS-GC, we consider two strategies for computing transition probabilities. First, we use a very simple strategy by dividing the old probabilities of AUGUSTUS equally for three states of high, medium, and low G+C contents. This strategy can be used when we have very limited training data. Our hypothesis is that they start with equal probabilities. Figure 3.2 presents the transition probabilities of first strategy. The second strategy is a more standard method.

We consider training data and compute the transition probabilities using the maximum likelihood estimation. Let a_{kl} be the transition probability for $k, l \in Q$, A_{kl} be the number of times that transits from the state k to state l in training data. The maximum likelihood estimator is defined as

$$a_{kl} = \frac{A_{kl}}{\sum_{q \in Q} A_{kq}} \tag{3.2}$$

For some states, there are many transitions. When we have many transitions, many parameters, we need a large number of training samples. Otherwise, the training is very biased. To avoid zero probabilities due to sparse/insufficient training data, we use pseudocounts by adding one extra counts to the observed frequencies to reflect prior biases regarding the probability values. Given pseudocounts r_{kl}, we define A'_{kl} as

$$A'_{kl} = A_{kl} + r_{kl} \tag{3.3}$$

Usually, in Laplace method, all r_{kl} equal to 1.

The performance comparisons of two strategies for computing transition probabilities are shown in Experiment Results section.

For each exon state on both forward strand and reverse strand, the exon length distribution of $E^{\mathrm{H}}_{\mathrm{single}}$, $E^{\mathrm{M}}_{\mathrm{single}}$, $E^{\mathrm{L}}_{\mathrm{single}}$, $E^{\mathrm{i}}_{\mathrm{init\ H}}$, $E^{\mathrm{i}}_{\mathrm{init\ M}}$, $E^{\mathrm{i}}_{\mathrm{init\ L}}$, $E^{\mathrm{i}}_{\mathrm{H}}$, $E^{\mathrm{i}}_{\mathrm{M}}$, $E^{\mathrm{i}}_{\mathrm{L}}$, $E^{\mathrm{H}}_{\mathrm{term}}$, $E^{\mathrm{M}}_{\mathrm{term}}$, $E^{\mathrm{L}}_{\mathrm{term}}$, $rE^{\mathrm{H}}_{\mathrm{single}}$, $rE^{\mathrm{M}}_{\mathrm{single}}$, $rE^{\mathrm{L}}_{\mathrm{single}}$, $rE^{\mathrm{i}}_{\mathrm{term\ H}}$, $rE^{\mathrm{i}}_{\mathrm{term\ M}}$, $rE^{\mathrm{i}}_{\mathrm{term\ L}}$, $rE^{\mathrm{i}}_{\mathrm{H}}$, $rE^{\mathrm{i}}_{\mathrm{M}}$, $rE^{\mathrm{i}}_{\mathrm{L}}$, $rE^{\mathrm{H}}_{\mathrm{init}}$, $rE^{\mathrm{M}}_{\mathrm{init}}$, and $rE^{\mathrm{L}}_{\mathrm{init}}$ for $i = 0, 1$, and 2 are computed on the corresponding exons, respectively. Similarly, different inhomogeneous kth-order Markov models (by default k=4) for each exon state are derived separately.

3.3 Experimental Results

To evaluate the performance of AUGUSTUS-GC, we tested AUGUSTUS-GC on three different test sets in plant genomes: *Arabidopsis Thaliana* [7], *Oryza Sativa* obtained from Childs Lab, Department of Plant Biology, Michigan State University and *Oryza Sativa* provided by Stanke et al. [99]. We compared our results to the regular AUGUSTUS *ab initio* gene finding program. Our method is designed for finding genes with G+C gradient. It has been investigated that many grass genes present a sharp $5'$-$3'$ decreasing gradient of G+C content.

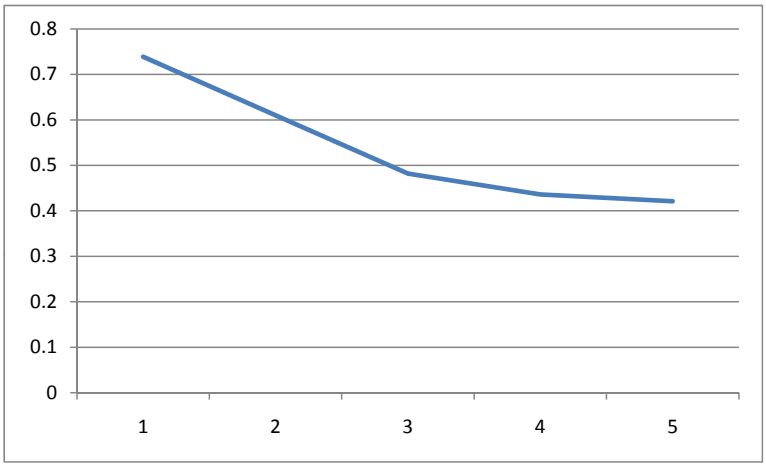


Figure 3.3: The example of a gene with G+C content gradient in *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content.

Figure 3.3 shows an example of descending slope of G+C content in *Oryza Sativa* data set by starting with high G+C content and ending with low G+C content. We also tested AUGUSTUS-GC on *Arabidopsis Thaliana* [7]. *Arabidopsis Thaliana* is not grass genome. It may not exhibit G+C content gradient. However, *Arabidopsis Thaliana* has a good quality of genome annotation. Thus, we focus on testing AUGUSTUS-GC in both *Arabidopsis Thaliana* and *Oryza Sativa*. We expect to see a bigger improvement of gene prediction performance in *Oryza Sativa* than in *Arabidopsis Thaliana*.

To quantify the performance of gene prediction tools, we evaluated the gene prediction accu-

racy by using two measures: sensitivity and specificity. We quantified the performance of gene prediction programs at three different levels: the nucleotide level, the exon level, and the gene level. We adopted commonly used metrics from Stanke and Waack [101]. The sensitivity and the specificity are defined as

$$Sensitivity(Sen) = \frac{TP}{TP+FN} \tag{3.4}$$

$$Specificity(Spe) = \frac{TP}{TP+FP} \tag{3.5}$$

where TP represents the number of correctly predicted features (coding nucleotides, exons, or genes). FN represents the number of annotated features that are not the predicted features. FP represents the number of predicted features that are not the annotated features. At each level, two basic metrics are being used. Sensitivity is the proportion of correctly predicted features in the set of all annotated features. Specificity is the proportion of correctly predicted features in the set of all predicted features. At the exon level, a predicted exon will be correct if both splice sites are identical to the true position of an exon. At the gene level, a predicted gene is considered correctly predicted if all coding exons are correctly identified, and no additional exons are identified in the gene. The predicted partial genes are considered as the predicted genes. The forward and reverse strands are considered as different sequences. Officially, the definition of specificity is defined as

$$Specificity(Spe) = \frac{TN}{TN+FP} \tag{3.6}$$

However, because the amount of non-coding nucleotides in a DNA sequence is much greater than the amount of coding nucleotides, TN becomes much larger than FP. Hence, specificity in gene prediction has typically been calculated by the first definition of specificity.

3.3.1 Gene Prediction in *Arabidopsis Thaliana*

To show the utility of AUGUSTUS-GC, the training data set was downloaded from Stanke's website [7]. We trained the model on *Arabidopsis Thaliana* data set. The data set contained 249 genbank sequences. There were two single exon genes and 247 multi-exon genes in the data set.

3.3.1.1 Training the HMM model

To train our HMM, we calculated G+C contents for all of the exons and classified them as high, medium, and low using specified cutoffs. For AUGUSTUS-GC, we have two cutoffs: LowT and highT. An exon will be classified as low G+C content group if the G+C content of exon is lower than specified LowT. Similarly, an exon will be classified as high G+C content group if the G+C content of exon is above the specified highT. An exon will be classified as medium G+C content group if the G+C content of exon is higher than or equal to specified lowT and lower than specified highT. Table 3.1 shows lowT and highT that we used in the experiments. There are three groups of exons according to G+C contents that we classified. For each group, it had different parameter sets. We used 10-fold cross-validation. It divided the data set randomly into 10 subsets. The evaluation method is repeated 10 times. Each round, one of 10 subsets is evaluated as the test set and the other nine subsets are put together for training. Then the average prediction accuracy of all 10 trials is calculated.

For computing transition probabilities, we tested on two strategies. First, we equally divided the probabilities of AUGUSTUS for three states of high, medium, and low G+C contents. The results by using the first strategy are shown in Table 3.1. Second, we used maximum likelihood estimation to calculate transition probabilities. The results by using the second strategy for computing the transition probabilities are shown in Table 3.2. Moreover, the length distribution for each new

exon state was computed. Correspondingly, we calculated kth-order Markov Model (by default k=4) for each new exon state. Figure 3.4 illustrates the distribution of exons by their G+C content for 1431 exons in *Arabidopsis Thaliana* data set.

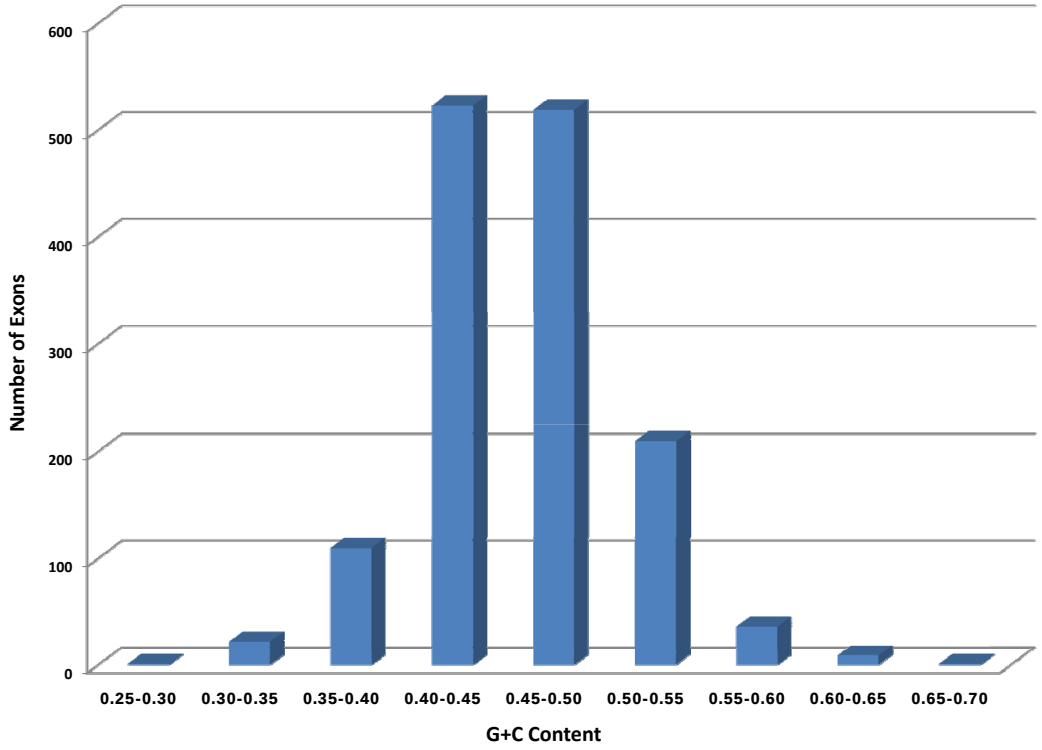


Figure 3.4: G+C Content of *Arabidopsis Thaliana* data set

3.3.1.2 Performance comparison between different gene prediction tools

We tested original AUGUSTUS and AUGUSTUS-GC on the testing data set of *Arabidopsis Thaliana* [7]. The testing data set was different from the training data set. There were 74 genbank sequences with 168 genes on forward and reverse strand. Our program was modified from AUGUSTUS version 2.4 downloaded from [6]. Both original AUGUSTUS and AUGUSTUS-GC were tested using default input parameters.

Table 3.1 shows the comparison of the accuracy of AUGUSTUS and AUGUSTUS-GC with different thresholds of G+C contents on the set of *Arabidopsis Thaliana*. We used the first strat-

Table 3.1: Performance comparison of gene prediction tools on *Arabidopsis Thaliana*. The transition probabilities were divided into three equal portions. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS.

Program		AUGUSTUS	AUGUSTUS-GC			
			lowT=0.47 highT=0.63	lowT=0.30 highT=0.60	lowT=0.30 highT=0.70	lowT=0.60 highT=0.70
Base	Sen	0.968	0.962	0.963	0.962	0.963
level	Spe	0.708	**0.709**	**0.71**	**0.71**	**0.71**
Exon	Sen	0.87	0.848	0.848	0.845	0.848
level	Spe	0.666	**0.669**	**0.679**	**0.68**	**0.679**
Gene	Sen	0.554	**0.565**	0.548	0.548	0.548
level	Spe	0.352	**0.36**	**0.354**	**0.354**	**0.354**

Table 3.2: Performance comparison of gene prediction tools on *Arabidopsis Thaliana*. The transition probabilities was trained by computing maximum likelihood estimation. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS.

Program		AUGUSTUS	AUGUSTUS-GC			
			lowT=0.47 highT=0.63	lowT=0.30 highT=0.60	lowT=0.30 highT=0.70	lowT=0.60 highT=0.70
Base	Sen	0.968	0.96	**0.972**	**0.972**	**0.972**
level	Spe	0.708	**0.711**	**0.709**	**0.709**	**0.709**
Exon	Sen	0.87	0.851	**0.882**	**0.882**	**0.882**
level	Spe	0.666	**0.674**	**0.677**	**0.677**	**0.677**
Gene	Sen	0.554	**0.56**	**0.565**	**0.565**	**0.565**
level	Spe	0.352	0.346	0.351	0.351	0.351

egy for computing the transition probabilities. The transition probabilities were equally separated into three states of high, medium, and low. These experimental results show that AUGUSTUS-GC achieved better sensitivity and specificity for gene level. For base level and exon level, AUGUSTUS-GC has higher specificity than AUGUSTUS. In addition, we showed different cutoffs for AUGUSTUS-GC. Changing cutoffs for G+C contents can improve the performance.

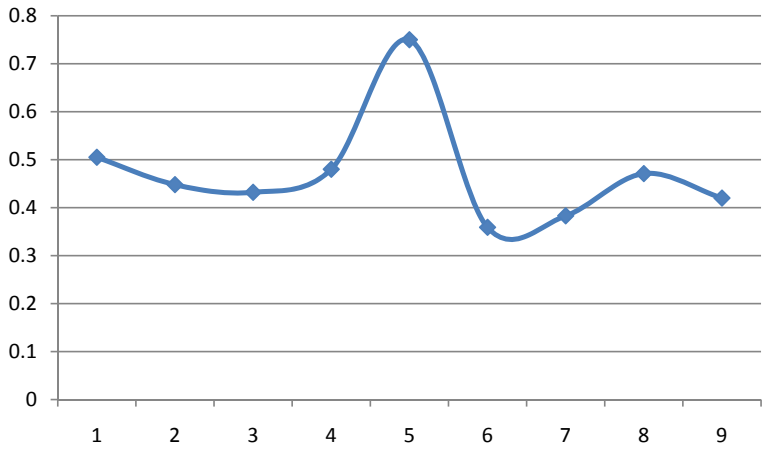


Figure 3.5: An example of a predicted gene with G+C content gradient of *Arabidopsis Thaliana* data set. This gene was predicted by AUGUSTUS-GC. X-axis represents exon index. Y-axis represents G+C content.

We further implemented the AUGUSTUS-GC by computing the transition probabilities based on the training set. The transition probabilities from intron to different exon states were computed using maximum likelihood estimation. Table 3.2 presents the comparison of accuracy of AUGUSTUS and AUGUSTUS-GC with different cutoffs and trained transition probabilities. Using maximum likelihood estimation for computing the transition probabilities gave the better overall performance. We conducted additional analysis using the results of lowT=0.30 and highT=0.60. We observed that, for some sequences, we can predict more genes and we can identify correctly position. For example, we can detect the gene in Figure 3.5 because it contained G+C content gradient. However, the regular AUGUSTUS cannot identify it correctly. Moreover, we compared the G+C change between the uniquely found genes by our tool and the common ones shared by AU-

GUSTUS and AUGUSTUS-GC. The results showed that there were 14 uniquely identified genes by our tool and 149 shared genes. We further examine whether our tool can find genes with more G+C change. We computed the standard deviation (*SD*) and the distance between the exon that has highest G+C content and the exon that has lowest G+C content for each protein-coding gene and reported the average of all genes. The experimental results demonstrated that the average *SD* of the uniquely predicted genes was 0.046. However, the average *SD* of the common ones was 0.033 which is smaller than the uniquely predicted genes. Also, the average distance between the exon that has highest G+C content and the exon that has lowest G+C content for each protein-coding gene of the uniquely found genes was 0.111 but the average distance of the common genes was 0.087. The average distance of our tool is wider than that of common ones.

3.3.2 Gene Prediction in *Oryza Sativa*

For *Oryza Sativa* data sets, we conducted two experiments. First *Oryza Sativa* data set was provided by Childs Lab, Department of Plant Biology, Michigan State University. Second *Oryza Sativa* data set was obtained from Stanke et al. [99]. These two data sets were constructed from different ways. The ground truth sequences of two data sets were different. The first data set is smaller than the second data set. Figure 3.6 presents the distribution of exons by their G+C content for 844 exons in first *Oryza Sativa* data set. Figure 3.7 shows the distribution of exons by their G+C content for 16199 exons in second *Rice* data set. According to Figure 3.6 and Figure 3.7, the G+C content change for the exons has wider range than that of *Arabidopsis Thaliana*. Our hypothesis is that in some grass genes such as rice, they have G+C gradient of exons rather than consistent. Thus, we conducted the experiments to test how our tool performed to handle the change of G+C content for each separated exons.

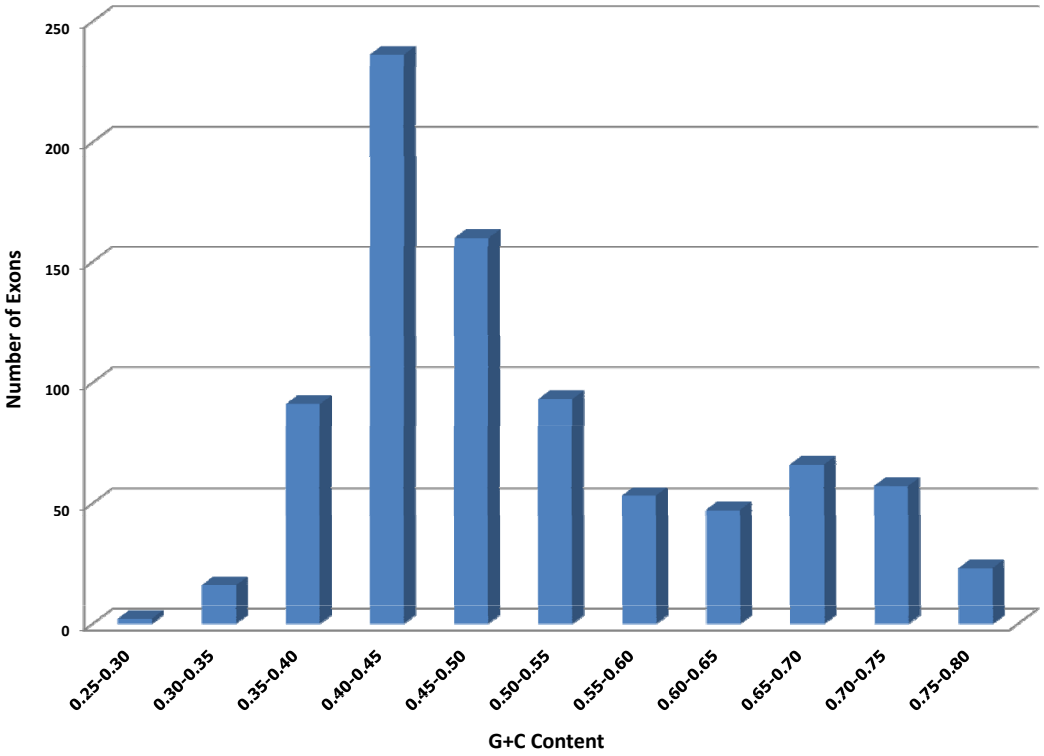


Figure 3.6: G+C Content of the first data set of *Oryza Sativa*

3.3.2.1 Gene Identification in *Oryza Sativa* obtained from Childs Lab, Department of Plant Biology, Michigan State University

In the first experiment of *Oryza Sativa* data set, we tested the performance of gene finding tools in *Oryza Sativa* data set obtained from Childs Lab, Department of Plant Biology, Michigan State University. This data set is a part of MSU Rice Genome Annotation Project [79, 114]. The training data set consisted of a set of 150 genbank sequences with 11 single-exon genes and 139 multi-exon genes on forward strand and reverse strand. Again, we used 10-fold cross validation strategy for training.

We examined the accuracy of gene prediction methods. There were 150 genbank sequences with 13 single exon genes and 137 multi-exon genes on forward and reverse strand for testing data set. In grass genomes, the sharp $5'$-$3'$ decreasing G+C content gradient is exhibited as we showed

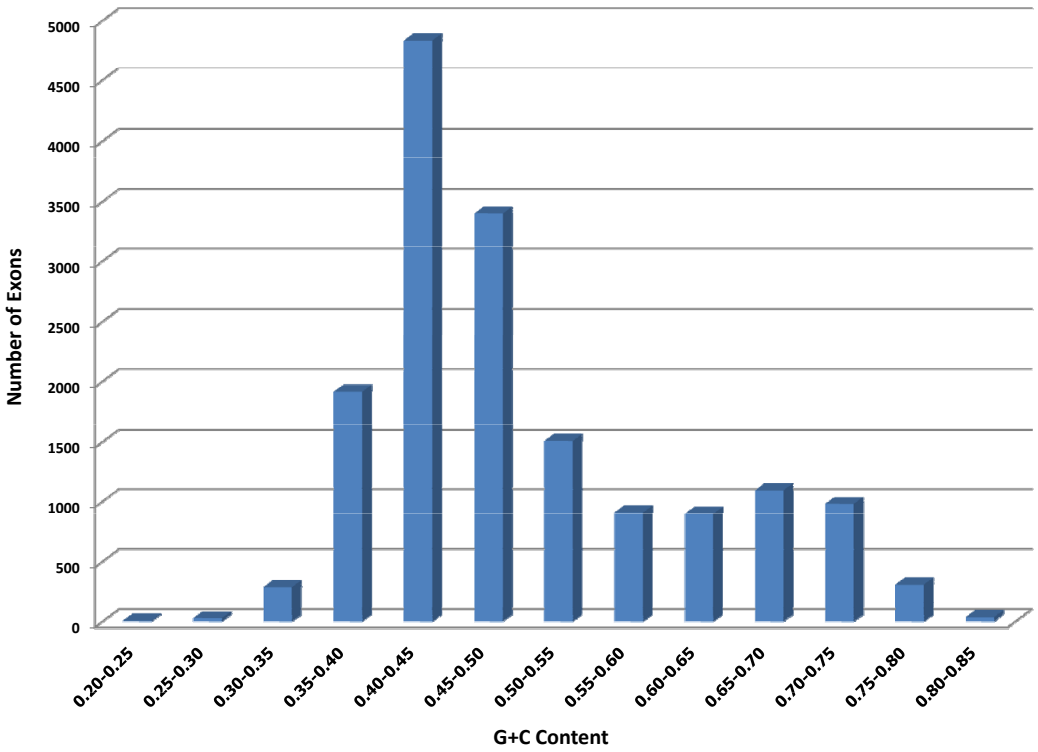


Figure 3.7: G+C Content of the second *Oryza Sativa* data set

in Figure 3.3. Therefore, we assessed the performance on *Oryza Sativa* data set to demonstrate the utility of our AUGUSTUS-GC.

We compared AUGUSTUS-GC with AUGUSTUS in Table 3.3. In this experiment, we observed changing patterns of G+C content of exons inside each gene. For example, some genes tend to start with high G+C content of exon and end with low G+C content of exon. As the result, AUGUSTUS-GC achieved higher sensitivity than AUGUSTUS. AUGUSTUS-GC can improve both the sensitivity and the specificity at all levels using a cutoff of low G+C content(0.40) and the cutoff of high G+C content(0.60). With training transition probabilities using maximum likelihood, the results are shown in Table 3.4. Furthermore, we compared changes in G+C contents of the uniquely identified genes by AUGUSTUS-GC and the common genes shared by AUGUSTUS and AUGUSTUS-GC. There were four uniquely predicted genes by AUGUSTUS-GC and 143 common genes. We compared the uniquely identified genes and common ones in terms of *SD*

Table 3.3: Performance comparison of gene prediction approaches on *Oryza Sativa* obtained from Childs Lab. The transition probabilities were divided into three equal parts. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS.

Program		AUGUSTUS	AUGUSTUS-GC			
			lowT=0.39 highT=0.61	lowT=0.35 highT=0.61	lowT=0.50 highT=0.60	lowT=0.40 highT=0.60
Base level	Sen	0.839	**0.84**	0.831	**0.921**	**0.841**
	Spe	0.892	**0.902**	**0.898**	0.883	**0.901**
Exon level	Sen	0.613	**0.617**	0.589	**0.698**	**0.617**
	Spe	0.694	**0.733**	**0.715**	0.692	**0.725**
Gene level	Sen	0.26	**0.28**	**0.267**	**0.267**	**0.287**
	Spe	0.235	**0.261**	**0.25**	0.234	**0.267**

Table 3.4: Performance comparison of gene prediction approaches on *Oryza Sativa* obtained from Childs Lab. The transition probabilities was calculated by using maximum likelihood estimation. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS.

Program		AUGUSTUS	AUGUSTUS-GC			
			lowT=0.39 highT=0.61	lowT=0.35 highT=0.61	lowT=0.50 highT=0.60	lowT=0.40 highT=0.60
Base level	Sen	0.839	**0.85**	**0.847**	**0.923**	**0.847**
	Spe	0.892	**0.898**	**0.898**	0.876	**0.899**
Exon level	Sen	0.613	**0.633**	**0.62**	**0.707**	**0.629**
	Spe	0.694	**0.732**	**0.716**	0.67	**0.727**
Gene level	Sen	0.26	0.253	0.253	**0.267**	**0.267**
	Spe	0.235	**0.235**	**0.235**	0.227	**0.247**

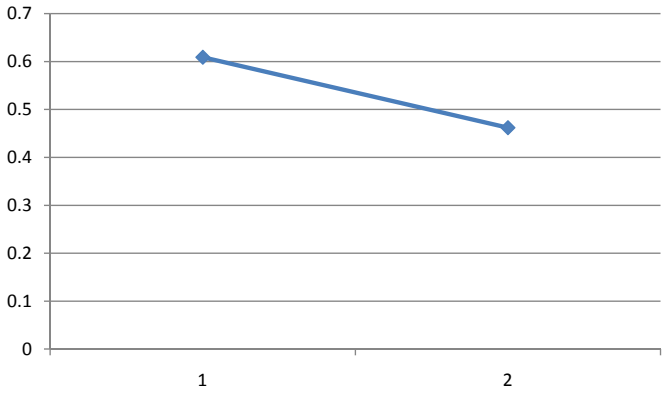


Figure 3.8: A predicted gene on forward strand with G+C content gradient of the first *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content.

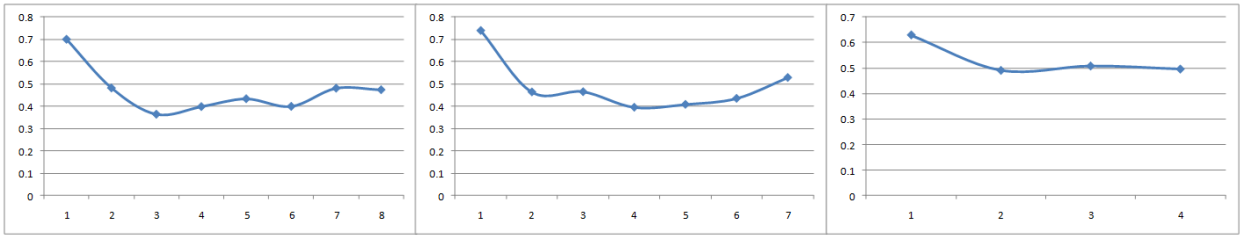


Figure 3.9: Three predicted genes on reverse strand with G+C content gradient of the first *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content.

and the distance between the exon that has highest G+C content and the exon that has lowest G+C content for each protein-coding gene. The results showed that the average *SD* of the uniquely predicted genes (0.098) was higher than that of commmon genes (0.075). Also, the average distance of the uniquely found genes by AUGUSTUS-GC (0.241) was wider than that of common genes (0.171).

Figure 3.8 and Figure 3.9 illustrates the genes that AUGUSTUS-GC can detect accurately, but the regular AUGUSTUS cannot identify it correctly. Because they contained G+C content gradient inside each gene, the G+C content of exon change rather than similar. The existing gene prediction tools do not work well for this case because they assumed the G+C contents of exons inside a gene are consistent.

3.3.2.2 Finding genes in *Oryza Sativa* retrieved from Stanke et al

In the second experiment of *Oryza Sativa* data set, the *Rice* test set was retrieved from Stanke et al. [99]. This is the brief description about how they picked those genes. First, they made a genbank file from the genome and the gff file. Second, they constructed a set with both UTRs annotated. Then, they generated selected genes with both UTRs and CDS plausible (from a visual inspection in JBrowse against panicle and leaf RNA-Seq STAR alignments). Fourth, they trained the data set to identify genes with errors only and removed the sequences with errors. Finally, the 1000 genes consist of 128 manually selected and 872 randomly chosen ones (from among filtered genes with both UTRs annotated). The *Oryza Sativa* parameter set was trained on a selection of 800 genes with UTRs from phytozome. For training data set, there were 187 single exon genes and 613 multi-exon genes on forward and reverse strand.

To assess the performance, 200 genes which were a subset of the 1000 genes were tested. The testing file consisted of 40 single-exon genes and 160 multi-exon genes on forward and reverse strands. Table 3.5 shows the summary of the accuracy results of AUGUSTUS and AUGUSTUS-GC on the test set. By applying AUGUSTUS-GC, the sensitivity of AUGUSTUS-GC at base level was enhanced from 0.859 to 0.942 for 0.31 low G+C content cutoff and 0.52 high G+C content cutoff. However, specificity of AUGUSTUS is slightly better than that of AUGUSTUS-GC for 0.31 low G+C content cutoff and 0.52 high G+C content cutoff. Moreover, AUGUSTUS-GC had the better sensitivity than AUGUSTUS at exon level. At exon level, the sensitivity of AUGUSTUS-GC was improved from 0.67 to 0.768 for 0.31 low G+C content cutoff and 0.52 high G+C content cutoff. At the gene level, AUGUSTUS-GC had better sensitivity and specificity (0.4 and 0.211, respectively). We computed the transition probabilities using maximum likelihood in Table 3.6. The sensitivity and specificity of AUGUSTUS-GC were improved.

Table 3.5: Performance comparison of gene prediction tools on *Oryza Sativa* provided by Stanke et al. The transition probabilities were divided into three equal parts. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS.

Program		AUGUSTUS	AUGUSTUS-GC			
			lowT=0.31 highT=0.52	lowT=0.49 highT=0.52	lowT=0.30 highT=0.50	lowT=0.60 highT=0.70
Base	Sen	0.859	**0.942**	**0.95**	**0.937**	0.84
level	Spe	0.619	0.607	0.597	0.59	**0.622**
Exon	Sen	0.67	**0.768**	**0.781**	**0.748**	0.63
level	Spe	0.552	0.546	**0.553**	0.52	**0.559**
Gene	Sen	0.355	**0.4**	**0.4**	**0.355**	**0.365**
level	Spe	0.191	**0.211**	**0.205**	0.177	**0.204**

Table 3.6: Performance comparison of gene prediction tools on *Oryza Sativa* provided by Stanke et al. The transition probabilities was trained using maximum likelihood estimation. **Bold number** indicates that sensitivity or specificity of AUGUSTUS-GC are higher than those of AUGUSTUS.

Program		AUGUSTUS	AUGUSTUS-GC			
			lowT=0.31 highT=0.52	lowT=0.49 highT=0.52	lowT=0.30 highT=0.50	lowT=0.60 highT=0.70
Base	Sen	0.859	**0.955**	**0.945**	**0.948**	0.858
level	Spe	0.619	0.607	0.601	0.586	**0.62**
Exon	Sen	0.67	**0.798**	**0.769**	**0.765**	0.665
level	Spe	0.552	**0.565**	0.544	**0.572**	0.547
Gene	Sen	0.355	**0.425**	**0.36**	0.35	**0.37**
level	Spe	0.191	**0.217**	0.186	**0.17**	**0.204**

We conducted additional analysis using the results of lowT=0.31 and highT=0.52. We found that AUGUSTUS-GC can detect the genes containing G+C content gradient. In addition, there were 26 uniquely predicted genes by AUGUSTUS-GC and 163 common genes shared by AUGUSTUS and AUGUSTUS-GC. Comparing uniquely predicted genes with common genes by average *SD* and the average distance between the exon that has highest G+C content and the exon that has lowest G+C content. The experimental results showed that uniquely predicted genes had higher *SD* than common genes which were 0.099 and 0.080, respectively. Besides, the average distance between the exon that has highest G+C content and the exon that has lowest G+C content for each protein-coding gene of the uniquely found genes had larger than that of the common genes which were 0.211 and 0.171, respectively.

Figure 3.10 illustrates a gene predicted by AUGUSTUS-GC. However, regular AUGUSTUS cannot identify. This gene is on reverse strand with G+C content gradient. This is strong evidence that our tool can predict the gene as we expected. Figure 3.11, Figure 3.12, Figure 3.13, Figure 3.14, and Figure 3.15 are the examples of the genes that have G+C content gradient. These genes were predicted by our tool, AUGUSTUS-GC. However, the regular AUGUSTUS cannot identify them correctly.

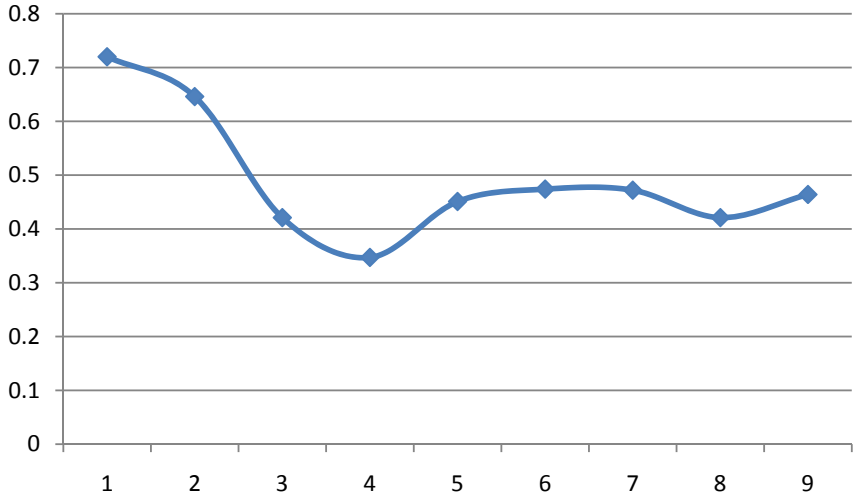


Figure 3.10: An example of a uniquely predicted gene on reverse strand with G+C content gradient of second *Oryza Sativa* data set. However, regular AUGUSTUS cannot detect it. X-axis represents exon index. Y-axis represents G+C content.

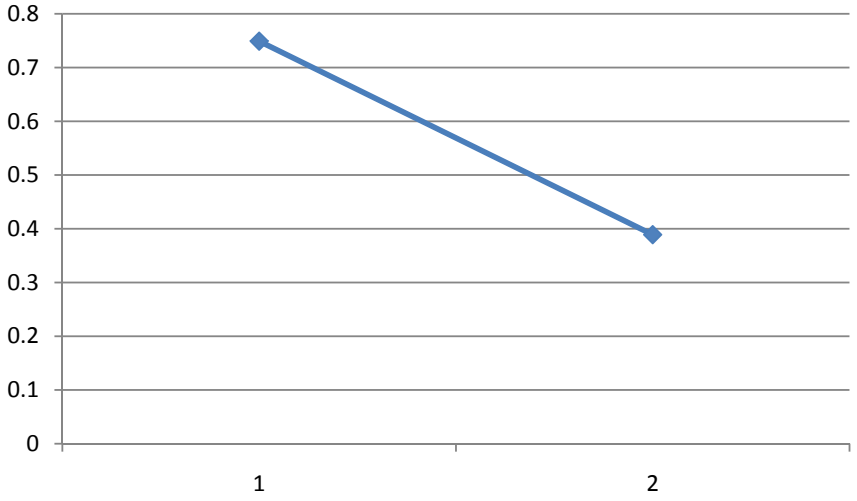


Figure 3.11: First example of a predicted gene on forward strand with G+C content gradient of second *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content.

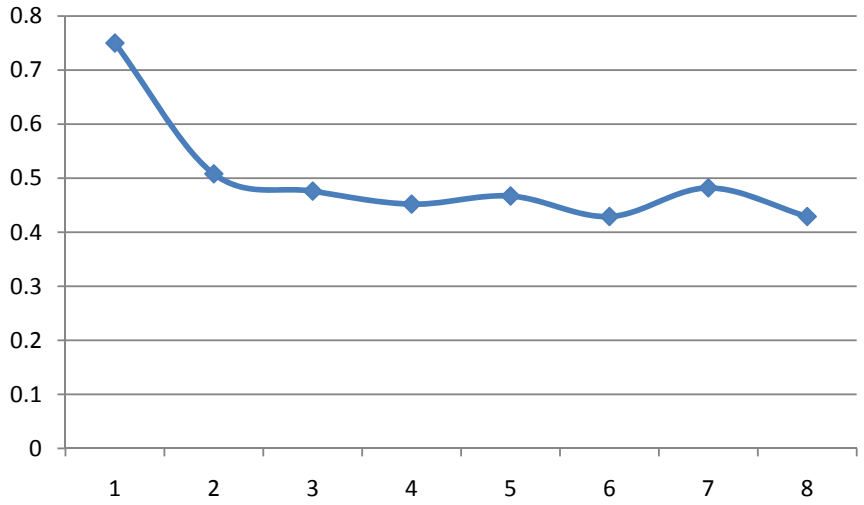


Figure 3.12: Second example of a predicted gene on forward strand with G+C content gradient of second *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content.

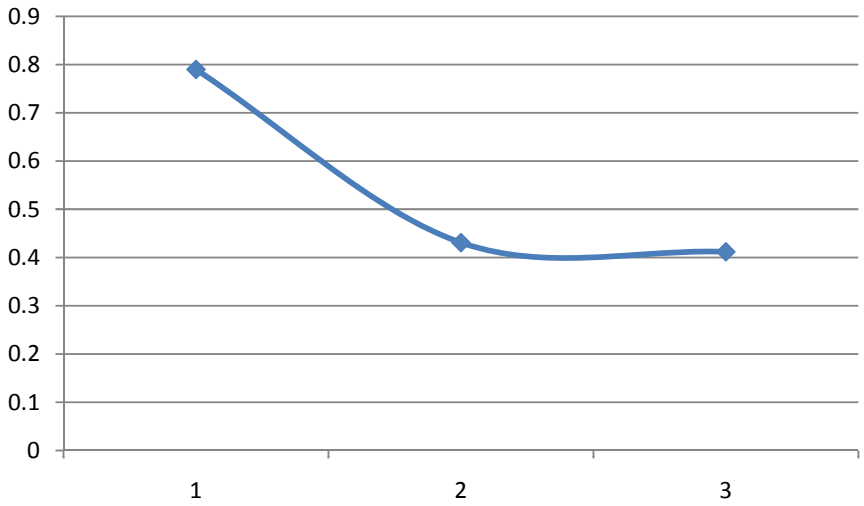


Figure 3.13: First example of a predicted gene on reverse strand with G+C content gradient of second *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content.

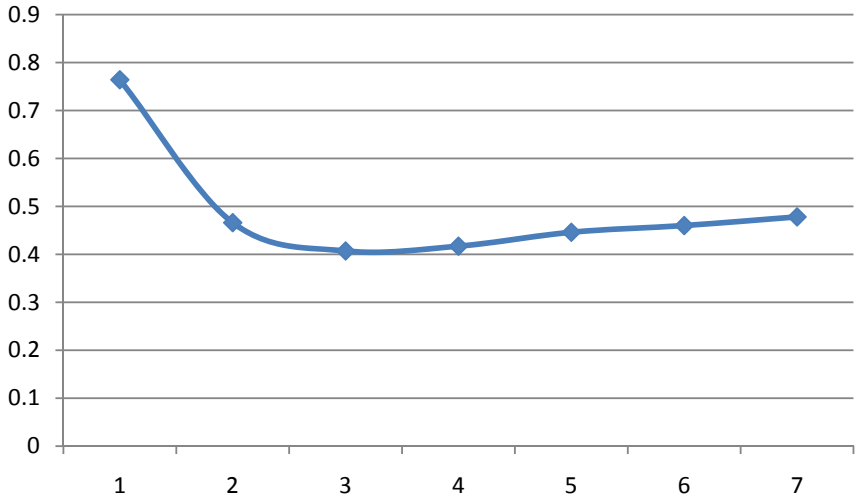


Figure 3.14: Second example of a predicted gene on reverse strand with G+C content gradient of second *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content.

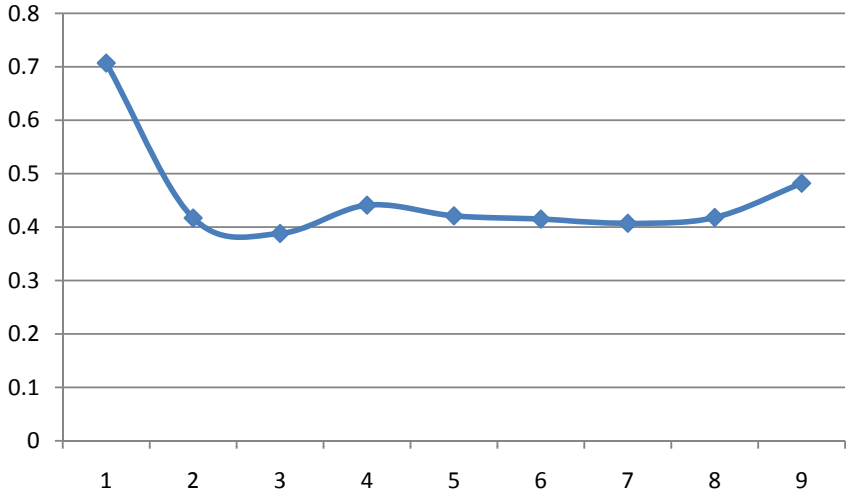


Figure 3.15: Third example of a predicted gene on reverse strand with G+C content gradient of second *Oryza Sativa* data set. X-axis represents exon index. Y-axis represents G+C content.

Chapter 4

Sensitive Short Read Homology Search for Paired-End Read Sequencing Data

4.1 Introduction

Homology search has been one of the most widely used methods for inferring the structure and function of newly sequenced data. For example, the state-of-the-art profile homology search tool, HMMER [29] has been successfully applied for genome-scale domain annotation. The major homology search tools were designed for long sequences, including genomic contigs, near-complete genes, or long reads produced by conventional sequencing technologies. They are not optimized for data produced by next-generation sequencing (NGS) platforms. For reads produced by pyrosequencing or more recent PacBio and nanopore technologies, frameshift caused by sequencing errors are the major challenges for homology search. For data sets produced by Illumina, short reads will lead to marginal alignment scores and thus many reads could be missed by conventional homology search tools. In order to apply homology search effectively to NGS data produced by Il-

Prapaporn Techa-Angkoon, Yanni Sun, and Jikai Lei, "Improve Short Read Homology Search using Paired-End Read Information", Proceedings of International Symposium on Bioinformatics Research and Applications (ISBRA 2016), Minsk, Belarus, June 5-8, 2016. (Appeared in Lecture Notes for Bioinformatics (LNBI) 9683)

Prapaporn Techa-Angkoon, Yanni Sun, and Jikai Lei, "Sensitive Short Read Homology Search for Paired-End Read Sequencing Data", Under review

lumina, many of which contain short reads, read mapping or *de novo* assembly [84, 76, 42, 117, 58] is first employed to assemble short reads into contigs. Then existing homology search tools can be applied to the contigs to infer functions or structures.

However, it is not always feasible to obtain assembled contigs from short reads. For example, complex metagenomic data poses serious computational challenges for assembly. Just 1 gram of soil can contain 4 petabase pairs (1×10^{15} bps) of DNA [115] and tens of thousands of species. Read mapping is not very useful in finding the native genomes or genes of these reads as most reference genomes are not available. *De novo* assembly also has limited success due to the complexities and large sizes of these data [108, 42, 117]. Besides metagenomic data, which usually lack complete reference genomes, RNA-Seq data of non-model species also faces similar computational challenges. Assembling short reads into correct transcripts without using any reference genome is computationally difficult.

Thus, in order to analyze the NGS data without reference genomes, a widely adopted method for functional analysis is to classify reads into characterized functional classes, such as protein/domain families in Pfam [31, 86], TIGRFAM [35], FIGfams [72], InterProScan [116], FOAM [85], etc. The read assignment is usually conducted by sequence homology search that compares reads with reference sequences or profiles, i.e., a family of homologous reference sequences. The representative tools for sequence homology search and profile homology search are BLAST [3] and HMMER [29], respectively. Profile homology search has several advantages over pairwise alignment tools such as BLAST. First, the number of gene families is significantly smaller than the number of sequences, rendering much faster search time. For example, there are only about 13,000 manually curated protein families in Pfam, but these cover nearly 80% of the UniProt Knowledgebase and the coverage is increasing every year as enough information becomes available to form new families [86]. The newest version of HMMER [29] is more sensitive than BLAST and is about 10%

faster. Second, previous work [27] has demonstrated that using family information can improve the sensitivity of a remote protein homology search, which is very important for metagenomic analysis because many datasets contain species remotely related to ones in the reference database.

4.1.0.3 Is HMMER good enough for short-read homology search?

HMMER has been successfully used in genome-scale protein domain annotation in many species. It has both high specificity and sensitivity in identifying domains. Thus, it is also widely adopted for profile homology search in a number of existing NGS analysis pipelines or websites (e.g. IMG/M [41], EBI metagenomics portal [74], CoMet [59], etc.). However, HMMER is not optimized for short-read homology searches. Short reads sequenced from regions of low conservation tend to be missed. We have quantified the performance of HMMER on several NGS datasets and the results showed that HMMER has much lower sensitivity than when applied to complete genes or genomes. Below we describe one of the experiments.

Table 4.1: The performance of HMMER under different cutoffs on classifying reads in the *Arabidopsis Thaliana* RNA-Seq dataset. There are 2,972,809 paired-end reads being uniquely mapped to 3,577 annotated protein domain families in the reference genome. However, HMMER under E-value cutoff 10 missed at least half of the reads when aligning these reads to input domains. The alignments by HMMER can be divided into three cases. Case 1: only one end can be aligned to one or multiple domains. Case 2: both ends can be aligned to one or multiple domains. Case 3: no end is aligned to any domain. In the table, the percentage of a case can be represented by each number. "HMMER w/o filtration" represents shutting off all filtration steps and running full Forward/Backward. "HMMER GA cutoff" represents applying HMMER with gathering thresholds.

Case	HMMER under E-value 10	HMMER w/o filtration under E-value 10	HMMER GA cutoff
Case 1	34.51%	32.83%	22.51%
Case 2	28.42%	31.58%	8.84%
Case 3	37.07%	35.59%	68.65%

We applied HMMER to annotate protein domains in reads sequenced from a normalized cDNA

library of *Arabidopsis Thaliana* [68, 119]. There were a total of 9,559,784 paired-end reads of 76 bp by Illumina. The known reference genome, the gene and domain annotations, and the read mapping results are combined to determine the true membership of reads. We downloaded all coding sequences (CDS) of *Arabidopsis Thaliana* from TAIR10 [106]. We identified the positions of domains in CDS using HMMER with gathering thresholds (GAs) by aligning CDS to 3,962 plant-related Pfam domains [31]. Then, bowtie allowing up to two mismatches [54] was used to map separately paired-end reads to CDS. We compared the positions of uniquely mapped reads in CDS to annotated domains in CDS. We chose all reads with both ends being uniquely mapped to a protein domain as the true positive set. Then the sensitivity of HMMER, which quantifies how many of these true positive reads can be identified, was computed. There were 2,972,809 read pairs in this true positive set. Their alignments by HMMER under different cutoffs can be divided into three cases. Case 1: only one end can be aligned. Case 2: both ends can be aligned to the corresponding protein family by HMMER. Case 3: neither end can be aligned. The results of this experiment were shown in Table 4.1. Case 2 is the ideal case. But it only takes about one-third of all read pairs. Turning off filtration does not improve the percentage of case 2 significantly. Using gathering thresholds (GA) cutoff is recommended for accurate domain annotation in genomes. However, near 70% of read pairs cannot be aligned under GA cutoff.

A closer look reveals that for many of those reads, the missing end is usually sequenced from a region with low sequence conservation with the underlying protein family. By using the whole-gene alignment against the protein family and the read mapping positions on the gene, we reveal the alignment of the reads against the model. One example is shown in Figure 4.1. In this example, one end r_1 can be aligned to the domain using HMMER with filtration on. However, the other end r_2 cannot be output by HMMER because of its poor alignment.

In order to improve the sensitivity, one may consider to use loose cutoffs such as a low score

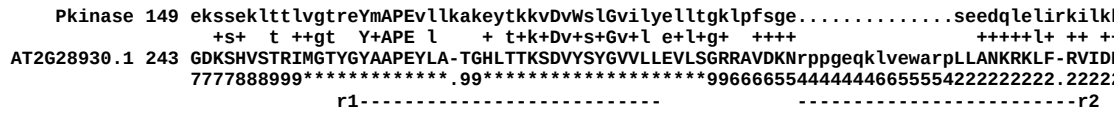


Figure 4.1: An example of a protein family, its alignment with a gene, and read mapping positions of a read pair against the gene. The Pkinase model had annotation line of consensus structure. The line beginning with Pkinase is the consensus of the query model. Capital letters show positions of the most conservation. Dots (.) in this line represent insertions in the target gene sequence with respect to the model. The midline represents matches between the Pkinase model and the AT2G28930.1 gene sequence. A + represents positive score. The line beginning with AT2G28930.1 is the target gene sequence. Dashes (-) in this line represents deletions in the gene sequence with respect to the model. The bottom line indicates the posterior probability of each aligned residue. A 0 represents 0-5%, 1 represents 5-15%, ..., 9 represents 85-95%, and * represents 95-100% posterior probability. The line starting with r_1 and ending with r_2 is read mapping regions on the gene sequence. A - indicates where the position of the read can be mapped to the gene sequence.

or high E-value cutoff. However, using loose cutoffs can lead to false positive domain alignments. In this work, we will describe a new method to improve the sensitivity of profile homology search for short reads without jeopardizing the alignment accuracy.

4.2 Methods

In this section, we describe a short read homology search method that incorporates properties of paired-end read sequencing. Paired-end sequencing is the preferred sequencing mode and is widely adopted by many sequencing projects. We have observed that for a large number of read pairs, only one end can be aligned by HMMER while the other end is missed. Thus, we exploit the sequencing property of paired-end reads to rescue the missing end.

Our probabilistic homology search model quantifies the significance of the alignment between a read pair and a protein domain family. The computation incorporates the distribution of fragment lengths (or insert sizes) of paired-end reads and the alignment scores. Similar approaches have been applied to mapping paired-end DNA reads to a reference genome [62, 97]. But to our knowledge,

this is the first time that an approximate Bayesian approach has been employed to align paired-end reads to protein families.

There are three major steps. In the first step, we will align each end (all-frame translations) to given protein families using HMMER under E-value cutoff 10. Note that although GA-cutoff is the recommended cutoff by HMMER for accurate domain annotation, only a small percentage of short reads can pass GA cutoff. Thus, we use E-value cutoff 10 in the first step in order to recruit more reads. As the reads are short, this step will usually align each read to one or multiple protein families. Not all of the alignments are part of the ground truth. In the second step, for all read-pairs where only one end is aligned by HMMER, we use the most sensitive mode of HMMER to align the other end to the protein families identified in the first step. Although the sensitive search mode of HMMER is slow, it is only applied to the specified protein families that are significantly fewer than total protein families in the dataset and thus will not become the bottleneck of large-scale homology search. In the last step, the posterior probability of the alignment between a pair of reads and a protein domain family is calculated.

The falsely aligned domains in the first step will be removed in the last through the computation of the posterior alignment probability. Figure 4.2 shows an example about determining the true protein family if both ends can be aligned to several families. In this example, M_1 is the most likely to be the native family due to the bigger alignment scores and the higher probability of the observed fragment length. We quantify the posterior probability of each read pair being correctly aligned to a protein family.

As the example in Figure 4.2 shows, in order to calculate the posterior probability of an alignment, we need to know the size distribution of fragments, from which paired-end reads are sequenced. Usually we may have the information about the range of the fragments (shortest and longest). However, the size distribution is unknown. For metagenomic data and RNA-Seq data of

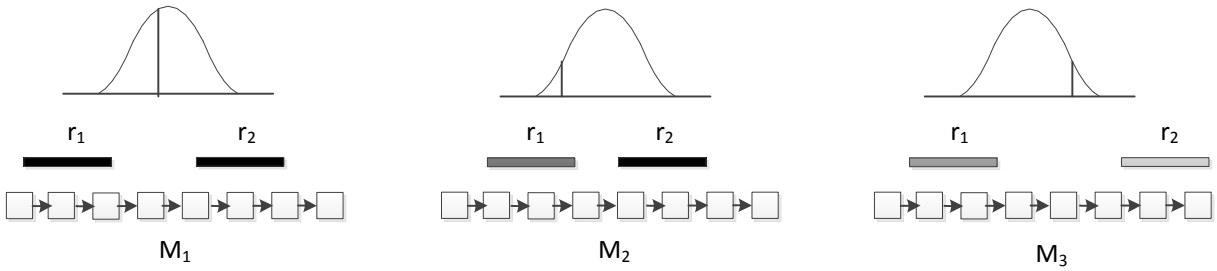


Figure 4.2: HMM alignments of a read pair. Paired-end reads r_1 and r_2 represented by two greyscale lines are aligned against models M_1, M_2, and M_3 with different scores of alignments. The darker lines represent bigger scores. The fragment size distribution is provided above each model. The distance between the two alignments is computed and is used to compute the likelihood of the corresponding fragment size. In this example, M_1 is most likely to be the native family.

non-models species whose complete or quality reference genomes are not available, it is not trivial to derive the fragment size distribution. In this work, we take advantage of the protein alignment and the training sequences to estimate the fragment size distribution. The next two sections will describe the details about computing fragment size distribution and the method to rank alignments using posterior probabilities.

4.2.1 Constructing fragment length distribution

Paired end reads are sequenced from the ends of fragments. When the reference genome is available, the fragment size can be computed using the distance between the mapping positions of the read pair. Thus, the distribution profile can be computed [62, 97] from a large-scale of read mapping positions. However, this method is not applicable to our work because we are focusing on the homology search of NGS data that lack reference genomes. For these data, we propose a model-based method to estimate fragment size distribution. The key observation is that if a read pair can be uniquely aligned to a protein family, it is very likely that this pair is sequenced from a gene that is homologous to the member sequences of the protein family. The homology is inferred

from statistically significant sequence similarity. Thus, we will use the alignment positions and the homologous seed sequences to infer the fragment size. This method is not accurate as we are not using any reference genomes/genes. However, our experimental results have shown that the estimated distribution is very close to the true distribution.

Figure 4.3 sketches the main steps of inferring a fragment's size from the alignment of a read pair against a protein family model. A read pair r_1 and r_2 are uniquely aligned to a protein family M. The alignment positions along the model M are from w to x and y to z, respectively. Model M is trained on a group of homologous sequences ("seed sequence 1" to "seed sequence N"). Note that the actual sequence from which r_1 and r_2 are sequenced is not in the training set of model M. The alignment positions along the model M will be first converted into the column indices in the multiple sequence alignment constructed by all seed sequences. Then after accounting for deletions and insertions, the column indices will be converted into positions along each seed sequence. As it is unknown which seed sequence shares the highest sequence similarity with the gene containing the fragment, we calculate the fragment size as the average of the distances between converted alignment positions.

The fragment size estimation is conducted for all paired end reads that are uniquely aligned to protein domain families. All the estimated fragment sizes are used to construct the fragment size distribution. We will compare the estimated fragment size distribution with the ones that are derived based on read mapping results.

4.2.2 Probabilistic model

For each aligned paired-end read, an approximate Bayesian approach [62, 97] is used to estimate the "alignment quality." The quality of alignment is defined as the probability of a pair of reads being accurately aligned to its native protein domain family. Because a pair of reads could be

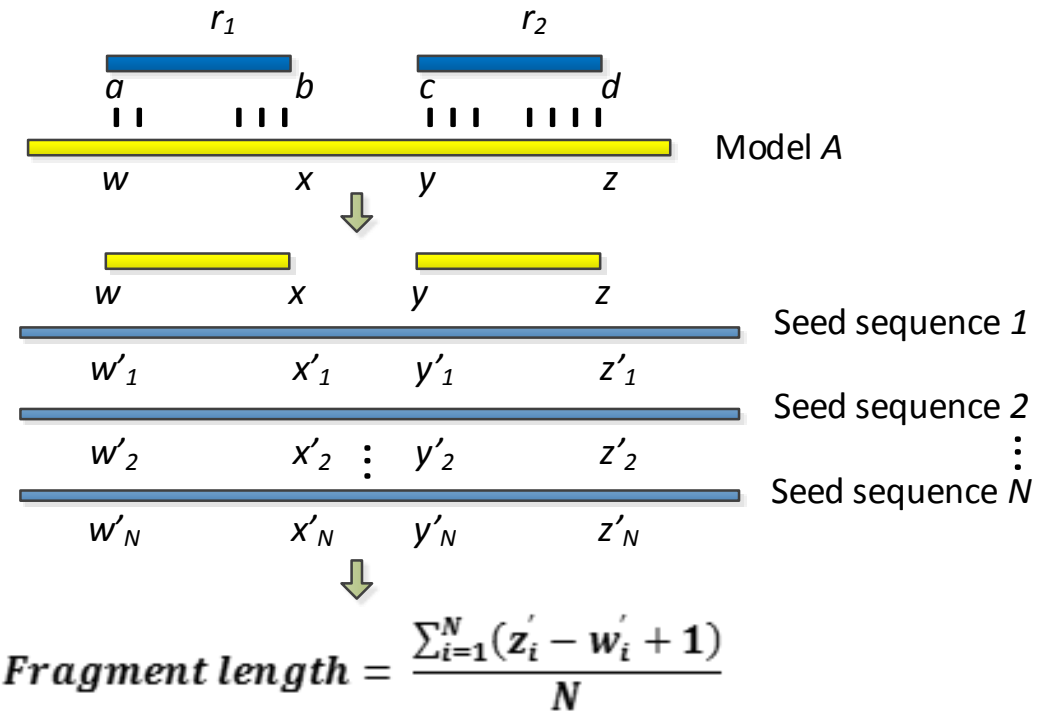


Figure 4.3: An example of fragment length calculation. The alignment positions along the profile HMM can be converted into positions in each seed sequences. The fragment size is computed as the average size of those mapped regions.

aligned to multiple domain families and some of them might not be in ground truth, we can rank all alignments using computed posterior probabilities and keep the alignments with high probability.

Let r_1 and r_2 be a read pair. Let A_1 and A_2 be the candidate alignment sets of r_1 and r_2 against one or more protein family models. For each alignment pair $a_1 \in A_1$ and $a_2 \in A_2$ with a_1 and a_2 being aligned to the same protein family M, we calculate the posterior probability of a_1 and a_2 being the true alignments generated by the read pair r_1, r_2 against M as:

$$Pr(a_1, a_2 | r_1, r_2) \propto e^{s_{a_1}/T} e^{s_{a_2}/T} Pr(f_{r_1,r_2}) \tag{4.1}$$

where $e^{s_{a_1}/T}$ is the target probability of generating an alignment score of a_1 against M [29, 45]. T is the scaling factor used in E-value computation. $Pr(f_{r_1,r_2})$ is the probability of observed fragment

size between r_1 and r_2. The posterior probability depends on the fragment length computed from a_1 and a_2 as well as their alignment scores.

We compute Equation (1) for each read pair's alignments and keep the alignments above a given threshold. For each read pair, suppose the maximum posterior probability of its alignments against all aligned models is p_{max}. We keep all alignments with probabilities above $p_{max} \times \tau$, where τ is 40% by default. Users can change τ to keep more or less alignments.

4.3 Experimental results

We designed profile-based homology search method for NGS data lacking reference genomes, including RNA-Seq data of non-model species and metagenomic data. In order to demonstrate its utility in different types of data, we applied our tool to a RNA-Seq dataset and a metagenomic dataset. In both experiments, we choose datasets with known reference genomes so that we can quantify the performance of homology search. It is important to note that the **ground truth** in this work is defined as the homology search results for complete genes. We are aware that computational protein domain annotation for complete genes or genomes are not always accurate. But whole-gene domain annotation has significantly higher sensitivity and accuracy than short read homology search and has been extensively tested in various species. Thus, our goal is to decrease the performance gap between short read homology search and whole-gene homology search.

HMMER can be run in different modes. In this work, we choose the most commonly used modes: HMMER with default E-value, HMMER with gathering thresholds (GAs) cutoff, and HMMER without filtration. GA cutoff is the recommended cutoff because of its accuracy. Turning off filtration will yield the highest sensitivity with sacrifice of speed.

The first dataset in our experiment is the RNA-Seq dataset of *Arabidopsis Thaliana*. The

second one is metagenomic dataset sequenced from *bacterial* and *archaeal* synthetic communities. We will first carefully examine whether our method and HMMER can correctly assign each read to its correct domain families. Then we will evaluate the performance of homology search from users' perspective. A user needs to know the composition of domains and also their abundance in a dataset. Thus we will compare HMMER and our method in both aspects.

4.3.1 Profile-based short read homology search in *Arabidopsis Thaliana* RNA-Seq dataset

The RNA-Seq dataset was sequenced from a normalized cDNA library of *Arabidopsis* using paired-end sequencing of Illumina platform [68]. There were 9,559,784 paired-end reads in total and the length of each read is 76 bp. The authors [68] indicated that the fragment lengths are between 198 and 801 bps. However, the fragment size distribution is unknown.

4.3.1.1 Determination for true membership of paired-end reads

The true membership of paired-end reads was determined using read mapping and domain annotation on complete coding sequences. First, all coding sequences (CDS) of *Arabidopsis Thaliana* genome were downloaded from TAIR10 [106]. Second, we downloaded 3,912 plant-related protein or domain models from Pfam [31]. We notice that some of these domain families are trained on genes of Arabidopsis. Thus, in order to conduct a fair evaluation of homology search performance, we removed all genes of *Arabidopsis* from the domain seed families and re-trained the Pfam profile HMMs. Third, CDS were aligned against Pfam domains [31] using HMMER with gathering thresholds (GAs) [29]. The alignment results contain the positions of domains in CDS. Note that it is possible that several domains are partially aligned to the same region in a coding se-

quence. This happens often for domains in the same clan [32] because these domains are related in structures and functions. In this case, we will keep all domain alignments passing the GA cutoff in the ground truth. Fourth, paired-end reads were mapped separately to CDS using Bowtie allowing up to 2 mismatches [54]. The positions of uniquely mapped reads in CDS were compared to annotated domains in CDS. If the mapping positions of read pairs are within annotated domain regions, we assigned the reads to those Pfam domains. The reads and their assigned domains constitute the true membership of these reads.

4.3.1.2 Performance of fragment length distribution

We compared our estimated fragment length distribution with the true fragment length distribution in Figure 4.4. The true fragment size distribution is derived by mapping all paired-end reads back to the reference genome. The comparison shows that, for a given length, the maximum probability difference between our fragment length distribution and the true fragment length distribution is 0.02, which slightly decreases the accuracy of the posterior probability calculation.

4.3.1.3 Our method can align significantly more reads

As shown in the first section, HMMER missed one end of at least half of the read pairs in the RNA-Seq dataset of *Arabidopsis*. By applying our tool, the percentage of case 2 (both ends) of paired-end read alignments increases from 28.42% to 62.51%. Table 1 presents the comparison. Importantly, the improvement is not achieved by sacrificing specificity. As we use the posterior probability to discard false alignments, the tradeoff between sensitivity and specificity is actually improved, as shown in the next section.

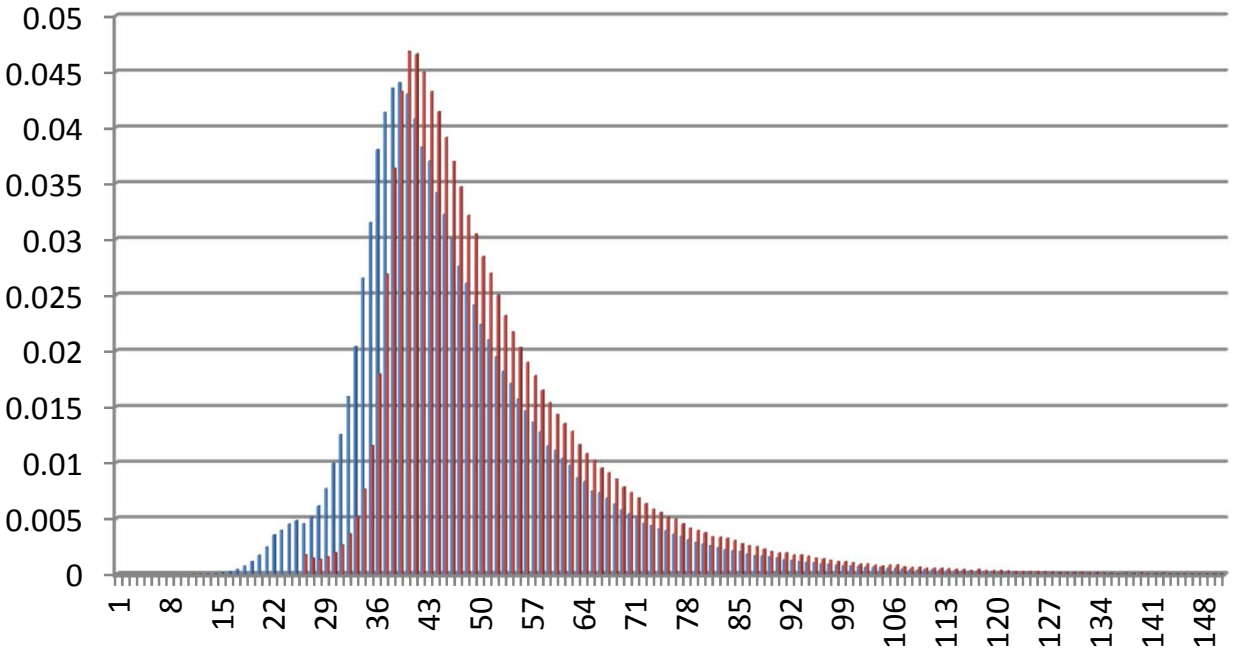


Figure 4.4: Comparing fragment length distribution of our method (blue) to fragment length distribution constructed from read mapping results (red). X-axis represents the length of fragment in **amino acids**. Y-axis represents probability of the corresponding fragment size.

Table 4.2: The percentages of all three cases of paired-end read alignments by HMMER and our tool for the RNA-Seq data. Case 1 represents one end. Case 2 represents both ends. Case 3 represents no end.

Case	HMMER, *E*-value 10	HMMER, w/o filtration, *E*-value 10	HMMER, GA cutoff	Our tool
Case 1	34.51%	32.83%	22.51%	0.42%
Case 2	28.42%	31.58%	8.84%	62.51%
Case 3	37.07%	35.59%	68.65%	37.07%

4.3.1.4 Sensitivity and accuracy of short read homology search

Although GA cutoff is the recommended threshold for domain annotation by HMMER, it yields low sensitivity for short read homology search. In order to align as many reads as possible, the default E-value cutoff is chosen. However, even for case 2, where both ends can be aligned by HMMER, these reads may be aligned to multiple domains and not all of them are correct. Our method can be used to improve the tradeoff between sensitivity and accuracy for both case 1 and

case 2.

In this section, the performance of profile-based homology search for each read is quantified by comparing its true protein domain family membership and predicted membership. For each read pair, suppose it is sequenced from domain set $TP = \{TP_1, TP_2, ..., TP_n\}$, which is derived from the read mapping results. The homology search tool aligns this read pair to domain set $C = \{C_1, C_2, ..., C_m\}$. The sensitivity and false positive (FP) rate for this read pair are defined using the following equations:

$$Sensitivity = \frac{|TP \cap C|}{|TP|} \tag{4.2}$$

$$FP\ rate = \frac{|C - TP|}{|TN|} \tag{4.3}$$

Note that TN represents the true negative domain set. Let $\mathscr{U}$ represent all domains we downloaded from Pfam ($|U|$=3,962). Then, for each read pair, $TN = U - TP$. In this section, the sensitivity and FP rate for each pair of reads are computed and then the average of all pairs of reads is reported using ROC curves.

4.3.1.4.1 Performance of case 1: There are 1,025,982 paired-end reads, where only one end can be aligned to one or multiple domain families by HMMER with filtration on. Figure 4.5 shows ROC curves of short read homology search using HMMER under different cutoffs and our method. For HMMER, we changed the E-value cutoff from 1000 to 10^{-5} with ratio 0.1. As some E-value cutoffs yield the same output, several data points overlap completely. For our method, each data point corresponds to different τ values (10% to 70%) as defined in Probabilistic Model Section. Unless specified otherwise, all the ROC curves are generated using the same configuration.

4.3.1.4.2 Performance of case 2: There are 844,796 paired-end reads with both ends being aligned by HMMER with filtration on. Some read pairs are aligned to false families. The falsely

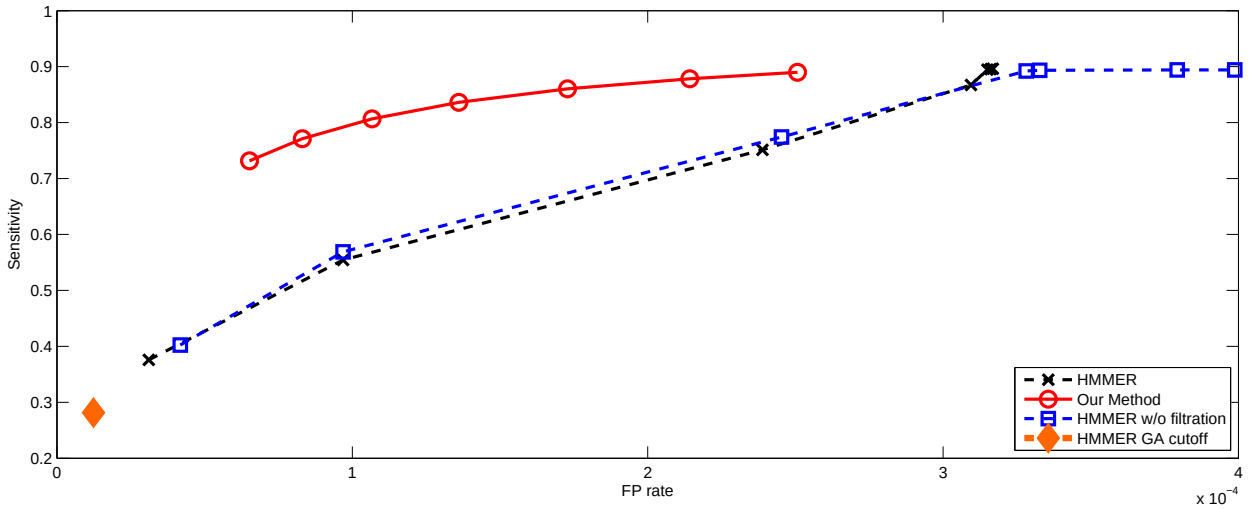


Figure 4.5: ROC curves of profile-based short read homology search for *Arabidopsis* RNA-Seq data. We compared HMMER and our tool on case 1, where one end can be aligned by HMMER with default E-value. Note that HMMER with GA cutoff has one data point.

aligned domain families can be removed by our method. Therefore, our method have better trade-off between sensitivity and false positive rate. In Figure 4.6, we plotted ROC curves of HMMER and our method.

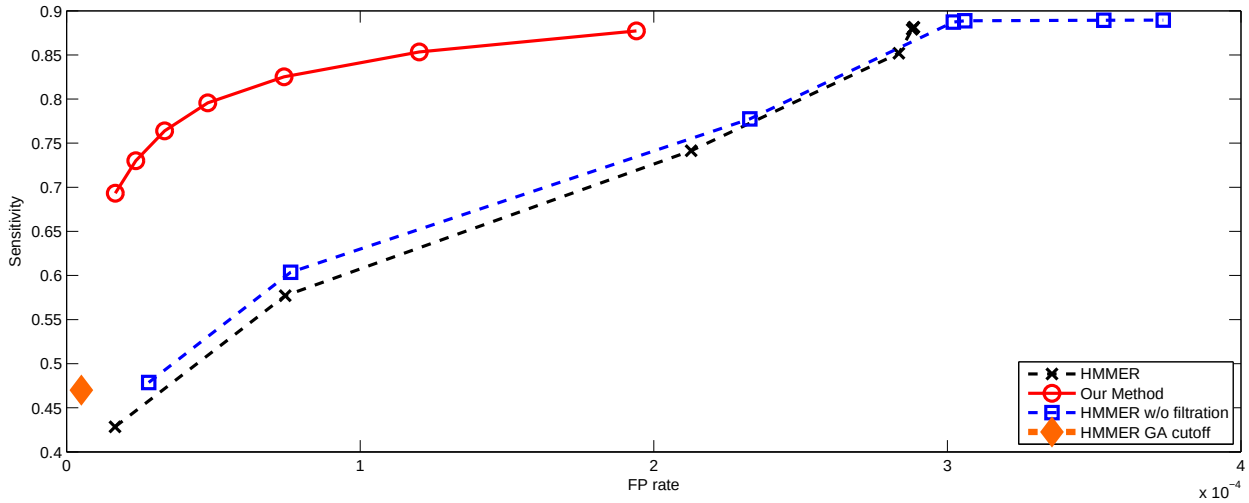


Figure 4.6: ROC curves of profile-based short read homology search for *Arabidopsis* RNA-Seq data. We compared HMMER and our tool on case 2, where both ends are aligned by HMMER with default E-value. Note that HMMER with GA cutoff has one data point. Using posterior probability helps remove false aligned domains and thus leads to better tradeoff between sensitivity and FP rate.

The results showed that HMMER with GA cutoff yields low sensitivity and low FP rate. Our method has better tradeoff between sensitivity and FP rate for both cases. We also computed other metrics including F-score ($\frac{2\times sensitivity\times PPV}{sensitivity+PPV}$) and PPV (Positive Predictive Value, $\frac{|TP\cap C|}{|C|}$). Comparing all tools in terms of F-Score and PPV under different thresholds for case 1, our tool achieves the highest F-Score 81.98%; the corresponding PPV is 80.41%. HMMER w/o filtration has the second highest F-Score 75.39% and its PPV is 65.17%. For case 2, our method has the highest F-score 86.33% with PPV 94.34%. HMMER with default E-value cutoff has the second highest F-Score 76.45% with PPV 67.50%.

4.3.1.5 Performance evaluation on domain-level

In order to assess the homology search performance on domain-level, we focused on comparing the set of domains found by HMMER and our tool. We further quantified the domain abundance, which is the number of reads classified in each domain by given tools. The predicted domain set and their abundance are also compared to the ground truth, which is derived using the read mapping results and the whole-gene domain annotation.

Our experimental results showed that the set of domains reported by HMMER under the default E-value cutoff and our method are almost identical. They only differ by 1 out of 3,962 domains. Both tools can identify almost all the ground-truth domains. The only exception is HMMER with GA cutoff, which returns 84% of true domains.

Although HMMER and our method reported near identical domain sets, they generated very different domain abundance. We compared the predicted abundance to the ground truth by computing their distance, which is the difference in the number of reads classified to a domain. According to the definition, small distance indicates higher similarity to the ground truth. For case 1, our tool has smaller distance to the ground truth than HMMER, with average distance being 65.39. Our

tool produced the same abundance as the ground truth for 1,185 domains. The average distances of HMMER, HMMER without filtration, and HMMER with GA cutoff are 107.60, 126.85, and 153.64, respectively. Figure 4.7 shows the distance of 377 domains for which our tool has distance above 86.

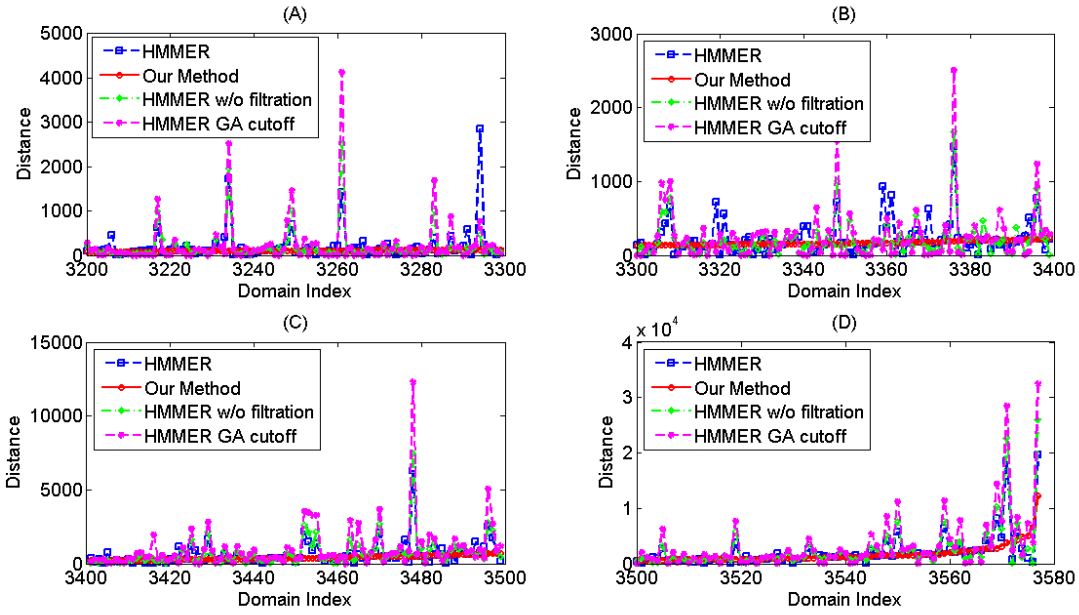


Figure 4.7: The distance comparison between our method and HMMER on case 1 of the RNA-Seq dataset of *Arabidopsis*. 377 domains with the largest distance values are listed in the four subplots. X-axis shows the indices of the domains. Smaller value indicates closer domain abundance to the ground truth. The average distances of HMMER, HMMER w/o filtration, HMMER GA cutoff, and our tool are 704.92, 781.80, 1,054.77, and 522.12, respectively.

Figure 4.8 illustrates the distance between the predicted domain abundance and the ground truth for case 2, where both ends can be aligned by HMMER under the default E-value cutoff. The average distances for HMMER, HMMER without filtration, HMMER with GA cutoff, and our tool are 121.61, 107.81, 139.56, and 96.34 respectively. Figure 4.8 only includes 358 domains for which our tool has the distance above 30.

In summary, being consistent with the results shown in Figures 4.5 and 4.6, our method can assign reads to their native domains with higher accuracy.

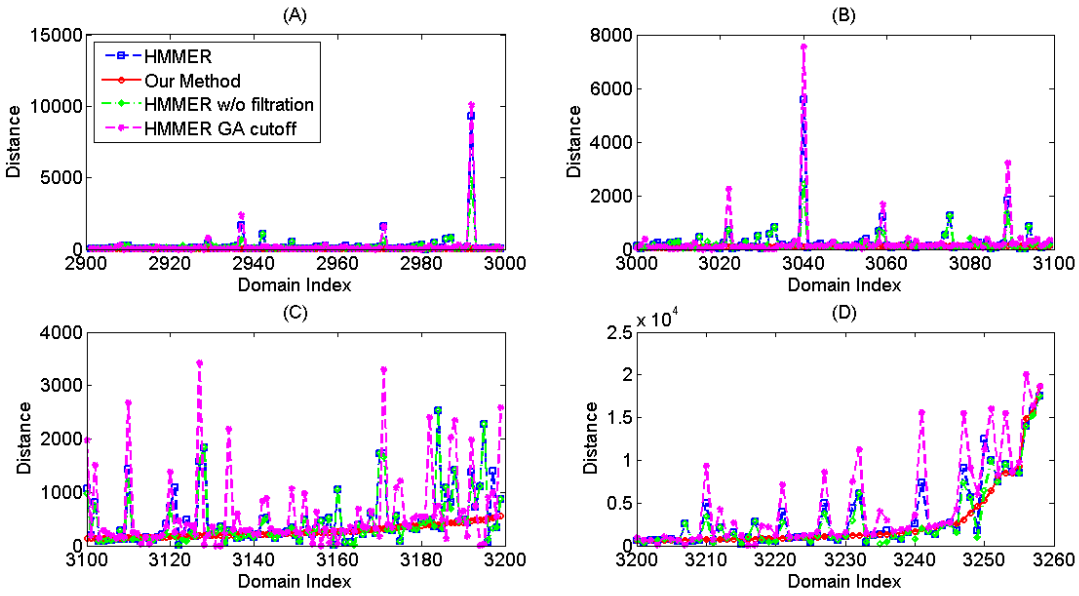


Figure 4.8: The distance comparison between our method and HMMER on case 2 of the RNA-Seq dataset of *Arabidopsis*. 358 domains with the largest distances are listed in the four subplots. X-axis shows the indices of the domains. Smaller value indicates closer domain abundance to the ground truth. The average distances of HMMER, HMMER w/o filtration, HMMER GA cutoff, and our method are 818.09, 704.65, 1084.50, and 558.60, respectively.

4.3.1.6 Running time analysis

We compared the running time of tested tools in Table 4.3. HMMER with GA cutoff is the fastest but yields low sensitivity. HMMER without filtration is computationally expensive and is the slowest. We are in between as we rely on the full Viterbi algorithm to align the missing end of a read pair.

Table 4.3: The running time of HMMER under different cutoffs and our tool on the *Arabidopsis Thaliana* RNA-Seq dataset. *Note:* The running time is the total running time of homology search tool to align 9,559,784 paired-end reads with 3962 domains.

Case	HMMER, E-value 10	HMMER, w/o filtration, E-value 10	HMMER, GA cutoff	Our tool
Time (m)	2,628.9	758,825.2	1,955.1	9,857.3

4.3.2 Profile homology search for short reads in a metagenomic dataset from synthetic communities

In the second experiment, we tested the performance of short read homology search in a metagenomic dataset. In order to quantify the performance of our method, we chose a mock metagenomic data with known composition.

4.3.2.1 Dataset

The chosen metagenomic data set is sequenced from diverse synthetic communities of *Archaea* and *Bacteria*. The synthetic communities consist of 16 *Archaea* and 48 *Bacteria* [96]. All known genomes were downloaded from NCBI. The metagenomic dataset of synthetic communities were downloaded from NCBI Sequence Read Archive (SRA) (accession No. SRA059004). There are 52,486,341 paired-end reads in total and the length of each read is 101 bp. All of reads are aligned against a set of single copy genes. These genes includes nearly all ribosomal proteins as well as tRNA synthases existed in nearly all free-living bacteria [26]. These protein families have been used for phylogenetic analysis in various metagenomic studies and thus it is important to study their composition and abundance in various metagenomic data. We downloaded 111 domains from Pfam database [31] and TIGRFAMs [35].

4.3.2.2 Determination of true membership of paired-end reads

The true membership of paired-end reads is determined based on whole coding sequence annotation and read mapping results. First, all coding sequences (CDS) of 64 genomes of *Archaea* and *Bacteria* were downloaded from NCBI. Second, CDS were aligned against 111 domains downloaded from TIGRFAMs [35] and Pfam database [31] using HMMER with gathering thresholds

(GAs) [29]. The positions of aligned domains in all in CDS were recorded. Third, paired-end reads were mapped back to the genomes using Bowtie [54]. The read mapping positions and the annotated domain positions are compared. If both ends are uniquely mapped within an annotated domain, we assign the read pair to the domain family. The true positive set contains all read pairs with both ends being uniquely mapped to a protein domain. We will only evaluate the homology search performance of chosen tools for these reads.

4.3.2.3 Performance of fragment length distribution

Again, we need to examine the accuracy of our fragment size computation. Figure 4.9 shows the fragment length distribution constructed from our tool and the fragment length distribution derived from the read mapping results. For a given length, the maximum probability difference between our method and the ground truth is 0.01, which slightly reduces the accuracy of posterior probability computation.

4.3.2.4 Our method can align more reads

In this experiment, the read length is longer than those in the first experiment. Consequently, HMMER can align more reads against their native domain families. Nevertheless, it still has one third of pairs of reads with one end being aligned to the protein domain families. By applying our tool, the percentage of case 2 (both ends) of paired-end read alignments is enhanced from 65.82% to 88.71%. The percentages of three cases by our tool and HMMER are shown in Table 4.4.

4.3.2.5 Sensitivity and accuracy of short read homology search

4.3.2.5.1 Case 1: one end is aligned by HMMER There were 213,668 paired-end reads with only one end being aligned to one or multiple domains. Figure 4.10 shows the ROC curves of short

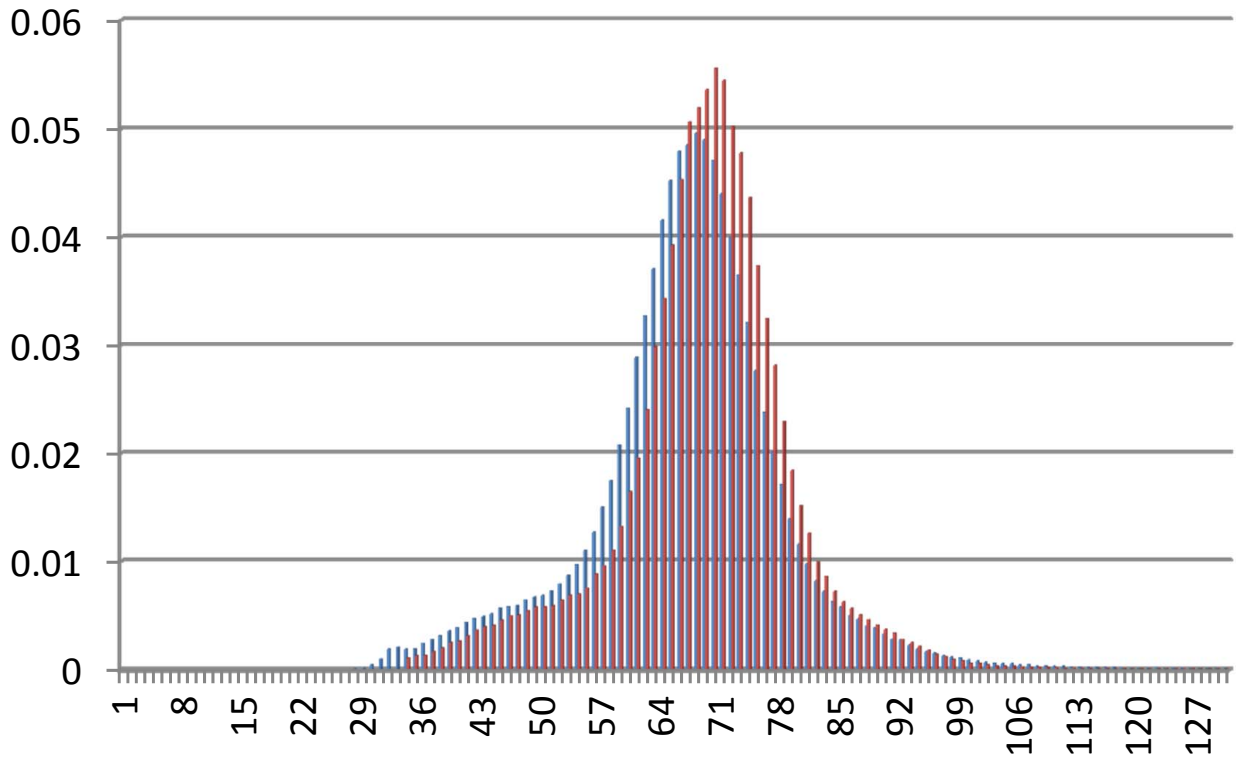


Figure 4.9: Comparing fragment length distribution of our tool (blue) to fragment length distribution constructed from read mapping results (red). X-axis represents fragment length in **amino acids**. Y-axis represents the probability of the corresponding fragment size.

Table 4.4: The percentages of all three cases of paired-end read alignments by HMMER and our tool for the synthetic metagenomic data. Case 1 represents one end. Case 2 represents both ends. Case 3 represents no end.

Case	HMMER, *E*-value 10	HMMER, w/o filtration, *E*-value 10	HMMER, GA cutoff	Our tool
Case 1	23.15%	21.63%	3.76%	0.26%
Case 2	65.82%	68.46%	2.46%	88.71%
Case 3	11.03%	9.91%	93.77%	11.03%

read homology search using HMMER and our tool. HMMER with GA cutoff has the lowest FP rate (0.0). However, the sensitivity of HMMER with GA cutoff is only 4.11%. In addition, we further computed PPV and F-Score of each data point in ROC curves. Comparing all tools, our tool has the highest F-Score and PPV (90.87% and 88.01%, respectively). HMMER with E-value 10 has the next highest F-score and PPV (64.79% and 48.07%, respectively).

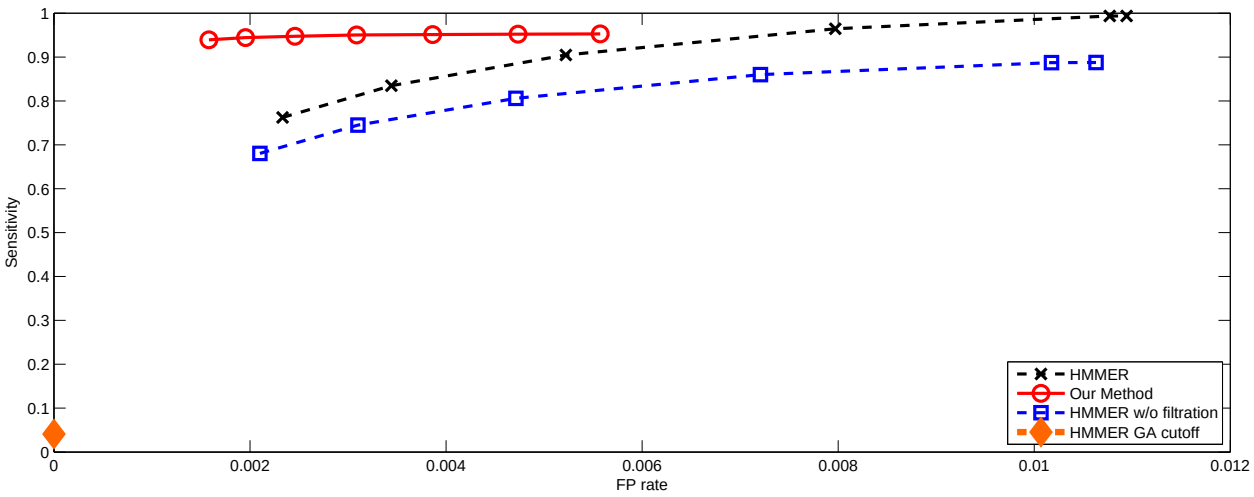


Figure 4.10: ROC curves of profile-based short read homology search for the synthetic metagenomic data set. We compared HMMER and our tool on case 1, where one end can be aligned by HMMER with default E-value. Note that HMMER under GA cutoff has one data point.

4.3.2.5.2 Case 2: both ends are aligned by HMMER

607,558 paired-end reads were classified to case 2. We divided data into two groups: 1) both ends being aligned to one domain and 2) both ends being aligned to multiple domains. There were 515,586 paired-end reads and 91,972 paired-end reads, respectively. When both ends are aligned to one single domain, the classification is usually correct. Thus, we focus on evaluating the performance of the second group, where read pairs are aligned to more than one domain. Figure 4.11 shows the average performance comparison between HMMER and our tool on 91,972 paired-end reads. Comparing all tools in term of F-Score and PPV, our tool achieves the highest F-Score of 96.05% and its PPV is 92.42%. HMMER w/o filtration achieves the second highest F-Score 80.28% with PPV 80.45%.

4.3.2.6 Domain-level performance evaluation

For whole dataset, we compared the set of domains identified by HMMER and our tool. The results showed that every tool identified all ground truth domains (111 domains) except HMMER with GA cutoff, which only found 26 domains.

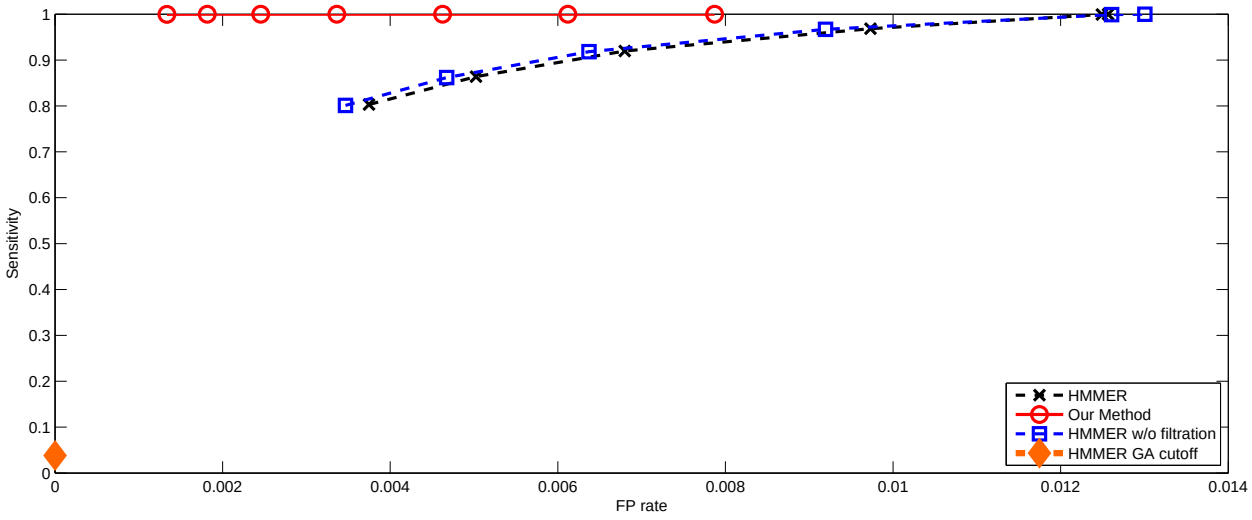


Figure 4.11: ROC curves of profile-based short read homology search for the synthetic metagenomic data. We compared HMMER and our tool on case 2, where both ends are aligned by HMMER under default E-value. Note that HMMER under GA cutoff has one data point. Using posterior probability helps remove false aligned domains and thus leads to better tradeoff between sensitivity and FP rate.

In addition, the domain abundance was quantified and compared to the ground truth. For each domain, we compute the "distance", which is the difference in the number of reads classified to a domain by a tool and in the ground truth. Smaller distance indicates closer domain abundance to the ground truth. For case 1, the average distances of HMMER, HMMER w/o filtration, HMMER with GA cutoff, and our method are 272.74, 280.65, 505.56, and 178.60, respectively. Our tool has the same abundance as the ground truth in 43 domains. We removed those 43 domains and showed distance of other domains in Figure 4.12.

For case 2, where both ends can be aligned, all tools have worse domain abundance estimation. The average distances of HMMER, HMMER w/o filtration, HMMER with GA cutoff, and our method are 702.39, 1698.79, 1831.55, and 666.96, respectively. Our tool still has the closest domain abundance to the ground truth. It has the same domain abundance as the ground truth for 68 domains. We removed the 68 domains and plotted the distances of other domains in Figure 4.13.

Although the read lengths of this data set are longer than the first data set, the average sequence

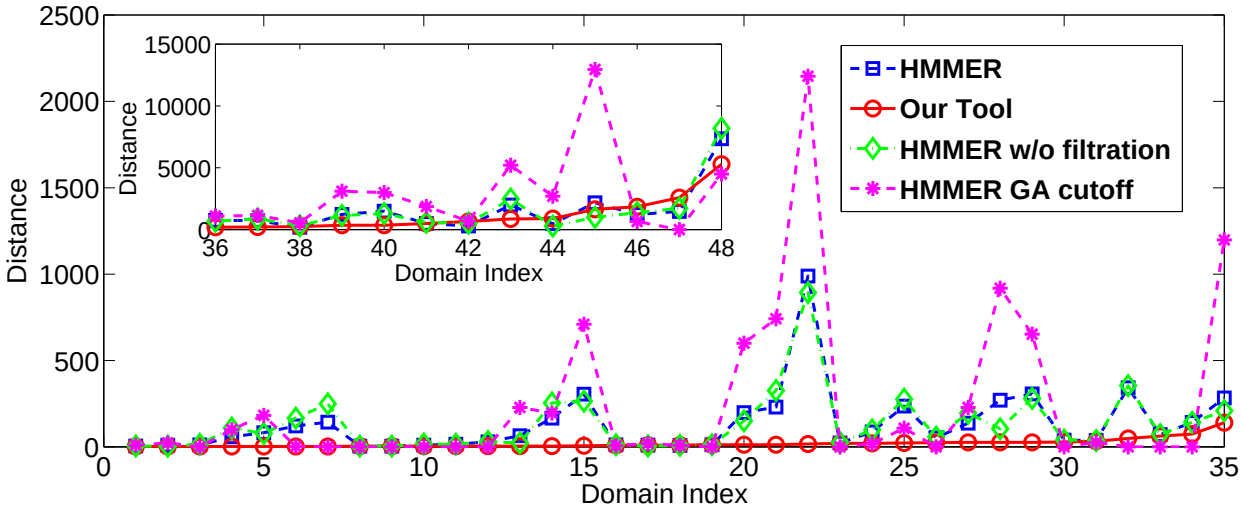


Figure 4.12: The distance comparison between our method and HMMER on case 1 of the metagenomic data set. X-axis shows the indices of the domains. Smaller value indicates closer domain abundance to the ground truth. Domains are sorted based on the distance of our tool. Due to scaling issues, domains with the largest distances are plotted in the embedded window.

conservation of the domain families is as low as 30%. The poorly conserved families contain large numbers of substitutions, long insertions and deletions, leading to either over-prediction or under-prediction of the tested tools. HMMER with E-value cutoff 10, HMMER w/o filtration, and our method all classified significantly more reads into the domain families than ground truth. HMMER with GA cutoff significantly under-classified short reads into the underlying families. Thus, the distances of all these tools are large.

4.3.2.7 Running time on a metagenomic dataset of synthetic communities

The running times of HMMER under different cutoffs and our tool are compared in Table 4.5. As expected, HMMER with filtration is the fastest. Our method is slower than HMMER with filtration but much faster than HMMER w/o filtration.

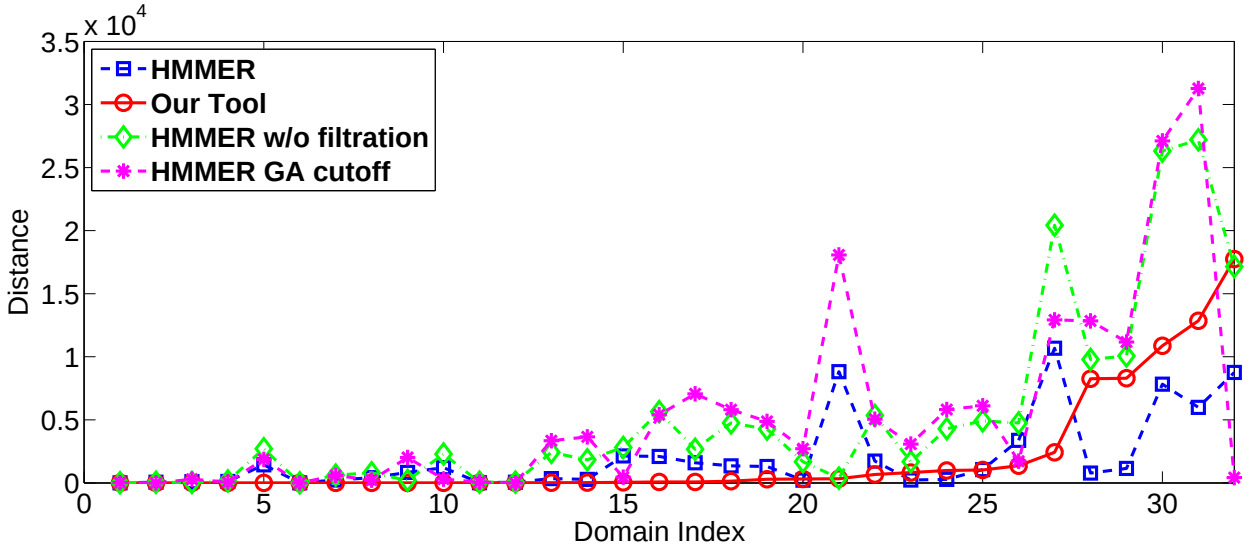


Figure 4.13: The distance comparison between our method and HMMER on case 2 of the metagenomic data set. X-axis shows the indices of domains. Smaller value indicates closer domain abundance to the ground truth.

Table 4.5: The running time of HMMER under different cutoffs and our tool on a metagenomic dataset of synthetic communities. *Note:* The running time is the overall running time of homology search tool to align 52,486,341 reads with 111 domains.

Case	HMMER, *E*-value 10	HMMER, w/o filtration, *E*-value 10	HMMER, GA cutoff	Our tool
Time (m)	353.13	152,864	378.39	4,468.40

4.4 Discussion and conclusion

Homology search has been widely used for sequence-based functional analysis in various NGS sequencing projects. In particular, for gene-centric analysis, reads are classified into characterized protein/domain families using profile-based homology search. While HMMER is the state-of-the-art tool for profile homology search, its performance on short reads has not been systematically examined. Our test of HMMER in various NGS data containing short reads shows that it could miss a large number of short reads. In this work, we described a probabilistic homology search model for paired-end reads. The goal is to improve the performance of short read homology search.

It is built on HMMER and can be used as a complementary tool to HMMER for more sensitive read classification.

One future direction is to improve the short read homology search performance for poorly conserved families. Near 4,000 domain families in the first experiment have higher average sequence identity and thus lead to reasonable domain abundance estimation. The 100+ families in the second experiment have low sequence identity and the tested tools tend to either over-classify or under-classify heavily for some families. Thus, better methods need to be designed to align short reads to poorly conserved protein families.

The advances of NGS technologies enable output of longer reads. The increased length will lead to better sensitivity of HMMER. However, before the reads reach the length of near complete transcripts or genes, there is still a need for improving short read homology search. In addition, existing sequencing projects are still heavily relying on today's sequencing technologies. We expect our tool can be used to improve the functional analysis.

Chapter 5

glu-RNA: aliGn highLy strUctured ncRNAs using only sequence similarity

5.1 Introduction

Noncoding RNAs (ncRNAs), which function directly as RNAs without translating into proteins, is highly important to modern biology. Different types of ncRNA have diverse and important biological functions. For example, house-keeping RNAs such as tRNAs and rRNAs, and small regulatory RNAs including miRNAs, have been extensively studied because of their important biological roles. In particular, the development of next-generation sequencing (NGS) technologies sheds lights on more reliable and comprehensive ncRNA annotation. Deep sequencing of transcriptoms of various organisms has revealed that a large portion of transcriptomic data cannot be mapped back to annotated protein-coding genes in the reference genome, indicating that those transcripts may contain transcribed ncRNAs.

The functions of many types of ncRNAs are associated with both their sequences and secondary structures, which contain interacting base pairs such as Watson-Crick base pairs and G-U. For

Prapaporn Techa-angkoon and Yanni Sun, "glu-RNA: aliGn highLy strUctured ncRNAs using only sequence similarity", Proceedings of ACM Conference on Bioinformatics, Computational Biology and Biomedical Informatics (ACM BCB 2013), Washington DC, USA, 2013.

example, functional pre-miRNAs in various species fold into hairpin structures. A majority of tRNAs share the characteristic clover-leaf structure. Thus, ncRNA gene-finding should incorporate information from both sequences and secondary structures.

Successful genome-scale ncRNA gene finding tools can be roughly divided into two groups. The first category includes tools for "known ncRNA search", which compare query sequences with annotated ncRNAs or ncRNA families and use their sequence and structural similarity for ncRNA annotation. As characterized ncRNAs or ncRNA families are needed as reference, this type of tools are limited to "known ncRNAs", such as tRNA, rRNAs, annotated miRNAs etc. The second category includes *de novo* ncRNA gene finding tools that are able to identify novel ncRNAs. Efficient implementations in this category accept sequence alignments as input and examine how likely the input sequences encode ncRNAs. Popular tools in this category include RNAz [110], EvoFold [83], QRNA [89], etc. For example, RNAz and EvoFold have been used for ncRNA gene finding in human genome [30]. Other tools that do not rely on alignments are also available for novel ncRNA identification. However, conducting structural alignment makes these tools highly expensive for genome-scale ncRNA search [107, 38].

A majority of popular de novo ncRNA gene finding tools in the second category require pairwise or multiple alignment as input. The qualify of the alignment directly affects the accuracy of ncRNA gene finding. There exist accurate structural alignment algorithms that maximize both sequence and secondary structure similarity [95, 104, 69]. Using structural alignments leads to more reliable ncRNA gene finding. However, most structural alignment tools cannot be conveniently applied to whole genomes or NGS data because of two reasons. First, genome scale ncRNA gene finding needs local alignment tools while many structural alignment programs are either global or use a simple sliding window strategy to generate local alignments. Second, the time complexity of structural alignment is still significantly higher than sequence alignment. Generating local

structural alignments between large genomes or large number of short reads generated by NGS technologies is not practical. It is thus desirable to have an alignment program that can achieve similar accuracy to structural alignment tools while keeping the same order of running time complexity as sequence alignment tools.

In this work, we designed and implemented an accurate ncRNA alignment program that only relies on sequence similarity. We applied this tool to 360 pairs of highly structured ncRNAs including tRNAs and riboswitch elements. The experimental results demonstrate that our alignment tool can generate more accurate ncRNA alignments than several popular sequence alignment tools. In particular, for highly structured ncRNAs with low sequence similarity ($<\%40$), we achieved better alignment accuracy than a popular structural alignment program while keeping the same running time complexity as typical sequence alignment programs.

5.2 Related Work

Recently, Will et al. [111] reported many novel ncRNAs using structured-based whole genome realignment. In their method, existing whole genome sequence alignment was used as the base alignment. Regions that were poorly aligned were realigned using a banded structural alignment algorithm, which conducts structural alignment in a restricted search space around the sequence alignment. The newly generated structural alignments were used for ncRNA examination. It was found that many of them may contain novel ncRNAs. Our tool shares the similar framework, which intends to generate a better alignment using a sequence alignment as the starting point. However, there are several major differences. First, the base alignment we use is generated using posterior probability, which is expected to be more reliable for sequences with low similarity. Second, unlike Will et al.'s work that relies on a banded structural alignment, we use sequence similarity and a

machine learning approach to convert the base alignment into a new alignment. No structural information is used.

The procedure of modifying a base alignment in our work can also be employed to create a banded dynamic programming space. The idea of banded alignment has been used in both sequence alignment and structural alignment programs [69, 111, 37, 24, 19]. By applying dynamic programming in banded search space, the running time complexity of alignment programs can be reduced. The most related work is Dynalign [69], which fixed a window of width M around the alignment diagonal. Thus, a position in sequence x is restricted to align within M-distance of the same position in sequence y. This space reduction significantly reduces the running time complexity of structural alignment. This heuristics is adopted by other structural alignment programs [38, 111, 24]. For example, a constant deviation (Δ) is used in Will et al.'s work to define a banded region for structural alignment.

The value of M is empirically determined in Dynalign. In an updated version of Dynalign [37], the authors redefines the search space by applying a cutoff to posterior alignment probabilities. Only position pairs with posterior alignment probabilities above the given cutoff will be used in search. Thus, different values of M are used for each row.

Our work differs from Dynalign in both the application and the method. First, our work aims to generate accurate ncRNA alignment without using structural information. Dynalign conducts structural alignment within reduced search space. Second, our work employs posterior probability and more features in a machine learning approach to produce a new alignment. Dynalign only uses posterior probability to define a search space.

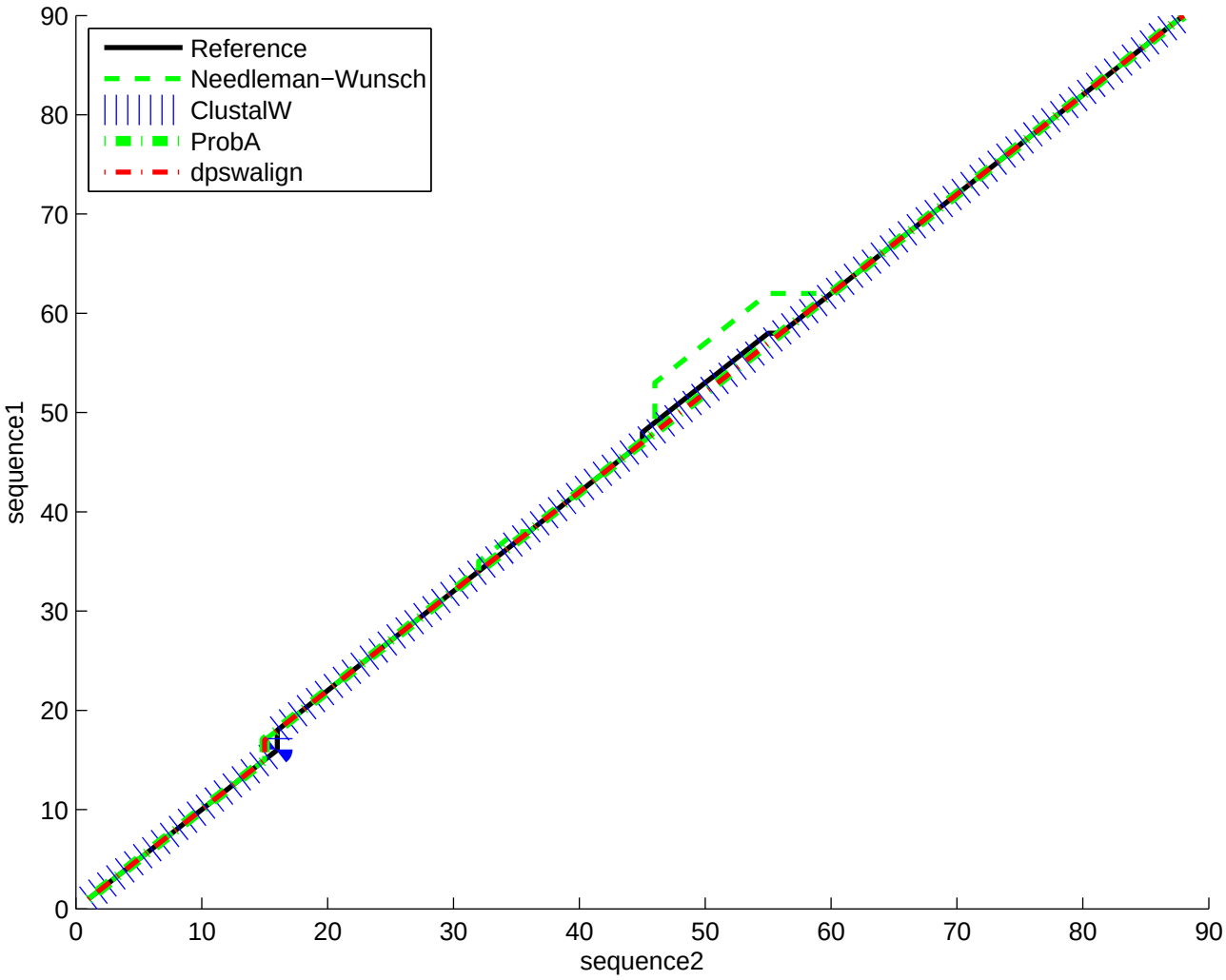


Figure 5.1: Alignments generated by four alignment programs for a pair of tRNAs with sequence similarity 80%. X-axis represents sequence 2 and Y-axis represents sequence 1. The alignment starts with (0,0).

5.3 Method

Our ncRNA alignment program is built on several key observations. First, for ncRNAs with high sequence similarity, the sequence alignment is usually highly consistent with structural alignment [33]. For highly structured ncRNAs that lack strong sequence similarity, the divergence between the sequence and structural alignments tends to be large. Second, different sequence alignment tools can generate different alignments, depending on the choice of scoring matrices,

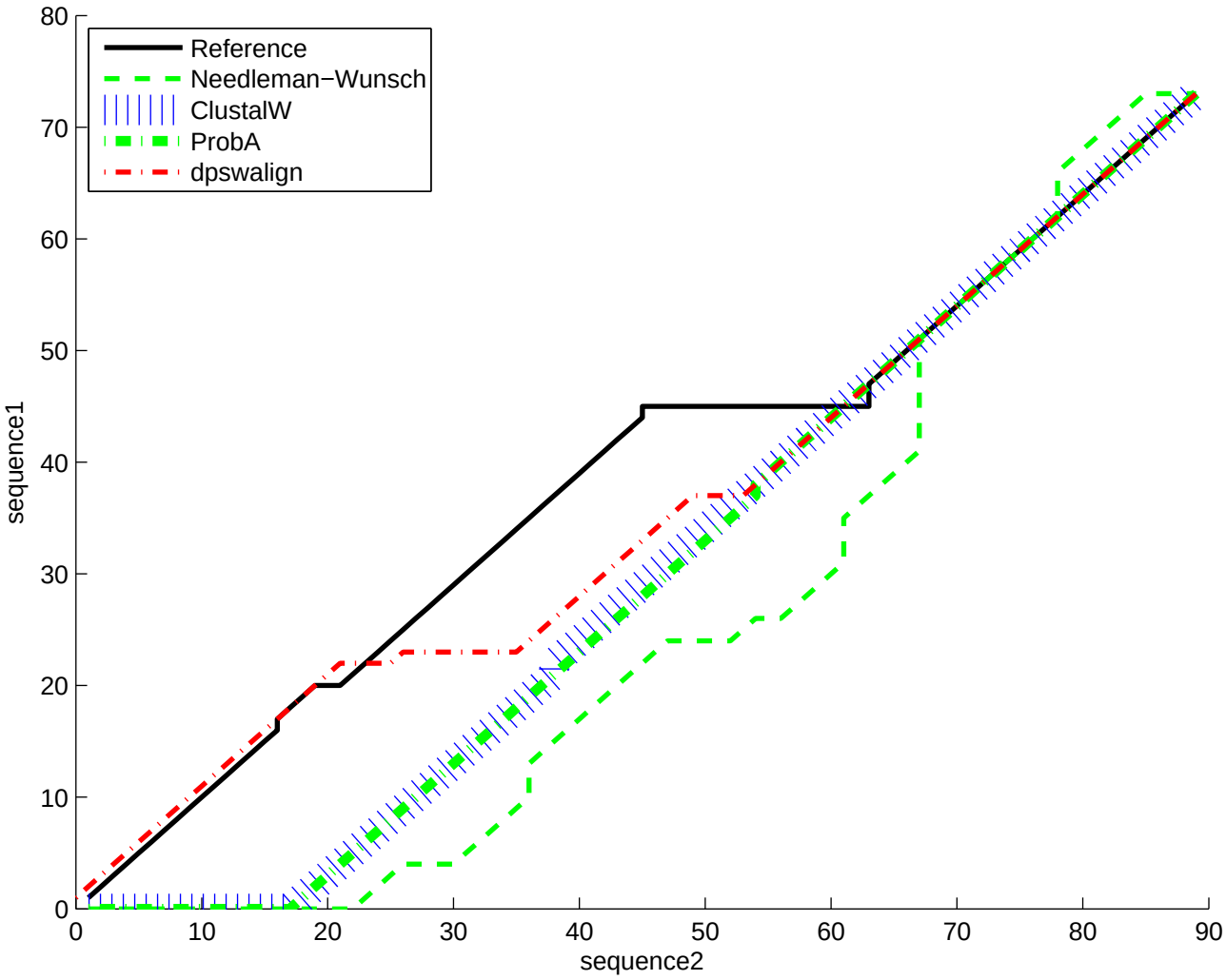


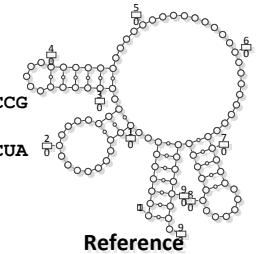
Figure 5.2: Alignments generated by four alignment programs for a pair of tRNAs with sequence similarity 40%. X-axis represents sequence 2 and Y-axis represents sequence 1. The alignment starts with (0,0).

gap penalties, and alignment algorithms. Previous work [37, 21, 70, 40, 57] suggested that alignments generated using posterior probabilities tend to be more accurate than alignments generated using a scoring table. Third, when different alignment programs agree with each other, there is high chance that the aligned bases are part of the true alignment.

The above observations can be visualized in Figure 5.1 and 5.2 through two pairs of homologous ncRNAs. In both figures, every path corresponds to an alignment. Besides the reference alignment, which is the true alignment from reliable ncRNA annotation databases, the alignments pro-

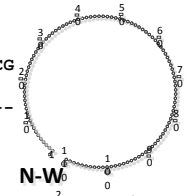
duced using four sequence alignment programs including Needleman-Wunsch [78], ClustalW [25], ProbA [70], and dpswalign [40] are presented. All paths significantly overlap in Figure 5.1 because the sequence similarity is high. For ncRNAs with low sequence similarity (Figure 5.2), two alignment programs that use scoring matrix (Needleman-Wunsch and ClustalW) produced highly different alignments. Similarly, although ProbA and dpswalign both employ posterior probabilities, they produced divergent alignments. In addition, the difference between sequence alignments and the reference alignment is pronounced. Moreover, the difference is not a constant along the path. It is changing for different rows in the matrix graph. Towards the end of the alignment paths, different alignment programs converge; the divergence between sequence alignments and the reference alignment is diminished.

```
Reference Alignment
>K01389.1_345-433
GGAGAGCUGUCCGAGU-GGUCGAAGGAGCACGAUUGGAAAUCGUGU-AGGCGGUCAACUCCGUCU--CAAGGGUUCGAAUCCCUUGCUCUCCG
>X02172.1_335-407
AGGAUAUUAGCUUAAUUGGU--AUAGCAUUCGUUUUACACACGAAAG-----------------AUUAUAGGUUCGAAUCCUAUAUUUCCUA
```




```
dpswalign Alignment
>K01389.1_345-433
-GGAGAGCTGTCCGAGTGGTCGAAGGAGCACGATTGGAAATCGTGTAGGCGGTCAACTCCGTCTCAAGGGTTCGAATCCCTTGCTCTCCG
>X02172.1_335-407
AGGATATTAGCTTAATTGGTAT----A---------GCATTCGTTTTACA----CACGAAAGATTATAGGTTCGAATCCTATATTTCCTA
```


```
Needleman-Wunsch Alignment
>K01389.1_345-433
GGAGAGCUGUCCGAGUGGUCGAAGGAGCACGAUUGG---AAAUCGUGUAGGCGGUCAACUCCGU----CUCAAG----------GGUUCGAAUCC----CUUGCUCUCCG
>X02172.1_335-407
---------------------AGGA----UAUUAGCUUAAUUGGUAUAG-----CA--UUCGUUUUACACACGAAAGAUUAUAGGUUCGAAUCCUAUAUUUCCUA----
```




```
ClustalW Alignment
>K01389_1_345-433
GGAGAGCUGUCCGAGUGGUCGAAGGAGCACGAUUGGAA-AUCGUGUAGGCGGUCAACUCCGUCUCAAGGGUUCGAAUCCCUUGCUCUCCG
>X02172_1_335-407
-----------------AGGAUAUUAGCUUAAUUGGUAUAGCAUUCGUUUUACACACGAAAGAUUAUAGGUUCGAAUCCUAUAUUUCCUA
```


```
ProbA Alignment
>K01389.1_345-433
GGAGAGCUGUCCGAGUGGUCGAAGGAGCACGAUUGGAAAUCGUGUAGGCGGUCA-ACUCCGUCUCAAGGGUUCGAAUCCCUUGCUCUCCG
>X02172.1_335-407
-----------------AGGAUAUUAGCUUAAUUGGUAUAGCAUUCGUUUUACACACGAAAGAUUAUAGGUUCGAAUCCUAUAUUUCCUA
```

Figure 5.3: The reference alignment and alignments generated by four alignment programs for a pair of tRNAs with sequence similarity 40%. The predicted secondary structures are shown when each alignment is used as input to Pfold, a structure prediction program.

Errors in alignments will be propagated into downstream analysis such as ncRNA gene finding and secondary structure prediction. A number of secondary structure prediction programs take alignments as input and output the consensus secondary structure. Erroneous alignments lead to wrong structure prediction. Figure 5.3 details the alignments corresponding to the paths in Figure 5.2 and the predicted structures using these alignments as input to Pfold [46]. The structure prediction can output the correct clover-leaf structure for the reference alignment. Other alignments did not lead to correct secondary structures. Thus, it is important to have accurate alignments.

In this work, our goal is to develop a sequence alignment program that can produce alignments as close to the reference alignment as possible. We achieved this goal by first choosing an existing sequence alignment as the base alignment. Then, we employed machine learning methods to determine how we should change the base alignment so that it overlaps or gets closer to the reference alignment. The features used in the machine learning methods were generated from the key observations, including sequence similarity and the consistency between various sequence alignment programs.

The remaining of the Method section is organized as follows. First, we formally define the alignment path and the quantification of the similarity between alignments. Then, we will briefly introduce posterior probability and our choice of the base alignment. Finally, we focus on the method of modifying and improving the base alignment.

5.3.1 Alignment path and similarity between two alignments

For a pair of sequences x and y, a directed matrix graph G of size $m \times n$ can be created, where m and n are the sizes of x and y, respectively. The ith base in x and the jth base in y defines a node, represented by a tuple, (x,y) in G. Three types of directed edges exist, corresponding to match/mismatch, insertion, and deletion in an alignment. Thus, a node with index (x,y) has three

directed edges to nodes $(x+1, y+1)$, $(x+1, y)$, and $(x, y+1)$. Any legal alignment between x and y can be visualized as a path in G. The similarity between two alignments can thus be represented by the distance of their corresponding paths, as suggested by Will et al. [111].

For each row r in the matrix graph, suppose the nodes have position (r, c) and (r, c') in alignment 1 and 2, respectively. The distance from alignment 1 to 2 at row r is thus $c' - c$, which also defines the extension needed to convert one alignment into the other one at row r. More generally, both alignments may span multiple columns for a row r. Let the node set of alignment 1 at row r be $\{(r, c_1), (r, c_2), \ldots, (r, c_m)\}$; the node set for alignment 2 at row r be $\{(r, c'_1), (r, c'_2), \ldots, (r, c'_n)\}$. The distance from alignment 1 to alignment 2 at row r is defined as:

$$d_r = max_{i=1\ to\ m}(min_{j=1\ to\ n}(c'_j - c_i)), \tag{5.1}$$

which mimics the extension needed between consecutive insertion blocks in sequence 1. In all experiments, we used the longer sequence as the Y-axis and thus maximumly avoided this case.

As one alignment contains multiple rows (equal to the length of one input sequence), the final distance between two alignments is :

$$sum_{r=1\ to\ m} d_r \tag{5.2}$$

where m is the number of rows. Intuitively, this distance is similar to edit distance between two sequences. It approximates the number of movements needed to convert one alignment path into the other. For example, Figure 5.4.(B) shows two alignment paths, one consisting of nodes with shape $\times$ and the other consisting of nodes $\bigcirc$ and $\square$. According to the above definition, the distance from the alignment represented by $\times$ to the other is thus $(3-1)+(4-1)+(5-2)+(7-2)+(7-2) = 18$. Note that for row 3, the distance is 3 according to the definition of d_r.

5.3.2 Choosing the base alignment

Our algorithm needs to use a sequence alignment as the starting alignment, which is referred to as "base alignment". Previous work [37, 21, 57] suggested that sequence alignments produced using posterior probabilities share the smallest distance with the true structural alignments. We re-evaluate this conclusion on highly structured ncRNAs. In particular, we compare various sequence alignment programs and choose the one with the best accuracy. In this paragraph, we briefly introduce the posterior probability and two alignment programs we chose for generating this alignment.

For input sequences x and y, the posterior probability of the ith base of x (i.e. x_i) being aligned with the jth base of y (i.e. y_j) is denoted as

$$Pr(x_i \sim y_j | x, y), \tag{5.3}$$

where $\sim$ represents an alignment. This probability can be computed using two methods. One is based on a pair-HMM [27] and the other is based on partition function. Both methods compute the posterior probability for all pairs of bases in x and y. The posterior probability table replaces the scoring matrix during the dynamic programming equations for conducting sequence alignment. In addition, the gap penalty is usually set to zero. Both global and local posterior probability-based alignment algorithms exist. The running time complexity for global alignment is $O(|x||y|)$, where $|x|$ and $|y|$ are the sizes of sequence x and y, respectively. As there are literatures available for describing both methods, we refer users to [27] for details about computing posterior probabilies.

In this work, we tested two different implementations of posterior probability-based sequence alignment. One is probA [70], which is based on partition function. The other is dpswalign [40], which is based on pair-HMMs. Because of different parameters and methods used for calculating

Table 5.1: Number of best alignments generated by four sequence alignment tools for 662 pairs of tRNAs and Riboswitches. Note that N-W is Needleman-Wunsch. ClustalW, a multiple sequence alignment tool, can also be applied to a pair of sequences.

Alignment program	# best alignments
Needleman-Wunsch	72
ClustalW	107
ProbA	73
dpswalign	289
N-W and ProbA	3
ProbA and ClustalW	6
ProbA and dpswalign	41
dpswalign and ClustalW	22
ProbA, dpswalign, and ClustalW	41
N-W, ProbA, and dpswalign	1
N-W, ClustalW, ProbA, and dpswalign	7

posterior probabilities, they tend to output different alignments.

We compared four sequence alignment programs on 662 pairs of homologous tRNAs and Riboswitches. We used tRNAs and Riboswitches with consistent secondary structure annotation from Rfam [16] and BRAliBase2.1 [112]. Of the four alignments generated for each pair by four programs, the best alignment is defined as the one with the smallest distance to the reference alignment. Table 5.1 lists the number of best alignments generated by each tool. In some pairs of sequences, some tools give the same smallest distance to reference alignment. For example, there are three pairs of tRNAs and Riboswitches that Needleman-Wunsch (N-W) and ProbA have the same smallest distance to the reference alignment. In Table 5.1, these numbers show that posterior probability-based alignments tend to be closer to the reference alignments. Of ProbA and dpswalign, the latter yields better performance. Thus, we choose the alignment generated by dpswalign as the base alignment.

5.3.3 Improving alignment accuracy using machine learning methods

The base alignment shares similarity with the reference alignment. We modify the base alignment to construct a new alignment that is expected to largely overlap the reference alignment. The modification is conducted by estimating the distance between the base alignment and the reference alignment for each row in the matrix graph. The distance is computed using machine learning methods by incorporating the key observations described at the beginning of Section 5.3. Specifically speaking, let r be a row index in the matrix graph. Suppose that the node at row r in the base alignment path is (r,x). The node at row r in the reference alignment path is (r,y). The goal is to change x into x' so that the distance between the new node (r,x') and the node (r,y) in the reference alignment is minimized. In the ideal case, if $x' == y$, the distance is zero, indicating the overlap of the nodes at row r. We applied machine learning methods to training data, which contains both base alignments and reference alignments and thus the known distance $y-x$ for each row. The trained model is then applied to test data to construct new alignments.

There are two steps in constructing a new alignment. First, features extracted from these information are used as input to machine learning methods and the specific extension is then calculated for each row. All these extensions roughly sketch a new alignment. However, some of them may not be part of a legal alignment path. For example, the extension in row 3 in Figure 5.4 is not part of a legal alignment path. Thus, in the second step, we fix extensions to construct a legal alignment.

There are 4 features we used as input to classification or regression models.

- F1: sequence similarity, a number between 0 to 1.
- F2-F4: pairwise distance between the three alignments generated by ClustalW, ProbA, and dpswalign. There are three distances between three alignments. These features are row specific. For example, for row r in the matrix graph, let the nodes in paths corresponding to

the alignments generated by ClustalW and dpswalign be (r,w) and (r,z), respectively. We have $F2 = w - z$. Note that the direction of the distance is also included. For the training data, we found that this number can be quite large for some input pairs. The largest distance for a row is 71, which means that two alignments are very far away from each other.

For each pair of sequences, F1 is evaluated by counting the ratio of identical bases in the global alignment generated by dpswalign. Features F2, F3, and F4 are computed for each row. The trained model is applied to each row in a matrix graph. The output value will be the extension value for the node in the base alignment.

Intuitively, the smaller the sequence similarity is, the larger the distance between the sequence alignment and the reference alignment paths is. We only chose three sequence alignment programs because of their overall good performance. Moreover, the three programs have very different implementations and thus complement with each other. When the pairwise distances between the three programs are all small, three different programs have a consensus on the alignment and thus the distance between the base and reference alignments tends to be small.

Theoretically speaking, both classification models and regression models can be trained. For classification models, we used 10 classes with labels: ± 1, ± 2, ..., ± 10. Note that the actual difference can be larger than 10. However, a majority of them should be smaller than 10. We tested multiple classification models including decision tree, Naive Bayes, and SVM. The 10-fold cross-validation training shows that a simple decision tree works best for this problem. Thus, we report the results using decision tree. For regression models, we used the values for F1 to F4 as input and the output is used as the extension. We tested different types of simple and complicated regression models. Here, we report the result of linear regression model because it yields better accuracy than other regression models. All model training was conducted using WEKA package [36].

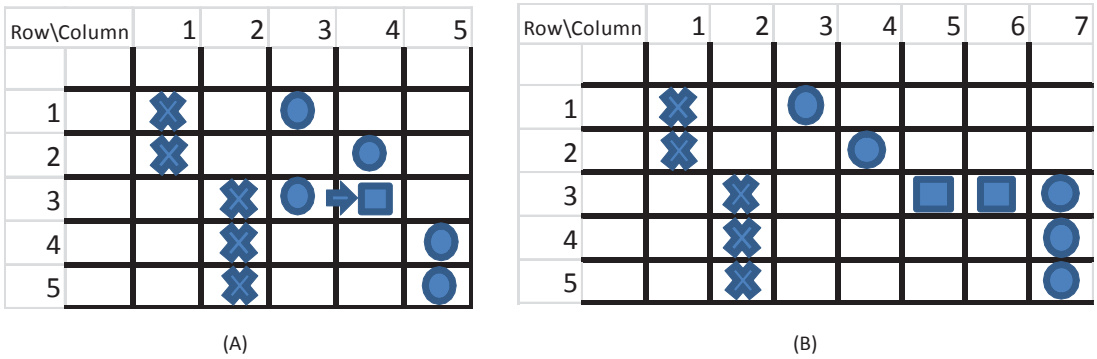


Figure 5.4: Construct a legal alignment using local search. $\times$ represent nodes in the base alignment. ◯ represent new positions of nodes in base alignment after extension. (A). node (3,3) is moved to (3,4), which is represented by □. (B). □ represent inserted nodes.

5.3.3.1 Constructing a legal alignment

The above machine learning methods generate a new position for each node in the base alignment if the computed extension value is not zero. The extension step does not evaluate whether the new positions are part of a legal alignment. It is possible that the new nodes are not in any legal alignment path. For example, Figure 5.4 shows two cases of illegal alignments. In both panels, the nodes indicated by circles are the results of extension. They do not form legal alignment paths. Moreover, the two examples represent two general cases. According to the standard glossary of graph theory, a node u is the direct successor of a node v if there is an edge from v to u in the path of interest. Figure 5.4.(A) illustrates the case that the new position of a node is smaller than the column index of the expected direct successor node. Specifically, node (3,2) is changed into node (3,3) because the trained model outputs an extension value of 1 for row 2. However, node (2,4) only accepts three possible positions for its successor node: (3,5), (2,5), or (3, 4). Figure 5.4.(B) shows the case that the extended node has much larger column index than the expected direct successor node. Specifically, for row 3, the expected direct successor node of (2,4) should be one of $(3,5),(2,5),(3,4)$. However, the extension creates a new position (3,7). We fix the illegal path using different strategies for these two cases. For case 1, we change the position of the

node that is not part of the legal alignment. In particular, we move it to right because its column index is smaller than the corresponding node in a legal alignment. Thus, node (3,3) is moved to node (3,4), as indicated by the square node in Figure 5.4.(A). For case 2, as the column index is much larger than the expected node, we insert new nodes, as indicated by the two square nodes in Figure 5.4.(B).

5.3.4 Running time analysis

The running time of this method includes three parts. The first part depends on the program that generates the base alignment using posterior probabilities. The second part is to apply the trained regression or classification model to compute the extension value for each row. Because we only have four features, this step is efficient. The running time is decided by the length of the alignment path. The third step is to examine whether the extended nodes form a legal alignment. If not, the nodes will be modified to form a legal alignment. The time complexity of the last step is linear to the size of the alignment path. Overall, the first step dominates the running time of glu-RNA. As we use posterior probability to generate the base alignment, the overall running time complexity is $O(mn)$, where m and n are the sizes of input sequences, respectively.

5.4 Experimental Results

5.4.1 Data

For ncRNAs with high sequence similarity, such as homologous mature miRNAs, existing sequence alignment programs can produce reliable alignments. Our focus is highly structured ncRNAs that may lack strong sequence similarity. For these types of ncRNAs, their sequences may

evolve faster than their secondary structures and thus pose hard cases for sequence alignment programs. For example, in the recent realignment work by Will et al. [111], the whole genome alignments were examined. Only regions that showed low alignment quality will be re-aligned. Thus, we focus on testing our method on highly structured ncRNAs.

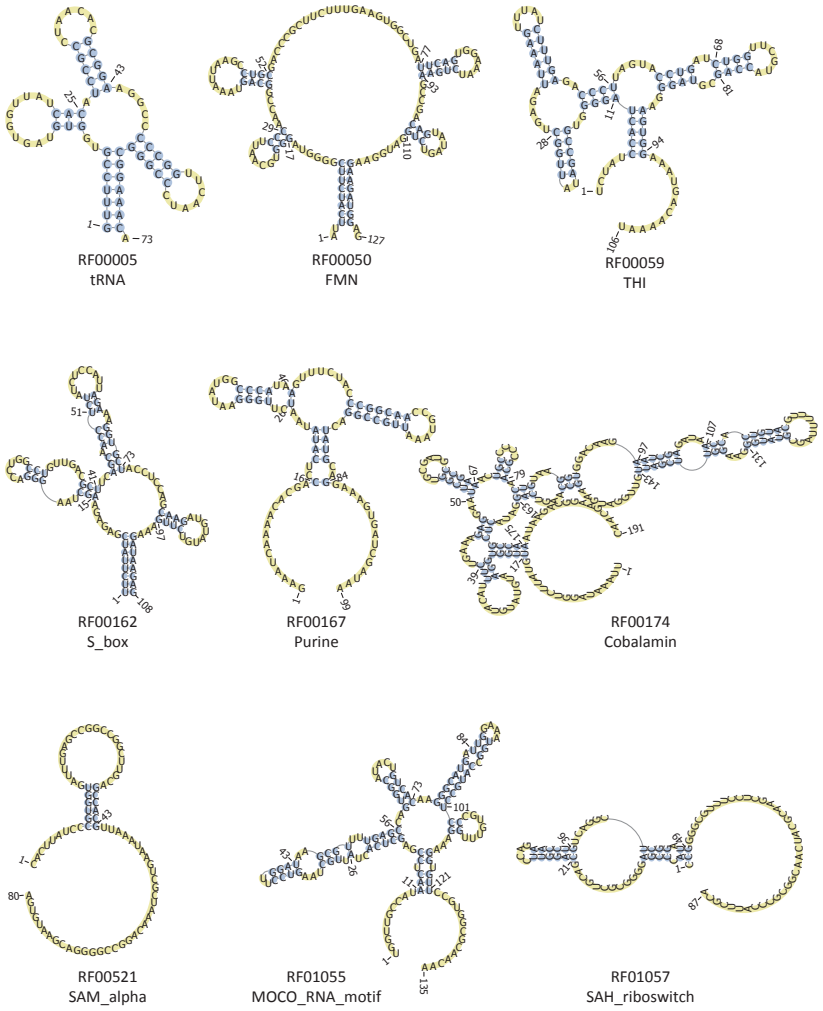


Figure 5.5: Types of ncRNAs and their consensus secondary structures (tRNA and riboswitches).

Table 5.2: Information of the Training Data Set

Rfam Family	ncRNA Type	Number of ncRNA pairs	Sequence Identity	Average Length
RF00005	tRNA	50	25%-65%	74.01
RF00050	FMN	100	35%-85%	134.50
RF00059	THI	50	20%-85%	109.99
RF00162	S_box	12	50%-85%	108.08
RF00167	Purine	100	45%-80%	100.85
RF00174	Cobalamin	50	30%-95%	203.17
RF00521	SAM_alpha	100	55%-95%	79.00
RF01055	MOCO_RNA_motif	100	25%-95%	141.90
RF01057	SAH_riboswitch	100	25%-90%	83.50

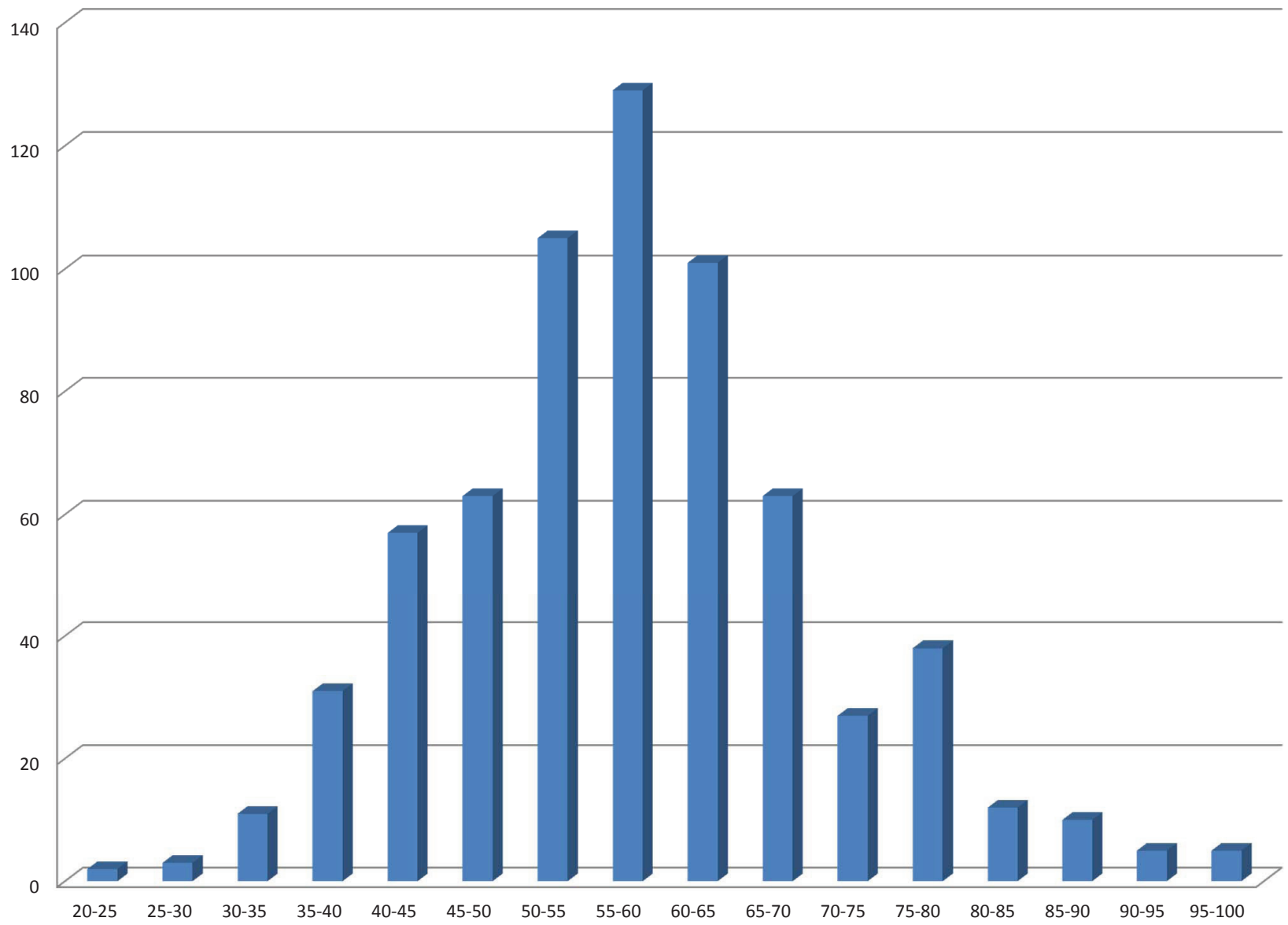


Figure 5.6: Sequence similarity histogram of training data set. X-axis represents the sequence similarity. Y-axis represents the number of ncRNA pairs.

5.4.1.1 Training data

To train the supervised learning models, we constructed our training data set using nine types of structured ncRNAs from BRAliBase2.1 [112] and Rfam 11.0 [16]. In BRAliBase 2.1, four types of ncRNAs were selected: tRNA, THI, S_box, and Cobalamin. As Rfam contains more structured ncRNAs, we chose five more riboswitch elements from seed alignment of Rfam. In total, we have tRNA and other riboswitch structures from FMN, THI, S_box, Purine, Cobalamin, SAM_alpha, MOCO_RNA_motif, and SAH_riboswitch. Riboswitch elements play an important role in transcription and translational regulation in prokaryotes. In addition, riboswitch families have complicated secondary structures and overall low sequence similarity [103, 118]. Thus, they were included in our training data. The consensus secondary structures of the nine types of families are shown in Figure 5.5. The numbers of pairs of sequences of the nine types of ncRNAs in our training data set are shown in Table 5.2. The pairwise sequence similarity histogram of the training data is presented in Figure 5.6.

For each pair of sequences in the training data, the true structural alignment is needed. BRAliBase provides quality alignments. For alignments extracted from the seed alignment of Rfam, we have high confidence in their pairwise alignments. First, these pairwise alignments are extracted from the multiple alignment of seed sequences in Rfam. Second, except SAM_alpha, the secondary structures of all other types of ncRNAs are published (as shown in Rfam alignment file). Thus, being part of qualify multiple sequence alignment, these pairwise alignments are not generated by any of the tested sequence alignment program and should not be biased towards any alignment program.

5.4.1.2 Testing data

To evaluate the performance of glu-RNA on quality alignments only, we constructed our testing data set from BRAliBase2.1. Five types of ncRNAs were chosen: tRNA, THI, Cobalamin, gcvT, and U2. Three types of ncRNAs (tRNA, THI, and Cobalamin) have the same types as training data set. However, all pairs of ncRNAs from this testing data set are different from those in the training data set. We chose tRNA, THI, and Cobalamin because of their complicated secondary structure. Moreover, the sequences of THI and Cobalamin are long. Two types of ncRNAs (gcvT and U2) were new and were not included in training data set. The secondary structures of gcvT and U2 are shown in Figure 5.7. The numbers of pairs of sequences of the five types of ncRNAs in our testing data set are presented in Table 5.3.

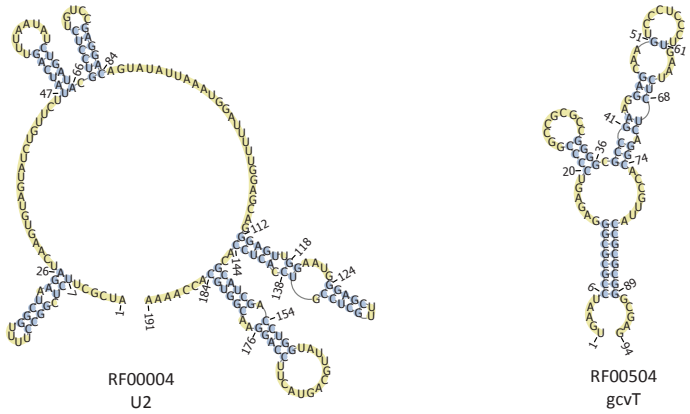


Figure 5.7: Consensus secondary structures of U2 and gcvT.

5.4.2 Performance comparison between different classification and regression models

There are multiple classification and regression models to choose from. Without knowing which model performs best for this problem, we used the output of cross-validation on the training data to

Table 5.3: Information of the Testing Data Set. Note that there is no overlap between the training and testing data.

Rfam Family	ncRNA Type	Number of ncRNA pairs	Sequence Identity	Average Length
RF00004	U2	60	40%-95%	185.82
RF00005	tRNA	60	25%-85%	74.76
RF00059	THI	120	20%-75%	108.55
RF00174	Cobalamin	50	25%-70%	204.09
RF00504	gcvT	70	25%-80%	99.72

choose the best model. All models were trained using Weka 3.6 [36]. For each pair of sequences in the training set, their sequence alignments were generated using dpswalign, ProbA, and ClustalW. For each row in the matrix graph, the distance between the node in the base alignment, generated by dpswalign, and the node in the reference alignment, was computed and used to evaluate the prediction performance of different methods.

We used 10-fold cross-validation, where 10 is the number of data set partitions. The training data in each fold must contain data of all classes. During each run, nine partitions were used for training and the remaining one was used for testing. This procedure was repeated 10 times so that each partition was used for testing exactly one. The accuracy for each run was reported using the following equation:

$$Accuracy = \frac{X}{Y} \tag{5.4}$$

where X is the number of rows for which the predicted distance (i.e. predicted extension) is the same as the known distance. Y is the total number of rows of all alignments in the training data set. The average accuracy was reported for 10-fold cross-validation.

We conducted the experiments with four standard classification models: Decision Trees, Naive Bayes, k-Nearest Neighbor (k-NN), and Support Vector Machines (SVM). As indicated in the Methods Section, the classes include 0, ± 1, ± 2, etc. For regression models, we did the experiments

Table 5.4: The accuracy of different models.

Machine Learning Model	Accuracy
Decision Tree	84.64%
Naive Bayes	70.14%
k-NN	79.95%
SVM	76.20%
Linear regression	80.99%
Isotonic regression	79.08%
Pace regression	73.37%

using Linear regression, Isotonic regression, and Pace regression. We used Weka to evaluate the performance of classification models and regression models with default parameters. The accuracy is summarized in Table 5.4.

According to Table 5.4, decision tree has the best accuracy among Naive Bayes, k-NN, and SVM. Thus, we used decision tree to determine the row specific extension in our implementation named glu-RNA. For regression models, linear regression has higher accuracy than isotonic regression and pace regression. To distinguish the implementation of our alignment program using linear regression model from decision tree, we name the former REglu-RNA, where RE represents regression model.

5.4.3 Performance comparison of different alignment programs

In this experiment, we benchmarked the performance of glu-RNA with other popular alignment programs on the testing data set. We applied glu-RNA, REglu-RNA, EMBOSS-Needle [78], ClustalW [25], ProbA [70], dpswalign [40], and Dynalign [37] to 360 ncRNA pairs of our testing data set presented in Table 5.3. EMBOSS Needle, ClustalW, ProbA, and dpswalign are global sequence alignment tools. Dynalign is a global structural alignment tool. For standard Needleman-Wunsch, we utilized EMBOSS Needle in EMBOSS package version 6.4.0 with gap opening penalty of 10 and gap extension penalty of 0.5. ClustalW was designed to generate global

multiple sequence alignment. In this experiment, we applied ClustalW to a pair of ncRNA sequences. We used ClustalW2 from ClustalW package version 2.1 with default parameters. ProbA is a pairwise sequence alignment tool using partition function-based posterior probabilities. We employed probA version 0.1.1 with default parameters. dpswalign is another posterior probability-based sequence alignment. However, it uses a pair-HMM to compute the posterior probabilities. We used dpswalign program from Ian Holmes' Dart package [40] with default parameters and two options: -pt and -oa. "-pt" option produces posterior-probability table between two sequences. "-oa" is used to generate the optimal accuracy alignment. As ClustalW, ProbA, and dpswalign are features in our models, we compared glu-RNA and REglu-RNA with these tools. The goal is to evaluate whether glu-RNA and REglu-RNA can generate more accurate alignment than these tools. Dynalign simultaneously predicts structure and alignment. It has the advantage of employing the mutual information of a pair of sequences to restrict the prediction of secondary structure. We used Dynalign program from RNAstructure package version 5.3 and ran Dynalign in the global mode. Note that there are a number of structural alignment programs. We chose Dynalign because it had carefully designed banding strategy for fast structural alignment. Thus, it is the most relevant structural alignment program for this work.

To evaluate the performance of different alignment programs, we used the the distance between the produced alignment and the reference alignment as defined in Section 5.3.1. The distance evaluates how close the predicted alignment to reference alignment. Small distance indicates high accuracy. There are 360 pairs of ncRNAs in the testing data set. Each program produces an alignment for each pair. We recorded the average distance on all pairs. glu-RNA has the best overall performance on 360 pairs, with average distance 273.219. The next lowest average distance is REglu-RNA (283.958). dpswalign and Dynalign have the third and fourth lowest average distance: 298.606 and 377.397. The average distances of different alignment programs on each family are

shown in Table 5.5. We have 4 out of 5 families that glu-RNA and REglu-RNA achieved the best performance.

Table 5.5: Average distances of different alignment programs on 360 pairs of ncRNAs in the testing data set. **Bold number** represents the lowest average distance for an ncRNA family. Underlined number shows the second lowest average distance for an ncRNA family.

Type	# test pairs	Needleman-Wunsch	ClustalW	probA	dpswalign	Dynalign	glu-RNA	REglu-RNA
U2	60	397.100	282.433	435.233	159.033	196.950	**155.217**	157.817
tRNA	60	440.967	184.800	232.167	156.400	**104.183**	133.583	134.633
THI	120	373.892	430.283	520.083	256.742	396.742	**234.125**	239.225
Cobalamin	50	1007.580	1323.960	1620.100	762.500	902.300	**645.120**	733.020
gcvT	70	421.200	608.429	670.943	280.543	358.157	295.429	**276.000**

Table 5.6: Average distances of different alignment programs on 58 pairs of sequences in five families with sequence similarity below 40%. **Bold number** represents the lowest average distance. Underlined number shows the second lowest average distance.

# test pairs	Needleman-Wunsch	ClustalW	probA	dpswalign	Dynalign	glu-RNA	REglu-RNA
58	1182.655	1510.793	1626.431	726.121	1101.517	**605.534**	667.345

Moreover, we tested the performance of alignment programs on sequences with sequence similarity below 40%. There are 58 pairs of sequences with sequence similarity below 40%. The average distances of tested alignment programs on these pairs are shown in Table 5.6. This experiment shows that glu-RNA can generate more accurate ncRNA alignments for sequence pairs with similarity below 40% than tested sequence alignment programs. In addition, all the above experiments show that glu-RNA generated more accurate alignments than Dynalign, which is a popular structural alignment program.

5.4.4 Discussion

Our program, glu-RNA, benefits from incorporating posterior probability and a machine learning approach. Figure 5.8 shows the alignments of a THI pair with sequence similarity 45% generated by glu-RNA, dpswalign, and Dynalign. We use the alignment generate by dpswalign as the base alignment and improve the accuracy by using trained models. This figure is a representative example showing that glu-RNA can generate more accurate alignments than dpswalign and Dynalign.

Our alignment program, glu-RNA, can generate an accurate ncRNA alignment relying only on sequence similarity. Thus, it has the same running time complexity as other sequence alignment programs. Here, we briefly compare glu-RNA and the structural alignment program Dynalign. On short ncRNAs such as tRNA in the testing data set, the average running time of glu-RNA is 32.88s while the average running time of Dynalign is 490.49s. When the length of sequence increases, the average running of Dynalign increases fast. In the Cobalamin data set, the average running time of glu-RNA is 36.26s. However, the average running time of Dynalign is 143,014s. Thus, for large input data such as those from metagenomics, applying glu-RNA is expected to be more promising to identify novel ncRNA clusters.

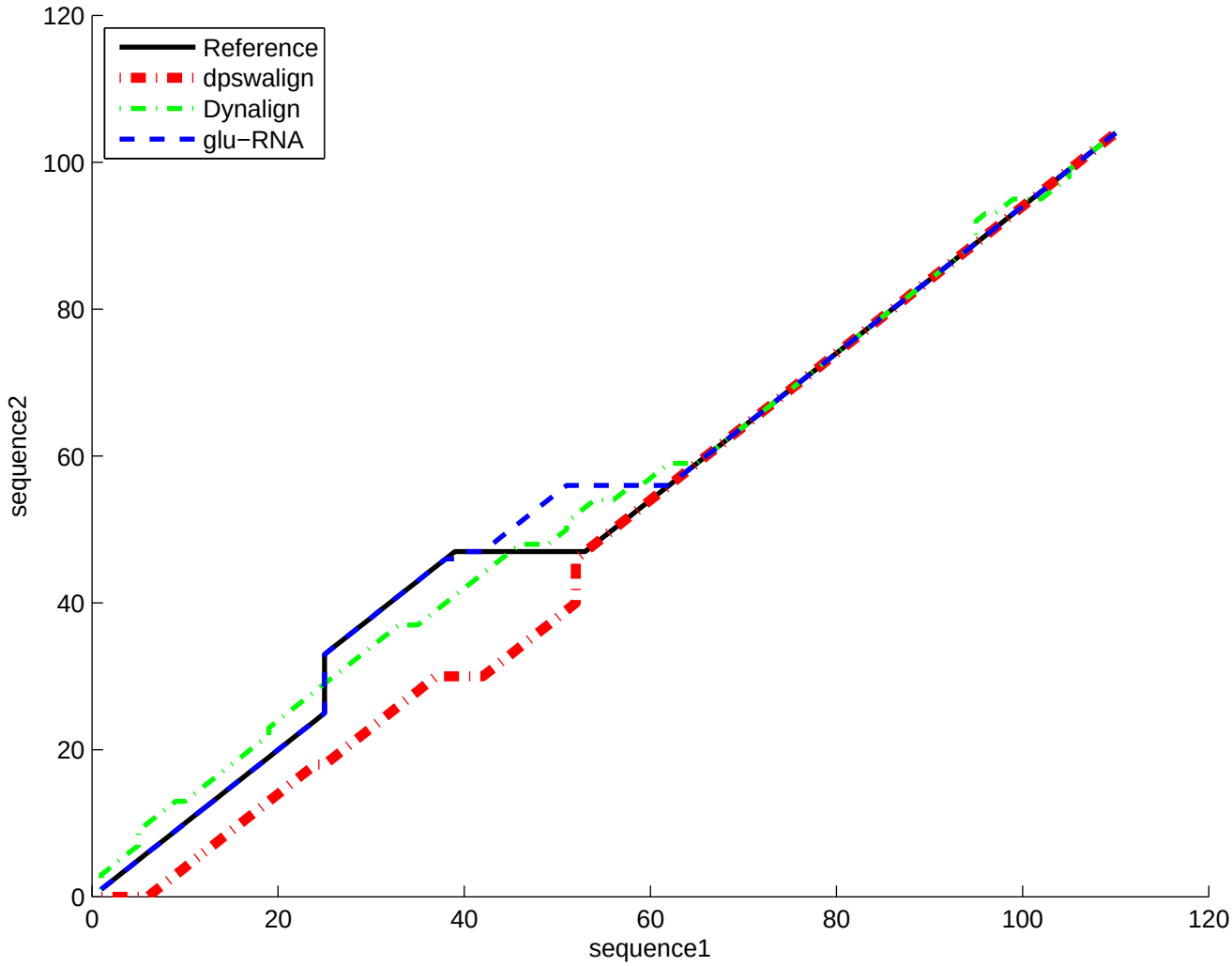


Figure 5.8: Comparing the alignments generated by glu-RNA, dpswalign, and Dynalign to the reference alignment of a THI pair with sequence similarity 45%. glu-RNA can produce more accurate alignment than dpswalign and Dynalign.

5.5 Conclusion and future work

We presented an accurate ncRNA alignment program that only relies on sequence similarity. Although it is conceptually easy, it produces alignments with higher accuracy than commonly used sequence alignment programs. Even without using any structural information, our program achieved slightly better performance than a widely used structural alignment program. The alignment program has quadratic running time complexity, which is both theoretically and practically

faster than standard structural alignment programs. The alignments can be used as input to ncRNA gene finding tools and can also be used for ncRNA secondary structure prediction. For example, the poorly aligned regions in existing pairwise genome-scale alignments can be realigned by our method before ncRNA gene finding tools are applied.

Our program can be further improved. First, we used the classification models and regression models conveniently incorporated in Weka. A more rigorous study about choosing the best model for this problem is needed. Second, the current method treats each row separately, which is not true in true alignments. The predicted extension values for consecutive rows have strong dependence with each other. Thus, a kth-order Markov model should be incorporated to improve glu-RNA. Third, the current implementation of our method can only be applied to a pair of sequences. We also plan to extend this method to handle multiple sequences simultaneously.

As part of our future work, we plan to apply our pairwise alignment program glu-RNA to short sequences produced in NGS data. One convenient application is to apply glu-RNA to cluster sequences from metagenomic data with the goal of detecting novel structural ncRNAs.

Chapter 6

Conclusion and Future Work

In this dissertation, we first introduced gene prediction and homology search aligning a sequence to protein families which are two essential steps for annotating functional elements in newly sequenced genomes. Then, three challenges in the fields of gene prediction, homology search, and ncRNA search are discussed. First, traditional gene prediction tools are not carefully designed to identify coding regions that have $5'$-$3'$ changing patterns. Second, NGS produces plenty short reads, existing homology search tools are not sensitive enough to align short reads that are part of remote homologs. Third, most existing structural alignment programs cannot be conveniently applied to NGS data. In order to address the challenges, we have developed three tools and compared the performance with existing programs.

First, I developed a tool named AUGUSTUS-GC, which was designed to accurately predict protein-coding regions that have various G+C content and $5'$-$3'$ changing patterns. It is more accurate than the existing gene identification programs such as AUGUSTUS. Second, I developed an approach to align short reads using paired-end read information. It can recover the missing end and determine the true domains. Third, I developed a program named glu-RNA, which was designed to accurately align highly structured ncRNAs employing only sequence similarity.

According to the previous studies, several directions can be improved. For gene prediction, some existing gene prediction tools can identify $5'$UTR and $3'$UTR regions. Currently, AUGUSTUS-GC is not able to identify UTR regions. We plan to extend AUGUSTUS-GC to accurately predict

5'UTR and 3'UTR regions. Existing gene prediction tools demonstrated that using extrinsic evidence derived from matches to an EST or protein database can improve the accuracy of gene prediction. We will further improve AUGUSTUS-GC accuracy by using hints from external sources if the data is available. Finally, we plan to provide users with a systematical way to better estimate lowT and highT for G+C contents based on the training data. For homology search, our current method relies on HMMER 3.0 at the first step. We plan to apply banded-viterbi algorithm to improve the sensitivity of short read homology search tool. For ncRNA search, current glu-RNA employed classification models and regression models from Weka. We can further improve glu-RNA by finding the better way to choose the best model for this problem. Furthermore, the current glu-RNA can be conducted pairwise alignment, we plan to extend to conduct multiple sequence alignments. We also plan to apply glu-RNA to reads generated by NGS data. For example, we can use glu-RNA to cluster short sequences from metagenomic data to detect novel structural ncRNAs.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] M. Alexandersson, S. Cawley, and L. Pachter. SLAM: Cross-Species Gene Finding and Alignment with a Generalized Pair Hidden Markov Model. *Genome Research*, 13(3):496–502, 2003.

[2] O. Aljawad, Y. Sun, A. Liu, and J. Lei. NcRNA Homology Search Using Hamming Distance Seeds. In *Proceedings of the 2Nd ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, BCB '11, pages 209–217, New York, NY, USA, 2011. ACM.

[3] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, Oct 1990.

[4] J. A.Martin and Z. Wang. Next-generation transcriptome assembly. *Nat Rev Genet*, 12(10):671–682, 2011.

[5] An Introduction to Next-Generation Sequencing Technology. http://www.illumina.com.

[6] Augustus download. http://augustus.gobics.de/binaries/old/.

[7] Augustus server. http://augustus.gobics.de/datasets/.

[8] V. Bafna and D. Huson. The conserved exon method for gene finding. *Proc Int Conf Intell Syst Mol Biol.*, 8:3–12, 2000.

[9] L. E. Baum, J. A. Eagon, et al. An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bull. Amer. Math. Soc*, 73(3):360–363, 1967.

[10] L. E. Baum and T. Petrie. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *The Annals of Mathematical Statistics*, 37(6):1554–1563, 1966.

[11] J. Besemer, A. Lomsadze, and M. Borodovsky. GeneMarkS: a self-training method for prediction of gene starts in microbial genomes. Implications for finding sequence motifs in regulatory regions. *Nucleic Acids Research*, 29(12):2607–2618, 2001.

[12] E. Birney and R. Durbin. Dynamite: a flexible code generating language for dynamic programming methods used in sequence comparison. *Proc Int Conf Intell Syst Mol Biol.*, 5:56–64, 1997.

[13] J. Buhler, U. Keich, and Y. Sun. Designing Seeds for Similarity Search in Genomic DNA. In *Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology*, RECOMB '03, pages 67–75, New York, NY, USA, 2003. ACM.

[14] C. Burge and S. Karlin. Prediction of complete gene structures in human genomic dna. *Journal of Molecular Biology*, 268(1):78 – 94, 1997.

[15] C. B. Burge and S. Karlin. Finding the genes in genomic dna. *Current Opinion in Structural Biology*, 8(3):346 – 354, 1998.

[16] S. W. Burge, J. Daub, R. Eberhardt, J. Tate, L. Barquist, E. P. Nawrocki, S. R. Eddy, P. P. Gardner, and A. Bateman. Rfam 11.0: 10 years of rna families. *Nucleic Acids Research*, 41(D1):D226–D232, 2013.

[17] F. Casals, Y. Idaghdour, J. Hussin, and P. Awadalla. Next-generation sequencing approaches for genetic mapping of complex diseases. *Journal of Neuroimmunology*, 248(1-2):10 – 22, 2012. Special Issue on New technologies for biomarker discovery in multiple sclerosis.

[18] S. E. Cawley, A. I. Wirth, and T. P. Speed. Phat–a gene finding program for Plasmodium falciparum. *Molecular and Biochemical Parasitology*, 118(2):167 – 174, 2001.

[19] K.-M. Chao, W. R. Pearson, and W. Miller. Aligning two sequences within a specified diagonal band. *Comput Appl Biosci.*, 8(5):481–7, 1992.

[20] S. Chikkagoudar, D. R. Livesay, and U. Roshan. PLAST-ncRNA: Partition function Local Alignment Search Tool for non-coding RNA sequences. *Nucleic Acids Research*, 38(Web-Server-Issue):59–63, 2010.

[21] S. Chikkagoudar, D. R. Livesay, and U. Roshan. PLAST-ncRNA: Partition function Local Alignment Search Tool for non-coding RNA sequences. *Nucleic Acids Research*, 38:W59–W63, 2010.

[22] P. G. Chis, Tiberiuand Harrison. *Analysing and Predicting Patient Arrival Times*, pages 77–85. Springer International Publishing, Cham, 2013.

[23] A. L. Delcher, D. Harmon, S. Kasif, O. White, and S. L. Salzberg. Improved microbial gene identification with GLIMMER. *Nucleic Acids Research*, 27(23):4636–4641, 1999.

[24] R. D. Dowell and S. R. Eddy. Efficient pairwise RNA structure prediction and alignment using sequence alignment constraints. *BMC Bioinformatics*, 7(400), 2006.

[25] J. D.Thompson, D. G.Higgins, and T. J.Gibson. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22(22):4673–4680, 1994.

[26] C. Dupont, D. Rusch, S. Yooseph, M. Lombardo, R. Richter, R. Valas, M. Novotny, J. Yee-Greenbaum, J. Selengut, D. Haft, A. Halpern, R. Lasken, K. Nealson, R. Friedman, and J. Venter. Genomic insights to SAR86, an abundant and uncultivated marine bacterial lineage. *The ISME Journal*, 6(6):1186–1199, 2012.

[27] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, UK, 1998.

[28] L. Duret, D. Mouchiroud, and C. Gautier. Statistical analysis of vertebrate sequences reveals that long genes are scarce in gc-rich isochores. *Journal of Molecular Evolution*, 40(3):308–317, 1995.

[29] S. R. Eddy. Accelerated Profile HMM Searches. *PLoS Comput Biol*, 7(10):e1002195, 10 2011.

[30] E. B. et al. Identification and analysis of functional elements in 1% of the human genome by the encode pilot project. *Nature*, 447(7146):799–816, 2007.

[31] R. D. Finn, A. Bateman, J. Clements, P. Coggill, R. Y. Eberhardt, S. R. Eddy, A. Heger, K. Hetherington, L. Holm, J. Mistry, E. L. L. Sonnhammer, J. Tate, and M. Punta. Pfam: the protein families database. *Nucleic Acids Research*, 42(D1):D222–D230, 2014.

[32] R. D. Finn, J. Mistry, B. Schuster-Bockler, S. Griffiths-Jones, V. Hollich, T. Lassmann, S. Moxon, M. Marshall, A. Khanna, R. Durbin, S. R. Eddy, E. L. L. Sonnhammer, and A. Bateman. Pfam: clans, web tools and services. *Nucleic Acids Research*, 34(suppl 1):D247–D251, 2006.

[33] P. P. Gardner, A. Wilm, and S. Washietl. A benchmark of multiple sequence alignment programs upon structural RNAs. *Nucleic Acids Research*, 33(8):2433–2439, 2005.

[34] S. S. Gross and M. R. Brent. Using Multiple Alignments to Improve Gene Prediction. *Journal of Computational Biology*, 13(2):379–393, 2006.

[35] D. H. Haft, J. D. Selengut, and O. White. The TIGRFAMs database of protein families. *Nucleic Acids Research*, 31(1):371–373, 2003.

[36] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[37] A. Harmanci, G. Sharma, and D. Mathews. Efficient pairwise RNA structure prediction using probabilistic alignment constraints in Dynalign. *BMC Bioinformatics*, 8(1):130, 2007.

[38] J. H. Havgaard, R. B. Lyngso, and J. Gorodkin. The FOLDALIGN web server for pairwise structural RNA alignment and mutual motif search. *Nucl. Acids Res.*, 33(suppl. 2):W650–653, 2005.

[39] J. HENDERSON, S. SALZBERG, and K. H. FASMAN. Finding Genes in DNA with a Hidden Markov Model. *Journal of Computational Biology*, 4(2):127–141, 1997.

[40] I. Holmes. Studies in probabilistic sequence alignment and evolution, 1998.

[41] IMG: Integrated Microbial Genomes. `http://img.jgi.doe.gov/`, 2011.

[42] A. M. Jeffrey and W. Zhong. Next-generation transcriptome assembly. *Nature Reviews Genetics*, 12:671–682, 2011.

[43] N. Jiang, A. A. Ferguson, R. K. Slotkin, and D. Lisch. Pack-Mutatorlike transposable elements (Pack-MULEs) induce directional modification of genes through biased insertion and DNA acquisition. *Proceedings of the National Academy of Sciences of the United States of America*, 108(4):1537–1542, 2011.

[44] N. C. Jones and P. Pevzner. *An introduction to bioinformatics algorithms*. Computational molecular biology. MIT Press, Cambridge (Mass.), London, 2004.

[45] S. Karlin and S. F. Altschul. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc Natl Acad Sci U S A.*, 87(6):2264–2268, 1990.

[46] B. Knudsen and J. Hein. Pfold: RNA secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Res*, 31(13):3423–3428, 2003.

[47] I. Korf. Gene finding in novel genomes. *BMC Bioinformatics*, 5(1):59, 2004.

[48] I. Korf, P. Flicek, D. Duan, and M. R. Brent. Integrating genomic homology into gene structure prediction. *Bioinformatics*, 17(suppl 1):S140–S148, 2001.

[49] L. Krause, A. C. McHardy, T. W. Nattkemper, A. Puhler, J. Stoye, and F. Meyer. GISMO-gene identification using a support vector machine for ORF classification. *Nucleic Acids Research*, 35(2):540–549, 2007.

[50] A. Krogh. Two methods for improving performance of an HMM and their application for gene finding. *Proc Int Conf Intell Syst Mol Biol*, 5:179–186, 1997.

[51] A. Krogh. Using Database Matches with HMMGene for Automated Gene Detection in Drosophila. *Genome Research*, 10(4):523528, 2000.

[52] A. Krogh, M. Brown, I. S. Mian, K. Sjlander, and D. Haussler. Hidden Markov models in computational biology: applications to protein modeling. *JOURNAL OF MOLECULAR BIOLOGY*, 235:1501–1531, 1994.

[53] D. Kulp, D. Haussler, M. Reese, and F. Eeckman. A generalized hidden Markov model for the recognition of human genes in DNA. *Proc Int Conf Intell Syst Mol Biol*, 4:134–142, 1996.

[54] B. Langmead, C. Trapnell, M. Pop, and S. Salzberg. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biology*, 10(3):R25, 2009.

[55] T. S. Larsen and A. Krogh. EasyGene – a prokaryotic gene finder that ranks ORFs by statistical significance. *BMC Bioinformatics*, 4(1):21, 2003.

[56] A. L. Lehninger, D. L. Nelson, and M. M. Cox. *Lehninger principles of biochemistry*. W.H. Freeman, New York, 2013.

[57] J. Lei, P. Techa-angkoon, and Y. Sun. NCRNA homology search based on an extended two-dimensional chain algorithm. *IEEE/ACM Trans Comput Biol Bioinform.*, 2012.

[58] D. Li, C.-M. Liu, R. Luo, K. Sadakane, and T.-W. Lam. MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. *Bioinformatics*, 31(10):1674–1676, 2015.

[59] T. Lingner, K. P. Aßhauer, S. F., and P. Meinicke. CoMet - a web server for comparative functional profiling of metagenomes. *Nucleic Acids Research*, 39(suppl_2):W518, 2011.

[60] L. Liu, Y. Li, S. Li, N. Hu, Y. He, R. Pong, D. Lin, L. Lu, and M. Law. Comparison of Next-Generation Sequencing Systems. *Journal of Biomedicine and Biotechnology*, 251364, 2012.

[61] A. V. Lukashin and M. Borodovsky. GeneMark.hmm: New solutions for gene finding. *Nucleic Acids Research*, 26(4):1107–1115, 1998.

[62] G. Lunter and M. Goodson. Stampy: A statistical algorithm for sensitive and fast mapping of Illumina sequence reads. *Genome Research*, 21(6):936–939, 2011.

[63] B. Ma, J. Tromp, and M. Li. PatternHunter: faster and more sensitive homology search. *BIOINFORMATICS*, 18(3):440–445, 2002.

[64] S. Mahony, J. O. McInerney, T. J. Smith, and A. Golden. Gene prediction using the Self-Organizing Map: automatic generation of multiple gene models. *BMC Bioinformatics*, 5(1):23, 2004.

[65] W. H. Majoros, M. Pertea, C. Antonescu, and S. L. Salzberg. GlimmerM, Exonomy and Unveil: Three ab Initio Eukaryotic Genefinders. *Nucleic Acids Research*, 31(13):36013604, 2003.

[66] W. H. Majoros, M. Pertea, and S. L. Salzberg. TigrScan and GlimmerHMM: two open source ab initio eukaryotic gene-finders. *Bioinformatics*, 20(16):2878–2879, 2004.

[67] E. R. Mardis. Next-Generation DNA Sequencing Methods. *Annual Review of Genomics and Human Genetics*, 9(1):387–402, 2008. PMID: 18576944.

[68] Y. Marquez, J. W. Brown, C. Simpson, A. Barta, and M. Kalyna. Transcriptome survey reveals increased complexity of the alternative splicing landscape in Arabidopsis. *Genome Research*, 22(6):1184–1195, 2012.

[69] D. H. Mathews and D. H. Turner. Dynalign: an algorithm for finding the secondary structure common to two RNA sequences. *Journal of Molecular Biology*, 317(2):191 – 203, 2002.

[70] U. Mckstein, I. L. Hofacker, and P. F. Stadler. Stochastic pairwise alignments. *Bioinformatics*, 18(suppl 2):S153–S160, 2002.

[71] M. L. Metzker. Sequencing technologies - the next generation. *Nat Rev Genet*, 11(11):31–46, 2010.

[72] F. Meyer, R. Overbeek, and A. Rodriguez. FIGfams: yet another set of protein families. *Nucleic Acids Research*, 37(20):6643–6654, 2009.

[73] I. M. Meyer and R. Durbin. Comparative ab initio prediction of gene structures using pair HMMs. *Bioinformatics*, 18(10):1309–1318, 2002.

[74] A. Mitchell, F. Bucchini, G. Cochrane, H. Denise, P. t. Hoopen, M. Fraser, S. Pesseat, S. Potter, M. Scheremetjew, P. Sterk, and R. D. Finn. EBI metagenomics in 2016 - an expanding and evolving resource for the analysis and archiving of metagenomic data. *Nucleic Acids Research*, 2015.

[75] B. Morgenstern, O. Rinner, S. Abdeddam, D. Haase, K. F. X. Mayer, A. W. M. Dress, and H.-W. Mewes. Exon discovery by genomic sequence alignment. *Bioinformatics*, 18(6):777–787, 2002.

[76] T. Namiki, T. Hachiya, H. Tanaka, and Y. Sakakibara. Metavelvet: an extension of velvet assembler to de novo metagenome assembly from short sequence reads. *Nucleic Acids Research*, 40(20):e155, 2012.

[77] E. P. Nawrocki and S. R. Eddy. Infernal 1.1: 100-fold faster RNA homology searches. *Bioinformatics*, 29(22):2933–2935, 2013.

[78] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–53, 1970.

[79] S. Ouyang, W. Zhu, J. Hamilton, H. Lin, M. Campbell, K. Childs, F. Thibaud-Nissen, R. L. Malek, Y. Lee, L. Zheng, J. Orvis, B. Haas, and J. Wortman. The TIGR Rice Genome Annotation Resource: improvements and new features. *Nucleic Acids Research*, 35:D883–D887, 2007.

[80] J. Park, K. Karplus, C. Barrett, R. Hughey, D. Haussler, T. Hubbard, and C. Chothia. Sequence Comparisons Using Multiple Sequences Detect Three Times as Many Remote Homologues As Pairwise Methods. *jmb*, 284(4):1201–1210, 1998.

[81] G. Parra, P. Agarwal, J. F. Abril, T. Wiehe, J. W. Fickett, and R. Guig. Comparative Gene Prediction in Human and Mouse. *Genome Research*, 13(1):108–117, 2003.

[82] G. Parra, E. Blanco, and R. Guigo. GeneID in Drosophila. *Genome Research*, 10(4):511–515, 2000.

[83] J. S. Pedersen, G. Bejerano, A. Siepel, K. Rosenbloom, K. Lindblad-Toh, E. S. Lander, J. Kent, W. Miller, and D. Haussler. Identification and classification of conserved RNA secondary structures in the Human Genome. *PLoS Comput Biol*, 2(4):251–262, 2006.

[84] Y. Peng, H. C. M. Leung, S. M. Yiu, and F. Y. L. Chin. Meta-IDBA: a de novo assembler for metagenomic data. *Bioinformatics*, 27(13):i94–i101, 2011.

[85] E. Prestat, M. M. David, J. Hultman, N. Tas, R. Lamendella, J. Dvornik, R. Mackelprang, D. D. Myrold, A. Jumpponen, S. G. Tringe, E. Holman, K. Mavromatis, and J. K. Jansson. FOAM (Functional Ontology Assignments for Metagenomes): a Hidden Markov Model (HMM) database with environmental focus. *Nucleic Acids Research*, 2014.

[86] M. Punta, P. C. Coggill, R. Y. Eberhardt, J. Mistry, J. Tate, C. Boursnell, N. Pang, K. Forslund, G. Ceric, J. Clements, A. Heger, L. Holm, E. L. L. Sonnhammer, S. R. Eddy, A. Bateman, and R. D. Finn. The Pfam protein families database. *Nucleic Acids Research*, 40(D1):D290–D301, 2012.

[87] L. Rabiner and B.-H. Juang. Fundamentals of speech recognition. 1993.

[88] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[89] E. Rivas and S. R. Eddy. Noncoding RNA gene detection using comparative sequence analysis. *BMC Bioinformatics*, 2(1):8–26, 2001.

[90] E. P. Rocha. Codon usage bias from tRNA's point of view: Redundancy, specialization, and efficient decoding for translation optimization. *Genome Research*, 14(11):2279–2286, 2004.

[91] A. A. Salamov and V. V. Solovyev. Ab initio Gene Finding in Drosophila Genomic DNA. *Genome Research*, 10(4):516–522, 2000.

[92] S. L. Salzberg, A. L. Delcher, S. Kasif, and O. White. Microbial gene identification using interpolated Markov models. *Nucleic Acids Research*, 26(2):544–548, 1998.

[93] F. Sanger and A. Coulson. A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase. *Journal of Molecular Biology*, 94(3):441 – 448, 1975.

[94] F. Sanger, S. Nicklen, and A. Coulson. DNA sequencing with chain-terminating inhibitors. *Proceedings of The National Academy of Sciences of The United States Of America*, 74(12):5463–5467, 1977.

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

[96] M. Shakya, C. Quince, J. H. Campbell, Z. K. Yang, C. W. Schadt, and M. Podar. Comparative metagenomic and rRNA microbial diversity characterization using archaeal and bacterial synthetic communities. *Environmental microbiology*, 15(6):1882–1899, 2013.

[97] A. M. S. Shrestha and M. C. Frith. An approximate Bayesian approach for mapping paired-end DNA reads to a reference genome. *Bioinformatics*, 29(8):965–972, 2013.

[98] M. Stanke. *Gene Prediction with a Hidden-Markov Model*. PhD thesis, Universitt Gttingen, 2003.

[99] M. Stanke, M. Diekhans, R. Baertsch, and D. Haussler. Using native and syntenically mapped cDNA alignments to improve de novo gene finding. *Bioinformatics*, 24(5):637–644, 2008.

[100] M. Stanke and B. Morgenstern. Augustus: a web server for gene prediction in eukaryotes that allows user-defined constraints. *Nucleic Acids Research*, 33(suppl 2):W465–W467, 2005.

[101] M. Stanke and S. Waack. Gene prediction with a hidden Markov model and a new intron submodel. *Bioinformatics*, 19(suppl 2):ii215–ii225, 2003.

[102] Y. Sun and J. Buhler. Designing multiple simultaneous seeds for DNA similarity search. In P. E. Bourne and D. Gusfield, editors, *RECOMB*, pages 76–84. ACM, 2004.

[103] Y. Sun, J. Buhler, and C. Yuan. Designing filters for fast-known NcRNA identification. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 9(3):774–787, 2012.

[104] Y. Tabei, K. Tsuda, T. Kin, and K. Asai. SCARNA: fast and accurate structural alignment of RNA sequences by matching fixed-length stem fragments. *Bioinformatics*, (22):1723–1729, 2006.

[105] L. Taher, O. Rinner, S. Garg, A. Sczyrba, M. Brudno, S. Batzoglou, and B. Morgenstern. AGenDA: homology-based gene prediction. *Bioinformatics*, 19(12):1575–1577, 2003.

[106] The Arabidopsis Information Resource (TAIR). http://www.arabidopsis.org.

[107] E. Torarinsson, M. Sawera, J. H. Havgaard, M. Fredholm, and J. Gorodkin. Thousands of corresponding human and mouse genomic regions unalignable in primary sequence contain common RNA structure. *Genome Research*, 16:885–889, 2006.

[108] T. Treangen, S. Koren, D. Sommer, B. Liu, I. Astrovskaya, B. Ondov, A. Darling, A. Phillippy, and M. Pop. MetAMOS: a modular and open source metagenomic assembly and analysis pipeline. *Genome Biology*, 14(1):R2, 2013.

[109] E. L. van Dijk, H. Auger, Y. Jaszczyszyn, and C. Thermes. Ten years of next-generation sequencing technology. *Trends in Genetics*, 30(9):418–426, 2014.

[110] S. Washietl, I. L. Hofacker, and P. F. Stadler. Fast and reliable prediction of noncoding RNAs. *Proc Natl Acad Sci USA*, 102(7):2454–2459, 2005.

[111] S. Will, M. Yu, and B. Berger. Structure-based whole genome realignment reveals many novel non-coding RNAs. *Genome Research*, 2013.

[112] A. Wilm, I. Mainz, and G. Steger. An enhanced RNA alignment benchmark for sequence alignment programs. *Algorithms for Molecular Biology*, 1:19, 2006.

[113] R.-F. Yeh, L. P. Lim, and C. B. Burge. Computational Inference of Homologous Gene Structures in the Human Genome. *Genome Research*, 11(5):803–816, 2001.

[114] Q. Yuan, S. Ouyang, A. Wang, W. Zhu, R. Maiti, H. Lin, J. Hamilton, B. Haas, R. Sultana, F. Cheung, J. Wortman, and C. R. Buell. The Institute for Genomic Research Osa1 Rice Genome Annotation Database. *Plant Physiology*, 138(1):18–26, 2005.

[115] I. Zarraonaindia, D. P. Smith, and J. A. Gilbert. Beyond the genome: community-level analysis of the microbial world. *Biol Philos*, 28(2):261–282, 2013.

[116] E. M. Zdobnov and R. Apweiler. InterProScan an integration platform for the signature-recognition methods in InterPro. *Bioinformatics*, 17(9):847–848, 2001.

[117] D. R. Zerbino and E. Birney. Velvet: algorithms for de novo short read assembly using de Bruijn graphs. *Genome Research*, 18(5):821–829, 2008.

[118] S. Zhang, I. Borovok, Y. Aharonowitz, R. Sharan, and V. Bafna. Sequence-based filtering method for ncRNA identification and its application to searching for riboswitch elements. *Bioinf.*, 22:e557–65, 2006.

[119] Y. Zhang, Y. Sun, and J. R. Cole. A Sensitive and Accurate protein domain cLassification Tool (SALT) for short reads. *Bioinformatics*, 29(17):2103–2111, 2013.

[120] Q.-Y. Zhao, Y. Wang, Y.-M. Kong, D. Luo, X. Li, and P. Hao. Optimizing *de novo* transcriptome assembly from short-read RNA-Seq data: a comparative study. *BMC Bioinformatics*, 12(Suppl 14):S2, 2011.

[121] X. Zhou, L. Ren, Q. Meng, Y. Li, Y. Yu, and J. Yu. The next-generation sequencing technology and application. *Protein and Cell*, 1(6):520–536, 2010.