FAST AND ACCURATE GENE PREDICTION BY PROTEIN HOMOLOGY

by

Rong She
Master of Science, Simon Fraser University, 2003
Bachelor of Engineering, Shanghai Jiaotong University, 1993

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

In the
School of Computing Science

SIMON FRASER UNIVERSITY

Spring 2010

APPROVAL

Name: **Rong She**

Degree: **Doctor of Philosophy**

Title of Thesis: **Fast and Accurate Gene Prediction by Protein Homology**

Examining Committee:

Chair: **Dr. Martin Ester, Professor, School of Computing Science**

_____

Dr. Ke Wang, Professor, School of Computing Science
Senior Supervisor

_____

Dr. Nansheng Chen, Associate Professor, Department of Molecular Biology and Biochemistry
Supervisor

_____

Dr. Cenk Sahinalp, Professor, School of Computing Science
Supervisor

_____

Dr. Jian Pei, Associate Professor, School of Computing Science
Internal Examiner

_____

Dr. Wyeth W. Wasserman, Professor, Department of Medical Genetics, UBC
External Examiner

Date Defended/Approved: Decembet 15, 2009

Declaration of
Partial Copyright Licence

ABSTRACT

The fast development of genome sequencing technologies has provided scientists with enormous amount of DNA sequences that keep increasing exponentially. The task of analyzing these DNA sequences and deducing useful knowledge from them remains challenging. One of the most important steps towards the understanding of genomes is gene prediction, which is determining the positions of genes and their components (including exons and introns) on the DNA sequence. There have been many attempts on computational gene prediction. The two main categories of gene prediction methods are *ab initio* methods and homology-based methods. The *ab initio* methods are usually sensitive in finding genes in novel genomes but often produce many false positives. The homology-based methods, on the other hand, usually have higher specificity, but are limited to finding genes that have homologous partners. With the accumulation of genome sequences of related species, there has been a growing demand for better and faster homology-based gene prediction programs.

In this thesis, I present a homology-based gene prediction framework that utilizes protein homology in determining positions of protein-coding genes. A protein sequence (the product of gene) is used as a query to help in finding genes that are homologous to the query protein. The framework consists of two major components. First, local alignments between the query protein and the genome are assembled into gene regions where potential homologous genes are

located. Next, each potential gene region is examined for gene signals and gene models are resolved by utilizing the alignment information provided by the local alignments. The experiments on genomes of two closely related species *Caenorhabditis elegans* and *Caenorhabditis briggsae* demonstrated that this method is both accurate and efficient. In particular, it runs hundreds of times faster than GeneWise, a popular homology-based gene prediction program, while being competitive in accuracy. Experiments have also been done on the human genome with a much larger size than *C. elegans* and *C. briggsae*, which showed similar performance behaviours of genBlast.

Keywords: gene prediction; protein homology; sequence alignment

DEDICATION

To my children

who always managed to put smiles on my face

ACKNOWLEDGEMENTS

This work could not have been done without the great support and continuous guidance from my senior supervisor, Dr. Ke Wang, who is always encouraging and has helped me in many ways. I am also deeply indebted to my supervisor, Dr. Nansheng Chen, who guided me with countless insightful discussions in every stage of the project development. I would also like to thank Dr. Cenk Sahinalp, my other supervisor, for his valuable feedbacks and comments. My gratitude also goes to Dr. Jian Pei and Dr. Wyeth Wasserman for serving on my thesis committee and giving me crucial guidance on improving the quality of this thesis. I would also like to thank all my fellow collaborators from Dr. Chen's Lab, Mr. Jeff Chu, Mr. Bora Uyar, Mr. Christian Frech, Mr. Ismael Vergara, for their great help and numerous feedbacks in collecting experimental results. Without the support from all these people, this work would never have been possible.

My special thanks also go to my two little kids, who at times can be the biggest challenges in my life, but somehow at the end of a very long day, they never fail to make me smile.

TABLE OF CONTENTS

Approval ii
Abstract iii
Dedication v
Acknowledgements vi
Table of Contents vii
List of Figures ix
List of Tables xi
Glossary xii

1: Introduction 1
1.1 Background and Motivations 1
1.1.1 Chromosome, DNA and Genes 2
1.1.2 Eukaryotic Protein-Coding Genes 4
1.2 Problem Definition and Challenges 7
1.3 A Novel Gene Prediction Framework 8
1.4 Thesis Organization 10

2: Gene Prediction: An overview 12
2.1 Ab Initio Gene Prediction 12
2.1.1 Content Sensors 13
2.1.2 Signal Sensors 14
2.1.3 Algorithms 15
2.1.4 Advantages and Problems 17
2.2 Homology-Based Gene Prediction 18
2.2.1 Homology-Based Methods Using Expressed Sequences (Proteins, cDNAs, ESTs) 19
2.2.2 Homology-Based Methods Using Comparative Genomics 22
2.2.3 Advantages and Limitations 22
2.3 Integrated Gene Prediction: Combining Ab Initio and Homology-based Predictions 23
2.4 Current Performances of Gene Prediction Programs 24
2.4.1 Evaluation Measures 24
2.4.2 Gene Prediction Evaluation 25
2.4.3 Summary 29
2.5 genBlast: A Novel Framework of Gene Prediction by Protein Homology 31
2.5.1 Motivation 31
2.5.2 genBlast Overview 32

3: genblastA: Finding Homologous Gene Regions 35
3.1 Finding Local Similarities 35

3.2 Filtering and Grouping HSPs 39
3.2.1 Challenges 39
3.2.2 Previous Attempts 41
3.2.3 genBlastA 43
3.3 genBlastA: The Methods 44
3.3.1 Problem Definition 44
3.3.2 HSP Groups 45
3.3.3 Graph Modelling 49
3.3.4 Finding the Best HSP Groups 54
3.3.5 Length Metrics 57
3.3.6 Graph Optimization 62
3.4 The Effectiveness of genBlastA 64
3.4.1 Test Genes and Experiment Setup 64
3.4.2 Resolving Paralogous Genes in Tandem Clusters 66
3.4.3 Searching for Orthologous Genes 70
3.4.4 Discussions 72

4: genblastG: Resolving Gene Structures 74
4.1 Problem Statement and Challenges 75
4.2 genBlastG Overview 76
4.3 Step 1: Determine Intron Regions 79
4.4 Step 2: Select Candidate Splice Sites 84
4.5 Step 3: Find Best Splice Sites 85
4.6 Step 4: Post Processing of Candidate Gene Structure 88
4.7 Discussions 91
4.8 Performance Evaluation 93
4.8.1 Experiment Setup 93
4.8.2 Results on *C. elegans* Genome 95
4.8.3 Results on *C. briggsae* Genome 99
4.8.4 Re-annotating the *C. briggsae* Genome 105
4.8.5 Results on the human genome 120

5: Conclusions 125
5.1 Discussions 125
5.2 Future Work 128

Bibliography 131
Appendix 1: genBlast Pseudocodes 144

LIST OF FIGURES

Figure 1. Eukaryotic genes on the DNA sequence 3
Figure 2. Functional structure of an eukaryotic protein-coding gene and its expression 5
Figure 3 GenBlast Overview 34
Figure 4 An example HSP (high-scoring segment pair) 36
Figure 5 Local Similarities Reported as HSPs 37
Figure 6 HSPs reported for a *C. elegans* gene C07F11.1 on Chromosome I 38
Figure 7 Challenges in Filtering and Grouping HSPs 41
Figure 8 Example HSPs 45
Figure 9 Example Groups of HSPs 47
Figure 10 The HSP Graph 51
Figure 11 The HSP Graph with logical edges identified 53
Figure 12 Example HSPs and the corresponding HSP graph 59
Figure 13 Grouping HSPs into groups representing homologous genes in tandem clusters 68
Figure 14 Comparison of genBlastA, ML and WU-BLAST in Resolving Tandem Genes 69
Figure 15 Query Coverage and Genomic Span Comparisons for Orthologous Gene Detection 73
Figure 16 HSPs and Their Exon Correspondences 80
Figure 17 Intron Regions between Adjacent HSPs 81
Figure 18 Intron Region inside a HSP 82
Figure 19 Intron Region between HSPs with Overlapping Query Segments 83
Figure 20 Finding the Best Pair of Donor and Acceptor 87
Figure 21 Adjusting the Initial Gene Structure 90
Figure 22 Running Time on *C. elegans* genome with Chromosome I genes as queries 97
Figure 23 Running Time on *C. briggsae* genome with *C. elegans* Chromosome I genes as queries 101
Figure 24 Comparisons of Query Alignment PIDs between genBlastG, WormBase, nGASP 104

Figure 25 Comparisons of Query Alignment PID between genBlastG and GeneWise 105
Figure 26 Gene Model Differences 106
Figure 27 Gene Model Split Cases 108
Figure 28 Gene Model Merge Cases 111
Figure 29 New Gene Models due to Isoforms 112
Figure 30 Gene Model Trimming/Extension Cases 114
Figure 31 Internal Exon Alteration Cases 117
Figure 32 Novel Genes 118
Figure 33 PCR Verification of Novel Genes 119
Figure 34 Running Time on the 75 human test genes 121
Figure 35 Query Alignment PID on the 75 human test genes 122
Figure 36 Average Running Time of genBlastG on the entire human genome 124

LIST OF TABLES

Table 1 Gene Prediction Accuracy in EGASP evaluations (human genome) [59] 28

Table 2 Gene Prediction Accuracy in nGASP evaluations (*C. elegans* Genome) [38] 30

Table 3 Length Distribution of *C. elegans* Chromosome I genes 96

Table 4 Accuracy Comparison on *C. elegans* Chromosome I genes (genBlastG vs. GeneWise) 99

Table 5 Accuracy Comparison on entire *C. elegans* genome (genBlastG vs. nGASP) 99

Table 6 Summary of 5 categories of gene revisions made by genBlastG (*C. briggsae* genome) 107

Table 7 Length Distribution of 75 test genes on the human genome 121

Table 8 Length Distribution of Genes on the entire human genome 123

GLOSSARY

acceptor	the end site of intron in a gene
alternative splicing	different ways of splicing on the same gene during gene expression
base	nucleotide, the basic building block of DNA sequence, represented by one of the 4 letters (A,T,G,C)
base pair	Two nucleotides on opposite strands of DNA are connected via hydrogen bonds, which is called a base pair. In particular, A usually pairs with T, and G usually pairs with C.
bp	The length of DNA is measured in bp, which is the number of base pairs in the DNA sequence.
cDNA	complementary DNA, experimentally obtained as a DNA copy of a mRNA by reverse transcription
CDS	coding region of a gene
chromsome	Chromosome is an organized structure that consists of DNA as well as DNA-bound protein (which serves to package the DNA and control its functions).
codon	3 adjacent bases on DNA strand
donor	the start site of intron in a gene
DNA	DNA is a double helix structure that consists of two strands in opposite directions to each other, with each strand being a sequence of nucleotides from the 4-letter alphabet (A,T,G,C).
EST	expressed sequence tags, one-shot subsequences of cDNA
exon	coding regions of a gene
frame	see “reading frame”

gene	basic unit of heredity that carries genetic instructions to synthesis proteins or other RNAs
genomic span	a region on DNA with a beginning and an end
genomic span similarity	the extent of overlap between two DNA regions R1 and R2, measured by Jaccard similarity (their intersection size divided by their union size): $\|R1 \cap R2\| / \|R1 \cup R2\|$
homolog, homologous gene	A homolog (homologous gene) is a gene that is similar to another gene due to common ancestry. In practice, the homology is often inferred by sequence similarity.
HSP	high-scoring segment pair, produced by sequence similarity search tools
intron	DNA regions between exons
isoform	different forms of protein that is coded by the same gene via alternative splicing
mRNA	messenger RNA, produced from gene transcription, used as templates for protein translation
multi-gene family	groups of genes from the same organism that encode proteins with similar sequences, usually have related functions
ortholog, orthologous gene	genes in different species that evolved from a common ancestral gene
paralog, paralogous gene	genes related by duplication within a genome
PID, Percentage of Identity	In sequence alignment, percentage of identity is the percentage of exact matches in the entire alignment.
protein, protein sequence	A protein is a chain of amino acids, usually represented as a sequence from 20-letter alphabet, each representing one type of amino acid. Proteins form the basis for most functions of cells.
reading frame	the start position of codon translation

sensitivity	the proportion of actual positives which are correctly identified as such
span similarity	see “genomic span similarity”
specificity	the proportion of true positives in the entire prediction
splice site	the splice junction between exon and intron, including donor site and acceptor site
splicing	the mechanism during gene expression, where the introns are removed from the pre-mRNA and produce mature mRNA
start codon	ATG
stop codon	TGA,TAA,TAG
tandem genes	duplicated genes that are located next to each other in close distances, forming tandem clusters
UTR	un-translated region in the transcribed mRNA

1: INTRODUCTION

1.1 Background and Motivations

The past decades have witnessed fundamental advances in genomics. In particular, genome sequencing projects have provided scientists with complete or essentially complete genome sequences of many organisms [5, 11, 22, 31, 37, 79, 91, 95, 113]. Since the publication of the first whole genome sequence of a free-living organism --- the bacterium *Haemophilus influenzae* [53] and the first animal --- the nematode *Caenorhabditis elegans* [31], genomes of more than a thousand species have been sequenced and thousands of more species are currently being sequenced, according to the Genome Online Database (http://genomesonline.org/).

Meanwhile, the cost and time for genome sequencing have been largely cut down by the fast development of new sequencing technologies since 2005 [16, 84, 86, 118]. The Human Genome Project [70, 79, 134] initiated in 1990 took 13 years to determine the human DNA sequences. Today, the “next generation” sequencing instruments are orders of magnitude faster and cheaper, which are capable of sequencing an entire human genome in a matter of weeks [17, 136, 140]. With instrumental advances in sequencing technologies, the volume of genomic data has skyrocketed. The amount of DNA sequences deposited into the sequence database *GenBank* has doubled every 18 months and continues to grow at an exponential rate [14, 15].

The accumulation of sequenced genomes is the first step towards deciphering genomes. The vast and ever-increasing amount of genome sequences available calls for development of sequence analysis tools that can quickly process these sequences and deduce meaningful knowledge [85, 114, 126]. One of the first and most fundamental tasks in understanding the sequenced genome of a species is gene prediction, which is the task of determining positions of genes and their components across the genome.

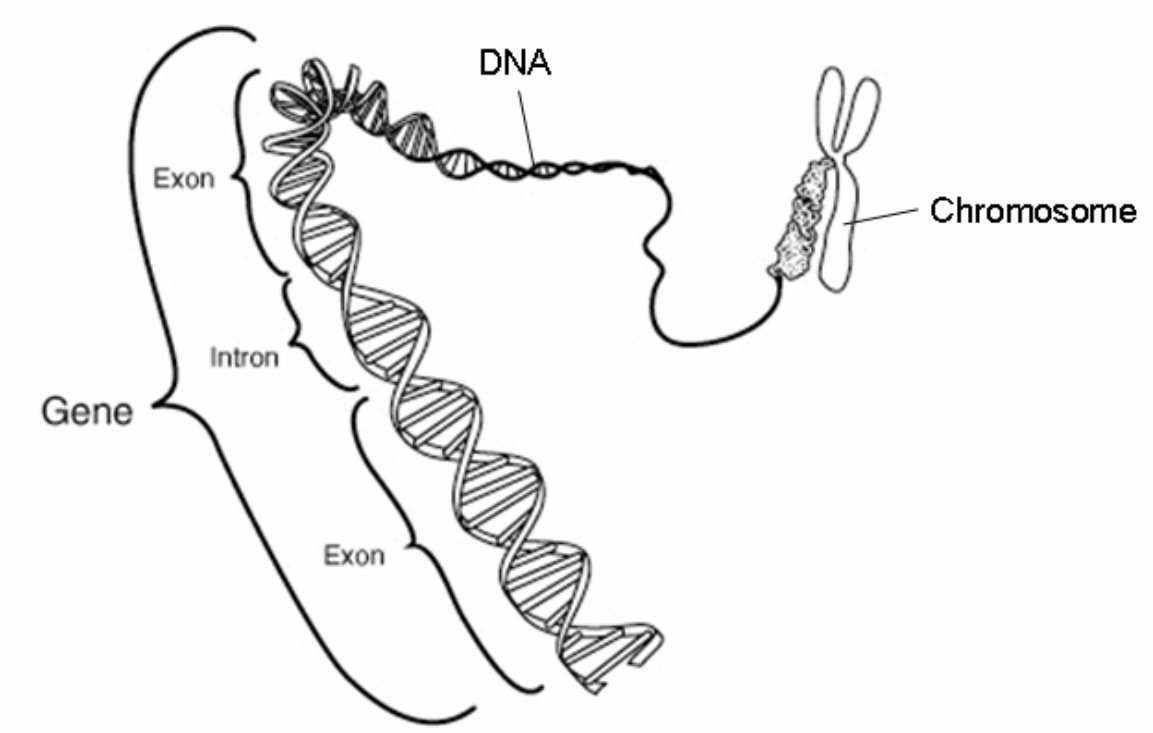
1.1.1 Chromosome, DNA and Genes

The genome sequence of an organism is the collection of all DNA sequences for each of the chromosomes in that organism. A *chromosome* is an organized structure that consists of a DNA sequence as well as DNA-bound protein (which serves to package the DNA and control its functions). For bacteria, which usually have just one chromosome, its genome is the DNA sequence of that chromosome [53]. On the other hand, humans, with 22 autosome pairs and 2 sex chromosomes [70, 79, 134], require 24 separate DNA sequences in order to represent the completed genome.

A DNA sequence is usually a double helix structure that consists of two strands in opposite directions to each other [138], as illustrated in Figure 1. Each DNA strand is made up of a sequence of nucleotides (or *bases*). There are four types of nucleotides that make up DNA sequences: adenine (A), thymine (T), guanine (G) and cytosine (C). Two nucleotides on opposite strands that are connected via hydrogen bonds are called a *base pair*. In particular, A usually pairs with T, and G usually pairs with C. Therefore, the two strands are

complementary to each other in that the sequence of one strand can be deduced from the sequence of its opposite strand, by the rule of base pairing. Thus each DNA sequence is represented by a single sequence from the four-letter alphabet (A,T,G,C). The size of a DNA sequence is commonly measured in base pairs (*bp*). The total number of base pairs is equal to the number of nucleotides in one of the strands. For example, the human genome consists of 3 billion base pairs in 22 autosomes and 2 sex chromosomes, with lengths ranging from 47 million to 247 million bp.

Figure 1. Eukaryotic genes on the DNA sequence[1]



The DNA sequence contains the genetic instructions used in the development and functioning of all known living organisms. The instructions

[1] modified from: http://en.wikipedia.org/wiki/File:Gene.png (in the public domain)

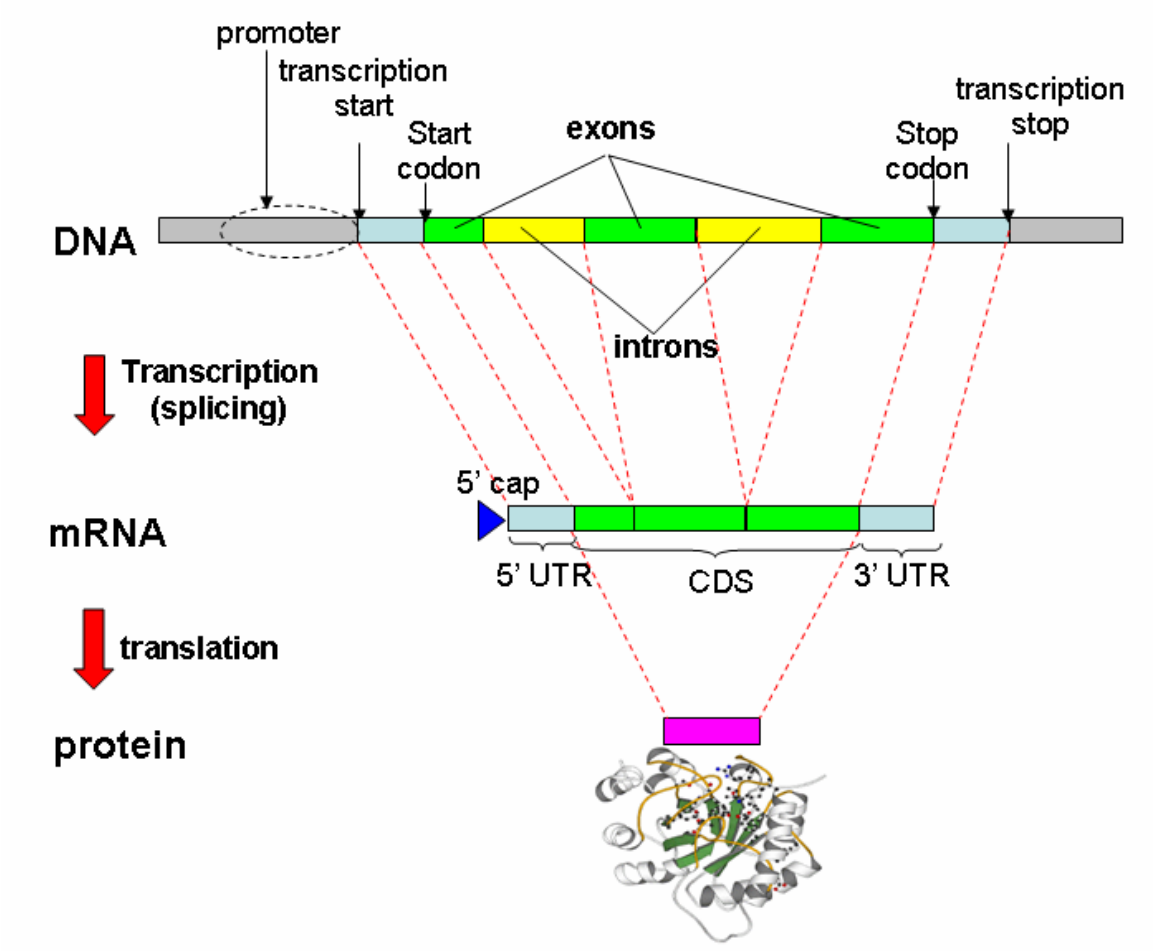
contained in DNA are needed to construct other components of cells, such as proteins and RNA molecules. The DNA segments that carry this genetic information are called *genes*. Genes are the basic units of heredity in living organisms and carry crucial information to build and maintain the biological function of cells and pass genetic traits to offspring. A DNA sequence can be millions of base pairs long, however, usually only a very small portion of the sequence contains genes in genomes of complex multi-cellular organisms, including humans [79].

Figure 1 illustrates the general concepts of chromosome, DNA and genes. A chromosome pair is shown. It contains two DNA sequences, and one of the DNA sequences is shown in more detail with a gene revealed as a DNA segment. A eukaryotic gene consists of both *exons* and *introns*, as opposed to the simpler structure of prokaryotic genes that do not contain introns. Thus predicting the gene structures of eukaryotic genes is more difficult and has been the focus of most gene prediction programs [29, 49, 82, 83, 87, 137, 144]. In this thesis, I will discuss the prediction of protein-coding genes in eukaryotes.

1.1.2 Eukaryotic Protein-Coding Genes

Protein-coding genes are the DNA segments that carry instructions to directly control the synthesis of *proteins*, which are critical parts of any organism and participate in virtually every biochemical process within cells. They make up the majority and most important class of genes, and thus have received the most attention in gene prediction studies. In the rest of this thesis, I will refer to the eukaryotic protein-coding genes simply as “genes”.

Figure 2. Functional structure of an eukaryotic protein-coding gene and its expression



1.1.2.1 Gene expression

Figure 2 shows the structure of a gene on a DNA sequence and the process of gene expression. The gene structure contains various components that play different roles during gene expression. The expression of the gene begins with its *transcription* into pre-mRNA, which then undergoes a process called *splicing*, in which introns (stretches of non-functional DNA within the gene) are removed and mature messenger RNA (mRNA) is produced. The mRNA is then used as a template for synthesizing protein in a process called *translation*, during which three nucleotides from the mRNA are read at a time and direct the

addition of a corresponding amino acid to the protein being synthesized according to the genetic code [42]. Thus the set of three adjacent nucleotides (or bases) is a unit of translation and is called a *codon*. The protein, the end product, is a sequence of amino acids, usually folded into a three-dimensional structure.

1.1.2.2 Gene structure

With the gene expression mechanism in mind, a gene can be defined as being composed of the transcribed region (between transcription start and transcript stop in Figure 2) and regulating regions (for example, the *promoter*, which provides a position that is recognized by the transcription machinery when a gene is transcribed and expressed). Within the transcribed region, the translated regions are called *CDS* (coding sequence) and the un-translated terminal regions at both sides of CDS are called *UTRs*. CDS consists of one or more *exons*. These are the regions that explicitly code for proteins. The regions between exons are called *introns*, which are removed during gene transcription. The junctions between exons and introns are called *splice sites*. There are two types of splice sites: the start site of an intron is called a *donor* site, and the end site of an intron is called an *acceptor* site.

In this thesis, the DNA position that corresponds to the translation start is called *gene start*. Protein-coding DNA usually starts with the specific 3-base sequence of “ATG”, which is called the *start codon*. The DNA position that corresponds to where the translation stops is signalled by other specific codons called *stop codons* (“TGA”, “TAG”, or “TAA”).

1.2 Problem Definition and Challenges

The gene prediction problem presented in this thesis is defined as the task of determining genomic positions of exons and introns in eukaryotic protein-coding genes. In this context, the gene region of interest is between the start codon and stop codon, which will be referred to as the positions of *gene start* and *gene stop*, respectively.

Identifying genes on the long DNA sequence with millions of base pairs is a challenging task, since genes occupy only a very small percentage of regions on the DNA sequence, especially in genomes of multi-cellular organisms. For example, protein-coding genes make up barely 2% of the human genome, which contains an absolute majority of DNA without an identified function [70, 79]. All gene prediction programs are limited by the current knowledge about genomes. Although scientists have made tremendous advances in the study of genes, much remains unknown about the exact machinery of genes and their expression.

The problem is further complicated by irregularities in gene structure. Exons are interrupted by introns, which may be very long. Exons may be very short. Some exons are as small as only 3 base pairs [32], which are easily missed by gene prediction programs. Genes may be long. For example, the largest human gene is composed of 79 exons spanning over 2 million base pairs. Over 99% of the gene is composed of introns, with some introns more than 100k base pairs long [98]. Genes may be within an intron region of another gene [65]. Genes may overlap on the same or opposite DNA strands [105]. *Alternative splicing* may take place during the expression of the same gene, producing

different gene products (*isoforms*) [21, 73, 88, 93]. Alternative splicing is a widespread form of regulation in eukaryotic cells, occurring in 80% of human genes [88]. These variations pose challenges to gene prediction.

1.3 A Novel Gene Prediction Framework

A protein is the gene product that is coded by the exon regions in the gene and has perfect correspondence with the CDS region, therefore, is sometimes loosely referred to as the "gene". With the existence of many genomes, many protein sequences have been derived from manual annotations or from full-length cDNAs that are experimentally obtained. These protein sequences can be used to aid the prediction of genes in other related genomes. Previous experimental studies have shown that gene prediction algorithms that make use of protein sequences generally perform better than other algorithms that do not take advantage of such data [38, 59]. In this thesis, I will present a novel gene prediction framework, *genBlast*, which makes use of homologous protein sequence in determining gene structures.

Homology between genes is usually inferred on the basis of sequence similarity. Local alignments between the protein sequence and the DNA sequence provide clues to genomic locations of exons. However, due to possible existence of multiple homologous genes, for example, genes in multi-gene families or tandem clusters [34, 108], the alignments may be scattered over many places and heavily overlap with one another. In addition, the local alignment algorithms are prone to producing irrelevant or extended alignments as the result of very conserved introns or other non-coding regions. Some local

alignments may simply reflect random noise aligned by chance, especially short sequences. On the other hand, because homologous genes may contain different DNA segments, especially in species that are not closely related, some exons may be missed by sequence similarity searches.

To tackle these challenges, genBlast consists of three main stages. First, with a protein sequence as the “query” and the genome where genes are to be predicted as the “target”, local similarities between the query and the target are found by using fast sequence similarity search tools [9, 10, 55, 75, 80, 104]. The output is a collection of local alignments called *HSPs* (High-scoring Segment Pair). Next, the large number of HSPs are then filtered and assembled into a ranked list of groups, with each group corresponding to one potential homologous gene on the target genome. The algorithm used in this stage has been published and named genBlastA [117]. Finally, for each HSP group, its gene structure, i.e. the exact positions of exons and introns, is resolved by utilizing the alignment information contained in the HSPs. The sequence similarity to the query protein is used to guide the search of the best possible splice sites that define the gene structure. The algorithm used in this stage is named genBlastG [111]. The details of the two core algorithms, genBlastA and genBlastG, will be discussed in the later chapters.

To evaluate the effectiveness of genBlast, the experiments were carried out on two genomes of closely related species: *Caenorhabditis elegans* (*C. elegans*) [31] and *Caenorhabditis briggsae* (*C. briggsae*) [127]. Using the *C. elegans* proteins as the queries, genBlast predicts their homologous genes on

both *C. elegans* and *C. briggsae* genomes. The predictions made by genBlast are then compared with existing annotations as well as predictions made by another popular homology-based gene prediction program, GeneWise [19, 20]. genBlast is shown to be more accurate than all other annotations. On the other hand, genBlast is orders of magnitude faster than GeneWise. Further experiments on the human genome show similar performance of genBlastG on large-scale genomes.

1.4 Thesis Organization

The rest of this thesis is organized as follows.

Chapter 2 lays out the foundation of gene prediction. It reviews previous attempts on computational gene prediction and outlines the current status. It categorizes the common gene prediction methods and discusses representative works in each category. It presents the common performance measures in gene finder evaluation. It also motivates and provides an overview of the genBlast framework.

Chapter 3 presents the first two stages in genBlast framework. In particular, it describes the genBlastA algorithm that is used to determine the approximate gene regions where the candidate homologous genes are located. Algorithm optimizations that allow the program to run efficiently are discussed. The preliminary version of this chapter was published in [117].

Chapter 4 discusses the final stage in genBlast, i.e. genBlastG. Given the approximate gene regions provided by genBlastA, it predicts gene structures in

those regions by utilizing HSP alignments extensively. It aims to maximize the sequence similarity between the query protein and the “spliced sequence” that is obtained by joining exon regions. The experimental results are presented and compared with other existing annotations as well as GeneWise results. Overall, genBlast is shown to be more accurate while being much faster than GeneWise.

Chapter 5 concludes the thesis and suggest some directions for future work.

2: GENE PREDICTION: AN OVERVIEW

Computational gene prediction has been an active area of research for the past two decades [49, 82, 83, 87, 119, 137, 144]. Many algorithms have been proposed and implemented to predict possible gene structures on the DNA sequences, with various degrees of success. Essentially, they can be categorized into two classes according to the types of information they use: *ab initio* approaches that use intrinsic information generated exclusively from the DNA sequences (also called “intrinsic methods”), and homology-based methods that utilize evidences from other extrinsic sources such as protein sequences, cDNAs (complementary DNAs) [6], ESTs (expressed sequence tags) [97], or other genomic sequences (also called “extrinsic methods”). Both have their advantages and limitations. The following sections provide a brief review on the general ideas behind both of these approaches.

2.1 Ab Initio Gene Prediction

Ab initio methods find genes by systematically examining the DNA sequences for certain signals including start codon, stop codon and donor and acceptor signals, as well as distinct patterns that distinguish different gene regions. These methods are the very first attempts at the gene prediction problem, especially when there is insufficient homology information available. At the time of review by Mathe et al. [87], it was estimated that only approximately half of the genes can be found by homology to other known genes or proteins.

The only solution then is to use predictive methods that make use of intrinsic information from DNA sequences only.

For any gene prediction program, some common syntactic constraints on the structure of a gene can be used as guidelines: (1) There are no overlapping exons in any single gene. (2) The length of all coding exons in a gene is a multiple of 3. (3) There is no in-frame stop codon in the CDS region, i.e., starting from the translation start site (start codon) and scanning only the coding exons in fixed 3bp step (one codon at a time, while skipping introns), one should not encounter any stop codon before hitting the translation stop site. However, these are only very general constraints and much more specific information is needed to locate the gene.

Ab initio programs generally make use of two different types of information to systematically examine the DNA sequence and identify genes: content sensors and signal sensors. *Content sensors* are measures that use characteristics of variable-length gene regions (exons, introns) to try to distinguish the regions. *Signal sensors* are measures that try to detect the presence of the functional fixed-length signals (start codon, stop codon, splice sites) specific to a gene.

2.1.1 Content Sensors

There are generally two types of content sensors: one for coding regions (exons), one for non-coding regions (introns, UTRs, intergenic regions). Coding regions generally have different statistical properties that can distinguish them

from non-coding regions [27, 43]. For example, nucleotide composition patterns are different between exon regions and intron regions. Exon regions (coding regions) contain more G+C content and introns (non-coding regions) are more A+T rich [99, 125]. Codon composition is also different in coding regions and non-coding regions [41, 128]. Thus codon bias can be used to help identify coding regions [89]. The most successful and frequently used coding measure is called hexamer (words of length six) frequency, which was shown to be the most discriminative variable between coding and non-coding regions [52]. Other measures include base occurrence periodicity which refers to the preferential spacing of nucleotides by certain distance. For example, coding regions often exhibit 3-base periodicity [131, 141].

2.1.2 Signal Sensors

The DNA sequence is made up of four types of base pairs: A, C, G, T. There are specific common subsequences that define the functional sites of a gene. Particularly, the initial exon in a gene starts with a three-base “ATG” codon (start codon). The last exon in a gene ends with one of the three stop codons: “TAG”, “TGA”, or “TAA”. The splice sites that define the exact boundaries of exons and introns also have particular signals. The donor site is usually signalled by “GT” and the acceptor site is usually signalled by “AG”. These signals are found to be common in most genes (with possible variations allowed) [30, 94]. However, these signals are weak and they spread out through the entire genome. Thus identifying the true signals is very challenging.

Various pattern recognition methods are used for identification of these signals, including consensus sequence (sequence motif) [110], position weight matrix (PWM) [26, 106, 122], weight array models (WAM) [143], maximal-dependence decomposition (MDD) donor matrices [28], and dependency graphs [36]. All these are statistical models that try to capture the intrinsic interdependency between base positions in splice sites.

2.1.3 Algorithms

Using signal sensors, one can accumulate evidence on signal occurrences in a sequence. In theory, each consistent pair of detected signals defines a potential gene region (initial exon, introns, internal exons, last exon). The number of such combinations is exponential. Most *ab initio* gene prediction programs use dynamic programming to identify most likely gene structures according to the evidences given by both content and signal sensors. Recent developments have largely converged to the probabilistic models based on hidden Markov models (HMMs) [82], such as GENSCAN [28], HMMgene [77], Genie [78], GeneID [101], FGENESH [112], GlimmerHMM [81], AUGUSTUS [125], GeneZilla [7], Genemark-ES [130], etc.

Briefly, HMM is a state-based generative model which emits symbols over a finite alphabet, with the generating states hidden and only the output from the model can be observed. The basic assumption is that the probability of appearance of a given base depends only on its k previous bases (k is the order of the Markov model). The model is defined by conditional probabilities $P(X|k$

previous bases), where X = A, T, C or G. The *k*-th order Markov model captures local dependencies in sequence, at the level of the *k*+1-mers.

In order to build a Markov model, a training set of sequences is needed to estimate the state transition and base emission probabilities. Once the model is built, given a genomic sequence, HMM outputs the most probable hidden state path that generates the observed sequence using the Viterbi dynamic programming algorithm [135] as follows: given a DNA sequence *S* of length *L* and a parse *Φ* also of length *L* (a parse defines the exact exon-intron structures on the sequence), the conditional probability of *Φ*, given that the sequence generated is *S*, can be computed using Bayes' Rule [13]:

$$P(\phi \mid S) = \frac{P(\phi, S)}{\sum_{\varphi \in \phi(L)} P(\varphi, S)}$$

where *Φ*(*L*) is the set of all parses of length *L*. Thus, given a particular DNA sequence *S*, the parse that maximizes the likelihood of generating *S* is predicted to be the most likely gene structure of that sequence.

The simplest Markov models are homogeneous zero-order Markov models which assume each base occurs independently with a given frequency. The larger the order of a Markov model, the finer it can characterize dependencies between adjacent base pairs. However, larger orders also require a much larger number of parameters and a much larger training set to reliably estimate the parameters, which may be problematic for newly sequenced genomes with small training sets. Most gene prediction methods now rely on a

5th-order Markov model, which exploits hexamer composition (words of length 6) in gene characteristics.

More sophisticated models include interpolated Markov models (IMMs) that combine statistics from several Markov models of different orders, and generalized HMMs (GHMMs) that improve on HMMs by abstracting the entire gene regions (such as exons, introns, UTRs) into single states and encapsulating syntactic and statistical properties of individual regions into each state. GHMMs have become the most widely used framework for gene prediction [7, 28, 78, 81, 125].

Alternative approaches have been investigated in *ab initio* gene finding. For example, algorithms that are based on discriminative machine learning methods, such as SVMs (support vector machines) [115], or conditional random fields [18].

2.1.4 Advantages and Problems

Ab initio methods deal strictly with the DNA sequence that needs to be annotated. They extract information regarding gene locations using statistical patterns inside and outside of gene regions and around gene boundaries, based on general features of genes. Thus they allow for prediction of novel genes. Such methods are indispensible to gene finding when there is limited homology information available.

However, these methods require known sequences as the training set in order to establish the statistical properties of various gene regions, which

inherently limits their applicability to sequences that, globally, behave in the same way as the learning set. In addition, although the statistical models allow for a good discrimination between large coding and non-coding regions, the identification of exact boundaries of coding segments remains difficult. Predicted coding region boundaries are often incorrect. The predicted structure frequently splits a single gene into several, or merges several genes into one, because distinguishing intergenic and intronic regions are difficult as they don't differ much and signals for predicting gene boundaries (gene regulating regions) are often too variable (can be degenerate and unspecific). Thus such methods often generate large number of false positives from overfitted models on small training sets. On the other hand, on large DNA sequences, gene prediction accuracy can drop significantly, due to decreased gene density and larger introns.

2.2 Homology-Based Gene Prediction

Homology-based methods (or extrinsic methods) look for genes by comparing segments of DNA sequence with those of known genes, proteins or other genomic sequences. The underlying principle inherent to the majority of homology-based gene finders is the combination of homology information with signal sensors. The additional sequences used for homology comparison are also called extrinsic content sensors.

The availability of genome sequences of related species has created growing demand for better and faster homology-based gene prediction programs, which gives rise to many developments in this area. Many *ab initio* methods are extended to incorporate extrinsic evidences and have thus become their extrinsic

versions. For example, TWINSCAN [76] is directly based on GENSCAN [28] and extends it to exploit homology between two related genomes. AUGUSTUS+ [123, 124] is the extrinsic version of AUGUSTUS [125]. SGP2 [102] is based on another *ab initio* gene finder GENEID [101]. Other representative homology-based programs include GeneWise [19, 20], Projector [92], exonerate [120], SLAM [47], N-SCAN [57], CONTRAST [58], etc.

Homology-based methods can be further categorized according to the type of evidence utilized: (1) expressed sequences of genes, including protein sequences, cDNA (complementary DNA, a DNA copy of a mRNA) or EST sequences (expressed sequence tags, one shot sequences from a whole cDNA library, essentially sub-sequences of cDNAs); or (2) DNA sequences of other related genomes. Some programs are able to deal with more than one type of extrinsic evidences [24, 48, 124]. Most homology-based approaches make heavy use of HMMs. A few notable examples are discussed below.

2.2.1 Homology-Based Methods Using Expressed Sequences (Proteins, cDNAs, ESTs)

Full length cDNAs are the most direct experimental evidence for gene structure. They are usually obtained by reverse transcription from mRNAs and are complete clones of targeted individual genes. cDNAs do not contain introns and are most relevant to establishing gene structures, especially if they come from the same or a closely-related genome. cDNAs can be aligned with its own gene perfectly (assuming there is no sequencing error), and can be used to align with DNA sequences from related species or from a different member of the

same gene family, which will give strong indication of a particular gene structure at such sequences. But experimentally obtained cDNA sequences often do not completely correspond to annotated genes, for example, because cDNAs also contain UTR regions and there may be alternative splice forms involved.

On the other hand, ESTs are subsequences of cDNAs and provide information that enables the identification of potentially partial exons. However, ESTs have some special characteristics which require careful treatment. First, they are redundant and in large numbers. Secondly, ESTs are error prone since they are generated from single reads. Thirdly, ESTs provide only local and limited information as they represent only partial mRNA sequences and even clusters of ESTs may not lead to identification of complete gene. Furthermore, the correct contribution of ESTs to an individual member of a gene family is not a trivial task. When using ESTs as extrinsic evidences, genes that are expressed under very specific conditions or at very low level are generally not present in the EST database, leading to false negative predictions.

Thus cDNAs or ESTs are usually used as additional evidences that can be combined with other intrinsic or genomic evidences, rather than being used alone, for example, in Exonhunter [24], AUGUSTUS_EST [123], TWINSCAN_EST and N-SCAN_EST [139], Genie_EST [107], etc.

A few programs have attempted to use protein sequences to find the homologous genes in the genomes of same or related species. Protein sequences are the gene products obtained after gene expression. In many cases,

correct protein sequences have been derived from manual annotation of the genes of interest or from full-length cDNAs.

GeneWise [20] is a popular homology-based gene predictor that uses protein sequences in addition to the intrinsic signal sensors to assist in gene finding. It combines a gene-prediction HMM with the protein-profile HMM into a single HMM to achieve simultaneous gene prediction and alignment. The process of predicting protein-coding gene structure and the process of the sequence alignment are represented by pair HMMs, which are HMMs that convert one sequence to another. The gene-predicting HMM converts a DNA sequence from the alphabet (A,C,G,T) to a protein sequence (gene product) from a different alphabet (20 amino acids). The second HMM, the protein-alignment HMM, maps the protein sequence to its homologous protein sequence, which is used to guide the gene prediction. The two HMMs are merged into one, with one state for each possible transition from the gene-predicting HMM to the protein-alignment HMM, so that the DNA sequence can be compared directly with the homologous protein sequence, while considering all possible intermediates of the predicted protein. GeneWise serves a critical role in the Ensembl [54] automated genome annotation pipeline. However, it is computationally intensive and requires preprocessing of the DNA sequence to much small regions [45]. It also has problems in predicting terminal exons which often contain short coding regions [20].

2.2.2 Homology-Based Methods Using Comparative Genomics

Knowledge of the genome of one species can be used to understand the genome of other species [62], based on the assumption that coding regions are more conserved than non-coding regions. When two genomes are closely related, the order of many genes, gene numbers, gene positions and even gene structures (exon-intron organization, splice site usage etc.) remain highly conserved. Thus new genes can be identified from genome comparisons. TWINSCAN [76, 133], SGP2 [102], Projector [92], SLAM [47], CONTRAST [58] are some notable examples of programs that utilize comparative genomics, all of which are based on various models of HMMs that make use of alignments between the reference genome and the genome of interest.

2.2.3 Advantages and Limitations

The important strength of these homology-based approaches is that the predictions are guided by accumulated pre-existing biological data, so that such predictions often achieve higher accuracy than pure *ab initio* methods. A single hit of sequence similarity is enough to detect the presence of a gene, even with non-canonical signals. Homology-based programs generally have good specificity and produce fewer false positives (than pure *ab initio* methods), because they are based on biological evidence (homology) to existing genes. Such methods are not species specific.

The biggest limitation of homology-based methods is that such approaches can only be used to find genes with homologs. This problem is alleviated with progress in genome sequencing, as more genes are found.

Sequence alignments by fast heuristic search tools may contain regions of low quality. Small exons are easily missed. Even when similarity is found, the regions are not always precise. This is especially true for comparative genomics approaches, which assume there is sufficient contrast in sequence similarity between coding regions and non-coding regions. However, similarity of the coding regions may not cover the entire gene. Closely-related species may exhibit similarity that extends to introns or other non-coding regions, in which case genomic comparisons will lead to false predictions. Therefore, homology-based methods based on protein sequence generally have better overall performances than comparative genomics approaches [38, 59].

2.3 Integrated Gene Prediction: Combining Ab Initio and Homology-based Predictions

While *ab initio* methods tend to overestimate gene numbers, homology-based gene finders may underestimate since they are limited to recognizing only those genes similar to prior examples. Integrated approaches have been proposed to combine both *ab initio* and homology-based approaches in order to obtain a consensus. Integrated approaches are generally more accurate than their constituent gene prediction programs. The representative programs include JIGSAW [7, 8], GLEAN [50], EVM [61], YACOP [129], as well as other programs based on neural networks [145] and Bayesian networks [103]. The success of these programs is based on the best efforts made by their constituent gene prediction programs, which are diverse methods so that they can compensate

one another and provide more accurate predictions when combined. Therefore, it is indispensable to develop gene finders using different models and algorithms.

2.4 Current Performances of Gene Prediction Programs

2.4.1 Evaluation Measures

The performance of gene prediction programs are commonly measured by *sensitivity* (*Sn*) and *specificity* (*Sp*). Sensitivity measures the proportion of actual gene structures on the genomic sequence which are correctly predicted as such. Specificity measures the proportion of those predicted gene structures that are actually true.

In standard classification problems, we have the following definitions:

- true positives (TP): cases that belong to class C and are correctly predicted;
- true negatives (TN): cases that do not belong to class C and are correctly predicted as not belonging to C;
- false positives (FP): cases that do not belong to class C but are mistakenly predicted as belonging to C;
- false negatives (FN): cases that belong to class C but are mistakenly predicted as not belonging to C.

Then we have: $Sn = \frac{TP}{TP + FN}$ and $Sp = \frac{TP}{TP + FP}$.

Furthermore, the accuracy of gene prediction programs is usually measured at three levels of granularity: base (nucleotide) level, exon level, and

gene level [29]. Each level may indicate some behaviour of the gene finder that other measures neglect. For example, at the base level, a true positive is a base that belongs to an exon (or intron) is correctly predicted to be in that region. At the level of exons, a true positive is an exon that was predicted exactly correct, with both of its boundaries (splice sites, or start codon, or stop codon) exactly identified. At the level of genes, a true positive is a gene that has all its component exons and introns being correctly identified. It is clear that the gene level accuracy is the most difficult to achieve because it requires all components in a gene to be perfectly predicted.

2.4.2 Gene Prediction Evaluation

Although there have been many reviews [29, 49, 56, 82, 83, 87, 87, 119, 137, 137, 144] in the area of gene prediction, a truly fair comparison of all prediction programs is impossible, because performances of many programs depend heavily on the specific training data that are used to develop them. Tools are often specialized for species, often with distinct statistical models. Many programs were developed in-house and were not accessible for independent evaluation. Existing annotations that are used as the ground truth in evaluation do not necessarily cover the whole truth, due to the limitations of current knowledge. For new programs, there is no standard for benchmark comparison. The following attempts at comprehensive evaluations give a partial picture of current gene prediction performances [12, 38, 59].

EGASP (the human ENCODE Genome Annotation Assessment Project) [59] is an influential community experiment on comprehensive evaluation of

protein-coding gene prediction programs. It tried to assess the state-of-the-art in gene finding by testing on the ENCODE (ENCyclopedia Of DNA Elements) regions of human genome. The ENCODE project [51] was a collaborative effort by many computational and laboratory-based scientists with the aim of identifying all functional elements in the human genome sequence. Its pilot phase focused on a selected 30 Mb of sequence within 44 selected regions, which represents approximately 1% of the human genome. The ENCODE team has produced a high quality annotation of the gene content of the ENCODE regions (GENCODE annotation) [63]. In the EGASP experiment, gene prediction programs are evaluated by comparing their results with the GENCODE annotation on the ENCODE regions. In addition, 11 pre-existing gene annotation tracks published in the UCSC Browser [74] were included in the comparison. The programs use any publicly available data before the evaluation deadline and were categorized into different classes according to the data that they used: (1) single-genome *ab initio* methods that use DNA sequences only; (2) homology-based methods, which include protein-, mRNA- and EST-based methods, as well as comparative genomics methods that use other genomes; (3) integrated methods.

For evaluation, the EGASP results are measured at four levels of granularities: base, exon, gene and isoform levels. Note that the gene level test in EGASP is slightly different from the standard gene-level measurement, which assumed only one gene model per gene, because many genes through alternative splicing produce different protein isoforms. The isoform level accuracy is the most stringent test. An isoform is considered correct only if all exons were

predicted accurately and no extra full or partial exons were predicted. The gene level accuracy was intermediate in stringency between the exon and isoform levels. A gene is correct if at least one of its isoforms is predicted correctly. Table 1 shows some of the results reported by EGASP [59] (other programs with clearly worse performances are omitted here). Programs that gave the best overall performances are bolded. The pre-existing annotations from the UCSC Browser are marked with asterisks (*).

In summary, programs that used expressed sequences (protein sequences and mRNA) and those that used integrated approaches were generally the most accurate for all measures. The *ab initio* programs and comparative genomics approaches are among the worst in general. At the base level, JIGSAW [8] and ENSEMBL(GeneWise) [45] both achieved greater than 90% for both sensitivity and specificity. However, the accuracy decreases considerably with increased level of granularity, especially at the isoform-level, which leaves much room for further improvement.

The similar performance behaviours of different types of programs were observed in later evaluation experiments. As the most recent effort in comprehensive evaluation, nGASP (nematode genome annotation assessment project) [38] evaluated gene finders on the well-annotated *C. elegans* genome [31]. About 10% of the *C. elegans* genome is used in the experiment, with the training set and testing set each comprising ten non-overlapping 1-Mb genomic sequence regions. Participants were given additional data that included multi-genome alignments between *C. elegans*, *C. briggsae* and *C. remanei*, and

Table 1 Gene Prediction Accuracy in EGASP evaluations (human genome) [59]

	Program		Base		Exon		Gene		Isoform	
			Sn	Sp	Sn	Sp	Sn	Sp	Sn	Sp
Ab initio Gene Prediction Tools	AUGUSTUS_ab_initio		78.65	75.29	52.39	62.93	24.32	17.22	11.09	17.22
	GENEMARK.hmm		76.09	62.94	48.15	47.25	16.89	7.91	7.70	7.91
	GENEZILLA		87.56	50.93	62.08	50.25	19.59	8.84	9.09	8.84
	GENEID (*)		76.77	76.48	53.84	61.08	10.47	8.78	4.78	8.78
	GENSCAN (*)		84.17	60.60	58.65	46.37	15.54	10.13	7.40	10.13
Homology-based Gene Prediction Tools	**Comparative Genomics**	AUGUSTUS-dual	88.86	80.15	63.06	69.14	26.01	18.64	12.33	18.64
		NSCAN	85.38	89.02	67.66	82.05	35.47	36.71	16.95	36.71
		SGP2 (*)	82.81	82.20	60.56	65.16	17.57	12.59	8.17	12.59
		TWINSCAN (*)	78.16	84.59	58.43	73.11	22.30	20.25	10.63	20.25
	Protein/mRNA/EST based	ACEVIEW	90.94	79.14	85.75	56.98	63.51	48.65	44.68	19.31
		AUGUSTUS-EST	92.62	83.45	74.10	77.40	47.64	37.01	22.50	37.01
		ENSEMBL (Genewise)	**90.18**	**92.02**	**77.53**	**82.65**	**71.62**	**67.32**	**39.75**	**54.64**
		EXOGEAN	84.18	94.33	79.34	83.45	63.18	80.82	42.53	52.44
		PAIRAGON+NSCAN_EST	87.56	92.77	76.63	88.95	69.59	61.71	39.29	60.64
		ECgene (*)	96.36	47.30	86.22	35.08	79.05	12.42	56.86	8.84
		ENSgene (*)	**91.39**	**91.92**	**77.71**	**82.39**	**73.99**	**68.30**	**40.52**	**54.09**
		MGCgene (*)	44.06	97.56	42.95	93.61	49.32	82.56	23.73	78.24
Integrated Gene Prediction Tools	**JIGSAW**		**94.56**	**92.19**	**80.61**	**89.33**	**72.64**	**65.95**	**34.05**	**65.95**
	PAIRAGON-any		87.77	92.78	76.85	88.91	69.59	61.32	39.29	60.34
	CCDSgene (*)		56.87	99.52	51.95	97.75	55.41	89.39	28.97	85.58
	KNOWNgene (*)		89.10	93.61	78.11	82.28	77.03	72.79	43.45	46.93
	REFgene (*)		**85.34**	**98.50**	**73.23**	**94.67**	**77.03**	**82.76**	**41.91**	**75.21**

alignments of ESTs, mRNAs and proteins to the *C. elegans* genome. The results were evaluated using reference gene sets drawn from WormBase (release WS160) [33, 109] and measured in the same way as in EGASP. Table 2 shows some of the best results reported by nGASP [38], with the programs that gave the best overall performances bolded. As in EGASP, the gene-level and isoform-level performances are significantly worse than those at base- and exon- levels. Again, the integrated method, JIGSAW [8], is the overall best performer, with base level sensitivity at 99% and specificity of more than 93%. Gene finders that used alignments of proteins, ESTs and mRNAs came in second. Note that GeneWise was not included in the nGASP evaluation.

The EGASP and nGASP results do not represent the whole picture of gene prediction research, because each tested only some selected regions in one particular organism. The datasets used in evaluation inevitably contain biases, for example, due to non-complete or unconfirmed annotations. Nevertheless, they provide a general idea on the state of the art in gene prediction and where improvements are to be made.

2.4.3 Summary

Current gene prediction methods have limitations. None of the programs is accurate enough to predict all gene structures adequately. In general, homology-based gene prediction methods outperform *ab initio* methods in accuracy when homology evidence is available. The accumulation of genome sequences keeps boosting the positive cycle of such good performances. Homology-based

Table 2 Gene Prediction Accuracy in nGASP evaluations (*C. elegans* Genome) [38]

	Program		Base		Exon		Gene		Isoform	
			Sn	***Sp***	***Sn***	***Sp***	***Sn***	***Sp***	***Sn***	***Sp***
Ab initio Gene Prediction Tools	AUGUSTUS_ab_initio		97.0	89.0	86.1	72.6	61.1	38.4	50.1	28.7
	CRAIG		95.6	90.9	80.2	78.2	43.8	37.8	35.7	36.3
	EUGENE_ab_initio		94.0	89.5	80.3	73.0	60.2	30.2	49.1	28.8
	Fgenesh		98.2	87.1	86.4	73.6	57.8	35.4	47.1	34.6
	GeneID		93.9	88.2	77.0	68.6	44.4	25.1	36.2	22.8
	GeneMark.hmm		98.3	83.1	83.2	65.6	46.3	24.5	37.7	24.0
	GlimmerHMM		97.6	87.6	84.4	71.4	58.0	30.6	47.3	29.3
	MGENE_ab_initio		97.2	91.5	84.6	78.6	54.8	42.3	44.6	40.9
Homology-based Gene Prediction Tools	**Comparative Genomics**	EUGENE v1	96.2	87.5	82.8	72.8	61.7	31.4	50.3	30.2
		MGENE	97.7	90.9	85.8	78.4	63.3	42.5	51.6	41.2
		N-SCAN	97.4	88.1	83.5	70.8	48.1	28.4	39.2	27.7
		SGP2	93.5	90.0	77.3	70.3	44.6	27.1	36.4	24.9
	Protein/mRNA/EST based	**AUGUSTUS+**	**99.0**	**90.5**	**92.5**	**80.2**	**80.1**	**51.8**	**68.3**	**47.1**
		ExonHunter v2	93.7	92.0	81.2	76.9	45.6	40.5	37.2	39.7
		Fgenesh++	97.6	89.7	90.4	80.9	78.3	54.2	65.5	53.4
		Gramene v1	98.2	95.4	88.5	71.8	48.7	37.2	41.7	19.6
		MGENE v3	98.7	91.9	91.0	80.6	70.6	51.1	57.7	48.0
Integrated Gene Prediction Tools	EUGENE v4		99.2	85.3	94.0	71.8	77.9	39.8	67.1	33.9
	Evigan		99.3	89.6	91.1	82.3	80.7	52.7	64.2	52.4
	Fgenesh++C		**98.7**	**89.7**	**91.1**	**82.7**	**80.3**	**57.1**	**66.1**	**56.3**
	GeneID v1		99.3	91.5	93.0	83.8	78.3	57.7	63.9	53.3
	GLEAN		98.9	87.3	88.3	75.4	64.7	37.6	51.4	37.0
	JIGSAW		**98.9**	**93.2**	**90.5**	**87.4**	**79.9**	**61.0**	**63.6**	**60.2**

methods based on protein sequences generally give better predictions than comparative genomics approaches.

2.5 genBlast: A Novel Framework of Gene Prediction by Protein Homology

2.5.1 Motivation

Homologous genes usually have conserved gene structures. For example, the comparative analysis of the mouse and the human genome [95] estimated that 99% of the mouse genes have a homologous human gene and 86% of the orthologous gene pairs are estimated to have the same number of coding exons. Most cases where there is a different number of exons can be explained by single exon fusion or exon splitting events. Proteins are gene products that can be seen as concatenation of coding exons. Thus homologous gene models can be predicted based on sequence similarity between a protein sequence and the predicted gene product. Proteins have been used in predicting homologous genes with competitive performance.

However, most current gene prediction programs that make use of protein sequences, such as GeneWise [20], AUGUSTUS [124], ExonHunter [24], are based on hidden Markov models (HMMs), which impose high computational cost. The running time of the Viterbi dynamic programming algorithm [135] used in HMM solutions increases rapidly with increased sequence length and number of states in the HMM. This makes them slow in annotating large scale of genome sequences. For example, in the ENSEMBL genome annotation system [45], genome sequences need to be pre-processed to refine the input sequence

before running GeneWise, the core gene prediction program in ENSEMBL. In our experiments, GeneWise needs more than one hour to predict a homologous gene model for a single gene that codes for proteins longer than 1,000 amino acids in size, even after the gene region has been identified. In addition, the accuracy of GeneWise still needs improvement, especially at the gene and isoform levels. GeneWise also tends to predict partial gene models and does not always return genes with stop codon at the end. Although with the possibility of parallelized computing and the use of massive computer farms, speed is a less important consideration for gene finders, an alternative algorithm with faster speed and better or even comparative accuracy is always useful. It is thus essential to explore new options that avoid complicated models, while achieving competitive accuracy.

2.5.2 genBlast Overview

In this thesis, I present a novel homology-based gene prediction framework, genBlast. Similar to GeneWise, genBlast takes the following biological sequences as input: a protein sequence (the query), and a genome where homologous genes are to be found (the target). The target genome may consist of one or more DNA sequences. In principle, the treatment for multiple DNA sequences is the same as for the single DNA, because each DNA sequence in the genome is independent and therefore processed independently. The ultimate goal is to predict the structures (positions of exons and introns) of the genes on the DNA sequences that are homologous to the query protein.

The general workflow of genBlast is shown in Figure 3. The input is a protein sequence and a DNA sequence on which to search for homologous genes. The sequence similarity between the query protein and the predicted protein product implies homology between the predicted gene and the query gene. genBlast can be seen as a gene prediction framework that consists of three stages as follows.

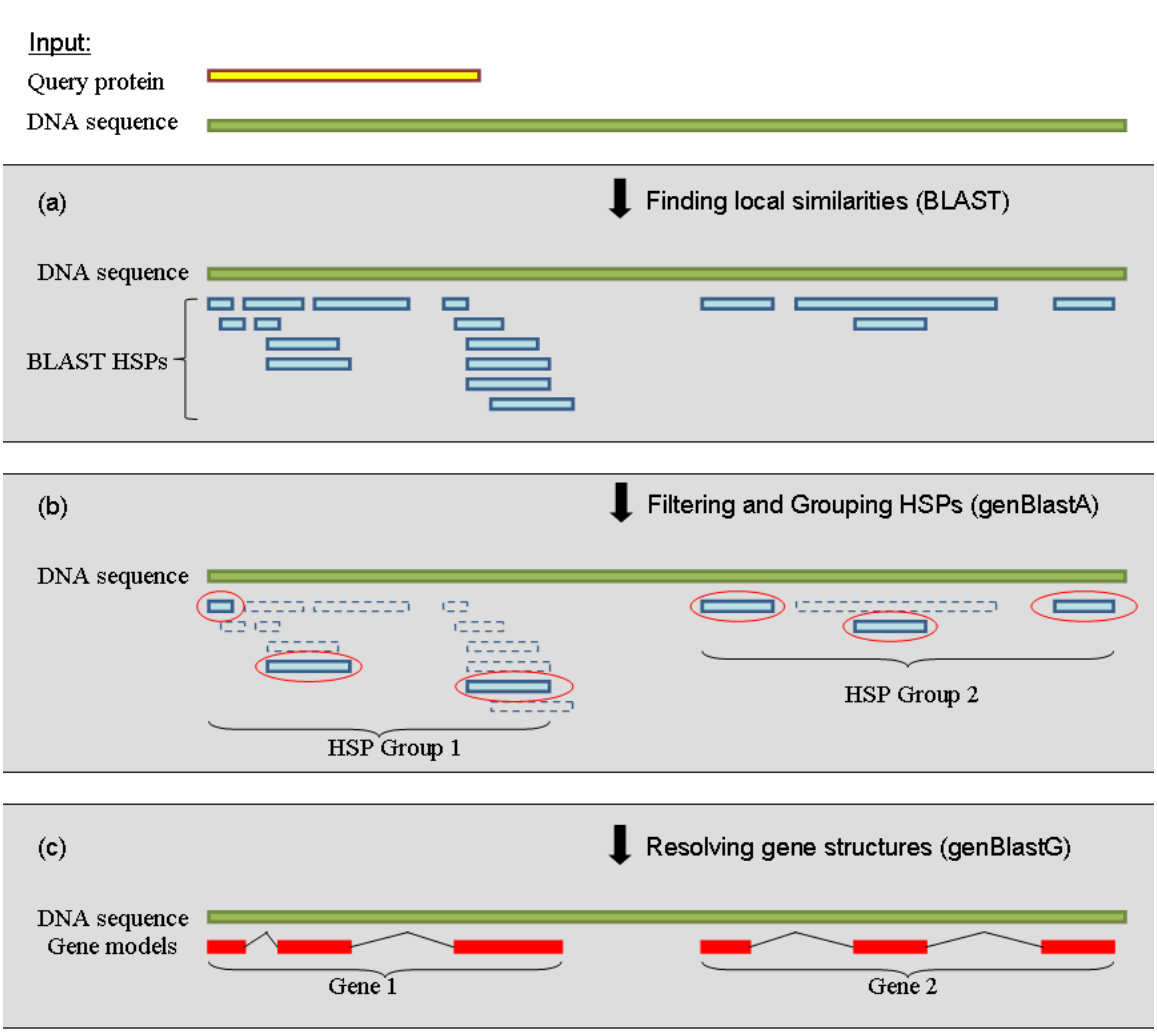
First, as the basis for finding homologous genes, local sequence similarities between the query protein and the DNA sequence are found by using fast sequence alignment tools such as BLAST [9]. This produces a list of local alignments called *HSPs* (High-scoring Segment Pairs) with each HSP containing a pair of sequence segments, one from the query protein, and the other from the target DNA sequence. Such HSPs may scatter all over the DNA sequence and may overlap with one another.

The next stage, genBlastA (A stands for assembly), is geared towards the finding of approximate gene regions on the DNA sequence where potential homologous genes are located. Irrelevant HSPs (noises) are filtered and the remaining HSPs are assembled into groups such that each group of HSPs forms a candidate homologous gene region. The algorithm of genBlastA has previously been published [117] and will be discussed in Chapter 3.

Finally, genBlastG (G stands for gene) [111] is used to determine the exact gene structure for each candidate gene region reported by genBlastA. As such, the purpose of genBlastA shares some similarity with the pre-processing step for GeneWise, where the approximate gene regions are found before

running GeneWise on the much shorter sequences [45]. However, genBlastA also provides detailed alignment information for the candidate gene regions that is essential for the actual gene prediction by genBlastG. genBlastG will be discussed in Chapter 4.

Figure 3 GenBlast Overview



The pseudocodes of genBlast are given in Appendix 1. In the following chapters, I will present the details of genBlastA and genBlastG as well as their roles in the gene prediction framework.

3: GENBLASTA: FINDING HOMOLOGOUS GENE REGIONS

This chapter discusses the first two stages of gene prediction, with the goal of locating the regions of possible genes that are homologous to the query protein.

3.1 Finding Local Similarities

In order to find possible regions of genes that are homologous to a given protein, it is essential to identify sequence similarities between the protein sequence and the genome. There are many sequence similarity search tools that can be used, such as BLAST [9], WU-BLAST [80], FASTA [104], sim4 [55], and BLAT [75], all of which are useful for homology studies. In general, these search tools work by identifying a list of sequence segments in a target genome sequence database that show similarity to a query sequence and report them as local alignments. For example, BLAST detects regions of similarity between the query sequence and target sequences in a database and reports local alignments between the two. It has been popular among biologists due to its speed and sensitivity.

In our context, the problem is to compare a protein query sequence against a DNA sequence database. This is usually achieved by first translating the DNA sequence to its corresponding protein sequences according to the genetic code [42], where a triplet codon in a DNA sequence is translated into a

single amino acid, as described in Chapter 1. Note that the initial position where translation starts is critical in protein translation. For example, the sequence GGGAAACCC can be broken down to codons of “GGG,AAA,CCC” or “GGA,AAC”, depending on the first nucleotide to be read. Different starting positions lead to different *reading frames* that produce different amino acid sequences. Thus programs such as tBLASTn (a protein-DNA alignment tool in BLAST program suite) compare a protein sequence against a DNA sequence database translated in all possible reading frames.

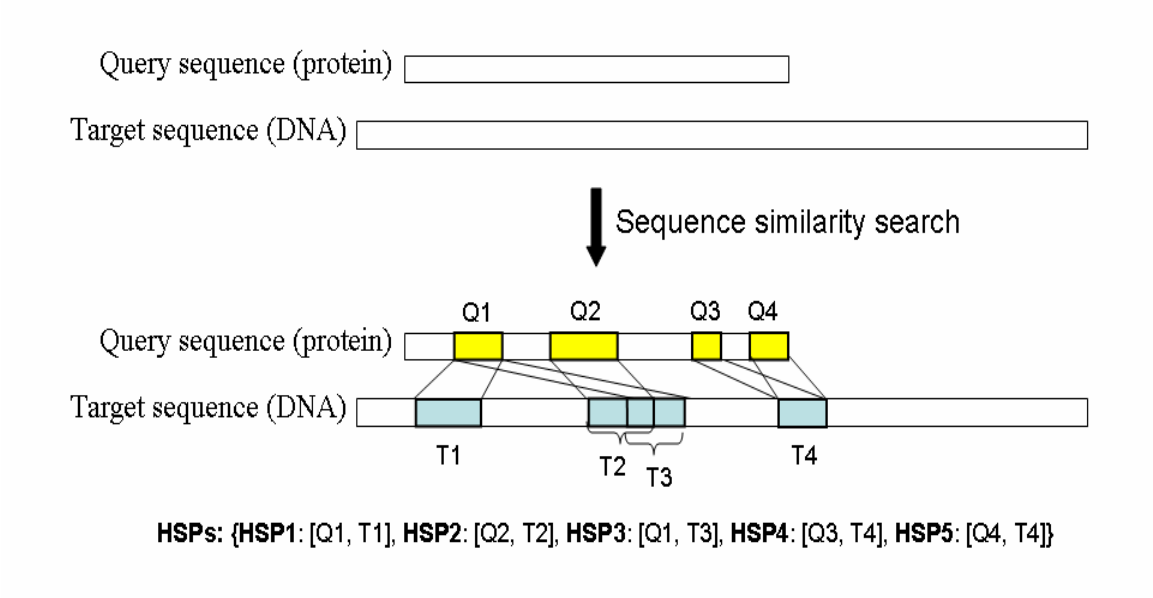
The sequence similarity search tools usually report the matches between the query sequence and the target database as *HSPs* (High-scoring Segment Pairs). An example HSP is given in Figure 4, which consists of a pair of sequence segments: a *query segment* from the query sequence, and a *target segment* from the target sequence. The center line shows the matching positions between query segment and target segment.

Figure 4 An example HSP (high-scoring segment pair)

```
Query:       1 MTISKNNKMMAHLNYRMKIILQDGRTFIGFFKAFDK--NILLAECEEHRQIKPKAGKKTD 58
               MTISKNNKMMAHLNYRMKIILQDGRTF+GFFKAFDK  NILLAECEEHRQIKPKAGKK D
Targt: 2972783 MTISKNNKMMAHLNYRMKIILQDGRTFVGFFKAFDKHMNILLAECEEHRQIKPKAGKKVD 2972962
```

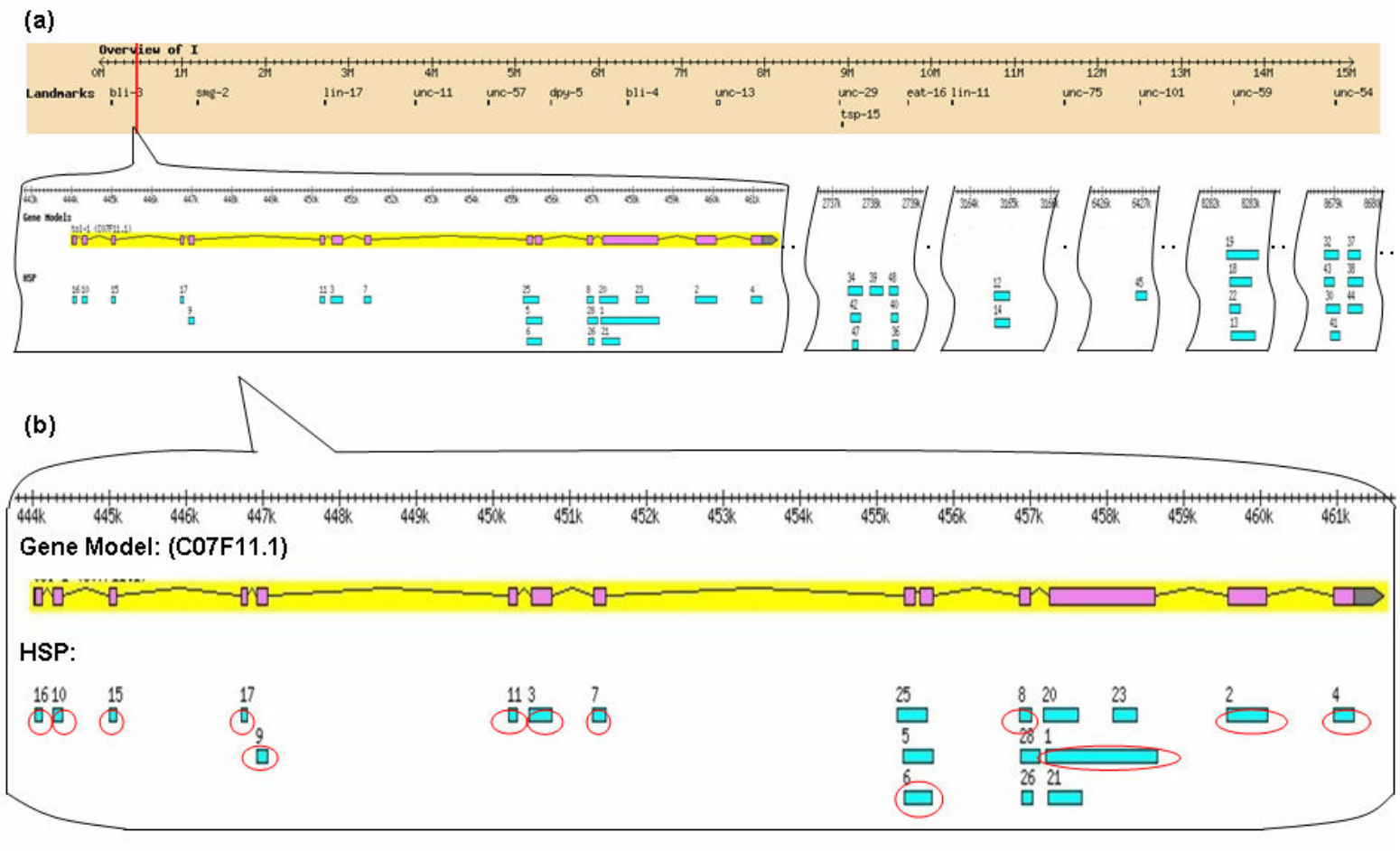
Figure 5 further illustrates a collection of HSPs, where each HSP consists of a pair of sequences: [Q,T], where Q is the query segment (shown in yellow) and T is the matching target segment (shown in blue). The correspondence between the query segment and the target segment is given by lines that connect the segments. HSPs may overlap on either query segments or target segments.

Figure 5 Local Similarities Reported as HSPs



When a similarity search returns numerous HSPs for a query gene (a protein sequence) in a genome, it suggests the existence of one or more homologous genes in that genome. For example, when the protein encoded by the *C. elegans* gene C07F11.1 is used as a query to BLAST against the *C. elegans* genome, many HSPs are reported, as shown in Figure 6(a). In this figure, the blue boxes at the bottom are HSPs at their corresponding genomic positions. They are scattered across many regions of the DNA sequence. Among these HSPs, some may represent candidate genes, while others are spurious hits. For example, the genomic region of the actual gene C07F11.1 (the region in yellow) is shown in the bottom left of Figure 6(a) and again in Figure 6(b), with the exons (purple boxes) and introns (lines connecting exons) shown above the HSPs in that region. Figure 6(b) is a magnified view of the same region. Only some of the HSPs are relevant and can provide biologists with a meaningful

Figure 6 HSPs reported for a *C. elegans* gene C07F11.1 on Chromosome I



starting point for further research, such as those HSPs that have correspondences with the exons in the gene model, as circled in Figure 6(b). The genomic region defined by those HSPs is a homologous gene region. For genes that are homologous to more than one other genes, multiple gene regions will be identified.

Although BLAST and other similarity searching tools produce lists of HSPs, they do not reveal which HSPs represent candidate genes, let alone reveal how many homologous genes exist in the target genome. This brings on the demand for developing new tools that can filter and organize HSPs into homologous gene regions.

3.2 Filtering and Grouping HSPs

The BLAST-like tools simply report all local alignments that exceed a user-specified threshold. For effective use of HSPs, a program is needed to organize them into groups so that further research can focus on regions that likely contain genes.

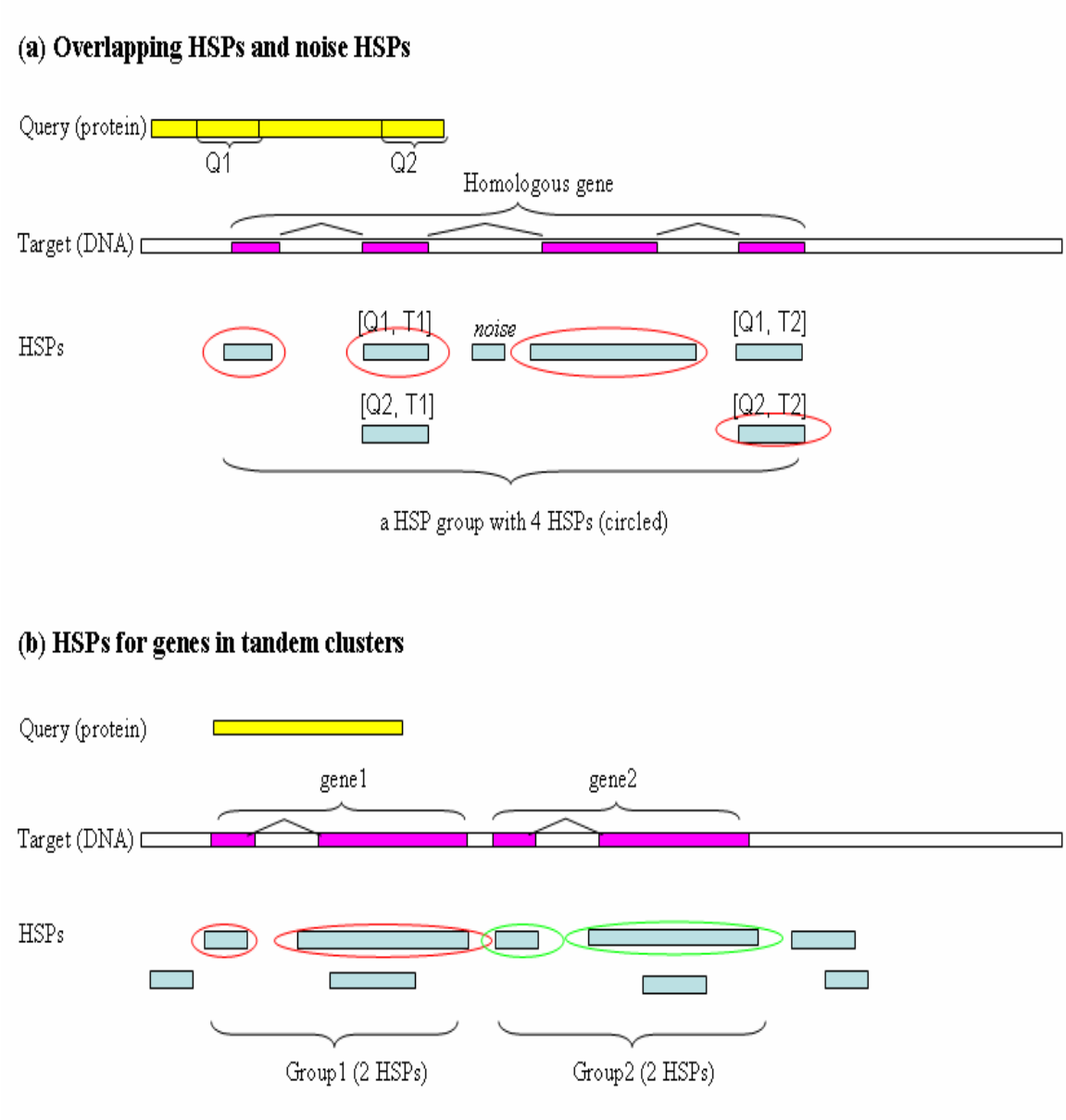
3.2.1 Challenges

The filtering and grouping of HSPs are not trivial tasks. Many genes contain “internal repeats”, which are fragments in a gene that show high similarity to each other. Thus a genomic region on the target sequence may align with multiple query regions and vice versa, i.e. HSPs may have overlapping target segments and/or query segments. As illustrated in Figure 7(a), the query gene contains internal repeats, with Q1 and Q2 highly similar to each other. This

results in multiple HSPs with same target segment that aligns with different query segments. For example, two HSPs are returned with the same target segment T1: [Q1,T1], [Q2,T1]; two other HSPs are returned with similar situation: [Q1,T2], [Q2,T2]. In addition, there may be random alignments (noise) that should be filtered, as shown in the intron region of the gene in Figure 7(a). All these factors must be taken into account when trying to identify the relevant HSPs that define a proper gene region, as shown by the circled HSPs in the figure.

Furthermore, the task is particularly challenging when the query gene belongs to a multi-gene family with many homologous genes, or has a large number of paralogous genes in tandem in the target genome (see glossary for “multi-gene family” and “tandem genes”). It is well known that a large number of genes in almost all sequenced genomes are parts of tandem homologous gene clusters. For example, in the nematode *C. elegans* genome, more than 1,400 chemosensory genes form many tandem gene clusters, each of which contains two or more homologous genes [108]. Figure 7(b) shows an example of a query gene with two homologous genes (gene 1 and gene 2) that are located in close genomic proximity. The boundary between the two homologous genes must be resolved so that HSPs corresponding to different genes are not erroneously grouped.

Figure 7 Challenges in Filtering and Grouping HSPs



3.2.2 Previous Attempts

Ad hoc approaches have been developed to filter and assemble HSPs into groups representing genes. These methods can resolve some genes, but fail in many cases.

The best-known program that provides the functionality of grouping HSPs is WU-BLAST [80], a BLAST-like program, which is also an extensively used local alignment tool for identifying sequence similarities. It provides a "topcomboE" option that categorizes HSPs into groups. The algorithm behind this option is not published. From our observation, within each group produced by WU-BLAST, HSPs are usually adjacent and collinear. Although WU-BLAST can group some HSPs into gene-like structures, for HSPs representing candidate genes within tandem clusters in the target genome, WU-BLAST fails badly. For these cases, WU-BLAST tends to group HSPs corresponding to different genes into the same group.

Cui and colleagues developed a new filtering and grouping algorithm that processes BLAST results to help identifying homologous genes [44]. They applied a three-step procedure to filter and group HSPs that represent candidate genes: (1) Filter all HSPs by discarding HSPs with scores lower than a heuristic threshold. (2) Group HSPs based on their physical distance along the chromosomes. (3) Further filter HSPs by estimating the genomic span of target regions. All HSPs that fall outside of the target regions are excluded from further analysis. This program is able to produce *some* HSP groups that represent tandem homologous genes. However, this program has an important weakness, which is its dependence on the physical distances (in step 2) between gene structures (groups of HSPs) to separate groups. It assumes that the distance between different HSP groups are significantly larger than the distance between HSPs within a group, which is not true for paralogous genes in tandem clusters.

The usage of *ad hoc* distance thresholds to separate adjacent genes makes this program fail in resolving many tandem genes. If the distance threshold value for separating genes is too large, HSPs corresponding to multiple genes will be lumped together into a large group. On the other hand, if the threshold value is too small, HSPs corresponding to a same gene could be divided into different HSP groups.

3.2.3 genBlastA

In this thesis, I present a newly developed graph-based algorithm, genBlastA, for the task of filtering and assembling HSPs into homologous gene regions. A distinctive feature of genBlastA is that it does not rely on *ad hoc* thresholds for filtering noise HSPs or on physical distance between target genes. Instead, genBlastA models the relationships and constraints among HSPs in a directed graph, called the *HSP graph*, and solves the HSP filtering and assembling problem by searching for the shortest paths in this graph. The novelty of this graph-based algorithm is an innovative edge length metric that reflects a set of biologically motivated requirements so that each shortest path corresponds to an HSP group representing a homologous gene. Unlike existing *ad hoc* grouping methods, this method filters and groups HSPs on the basis of optimizing the path length that captures the quality of a group of HSPs. Consequently, genBlastA is more robust and the solution it finds is optimal with respect to the given length metric. The details of the genBlastA algorithm are described in the following section.

3.3 genBlastA: The Methods

3.3.1 Problem Definition

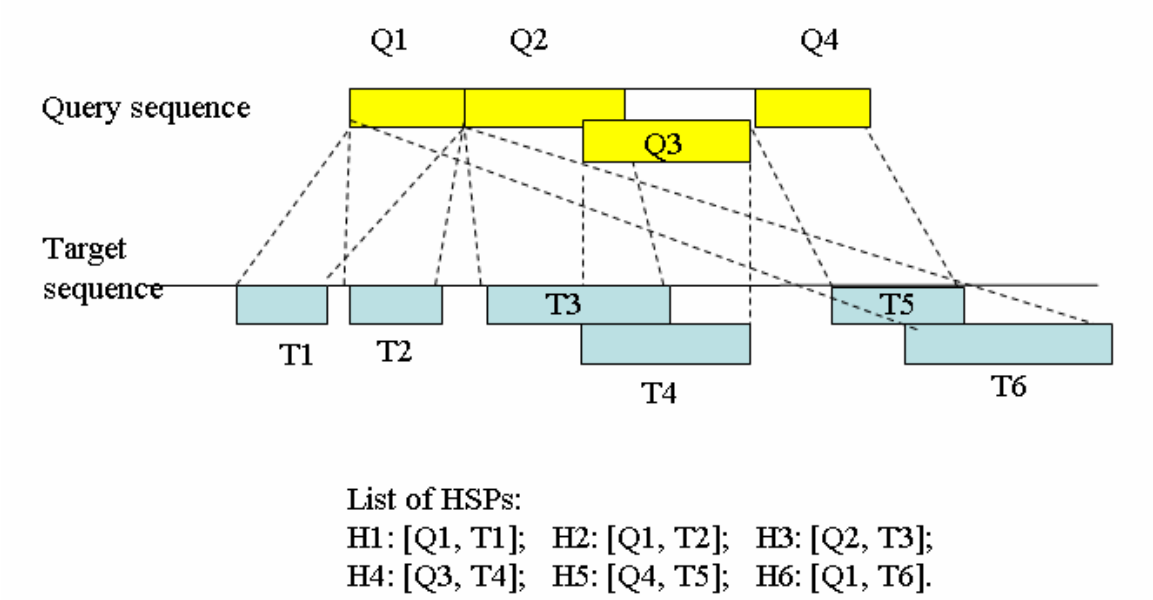
The problem we studied is this: Given a collection of HSPs, which are local alignments between a query protein sequence and some target DNA sequences, we want to identify all regions containing homologous genes on the target DNA sequences. With genBlastA, this is achieved by filtering and grouping the HSPs such that each group of HSPs forms a candidate gene region.

Note that each DNA sequence has two strands in reverse directions of each other. HSPs can be on any reading frame of any strand. HSPs on opposite strands differ only in the alignment directions of their target segments. genBlastA considers each strand as an independent target sequence that has its own list of HSPs. Thus genBlastA processes each target sequence separately in order to obtain the candidate gene regions on that sequence. Finally, all candidates on all target sequences are ranked into a single ranked list of candidate gene regions. For brevity, the following discussions of genBlastA algorithm will be based on single DNA sequence on the forward strand.

Each HSP contains the following information: (1) the target segment T and its location in the target sequence; (2) the corresponding query segment Q and its location in the query sequence; (3) a percentage of identity value (*PID*), which is the percentage of exact matches in the alignment, calculated as the number of exact matches over the length of the entire alignment. An example list of HSPs is shown in Figure 8. Note the HSPs shown in this figure are only for illustration purposes, although genBlastA is able to properly handle HSPs with various kinds

of relationships. For example, [Q1,T1] and [Q1,T2] represent two different HSPs with overlapping query segments. [Q2,T3] and [Q3,T4] are two other HSPs that overlap on both query and target segments. This example will be used as a running example for the discussions in the following sections to explain the genBlastA algorithm.

Figure 8 Example HSPs



3.3.2 HSP Groups

With each HSP target segment matching a query segment, a sequential group of HSP target segments can collectively match a larger piece of the query sequence. We are interested in those groups of HSPs, which correspond to genes in the sense that they are homologous to the query gene. Such groups are termed *HSP groups*. In general, there are different numbers of HSP groups in the target sequence for each query gene. If the query gene is not conserved in the

target genome, then no HSP group can be found. If the query gene belongs to a multi-gene family (or the query gene has many paralogous genes), there will be multiple HSP groups in the target sequence, each representing a candidate region containing a paralogous gene.

Before grouping HSPs, it is important to understand some essential requirements that must be satisfied for a group of HSPs to represent a target gene. These requirements are summarized below.

Rationale I (Sequential Ordering): Because each gene is a sequence where the order of base pairs is critical, the sequential ordering must be preserved when grouping HSPs.

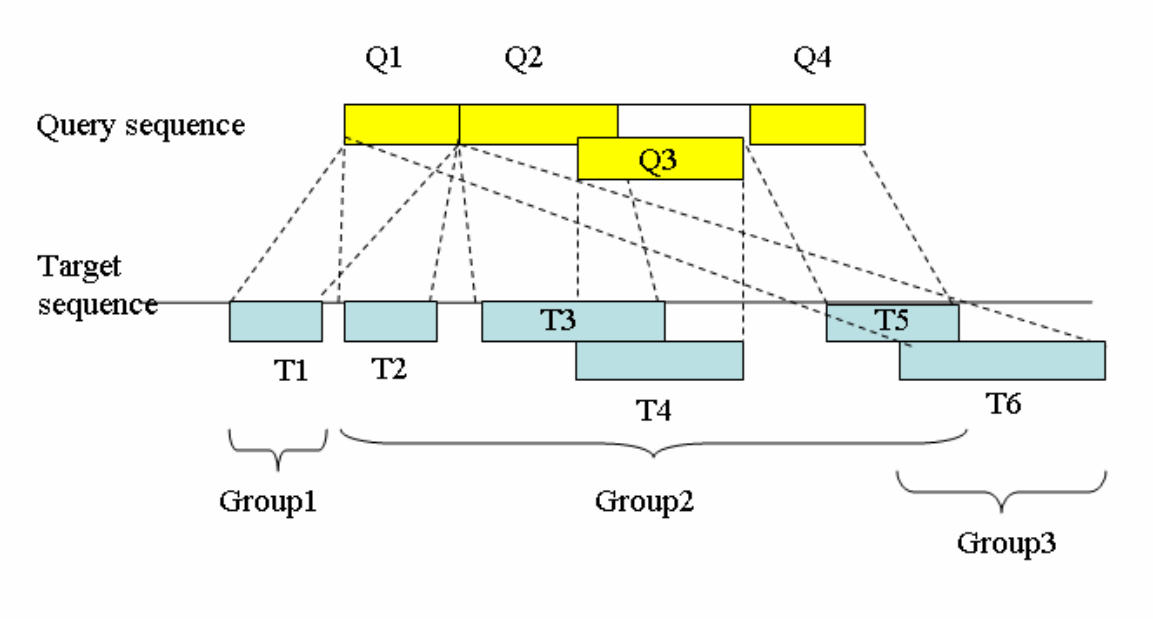
This rationale is justified by gene structure conservation among homologous genes [95]. Although genes may be formed by reordering of exons during evolution, they are considered as new genes that code for novel proteins [23] and thus not included in our consideration. We are interested in finding genes that code for similar proteins.

Rationale II (Co-linearity): For each HSP group, the target segments must be arranged in the same order as their corresponding query segments, i.e. the HSPs must be collinear.

Consider the example in Figure 8. T3 and T4 are in the same order as their query segments. So [Q3,T4] can be in the same group as [Q2,T3]. In fact, by merging T3 and T4 into one continuous target region, and merging their query segments into one continuous query region, we may have a larger, thus better

alignment. Figure 9 shows a possible grouping of HSPs that satisfies the sequential ordering and co-linearity requirements. Note that Group 1 and Group 3 have incomplete query coverage because a large portion of the query sequence is not covered by their query segments. In contrast, Group 2 covers the entire query sequence. A good HSP group should have large query coverage.

Figure 9 Example Groups of HSPs



Rationale III (Noise Skipping): Some HSPs may be random alignments (alignment by chance), called noise HSPs. Such HSPs should be dropped or skipped so that they are not in any HSP group. Although most noise HSPs are short and have low PID, some are relatively long or have high PID and may not be easily identified or removed. The determination of whether a HSP is noise or not depends also on other HSPs as a group for forming the gene region and cannot be easily determined by looking at each HSP individually.

Rationale IV (Proximity): Because genes occupy contiguous genomics regions, the HSPs within an HSP group, which corresponds to a target gene, should be close to each other on the target sequence as much as possible to avoid merging HSPs corresponding to adjacent tandem genes into one group.

When deciding whether an HSP is a noise and should be skipped, both Rationale III and IV should be considered because such skipping may produce HSP groups that span multiple gene regions, which is contradictory to Rationale IV. For instance, the earlier example of tandem genes in Figure 7(b) shows two homologous genes next to each other. Two groups of relevant HSPs are circled, and correspond to the two genes. By skipping the second HSP in group1 and the first HSP in group2, it may be possible to form another HSP group that still satisfies co-linearity (with the first HSP in group1 and the second HSP in group2), which is in fact undesirable because it contradicts the rationale of Proximity. There must be a systematic way of determining whether to skip a HSP or not.

Rationale V (Query Coverage): This is used to measure the quality of HSP groups. For a group of HSPs, the combined region of their query segments should cover the query sequence as much as possible. In Figure 9, Group 2 is better than either Group 1 or Group 3 because it covers a larger region of the query sequence.

Rationale VI (Single Group Membership): Since HSPs generally correspond to coding exons and it is extremely rare that different genes share coding exons [35], in this work, we require that each HSP belongs to at most one candidate gene region, thus, at most one HSP group.

Since there is some contention among Rationales III-V, it is not always clear how to best group HSPs such that each group satisfies the above requirements. This task becomes particularly challenging when a query has many homologous genes because there will be multiple HSP groups on the target sequences and the number of such groups is unknown *a priori*. genBlastA is designed to address all above requirements and challenges by modelling the HSPs with a directed graph.

3.3.3 Graph Modelling

An HSP graph is a graph representation that captures the above requirements on HSP groups. Each HSP is represented by a node, with edges that model the sequential ordering of the HSP target segments (Rationale I) and additional edges that model the skipping of HSPs (Rationale III). An HSP grouping is modelled by grouping the nodes on a path, such that each group covers as many query segments as possible while preserving co-linearity (Rationale II). In a later section, I will define a length metric of the edges that takes into account the quality of query coverage (Rationale III-V). By using this length metric, I will show that an optimal HSP group is a shortest path in the HSP graph. Rationale VI is enforced in a post-processing step that ranks all the HSP groups before output.

Before formally defining the HSP graph, let us first define some terminologies that describe the physical relationships between HSP target segments.

Definition 4.1 (Physical relationship of HSP target segments). Given HSP target segments T_m and T_n:

- *After/Before*: If T_n's starting position is larger than T_m's ending position, we say that T_n is *after* T_m, and T_m is *before* T_n;
- *Between*: If there exists another target segment T_k that is after T_m and before T_n, we say T_k is between T_m and T_n;
- *Adjacent*: If there is no other target segment between T_m and T_n, then we say T_m and T_n are *adjacent*. This includes the case where T_m and T_n overlap (i.e. T_m and T_n share some base pairs);
- *Later than/Ealier than*: If T_n's starting position is larger than T_m's starting position, we say T_n is *later than* T_m and T_m is *earlier than* T_n. This relationship compares two starting positions, thus, differs from T_n being *after* T_m. ■

Similarly, these relationships can be defined for HSP query segments based on their locations on the query sequence.

Definition 4.2 (HSP Graph). Given a collection of HSPs, each HSP is represented by a node in an HSP graph, with two types of physical edges constructed as follows. For two HSPs H_m:[Q_m,T_m] and H_n:[Q_n,T_n], where T_m and T_n are target segments and Q_m and Q_n are their corresponding query segments:

1. *Adjacent edges*: If T_m and T_n are adjacent and T_n is later than T_m, there is an edge $H_m \rightarrow H_n$;
2. *Skip edges*: If there is a path, but no direct edge, from T_m to T_n, then we add an edge $H_m \rightarrow H_n$ (i.e. transitive closure). ■

The adjacent edges in case (1) model the sequential ordering (Rationale I) where T_n follows T_m closely. The skip edges in case (2) model the possibility of skipping over noise HSPs (Rationale III). An edge $H_m \rightarrow H_n$ represents the possibility that H_n extends the HSP group that contains H_m. By following a skip edge, the group is extended without including all skipped nodes.

Figure 10 The HSP Graph

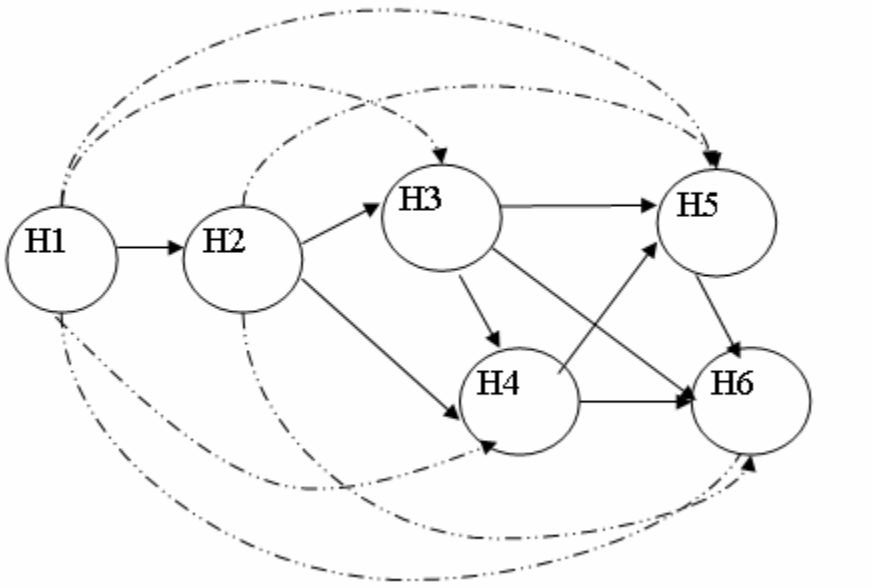


Figure 10 shows the HSP graph for the example HSPs in Figure 8. The solid edges are adjacent edges and the dotted edges are skip edges. Each path in the graph represents a way of selecting HSPs along the path. Such an HSP graph provides a complete search space for all possible groupings of HSPs. The number of skip edges can be very large. However, after introducing a length metric on edges (in “Length Metrics” section below), I will show that many skip edges can be removed without affecting the optimal grouping. genBlastA will not

construct such skip edges, thereby dramatically increasing the algorithm efficiency.

So far, an edge $H_m \rightarrow H_n$ indicates the sequential ordering of their target segments T_m and T_n, which is a necessary condition for extending the HSP group that contains H_m by H_n. Rationale II also requires that this order be consistent with the order of involved query segments (co-linearity). For example, in Figure 10, edge H1→H2 is between two HSPs H1: [Q1,T1] and H2: [Q1,T2], whose relationship is shown in Figure 8. T2 is later than T1 but Q1 is not later than Q1, therefore H2 cannot belong to the same group as H1. On the other hand, for edge H2→H3, H3's target segment T3 is later than H2's target segment T2, and H3's query segment Q2 is also later than H2's query segment Q1, therefore, this edge represents a valid group extension according to Rationale II.

The above discussion suggests that edges in an HSP graph can be labelled according to their *logical* functions: edges that represent group extensions and edges that end the current group and start a new group, due to violation of Rationale II.

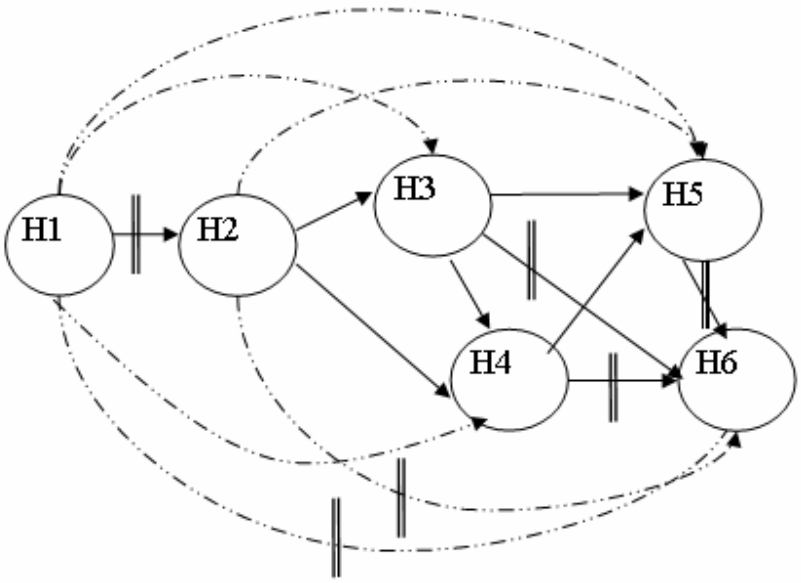
Definition 4.3 (Logical Edges in the HSP graph). For an existing edge $H_m \rightarrow H_n$ in an HSP graph, where H_m is $[Q_m,T_m]$ and H_n is $[Q_n,T_n]$,

- *Extension edge*: $H_m \rightarrow H_n$ is labelled as an extension edge if either (1) T_m and T_n overlap, and Q_m and Q_n are adjacent with Q_n being later than Q_m; or (2) if T_m and T_n do not overlap, Q_n is later than Q_m.

• *Separating edge*: If $H_m \rightarrow H_n$ is not an extension edge, then it is labelled as a separating edge. ■

Note that the definition of logical edges is independent of the physical edges and each type of physical edge can belong to either type of logical edges. Intuitively, an extension edge $H_m \rightarrow H_n$ means that the two HSPs H_m and H_n are collinear. In this case, the group that contains H_m may be extended by adding H_n after H_m. A separating edge $H_m \rightarrow H_n$ means that the HSPs are not collinear, therefore, H_n must belong to a different group from H_m.

Figure 11 The HSP Graph with logical edges identified



In Figure 11, to distinguish these two types of logical edges, we add a vertical bar to each separating edge. For example, H1→H2 is a separating edge, which means that its source node and destination node should belong to different

HSP groups. On the other hand, the skip edge H1→H3 is an extension edge, whereas another skip edge H1→H6 is a separating edge.

With the introduction of logical edges, each path in the HSP graph represents a way of filtering and grouping HSPs: as we traverse a path, following an extension edge extends the current HSP group to include the destination node, and following a separating edge ends the current HSP group at its source node and starts a new HSP group at its destination node. If an extension edge is a skip edge, following the edge will skip over the nodes on the paths that are shortcut by the edge. In this sense, the HSP graph provides a complete search space for filtering and grouping HSPs.

Note that in our implementation, an additional parameter of maximum intron length is used to impose an extra constraint on HSP groups, so that HSPs too far apart are not grouped together. This poses an upper limit on the distance between adjacent HSPs in the same group, which may affect the logical labelling of edges. It is different from the distance threshold used in other approaches [44], which tried to group HSPs based on distance. genBlastA works even without setting such limit. The use of a distance limit allows genBlastA to produce HSP groups with intron characteristics specific for the target species.

3.3.4 Finding the Best HSP Groups

To represent all groups in a uniform way, the HSP graph is augmented with two special nodes: node σ has an outgoing edge to all nodes with no incoming edges, and node τ has an incoming edge from all nodes with no

outgoing edges. All edges adjacent to σ and τ are regarded as separating edges. With this augmentation, every HSP group can be represented by an *extension path* of the form:

Extension path: $(H' \rightarrow H_1 \rightarrow H_2 \rightarrow \ldots \rightarrow H_k \rightarrow H'')$,

where $H' \rightarrow H_1$ and $H_k \rightarrow H''$ are separating edges and all other edges $H_i \rightarrow H_{i+1}$ are extension edges. This extension path represents the HSP group of *k* HSPs $(H_1, H_2, \ldots, H_k)$, where the target-query alignment of each H_i always satisfies the sequential ordering (Rationale I) and co-linearity (Rationale II). The addition of the two special nodes σ and τ makes sure that any regular HSP node has some other node before and after it in the HSP graph. Thus, in the case of the very first HSP node H_1 in the HSP graph, the first separating edge of its extension path is $\sigma \rightarrow H_1$; similarly, for the very last HSP node H_k in the graph, the last separating edge of its extension path is $H_k \rightarrow \tau$.

Suppose that we have a "length metric" on an extension path such that the shorter the extension path, the better HSP group it represents (as a candidate gene region). To find the best HSP groups, for every node H_1 that is the destination node of a separating edge $H' \rightarrow H_1$, we search for the shortest extension path $p=(H' \rightarrow H_1 \rightarrow H_2 \rightarrow \ldots \rightarrow H_k \rightarrow H'')$. This task is the single-source shortest path problem from node H_1. Node H_1 is called a *group starting node*. The shortest extension paths are well defined because the HSP graph is acyclic. Note that the choice of H' and H" does not have effect on the represented HSP group $(H_1, H_2, \ldots, H_k)$.

After all shortest extension paths for all group-starting nodes are found, they are then ranked by the length metric to obtain a ranked list of HSP groups for the current HSP graph. If there are multiple DNA sequences in the target genome, one HSP graph is constructed for each DNA sequence. The shortest extension paths are obtained from each HSP graph and the results are then ranked globally into a combined list.

From Rationale VI, each HSP can belong to at most one candidate gene region. Thus for nodes (HSPs) that are on the route of more than one shortest extension paths, a post processing step is performed to delete any shared HSP from all except the highest ranked group that contains the HSP. In other words, the HSPs that are already in a higher-ranked group will be removed from any lower-ranked groups.

The single-source shortest path algorithm for a directed acyclic graph can be done efficiently in O(E) time, where E is the number of edges [132]. Executing this algorithm once for each possible starting node H_1, the total running time is O(E•V), where V is the number of group starting nodes, i.e. the destination nodes of separating edges. V is bounded by the number of HSPs. The number of adjacent edges constructed by Definition 4.2(1) is not large because such edges are constrained by the requirement of physical adjacency between HSP target segments. However, the number of skip edges constructed by Definition 4.2(2) can be large. Fortunately, many skip edges do not need to be constructed, which will be discussed in the “graph optimization” section below. In order to establish such optimizations, let us first define the length metric in the HSP graph.

3.3.5 Length Metrics

The length metric is defined in line with the quality of a HSP group. Informally, an HSP group is good if it covers the query sequence as much as possible with a good match between target and query segments (Rationale V), does not contain noise HSPs (Rationale III), and does not skip non-noise HSPs (Rationale IV). Consider an HSP group represented by an extension path *p*=(H' $\rightarrow H_1 \rightarrow H_2 \rightarrow \ldots \rightarrow H_k \rightarrow$ H"). For any edge *e* on the path, there are possibly both rewards and penalties associated with the edge, based on the notion of quality of a HSP group. *P*(*e*) represents the penalty for possible missing coverage of the query sequence and possible skipping of non-noise HSPs when following the edge *e*. *R*(*e*) represents the reward for query regions that are covered by the source node of *e*. To minimize the penalty and maximize the reward, the length of edge *e* is defined to be:

$$Length(e) = P(e) - R(e) \tag{4.1}$$

And the length of path *p* is the total length of all edges on *p*:

$$Length(p) = \sum_{e \in p} Length(e) \tag{4.2}$$

The definition of *P*(*e*) and *R*(*e*) is based on the notion of *weight* for HSPs. Consider an HSP H: [Q,T] with the alignment length *Len* and percentage of identity *PID*. An HSP is considered of higher quality if it has a longer query segment and higher PID. So the *weight of an HSP H* is defined as:

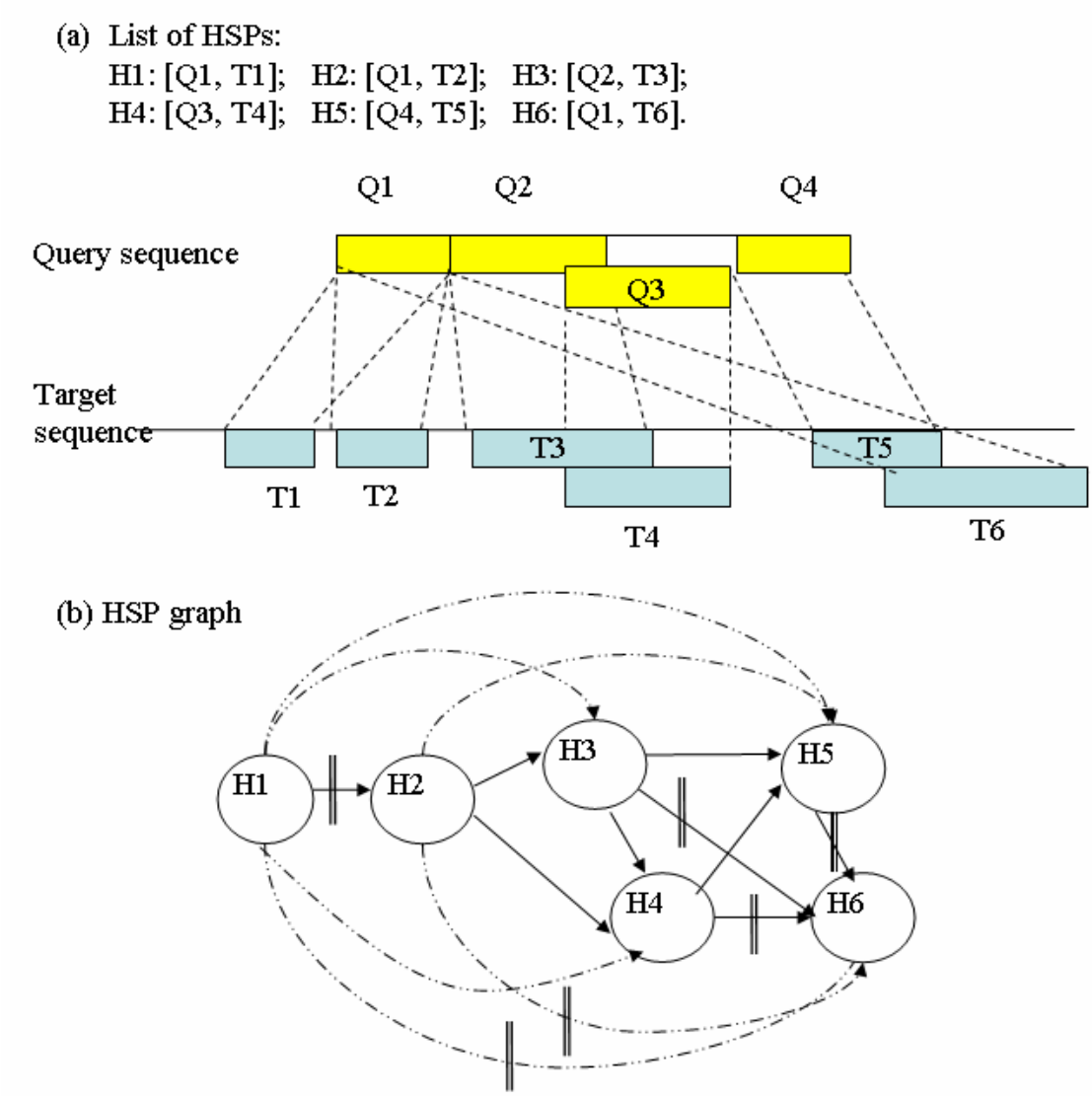
$$W_H = Len * PID \tag{4.3}$$

A query segment Q may be involved in several HSPs. Let *AVG_PID* be the averaged PID over all HSPs that have Q as the query segment. The *weight of a HSP query segment Q* is defined as:

$$W_Q = Len * AVG\_PID \tag{4.4}$$

Consider the HSPs in Figure 12(a) and the corresponding HSP graph in Figure 12(b). By following the edge H3→H5, we miss the part of query between Q2 and Q4, i.e. the part of Q3 that does not overlap with Q2, denoted as (Q3−Q2). Because the overlapped part of Q2 and Q3 has been covered by H3, when we compute the weight of this missed query coverage, *Len* in Equation (4.4) is the length of (Q3−Q2) and *AVG_PID* is the PID of H3 (since H3 is the only HSP that aligns with (Q3−Q2)). Similarly, for the overlapped part of Q2 and Q3, denoted as (Q2∩Q3), its *Len* is the length of (Q2∩Q3) and *AVG_PID* is the average of PIDs between H3 and H4 (since both HSPs align with (Q2∩Q3)).

Figure 12 Example HSPs and the corresponding HSP graph



I now present the definition of reward and penalty for edges, i.e. $R(e)$ and $P(e)$.

Definition 4.4 (Edge Reward and Edge Penalty). For any edge e, its edge reward $R(e)=W_H$, where H is the HSP at the source node of edge e. Note that, for the first edge $H' \rightarrow H_1$ on an extension path, $R(H' \rightarrow H_1)=0$ because H' is

not included in the HSP group. On the other hand, the edge penalty $P(e)$ is equal to the weighted sum of three penalties:

$$P(e) = \alpha * [P_{ext}(e) + P_{sep}(e)] + \beta * P_{skip}(e) \quad (4.5)$$

$P_{ext}(e)$, which applies to an extension edge e, captures the penalty on missing query coverage when following an extension edge e. For an extension edge e: $H_m \rightarrow H_n$, where H_m has the query segment Q_m and H_n has the query segment Q_n, $P_{ext}(e)$ is defined as the total weight of all missing parts of the query between Q_m and Q_n. In Figure 11, P_{ext} (H2→H4) is equal to the weight of the part of Q2 that does not overlap with Q3 because this part is missed when following this edge.

$P_{sep}(e)$, which applies to a separating edge e, captures the penalty on missing query coverage when following a separating edge e. Note that a separating edge is either the first or the last edge on an extension path. If e is the first edge $H' \rightarrow H_1$ of an extension path, the part of the query before H_1's query segment will be missed and $P_{sep}(e)$ is equal to the total weight of such missing query parts. If e is the last edge $H_k \rightarrow H''$ on an extension path, the part of the query after H_k's query segment will be missed and $P_{sep}(e)$ is equal to the total weight of such missing query parts. For example, in Figure 11, for every extension path with the separating edge H4→H6 as the last edge, where H4's query segment is Q3, the HSP group ends at H4, therefore all query parts following Q3 (i.e. the entire Q4) are missed by the group.

$P_{skip}(e)$, which applies to a skip edge e, captures the penalty on skipping HSPs when following a skip edge e, which may result in HSP groups that span multiple gene regions (counter to Rationale IV). Note that if e is also a separating edge, e does not extend any HSP group and $P_{skip}(e)=0$. If e is an extension edge $H_i \rightarrow H_{i+1}$, following e will skip the HSPs on any non-skip paths from H_i to H_{i+1} and $P_{skip}(e)$ is the minimum total weight of the HSPs on a single such path. In Figure 11, for the skip edge H2→H5, there are 3 paths with non-skip edges from H2 to H5: (H2→H3→H5), (H2→H3→H4→H5) and (H2→H4→H5), which skip HSPs (H3), (H3, H4), and (H4), respectively. P_{skip}(H2→H5) is equal to the minimum of W_{H3} and W_{H4}. ■

With the above definitions of length metric, the length of each extension path in the HSP graph can be measured and ranked. The driving force for following a skip edge e is the increased $R(e)$ (reward) if it extends a group with a better although not adjacent HSP. On the other hand, $P_{skip}(e)$ (penalty) will be increased as well. A skip edge is followed only when the reward is bigger than the associated penalty. By incorporating such a trade-off between reward and penalty, the decision of whether to skip a HSP is made systematically during the search for shortest paths. This approach is more robust than trying to remove noise HSPs by *ad hoc* score thresholds or separating HSP groups by *ad hoc* physical distances.

In formula (4.5), α and β are constants, where $\alpha+\beta=1$, representing the relative importance of the penalties for missing query coverage and skipping HSPs. The purpose of using these parameters is to add flexibility to the program,

which allowed us to study the effect of penalties. They are *not* rigid thresholds and do not depend on specific genomes. In fact, several settings for *α* and *β* have been tested (with α = 0.25, 0.5, 0.75, 0.9), and the results for all were similar despite different values of *α*. This shows that such parameters are different from *ad hoc* thresholds that are crucial in other programs such as [44]. In fact, genBlastA is quite insensitive to the settings of these parameters, demonstrating the robustness of genBlastA.

3.3.6 Graph Optimization

As discussed earlier, finding all shortest extension paths has a running time of O(E•V), where E is the number of edges in the HSP graph, V is the number of nodes, i.e. the number of HSPs. I now show that many of the skip edges are redundant in that they are never used by any shortest path, and therefore can be pruned before the search starts. This optimization does not affect the HSP groupings, but will shorten the running time of genBlastA.

Definition 4.5 (Redundant edges). An edge is redundant if its removal from the HSP graph does not affect the result of shortest extension paths. ■

I now present two theorems that are used to identify redundant edges. Unless otherwise specified, all nodes refer to the normal nodes representing actual HSPs, not the special nodes σ or τ.

Theorem 1. A skip edge $H_m \rightarrow H_n$ that is a separating edge is redundant.■

Proof: With $H_m \rightarrow H_n$ being a separating edge, an extension path *p* using this edge has two cases: (1) the path ends with $H_m \rightarrow H_n$, or (2) the path starts

with $H_m \rightarrow H_n$. We consider case (1) where p is $(H' \rightarrow H_1 \rightarrow \ldots \rightarrow H_m \rightarrow H_n)$, as case (2) is symmetric. For a skip edge $H_m \rightarrow H_n$, there must exist a path from H_m to H_n that contains only non-skip edges. Let x be the shortest prefix of this path that ends with a separating edge. Let path p' be the path p with $H_m \rightarrow H_n$ being replaced with x. If x is a single separating edge, $Length(p')=Length(p)$. If x contains at least one extension edge, p' extends p by more nodes without introducing new P_{skip} penalty, because all edges on x are non-skip edges. On the other hand, by including the additional nodes on x, p' has less penalties of P_{ext} and P_{sep}, and more rewards than p because it has more HSPs to cover more query segments. Thus, $Length(p')<Length(p)$, and removing the edge $H_m \rightarrow H_n$ has no effect on shortest extension paths. ■

The next theorem shows that skip edges that are transitive closure of extension edges are redundant.

Theorem 2. A skip edge $H_m \rightarrow H_n$ that is an extension edge is redundant if there is another node H_k such that $H_m \rightarrow H_k$ and $H_k \rightarrow H_n$ are extension edges. ■

Proof: Any HSP group produced by following the skip edge $H_m \rightarrow H_n$ can be represented by an extension path $p=(\ldots \rightarrow H_m \rightarrow H_n \rightarrow \ldots)$. Let p' be the modified path $(\ldots \rightarrow H_m \rightarrow H_k \rightarrow H_n \rightarrow \ldots)$ where the prefix and suffix remain unchanged. p' has more rewards than p since p' has one additional node H_k. Both P_{ext} and P_{skip} by following $H_m \rightarrow H_k \rightarrow H_n$ are less than by following $H_m \rightarrow H_n$, since H_k covers one additional query segment between H_m and H_n, and $H_m \rightarrow H_n$ skips more nodes

than $H_m \rightarrow H_k \rightarrow H_n$ does. Thus, *Length*(*p'*) < *Length*(*p*) and removing $H_m \rightarrow H_n$ has no effect on shortest extension paths. Thus $H_m \rightarrow H_n$ is redundant. ■

With these optimization strategies, the number of skip edges in the HSP graph is dramatically reduced, thereby increasing the efficiency of genBlastA. For example, on a moderate PC with Pentium IV 2.6GHz CPU, 1G memory and running Windows XP, it takes less than 60 seconds to process 300 *C. elegans* query genes with over 36,000 HSPs. On average, the total number of edges in the HSP graph in our experiments is less than 2 times of the number of HSPs, i.e. the number of nodes in the HSP graph. This shows the effectiveness of the graph optimization strategies.

3.4 The Effectiveness of genBlastA

genBlastA is designed to identify groups of HSPs that represent homologous genes. Its effectiveness has been tested by applying genBlastA to find HSP groups that represent orthologs (genes in different species but with same origin in evolution) and paralogs (genes duplicated within a species), using *C. elegans* [39] and *C. briggsae* [60] genomes.

3.4.1 Test Genes and Experiment Setup

The entire genomes of both *C. elegans* and *C. briggsae* have been sequenced. In particular, as the first multicellular organism whose genome was completely sequenced, *C. elegans* is a model organism that has been extensively annotated [66] and thus is often used to evaluate new algorithms. On the other hand, *C. briggsae*, a sister species of *C. elegans*, is not as well studied

and the existing annotations are usually generated by other gene prediction programs. It provides an excellent platform to study the effectiveness of gene detection in related species.

The test genes were obtained from WormBase (http://www.WormBase.org/), an integrated genome database for *C. elegans* and other nematode species including *C. briggsae* [33]. The C. elegans genome size is about 100 Megabases, containing 6 chromosomes and about 20,000 protein-coding genes in total. To test the effectiveness of genBlastA, 464 representative *C. elegans* genes are selected. The majority (300 genes) of these genes were taken from three representational contiguous regions of *C. elegans* chromosome I. These three regions are the left arm, the middle region, and the right arm of chromosomal regions, each containing 100 genes. To ensure that the test gene set contains representative genes of different complexities, additional 164 genes are added to the set, including genes with regions of internal repeats and genes that belong to large paralogous tandem clusters. The test gene set can be downloaded from http://genome.sfu.ca/projects/genBlastA/.

The test genes are used as queries to search against two target genomes: *C. elegans* and *C. briggsae*. The *C. elegans* genome is used to test the capability of genBlastA in identifying paralogous genes (called *EvsE test*). The *C. briggsae* genome is used to test genBlastA in identifying orthologous genes (called *EvsB test*). In addition, two different BLAST settings are tested when producing HSPs, “ungapped” and “gapped”, in order to test the effect of this setting on the final

grouping. The gapped HSPs are generally longer with more gaps and mismatches, and ungapped HSPs are generally shorter with higher PIDs.

The HSP groups produced by genBlastA were compared with those of two existing programs with similar functionalities --- WU-BLAST [80] and the program by Cui et al. [44]. WU-BLAST is available by an academic license. Since the HSP grouping functionality of the program by Cui et al. is not readily available, it is implemented based on their publication and denoted as ML in the following text. ML requires a distance threshold to resolve different HSP groups. This threshold is not described in detail in their publication; therefore, it was derived from several best attempts and a value that led to overall best performance of ML in our experiments was used, which was 1000bp.

3.4.2 Resolving Paralogous Genes in Tandem Clusters

This first experiment was designed to test the programs' capacity to resolve HSP groups that correspond to paralogous genes in multi-gene families or tandem clusters. For this purpose, 30 genes from the test gene set that belong to large gene families in tandem clusters were used as queries to search against the *C. elegans* genome. After HSP groups are produced by genBlastA, WU-BLAST, and ML, all candidate regions with query coverage $\geqslant$50% are retained. The HSP groups were then examined and compared against the annotations in the *C. elegans* genome database in WormBase. An HSP group is called "specific" if its corresponding genomic region contains only one annotated gene; otherwise it is called "nonspecific" if the region overlaps with multiple annotated genes. Intuitively, HSP groups with high query coverage and containing only

single genes are likely defining true paralogs. The programs' capabilities in resolving multiple paralogous genes are measured by the ratio of specific groups, i.e. the number of groups that are specific versus the total number of HSP groups examined.

Figure 13 illustrates an example, in which there are five paralogous genes in a tandem gene cluster. The annotated gene models are shown in the "Gene Models" track at the top. HSPs are shown as blue boxes in the "All HSPs" track, followed by the tracks of HSP groups reported by genBlastA, ML, and WU-BLAST, respectively. It can be seen that WU-BLAST had a hard time deriving sensible groups from HSPs, because the HSPs heavily overlap as a result of multiple similarities in a close genomic region. It managed to identify only one HSP group that matches a target gene and failed to produce groups corresponding to the rest four genes. ML produced three groups, two of which erroneously contain HSPs corresponding to other adjacent genes. ML missed groups for two target genes (T27B7.4 and T27B7.6a), and mistakenly grouped HSPs corresponding to T27B7.6a to the HSP group corresponding to T27B7.5. In contrast, genBlastA successfully resolved all five genes, producing five groups of HSPs that properly define the approximate regions of all paralogous genes.

Figure 14 shows the ratio of specific HSP groups among all predicted groups by the three programs. The error bars represent the standard error. The values show statistical significance by paired Student's T Test (with p-value < 0.001) [100]. In summary, when HSPs were produced by the ungapped setting in the *EvsE test*, the average ratio of specific HSP groups by genBlastA is around

Figure 13 Grouping HSPs into groups representing homologous genes in tandem clusters

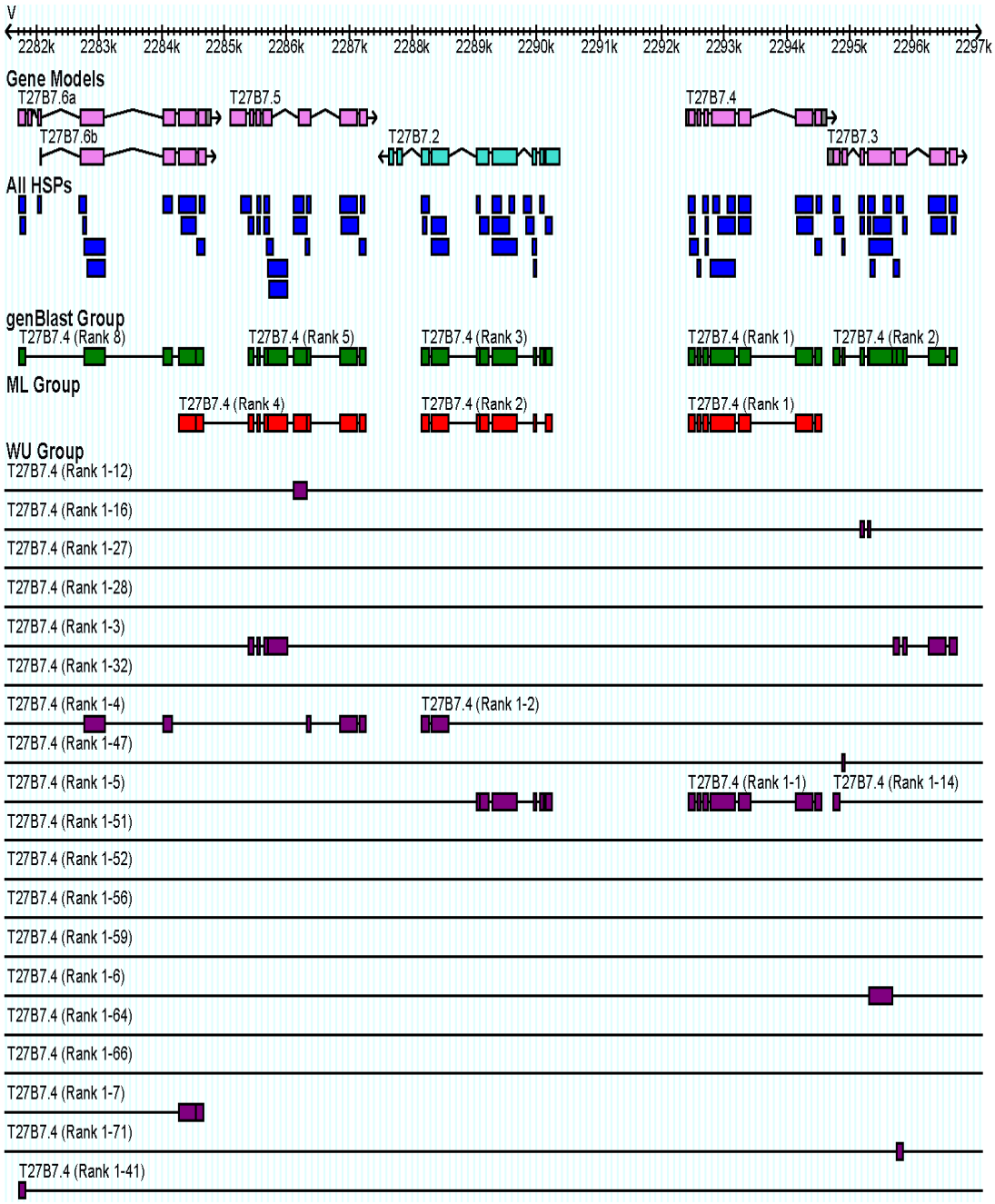
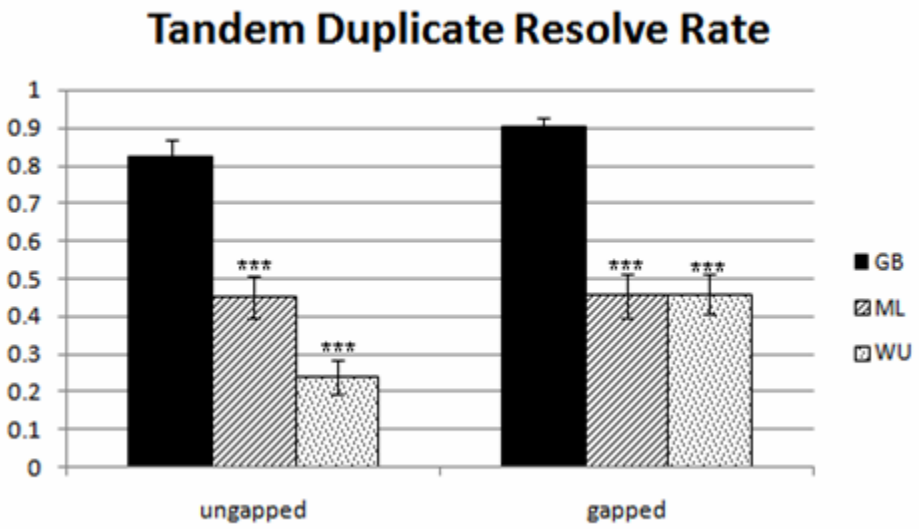


Figure 14 Comparison of genBlastA, ML and WU-BLAST in Resolving Tandem Genes



80%, which is significantly higher than that produced by WU-BLAST (about 20%) or ML (about 40%), as shown in Figure 14. With the gapped HSPs, genBlastA performs even better, with over 90% of HSP groups being specific. WU-BLAST often generates numerous HSP groups, but they usually span genomic regions of multiple genes (therefore nonspecific). Consequently, WU-BLAST frequently groups together tandem paralogous genes and cannot correctly identify regions of individual genes. ML also exhibited poor performance due to its use of a distance threshold, although being somehow better than WU-BLAST. In particular, as its distance threshold increases, the ability of ML to resolve closely spaced paralogous genes decreases. In all cases, genBlastA was able to resolve much more specific HSP groups in tandem clusters compared to either WU-BLAST or ML.

3.4.3 Searching for Orthologous Genes

This experiment was designed to test genBlastA identification of homologous gene regions that are most similar to the query. All 464 genes in the test gene set were used as queries. For each query, the top-ranked HSP group, i.e. the candidate ortholog of the query gene, is examined. Since the top-ranked group is expected to be the most similar to the query gene, in the *EvsE test*, it is expected to map to the query gene itself; in the *EvsB test*, it should map to its *C. briggsae* ortholog. The top-ranked HSP group is thus compared with the annotated gene in WormBase.

For accurate comparisons, the quality of the HSP group is measured by the following two criteria: (1) *query coverage*, and (2) *genomic span*. Query coverage measures the similarity between the HSP group and the query gene. It is defined as the proportion of the query sequence covered by the HSPs in the HSP group. The higher coverage generally implies a better HSP group. Genomic span measures the extent of overlap between the genomic region given by the HSP group and the expected gene region as annotated in WormBase. This is computed using the Jaccard similarity: given the annotated target gene region R_A and the gene region R_H that is defined by a HSP group, their similarity is $(|R_A \cap R_H|/|R_A \cup R_H|)$, i.e. their intersection size divided by their union size. This result is 0 when two regions do not overlap and 1 when the regions align perfectly.

3.4.3.1 Comparisons of Query Coverage

Figure 15(a, c) shows the average query coverage of all three programs for 464 query genes in the test gene set. Figure 15(a) shows the results in *EvsE test* and Figure 15(c) shows the results in *EvsB test*. In *EvsE test*, with both ungapped and gapped HSPs, genBlastA identifies HSP groups with close to 100% query coverage and significantly outperformed both WU-BLAST and ML. In the *EvsB test*, genBlastA has average gene coverage of about 90%, and continues to outperform the other programs.

3.4.3.2 Comparisons of Genomic Span

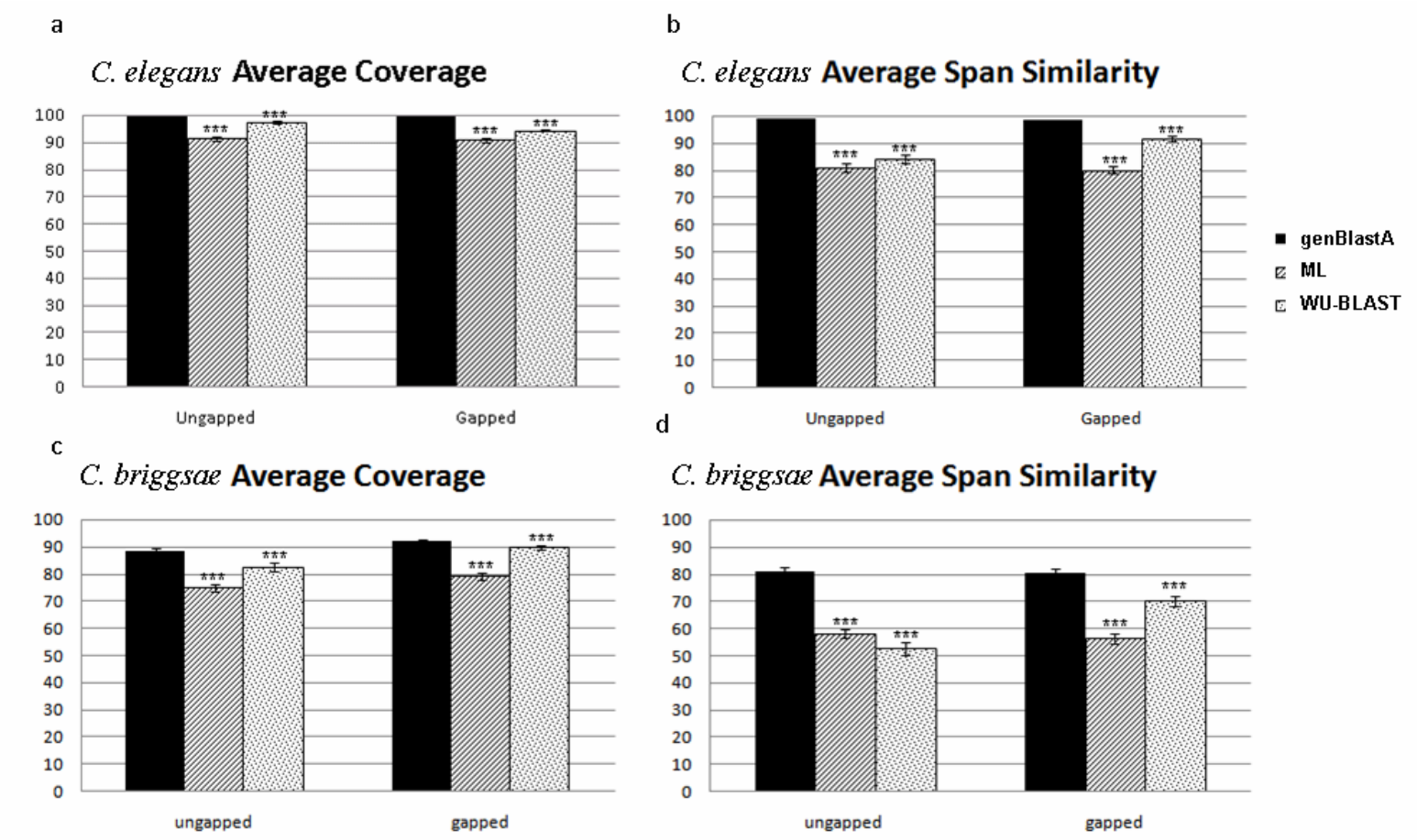
Figure 15(b, d) shows the average genomic span similarity of the three programs, with Figure 15(b) showing the results in *EvsE test* and Figure 15(d) showing the results in *EvsB test*. In *EvsE test*, genBlastA obtained close to 100% span similarity, outperforming both WU-BLAST and ML by large margins, suggesting that genomic regions predicted by WU-BLAST and ML are quite different from the real genomic regions. For *EvsB test*, genBlastA also outperformed both WU-BLAST and ML. When comparing WU-BLAST with ML, WU-BLAST shows better genomic span than ML in gapped setting and worse in ungapped setting. This may be due to the difficulty of WU-BLAST in assigning proper group memberships when there are many overlapping HSPs in ungapped setting.

3.4.4 Discussions

As shown in the above experiments, genBlastA outperformed both WU-BLAST and ML in identifying both paralogous and orthologous HSP groups. Its performance is quite stable across different settings of BLAST (either “gapped” or “ungapped”) that generated different set of HSPs. This demonstrates the robustness of the genBlastA algorithm.

With genBlastA, users can quickly interpret the large list of HSPs and effectively identify homologous gene regions as defined by the HSP groups. Because each group represents a full-length candidate gene, rather than fragments of a gene (HSPs), genBlastA provides insights on gene structures and allows users to focus on the interesting targets, which can then be explored further.

Figure 15 Query Coverage and Genomic Span Comparisons for Orthologous Gene Detection



4: GENBLASTG: RESOLVING GENE STRUCTURES

With a given query protein and the target genome, I have shown that genBlastA is able to effectively find the candidate regions where homologous genes are located. This is a good starting point. But these regions do not tell the exact structure of genes. One immediate task is to examine these candidate gene regions and resolve their gene structures, i.e. determining the exact position of the gene and its components (exons and introns).

There are two ways to do this. One way is to apply existing gene prediction tools on the candidate regions to predict possible gene structures, such as the popular tool GeneWise [20]. However, as discussed earlier, these HMM-based programs are usually slow. The other way is to directly make use of the HSP groups produced by genBlastA and try to predict gene structures based on the HSPs. This is possible because HSPs frequently have certain correspondence to the exon regions of the gene, as illustrated in Figure 6. Thus they can be exploited further for the task of gene prediction. This basic observation motivates the approach in this thesis and leads to the new gene prediction program, genBlastG [111], which is the topic of this Chapter. Although genBlastG does not use complex computational models, our experiments showed that it is able to achieve higher accuracy than other gene prediction methods.

4.1 Problem Statement and Challenges

genBlastA parses HSPs generated by sequence similarity search tools into HSP groups. Each HSP group defines a candidate homologous gene. The genomic positions of HSP target segments can be used to define exon positions of the gene contained in the corresponding HSP group. However, it is not straightforward to map the HSPs to exons due to the following reasons.

- First, appropriate splicing signals need to be resolved. The restrictions on gene start and gene stop with proper codons must also be considered.
- Second, HSPs often contain gaps and mismatches in their alignments and the boundaries of exons usually do not coincide with boundaries of HSPs.
- Third, due to the threshold based alignments of most local alignment tools (including BLAST), HSPs are extended as long as its score is above some given minimum threshold. Consequently, one HSP may correspond to the region that contains multiple exons, especially when the intron between exons is small.
- Due to mutations accumulated in evolution, target exons may not have precise correspondences with the query protein and it is possible for the region of one exon to be represented by multiple HSPs.
- Finally, some exons may not have correspondence with any HSP, especially for small exons that are easily missed by sequence similarity search tools.

To tackle these challenges, genBlastG utilizes the sequence alignment information contained in HSPs and makes adjustments when necessary. Given the process of gene expression, the DNA sequence that is translated into protein is a concatenation of all exons in a gene, which is obtained by removing introns and joining exons. Such sequence will be referred to as the *spliced sequence* in the following text. The basic intuition of genBlastG is to find the exon-intron structure that results in maximized sequence similarity between its spliced sequence and the query protein. Existing HSPs provide alignment information that can be directly used in this process.

4.2 genBlastG Overview

genBlastA returns a ranked list of HSP groups, with each group corresponding to a potential gene homologous to the query. Each group is independent of one another. Thus genBlastG examines each HSP group in the ranked order and process them independently. Usually only the top few ranks are of interest, because lower ranked groups carry much less sequence similarity to the query and usually do not represent complete or relevant genes. genBlast allows the user to control the number of ranks to be examined. Without loss of generality, the following presentation of the genBlastG algorithm discusses the processing of one HSP group.

Starting with the HSP group generated by genBlastA, genBlastG directly uses HSPs for gene prediction. The basic rationales behind the genBlastG algorithm are simple. First, the locations of exons are approximated by the genomic regions of HSPs, as defined by the genomic positions of HSP target

segments. However, this does not mean exon boundaries are easily defined, as the HSPs only provide approximate references. On the other hand, this does mean that exons must overlap at least partially with HSPs. Second, the sequence similarity between the spliced sequence and the query protein is used as the quality measure to guide the search for the best possible splice sites on the target genome. genBlastG identifies boundaries of introns and exons so that the spliced sequence show the maximized similarity to the query gene.

The task of resolving a gene model involves the determination of gene start, gene stop, and all splice sites, each of which will be discussed below.

Gene Start / Gene End. Given a group of HSPs, the approximate gene region is defined as the genomic region between the start of the first HSP and the end of the last HSP. In order to find genes that code proteins similar to the entire query sequence, the gene start should be searched close to the beginning of the gene region. Similarly, gene end should be searched close to the end of the entire gene region. There should be no in-frame stop codon between gene start and gene end that disrupts gene expression.

Gene start is usually signalled by a start codon of “ATG” sequence (which codes for methionine (M) and serves as an initiation site), with possible alternatives. genBlastG searches for a start codon at the beginning or adjacent to the first HSP. This start codon also needs to be in frame with the first HSP, in order for the first HSP to correspond to the first exon or at least a partial exon. This means the distance between the beginning of the start codon and the beginning of the first HSP must be a multiple of 3. In addition, there must be no

stop codon (either “TGA”, “TAG”, “TAA”) between the gene start and the first HSP. Thus the gene start is detected by searching from the beginning of the first HSP and going upstream, until hitting either a start or stop codon. The first start codon defines the exact location of the gene start. If a stop codon is encountered first, the beginning of the first HSP will be used as the gene start.

Gene end is always signalled by one of the three stop codons “TAG”, “TGA” or “TAA”. Similar to the search for gene start, genBlastG looks for the first in-frame stop codon from the end of the last HSP and going downstream. There is no need to monitor signals other than the stop codons.

Note that the sites of gene start and gene end found in this initial process are not necessarily the final sites. In a post-processing step of genBlastG, it is possible for either gene start or gene end to be adjusted according to additional evidences of sequence similarity in the beginning or end region of the gene. The post-processing in genBlastG will be discussed in a later section.

Splice Sites. Splice sites also have well-defined signals. A canonical intron start with the base pairs “GT” and end with the base pairs “AG”, which are referred to as the splice “donor” and splice “acceptor” signals, respectively. There are also non-canonical splicing signals [30]. However, the presence of these signals is not sufficient to identify the splice sites because there are many random pairs of donor/acceptor signals along the entire region of the gene. genBlastG tackles the splice site detection problem by dividing it into several smaller tasks: first, the approximate regions for each intron are determined; next, for each approximate intron region, some candidate donor sites and acceptor

sites in that region are selected; then, the best combination of donor and acceptor sites among these candidates is found for each intron region. Finally, once the initial gene structure is found, some post-processing is needed, during which the query coverage of the initial exons are examined and the gene structure is adjusted if necessary. These four steps are discussed in details in the next sections.

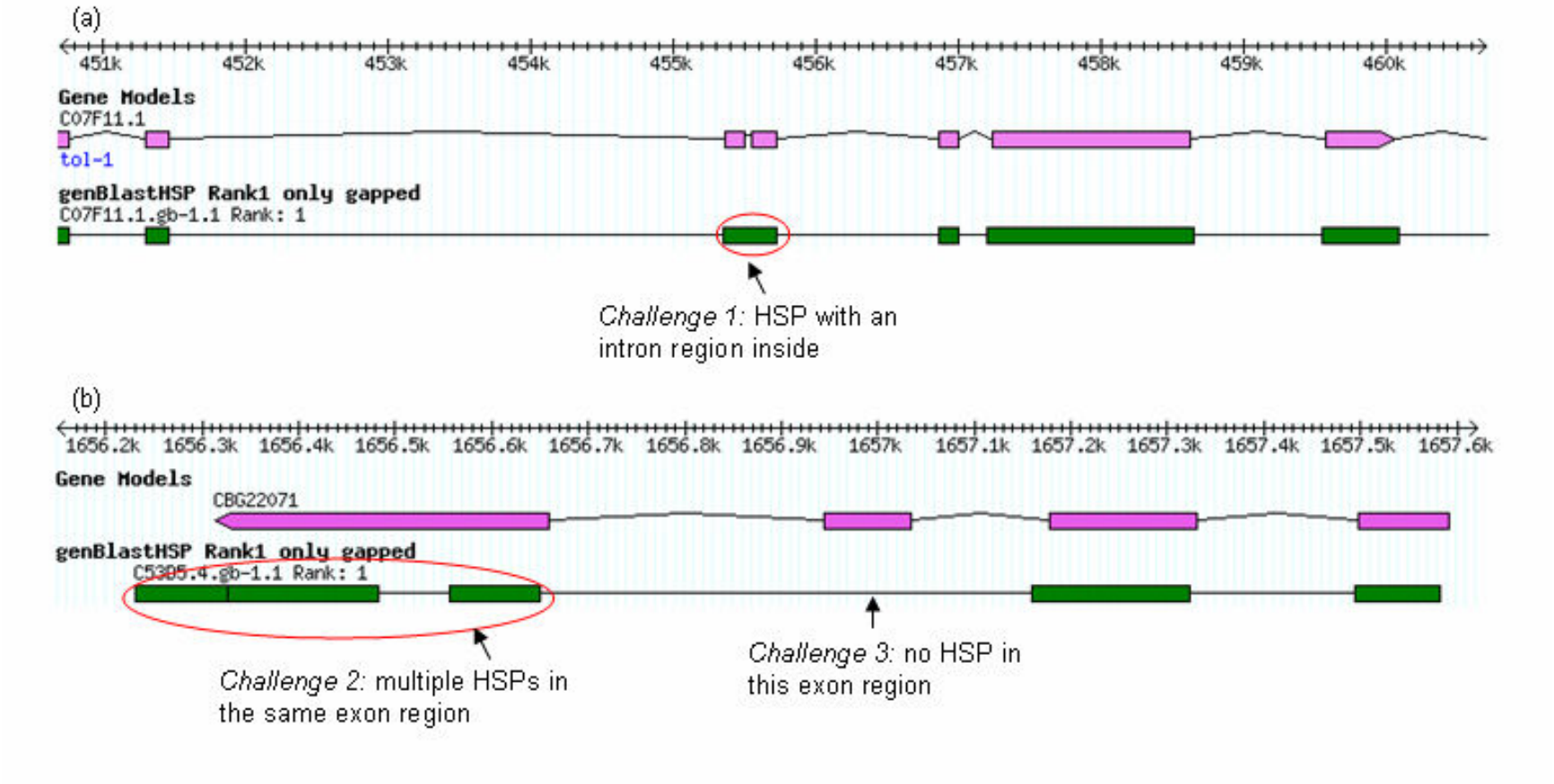
4.3 Step 1: Determine Intron Regions

In this step, we locate the approximate genomic regions for introns. With the entire gene region given by the HSP group, the simplest case is that HSPs correspond to coding exons and the genomic regions between adjacent HSPs represent introns. However, there is no simple one-to-one correspondence between HSPs and exons. As discussed earlier, there are many challenging exceptions, as illustrated in Figure 16, where HSPs are shown below the annotated gene models. This figure gives a concrete example of the correspondence between HSPs and exons. Although usually one HSP corresponds to one exon, there are many exceptions, such as:

- one HSP corresponds to multiple exons (Challenge 1); or
- multiple HSPs correspond to one exon (Challenge 2); or
- some exon regions may not have corresponding HSPs (Challenge 3), which requires some additional adjustment that will be discussed later.

To tackle these challenges, the following guidelines are identified. First, because the query is a protein product of the gene and is translated from exons

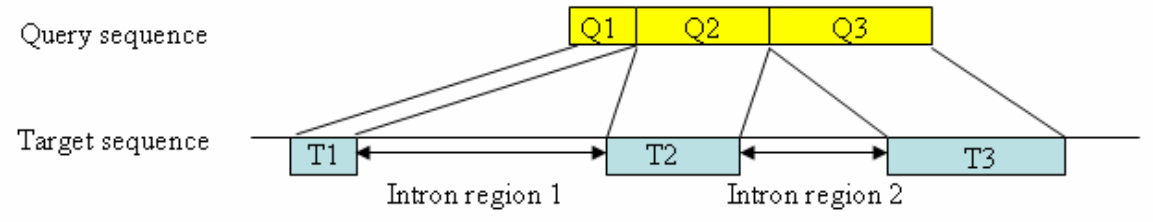
Figure 16 HSPs and Their Exon Correspondences



only, an intron should be the part of genomic sequence that has no query correspondence. Second, the genomic sequence formed by splicing the intron regions and joining the exons, i.e. the spliced sequence, should have high sequence similarity to the query protein sequence. These guidelines lead to the following heuristics of identifying intron regions.

(1) Introns between adjacent HSPs. First, as a common case, the genomic region between two adjacent HSPs is an intron region if it is more than certain length (Figure 17). This length is a user-defined threshold, MIN_INTRON_REGION_LEN, which represents the minimum length of an intron region and can be adjusted for different species, since studies have shown that intron lengths may differ for different species [46]. In our later experiments on *C. elegans* and *C. briggsae* genomes, this length is set to 15.

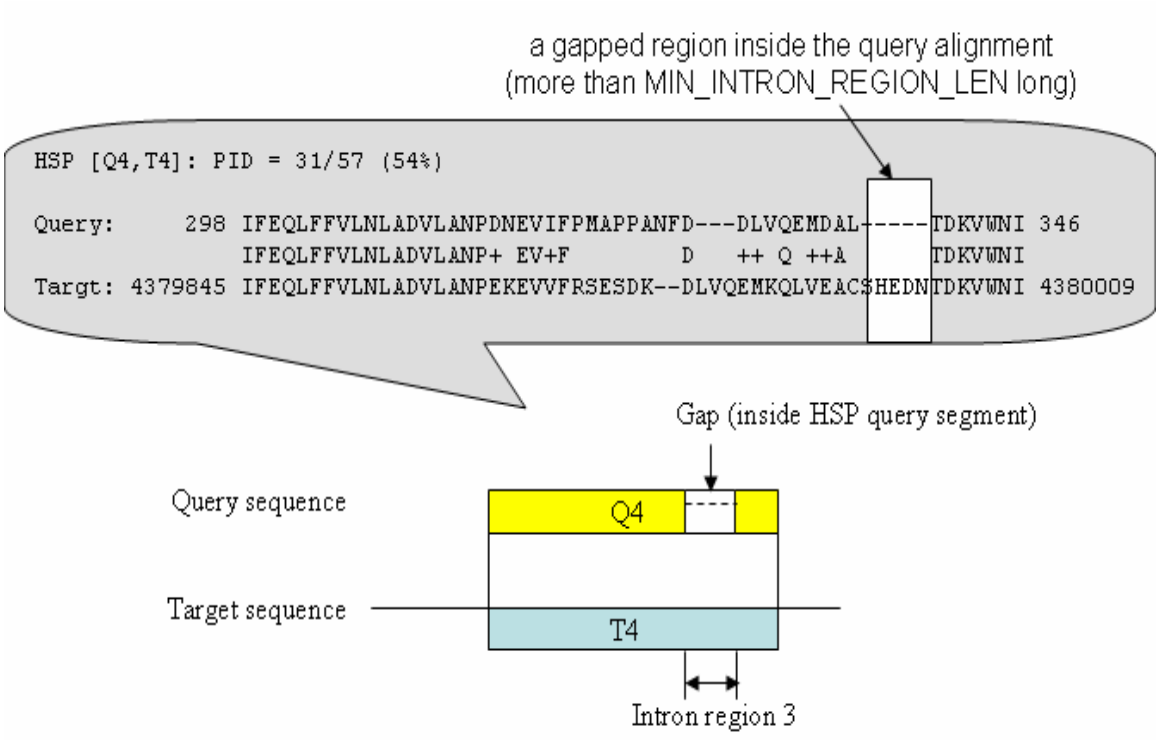
Figure 17 Intron Regions between Adjacent HSPs



(2) Introns within a HSP. As shown by challenge 1 in Figure 16, there are possible intron regions inside a HSP. Thus we need to examine the alignment of each HSP. Since we expect the intron region to have no query correspondence, the intron region inside a HSP should be aligned with gaps on the query sequence. Therefore, if there is a region in the HSP alignment where the query segment consists of continuous gaps that are longer than

MIN_INTRON_REGION_LEN, that region is considered to be a candidate intron region. For example, in Figure 18, given a HSP: [Q4, T4], the gap in its query alignment leads to the intron region 3, which is the region on target segment that is not aligned with any amino acid in the query.

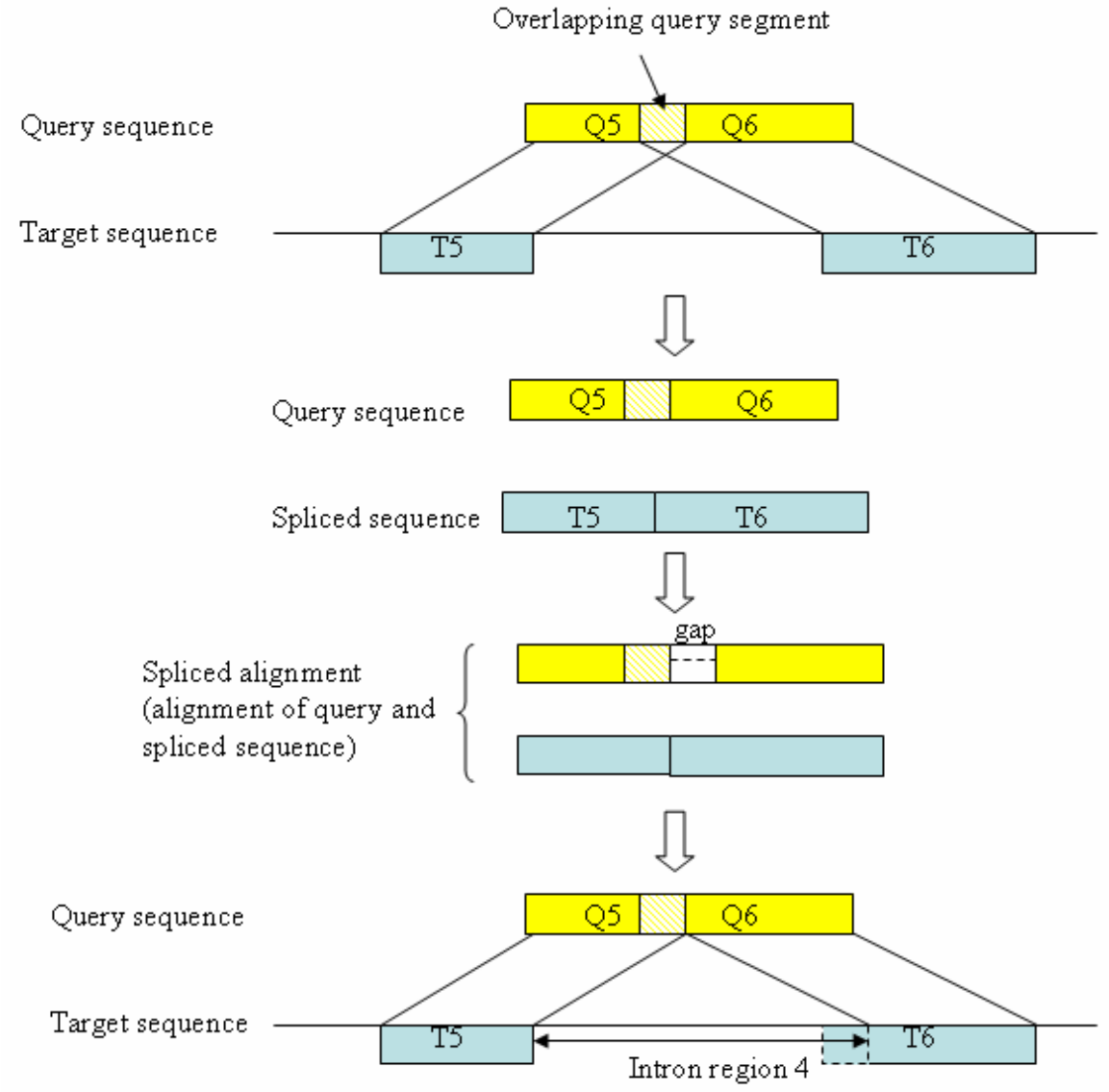
Figure 18 Intron Region inside a HSP



(3) Introns between adjacent HSPs with overlapping query segments.

The borders of the intron region between two HSPs may need to be adjusted if the two HSPs contain overlapping query segments. Since HSPs roughly correspond to exons, before exons are identified, a preliminary spliced sequence can be formed by gluing together HSP target segments. Because the entire group of HSPs is expected to represent one complete gene that is homologous to the query, the spliced sequence should align well with the query sequence.

Figure 19 Intron Region between HSPs with Overlapping Query Segments



Consider two adjacent HSPs with overlapping query segments shown in Figure 19. When aligning the spliced sequence against the query sequence, because there must be one-to-one correspondence in the sequence alignment, the overlapped part of the query segment can be aligned with only one of the HSP target segments. We chose to use the HSP target segment that has higher identity in the overlapping part. Thus the other HSP target segment will be

aligned with inserted gaps on the query (i.e. has no query correspondence), which in turn will be included in an intron region.

4.4 Step 2: Select Candidate Splice Sites

This step selects the candidate splice sites according to the intron regions given by the previous step. An intron region serves as an anchor that defines the approximate boundaries of a possible intron and a starting point to search for appropriate splicing signals. Once the intron regions are detected using the HSPs, the splice site detection problem is simplified to selecting splice sites that are close to the borders of intron regions. In particular, the upstream border of an intron region is the starting place to look for donors, and the downstream border is the place to start looking for acceptors. Because the intron regions are only approximate boundaries of introns, flexibility needs to be added by considering multiple choices of splice sites around the borders of intron regions.

Only canonical signals (“GT/AG”) are used in our current implementation. Note that it is easy to incorporate other signals to address variability in the sequence motif, by searching for additional signals in the intron region. For each intron region, the donor signal (“GT”) and the acceptor signal (“AG”) are searched independently. A number of splice signals that are closest to the borders are selected as the candidate splice sites. This is done by using a user-defined threshold (MAX_NUM_SPLICE_SITES) to control the number of candidates selected around each border, i.e. for the border of each intron region, the number of donors is at most MAX_NUM_SPLICE_SITES and so is the number of acceptors. Therefore, the selection of donors depends on the

existence of “GT” signals within the given region and their relative distances to the upstream border of the intron region. Similarly, the selection of acceptors depends on “AG” signals and their relative distances to the downstream border of the intron region.

The purpose of this step is to provide candidate sites that will be evaluated later. Therefore we can select as many candidates as possible, limited only by the available computing resources. In our current experiments, MAX_NUM_SPLICE_SITES is set to 20, which achieved reasonable performance in both speed and accuracy. The candidate donor sites are identified by at most 20 “GT” signals that are closest to the upstream border (may be at either side of such border). Similarly, acceptors are identified by “AG” signals around the downstream border. Therefore, for each intron region, we select at most 20 candidate donors and 20 candidate acceptors.

4.5 Step 3: Find Best Splice Sites

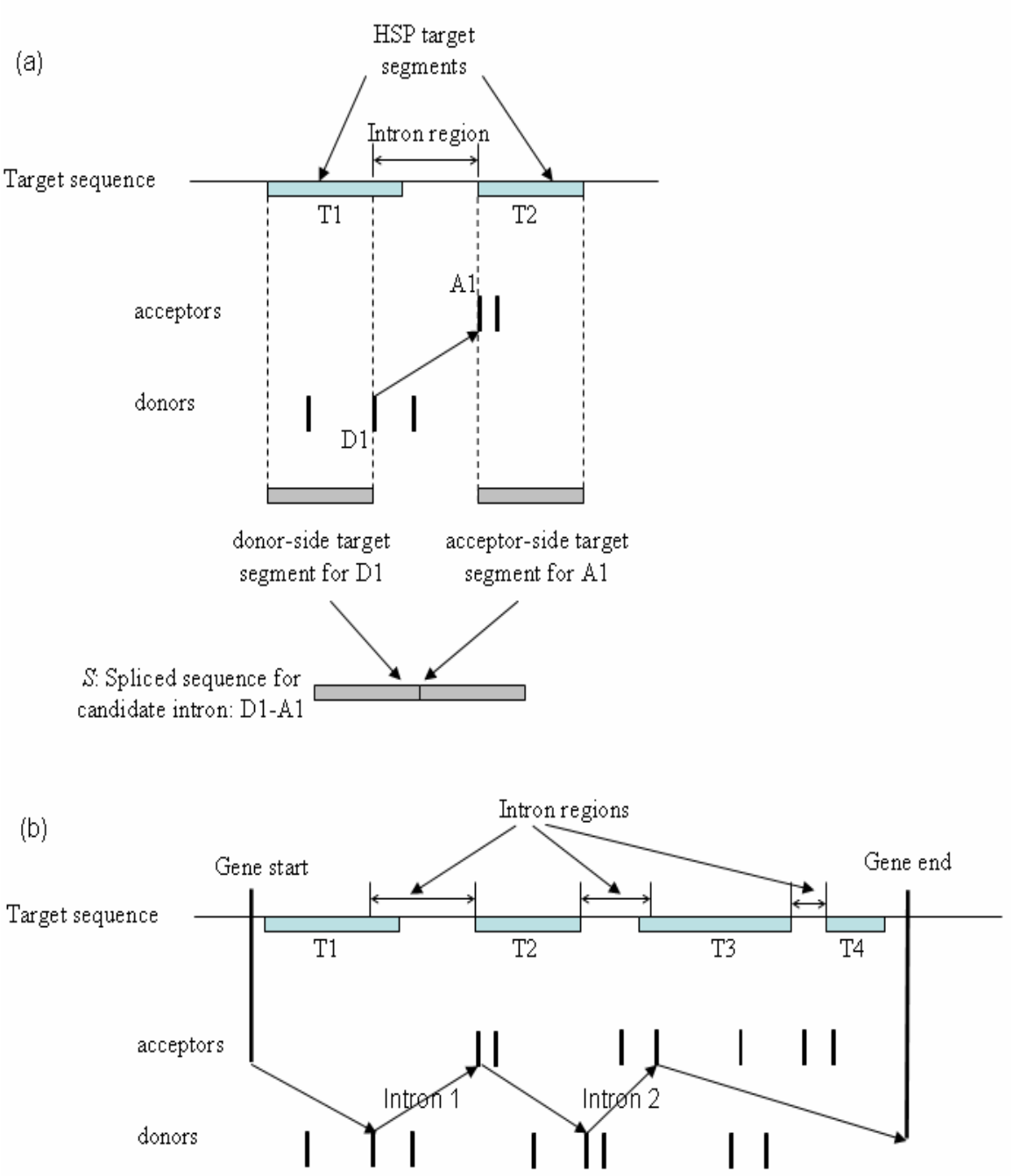
An intron is defined by a pair of donor and acceptor sites. We now determine the best pair of donor and acceptor for each intron region. The best pair of donor and acceptor is selected based on sequence similarity between the spliced sequence (obtained by splicing introns and joining exons) and the query, which is measured by the percentage of identity (PID) in the alignment, called *spliced alignment*. The pair of donor and acceptor that maximizes such alignment PID should be chosen as the best pair. In addition, adjacent exons must be in frame with each other and there should be no in-frame stop codon in the spliced sequence. The detailed procedure is as follows.

Consider an intron region *I*. *I* is associated with its own set of donors {D1, …, Di} and acceptors {A1, …, Aj}, as given by the previous steps. For a pair of donor and acceptor to be considered a *valid pairing*, they must be in-frame with each other and there is no in-frame stop codon in the corresponding spliced sequence *S*, which is formed by connecting the target segment of the HSP at the upstream side of a donor site (called "donor-side target segment") with the target segment of the HSP at the downstream side of an acceptor site (called "acceptor-side target segment"). It is possible that there exists no valid pair of donor and acceptor in an intron region, in which case no intron will be predicted for this region.

Figure 20(a) shows an example of such spliced sequence *S* (for donor D1 and acceptor A1), which is induced from two HSPs. *S* is the concatenation of two subsequences: one from the beginning of the first HSP (donor-side HSP) to the donor site, the other from the acceptor site to the end of the second HSP (acceptor-side HSP). For this spliced sequence, its corresponding query segment *Q* is the part of the query from the beginning of the donor-side HSP query segment to the end of the acceptor-side HSP query segment. The quality of the alignment between *S* and *Q* determines the selection of the best pair of donor and acceptor for the current intron region, i.e. the valid pair that results in the highest PID will be selected.

To address the challenge 2 in Figure 16, where several HSPs correspond to the same exon, we need to consider the possibility that some intron regions produced in Step 1 are in fact unnecessary. Thus we also examine the case in

Figure 20 Finding the Best Pair of Donor and Acceptor



which there is no intron in an intron region. In this case, there is no splicing and the spliced sequence is simply the DNA sequence from the beginning of donor-side HSP target segment to the end of acceptor-side HSP target segment. The alignment between such spliced sequence and the corresponding query

segments is computed in the same way as the spliced alignments for all other donor-acceptor pairs, and its PID is compared with all other PIDs. Therefore, the choice of predicting an intron or not depends only on the quality of spliced alignment. Note that it is possible to have more than one alignment with the maximum PID, in which case the alignments will be further compared on alignment scores. These scores can be computed based on a score matrix that measures the alignment significance between amino acids, in this context, we use the BLOSUM62 substitution matrix [64].

We evaluate the candidate splice sites in each intron region and select the best pair of donor and acceptor for each region one by one. Figure 20(b) illustrates a case with several intron regions. In particular, the last intron region generated no intron at the end, because the spliced alignment in the “no-intron” case gives the highest PID.

4.6 Step 4: Post Processing of Candidate Gene Structure

Once the best splice sites are selected for all intron regions, the initial gene structure is determined. However, there may still be exons missing from the model, especially small ones. This is because sequence similarity search programs often fail to pick up weak alignment, as shown by the challenge 3 in Figure 16(b). To address this problem, we need to “repair” the preliminary gene model by uncovering the missing alignments to maximize the similarity between the predicted gene model and the query. This is done as a post-processing step as follows.

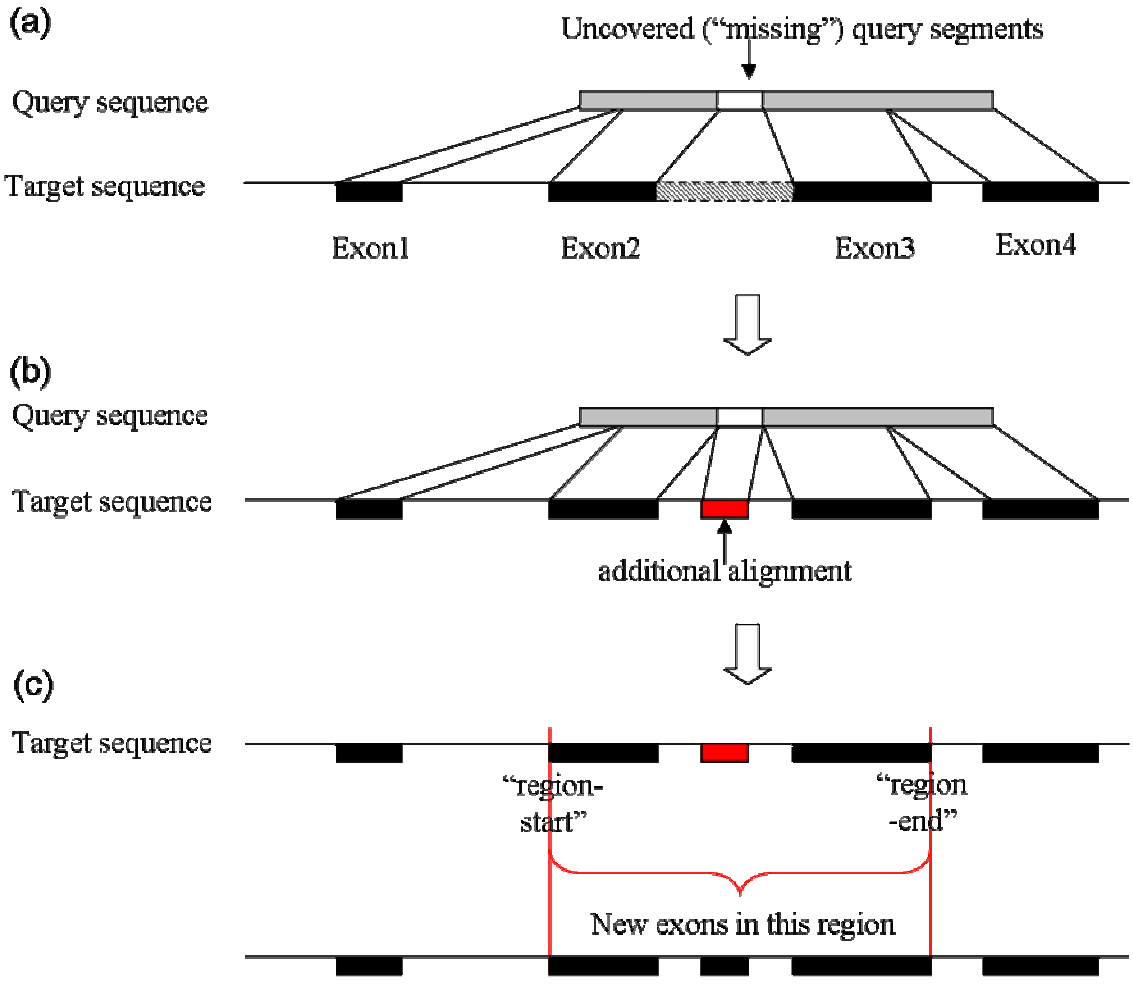
We examine the exons given in the initially predicted gene structure. The query correspondence of each exon is given by the alignment of the spliced sequence as computed in the previous steps. For each pair of adjacent exons, if there is a gap of more than certain length (again, a user-defined threshold) between their corresponding query segments, it signals the possibility that there may be missing exons in the DNA region between these two exons, in which case we will try to repair the initial prediction. Similarly, for the first and last exon, we also check their query correspondences and examine the possible missing query coverage before the first exon or after the last exon. Thus, any query segment that is not covered by initially-predicted exons calls for possible adjustments to such exons.

Figure 21(a) shows an example where a missing piece of the query is identified between Exon2 and Exon3, which is not covered by any HSP in the HSP group. The post-processing step will try to determine whether the two initial exons in this region (Exon2 and Exon3) need to be adjusted.

For each region that is subject to repair (either between two exons, before the first exon, or after the last exon), an optimal local alignment algorithm [121] is used to find the possible missing alignment in the genomic region between two adjacent exons; or in the case of the first or last exon, the genomic region of a certain length before the first exon or after the last exon, respectively. Figure 21(b) shows a new local alignment found between Exon2 and Exon3 that is aligned with the missing query segment in Figure 21(a). Note that this illustrates an ideal case where the missing piece is perfectly covered by the new alignment.

Usually the newly-found local alignment may cover only part of the missing query segment or extend to the query segments that are already covered by initial exons. Nevertheless, the treatment is the same. The new local alignments are used to locate the possible new set of splice sites within this region as follows.

Figure 21 Adjusting the Initial Gene Structure



The adjustment is done locally for each region that needs repair, i.e. the adjustment for one region does not need to be concerned with other regions.

Each region is treated as an independent gene region. Given the initially-predicted exons, if a region between two adjacent exons needs repair (as shown in Figure 21), the start of this gene region is fixed at the start position of its first exon (also called “region-start”), and the end of the region is fixed at the end position of the second exon (also called “region-end”). In other words, the initially-predicted exons are used as references to define the gene region. All local alignments (including the original HSPs and the newly-found alignments) that fall within this region (between “region-start” and “region-end”) are used to find a possibly new set of exons in this region, by following the same three steps as described above (determine intron regions, select candidate splice sites, find the best splice sites). The resulting new set of exons in this region is then compared with the initially-predicted exons in the same region. The set of exons that leads to higher PID (or in case of the same PID, higher alignment score) in its spliced alignment is chosen as the final exons.

4.7 Discussions

The modelling of gene prediction problem in genBlastG is designed to closely follow the biological intuitions on gene expression and sequence similarity. HSPs are used as the basic unit that carry the sequence similarity information. genBlastG is built to find the proper correspondence between HSPs and exons and to establish the gene structure by looking for gene signals at appropriate places. The biological constraints are incorporated into the search for maximized sequence similarity between the prediction and the query.

genBlastG has a few user-adjustable internal parameters, including maximum intron length, minimum intron length and minimum internal exon length, which are all extra restrictions on gene models to ensure their conformity to gene characteristics of the target genome. They are enforced as constraints when determining intron regions and valid pairing of donors and acceptors. They can be easily adjusted for different species. In addition, a few thresholds were used to control the process of genBlastG. A maximum number of splice sites (currently set to 20) is used to control the number of candidate splice sites around the border of each intron region. During the post-processing step, the missing query coverage in the initial gene model is checked and compared with a maximum allowed length of missing query segment. The initial exons are subject to repair only when the missing query coverage is more than the allowed length, signalling the necessity of repair. This value is set to 1 amino acid for the head or trailing exon and 6 amino acids for internal exons in our current experiments. For the DNA regions before the first exon and after the last exon in the initial gene model, a DNA length limit (1000 bp) is used to restrict the length of DNA on which to search for additional local alignments during post-processing. These thresholds are used to fine tune genBlastG and their values are determined pragmatically.

With the integrated design of genBlastA and genBlastG, the gene prediction framework of genBlast is both fast and accurate, which will be shown in the following experiments.

4.8 Performance Evaluation

4.8.1 Experiment Setup

The performance of genBlastG was tested by applying it to predict genes in the genomes of the popular model organism *C. elegans* and its sister species *C. briggsae* (WormBase release WS200 [3, 109]). We also tested genBlastG on the human genome [1].

The whole *C. elegans* genome has 23,973 protein products (including isoforms, where the same gene is spliced in different ways to form different mRNAs and code for different protein products). Using the *C. elegans* protein sequences as queries, we evaluated genBlastG for its performance in finding paralogous genes on the *C. elegans* genome. *C. briggsae* genome is a closely-related species to *C. elegans*, with unconfirmed gene annotations. Using the entire set of *C. elegans* protein sequences as queries and the *C. briggsae* genome as the target, genBlastG is evaluated for its capability in finding orthologous genes. In addition, the human genome is used to test the applicability of genBlastG on more complex organisms with large genomes.

The predictions made by genBlastG are compared with GeneWise [20], since both programs are homology-based methods using protein sequences. GeneWise represents the current state-of-the-art in gene prediction. It was shown to be one of the best performers in the EGASP evaluations, however, it was not included in the subsequent nGASP evaluations. Because the running time of GeneWise depends heavily on the length of DNA sequence to be

examined, we first use genBlastA to narrow gene regions for analysis. Therefore, both genBlastG and GeneWise work only on gene regions reported by genBlastA.

As the entire genome database is stored as one large file, in order to take advantage of the reduced genomic regions reported by genBlastA, GeneWise requires an extra step of extracting the genBlastA region and use this much smaller sequence as its input for each query. Note that the time of extracting the genomic regions for GeneWise is not included in GeneWise runtime. In genBlastG implementation, in order to obtain the genomic sequence of desired gene regions as reported by genBlastA, we build a small index upon the initial scan of the genome database. This index contains a record for each sequence in the genome database, in the form of <Seq, Pos>, where "Pos" is the start position in the database for sequence "Seq". Then, for each gene region reported by genBlastA, the index is looked up to find the start position of the desired sequence, so that the database scan is directly started from that position without scanning the entire database. The time spent on building the index is included in genBlastG runtime.

For *C. elegans* and *C. briggsae* genomes, we also compared genBlastG results with the existing gene models as previously predicted by the nGASP project [38], which were produced by the overall best performer, JIGSAW. The nGASP results are obtained from the WormBase ftp site [4].

All experiments were run on a computer with Intel Xeon E5430 2.66GHz CPU and 16G memory. The genBlast code was written in C++. WU-BLAST [80]

(without the grouping option) is used as the sequence similarity search tool to produce HSPs.

4.8.2 Results on *C. elegans* Genome

The *C. elegans* genome has been actively curated by the *C. elegans* research community and the WormBase curators [31, 33, 66]. In the WormBase WS200 release, 85.2% of all *C. elegans* genes are either confirmed or partially confirmed. Thus the *C. elegans* genome annotation is well suited for evaluating the accuracy of a new gene prediction program.

4.8.2.1 Speed Comparison

We compared the running time of genBlastG with GeneWise. For both methods, WU-BLAST and genBlastA were utilized to determine the gene regions, from which the top ranked region is used as input to both genBlastG and GeneWise. Therefore, in this context, genBlastG and GeneWise can be both regarded as the last step in the gene prediction framework as shown in Figure 3. The running time reported here refers only to the last step. The running time of genBlastA is reported separately.

We used all curated genes on Chromosome I of *C. elegans* as queries and run genBlastG and GeneWise to find genes in the entire *C. elegans* genome. For query genes with multiple isoforms, only the longest isoform is used. Thus the query set consists of totally 2,876 genes. In order to examine the effect of query length on the running time of different algorithms, we divided all Chromosome I genes into five categories depending on their lengths. The

number of genes that are tested in this experiment and their length distributions are shown in Table 3.

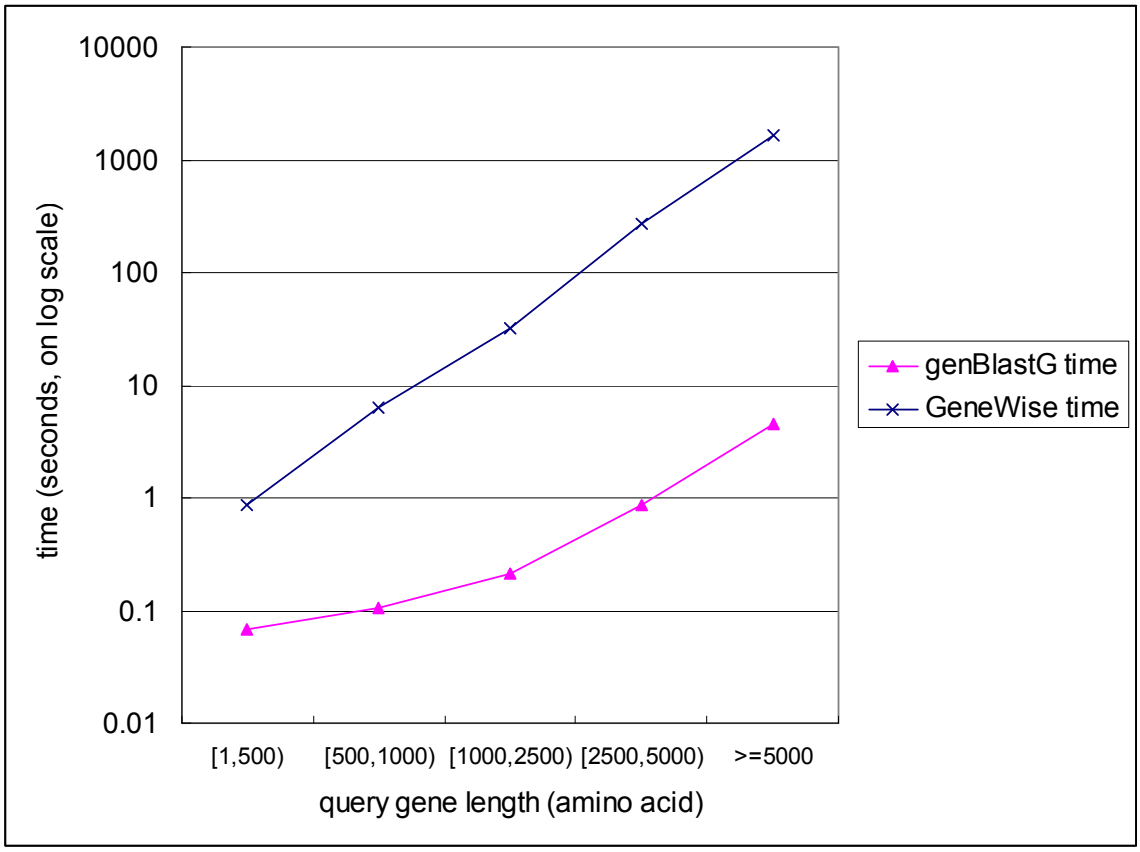
Table 3 Length Distribution of *C. elegans* Chromosome I genes

	Gene categories according to query length				
Query protein length	[1, 500)	[500, 1000)	[1000, 2500)	[2500, 5000)	Over 5000
# of genes	2047	644	164	19	2

Figure 22 shows the average running time of genBlastG and GeneWise on *C. elegans* with all Chromosome I genes as queries. Note that it is on logarithmic scale. The running time for each category is the average running time for all genes in that category. Figure 22 shows that genBlastG is faster than GeneWise, sometimes hundreds of times faster, especially for longer genes. It took 5 hours and 25 minutes for GeneWise to predict gene models for all Chromosome I query genes. In contrast, the total running time for genBlastG on the same gene set is just over 4 minutes. genBlastG took about 18 minutes to predict gene models for the entire *C. elegans* genome. It is estimated that GeneWise would take more than 1 day to finish the same task.

The total running time of genBlastA that is used as the pre-processing tool in this experiment (for Chromosome I genes) was just 5 minutes. To process all genes in the *C. elegans* genome, the total time spent on genBlastA alone was 1 hour and 17 minutes, given the large number of HSPs (more than 17.5 million HSPs) obtained by WU-BLAST.

Figure 22 Running Time on *C. elegans* genome with Chromosome I genes as queries



4.8.2.2 Accuracy Comparison

For accuracy comparison, we used the curated gene models in WormBase (release WS200) as the true models to evaluate all results. As discussed in Section 2.4, the accuracy of gene finders is commonly measured by specificity and sensitivity. Specificity (*Sp*) is the percentage of predictions that are correct; sensitivity (*Sn*) is the percentage of actual coding genes / exons / nucleotides that are predicted as such.

Because each isoform is used as an independent query to search for genes, both genBlastG and GeneWise produce a gene model for each isoform. Therefore, we can compare them at isoform, exon and base levels. On the other

hand, the pre-existing nGASP annotation has gene predictions for the entire genome, but with only one model per gene, thus it is compared with genBlastG at gene, exon and base levels on the entire *C. elegans* genome. As described in Section 2.4, a gene is considered correct so long as at least one of its isoforms is predicted correctly, i.e. the nGASP gene model is considered correct if it matches any of the WormBase isoforms.

Table 4 shows the accuracy results of genBlastG and GeneWise on Chromosome I genes. genBlastG outperforms GeneWise, especially at the isoform level. Table 5 shows the comparison between genBlastG and nGASP predictions for the entire *C. elegans* genome, where genBlastG exhibited much better accuracy. For completeness, the isoform-level accuracy of genBlastG is also shown (nGASP does not predict isoforms). These tables show that the performance of genBlastG is consistent on both Chromosome I genes and the entire genome. The specificity and sensitivity of genBlastG are both well above 90% at all levels.

Note that the accuracy of nGASP predictions differs from what was reported by the original nGASP project, because the gene regions that are used for testing are different. Our evaluation is based on the entire *C. elegans* genome, while nGASP evaluation used 10% of the genome. The WormBase models used for evaluation are also from different WormBase releases. In addition, the protein sequences used in our evaluation are more accurate compared with the data supplied by nGASP. In general, our experiments show that previous nGASP models are much worse than genBlastG models, especially at the gene level.

Table 4 Accuracy Comparison on *C. elegans* Chromosome I genes (genBlastG vs. GeneWise)

	Isoform level		Exon level		Base level	
	Sp. (%)	Sn. (%)	Sp. (%)	Sn. (%)	Sp. (%)	Sn. (%)
genBlastG	95.00	95.00	99.07	98.62	99.69	99.69
GeneWise	89.02	88.66	98.02	96.44	99.89	99.64

Table 5 Accuracy Comparison on entire *C. elegans* genome (genBlastG vs. nGASP)

	Isoform level		Gene level		Exon level		Base level	
	Sp. (%)	Sn. (%)	Sp. (%)	Sn. (%)	Sp. (%)	Sn. (%)	Sp. (%)	Sn. (%)
genBlastG	94.26	94.24	95.96	95.96	98.64	98.17	99.62	99.62
nGASP	n/a	n/a	72.68	58.97	92.17	71.43	97.03	94.02

4.8.3 Results on *C. briggsae* Genome

C. briggsae genome is a sister genome of *C. elegans* and has been widely used as a comparative platform for understanding the genome of *C. elegans*. These two species split approximately 80-120 million years ago [40, 127], around the same time as the human/mouse split [95]. Early comparative analysis between genes sets of *C. elegans* and *C. briggsae* revealed about 2,000 different genes [127].

Compared with the extensively annotated *C. elegans* genome, essentially all *C. briggsae* genes are still hypothetical and have not been validated since its publication. Recently, the *C. briggsae* genome has been re-annotated by the nGASP project as an attempt to improve its gene set [38]. Despite these efforts,

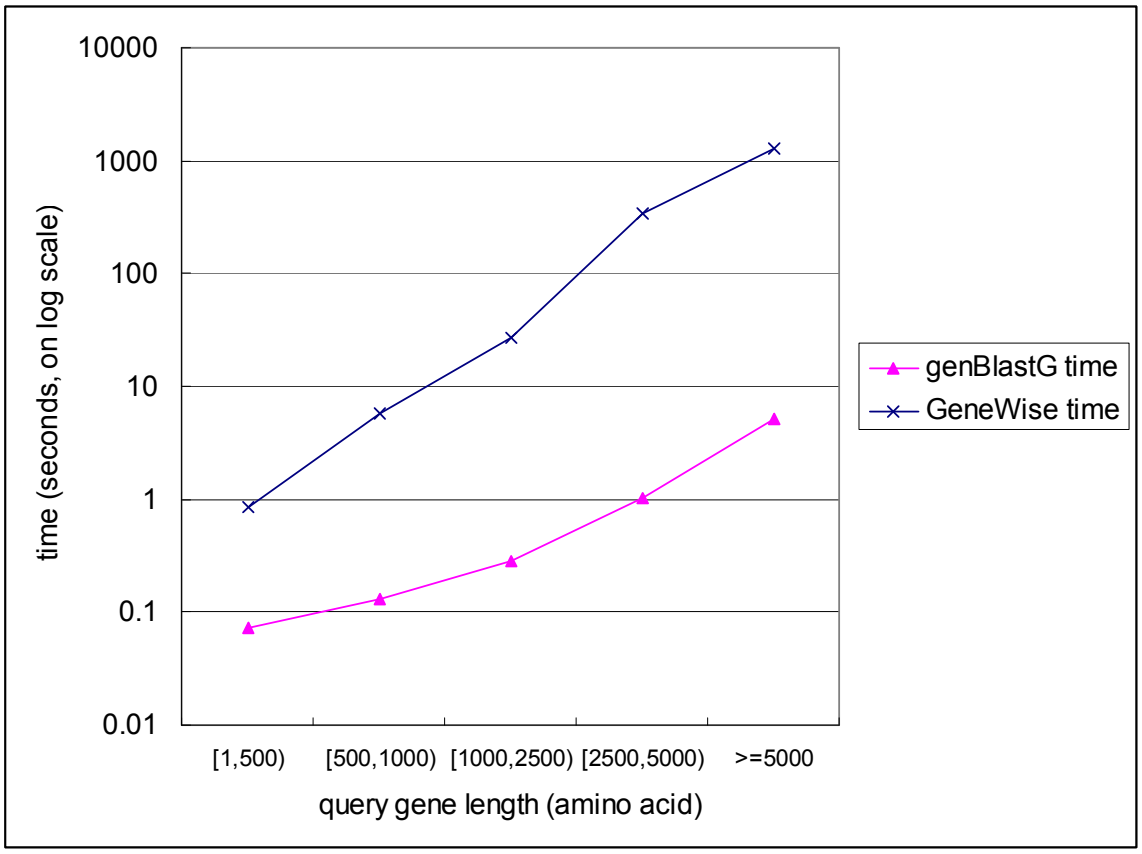
many gene models are obviously defective when compared with orthologous genes in *C. elegans*. In this thesis, genBlastG has been used to revise *C. briggsae* gene models based on their homology to *C. elegans* genes [72].

genBlastG was applied to predict *C. briggsae* genes using the same set of *C. elegans* proteins as queries (23,973 proteins in WS200 release [3], including isoforms) and the *C. briggsae* WS200 genomic sequences as the target genome. The predictions made by genBlastG are compared with GeneWise, WormBase and nGASP predictions.

4.8.3.1 Speed Comparison

Similar to the *C. elegans* experiments, for running time comparison between genBlastG and GeneWise, WU-BLAST and genBlastA were used as pre-processing tools and all *C. elegans* genes on Chromosome I were used as queries. Figure 23 shows the average running time of genBlastG and GeneWise for the five categories as in Table 3. It shows similar trends to Figure 22. GeneWise took 5 hours and 13 minutes to finish this experiment. In contrast, genBlastG took 5 minutes. For predicting all orthologous genes on the entire *C. briggsae* genome, genBlastG took 18 minutes.

Figure 23 Running Time on *C. briggsae* genome with *C. elegans* Chromosome I genes as queries



4.8.3.2 Comparison of Query Alignment PID

For the *C. briggsae* genome, because its current WormBase annotations are mostly based on other gene prediction programs and not validated by experimental evidence, there is no ultimate truth against which to compare gene models. We evaluated all predictions (genBlastG, GeneWise, nGASP, WormBase) based on the alignment identity (PID, percentage of identity in the alignment) between the predicted gene product and the query protein, which will be referred to as the *query alignment PID*. The alignment was done after the gene models are predicted and by using the optimal local alignment algorithm [121].

Note that it is not clear how the existing gene models annotated by WormBase and nGASP correspond to the new predictions made by genBlastG. Therefore, we compared the genBlastG model with these existing models only if they are in the same genomic region. If the WormBase/nGASP models overlap with more than one genBlastG model, then the comparison is made with the model that gives WormBase/nGASP the highest PID.

Figure 24 shows the PID comparisons between genBlastG, WormBase and nGASP. The query alignment PID of genBlastG models are plotted against the PID of corresponding WormBase or nGASP models. Each point represents a gene with PID from genBlastG and WormBase/nGASP. The X-axis represents PID from genBlastG; Y-axis represents PID from WormBase (Figure 24a) or nGASP (Figure 24b). The area below the diagonal line in the figure shows the cases (points) where genBlastG provides better PID than WormBase or nGASP predictions. Most of the cases fall in this area. Compared with WormBase predictions, about 25% of genBlastG predictions show much higher PID (PID differs by more than 10%) and only 2% of genBlastG predictions show PID of more than 10% lower. The overall comparison between genBlastG and nGASP is almost the same as that between genBlastG and WormBase, as shown in Figure 24. The gene models produced by genBlastG have the average query alignment PID of 72.7%, while the average PID of WormBase and nGASP models is 64.8% and 65.4%, respectively. genBlastG performs considerably better.

Figure 25 shows the PID comparison between genBlastG and GeneWise, based on *C. elegans* Chromosome I query genes. Models produced by

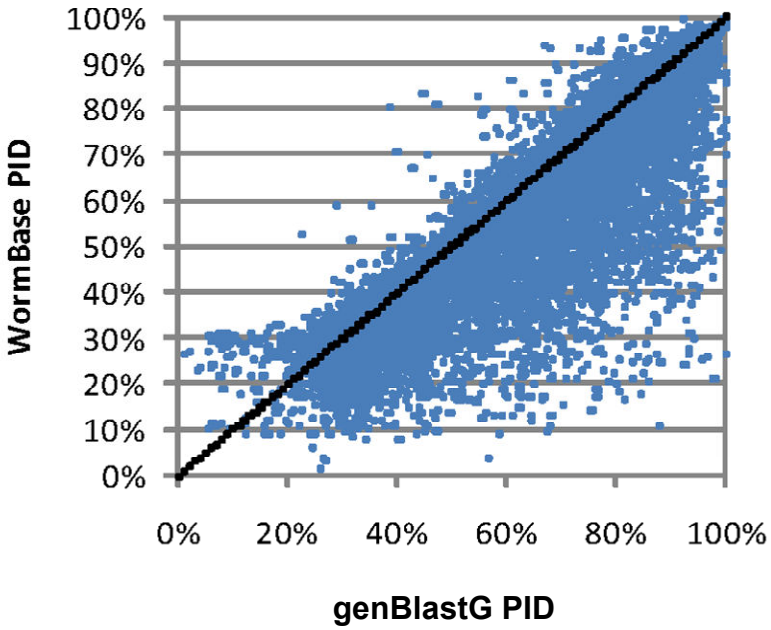
genBlastG show slightly higher PID than models from GeneWise. The average PID of genBlastG models is 73.8%, whereas the average PID of GeneWise models is 72.4%. About 8% of genBlastG predictions produced much higher PIDs than those of GeneWise (with PIDs of more than 10% difference) and 1.5% of genBlastG predictions have PIDs of more than 10% worse.

Note that GeneWise does not always produce a complete gene model, i.e. it sometimes gives a partial gene structure that does not have the proper stop codon where a gene should end. In contrast, genBlast always predicts a complete gene model that ends at a stop codon. In fact, in this experiment, 35% of GeneWise gene models do not end at a stop codon. These cases (points) are shown in red color in Figure 25. Such situation occurs much more frequently in cases where GeneWise showed higher PID than genBlastG. Among the GeneWise models whose PIDs are more than 10% higher than those of genBlastG, 68% of them do not have proper stop codons at the ends.

genBlast outperforms all other gene prediction tools in terms of alignment PIDs of the predicted model against the query, indicating that genBlastG models are more similar to their corresponding query genes. Although expressing the overall accuracy in terms of query alignment PID may be biased since not all orthologous genes in *C. briggsae* genome are highly similar to their *C. elegans* counterparts, we believe it is an appropriate choice as it shows an overall picture and insights into the general quality of gene models.

Figure 24 Comparisons of Query Alignment PIDs between genBlastG, WormBase, nGASP

(a) genBlastG vs WormBase



(b) genBlastG vs nGASP

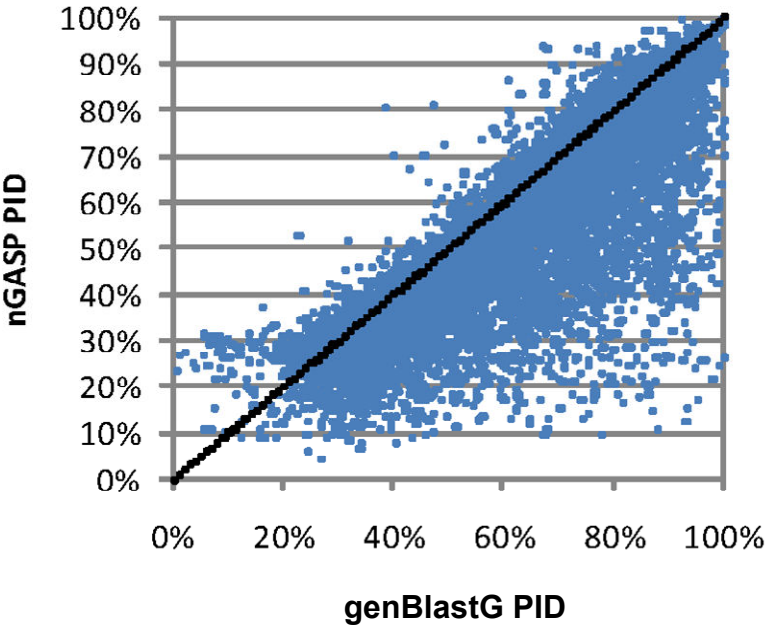
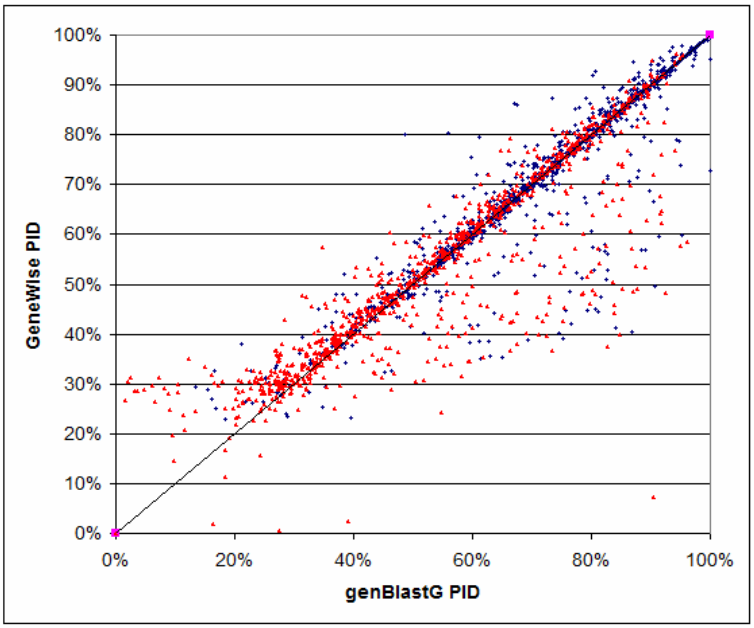


Figure 25 Comparisons of Query Alignment PID between genBlastG and GeneWise



4.8.4 Re-annotating the *C. briggsae* Genome

The performance evaluation based solely on query alignment PID is not conclusive with regards to the quality of the predicted gene models. Therefore, we also closely examined the new models predicted by genBlastG and compared them with current WormBase and nGASP models manually [72]. We focus on the genBlastG models that show above 60% of query alignment PID (11,480 models) and examined them in detail. Biological experiments were carried out in the lab in order to validate some of the predicted gene models. Particularly, PCR verification [96] was performed to confirm the predictions using full-length cDNAs.

The comparison with the current WormBase and nGASP gene models revealed four types of revisions that are made by genBlastG (Figure 26) [72]:

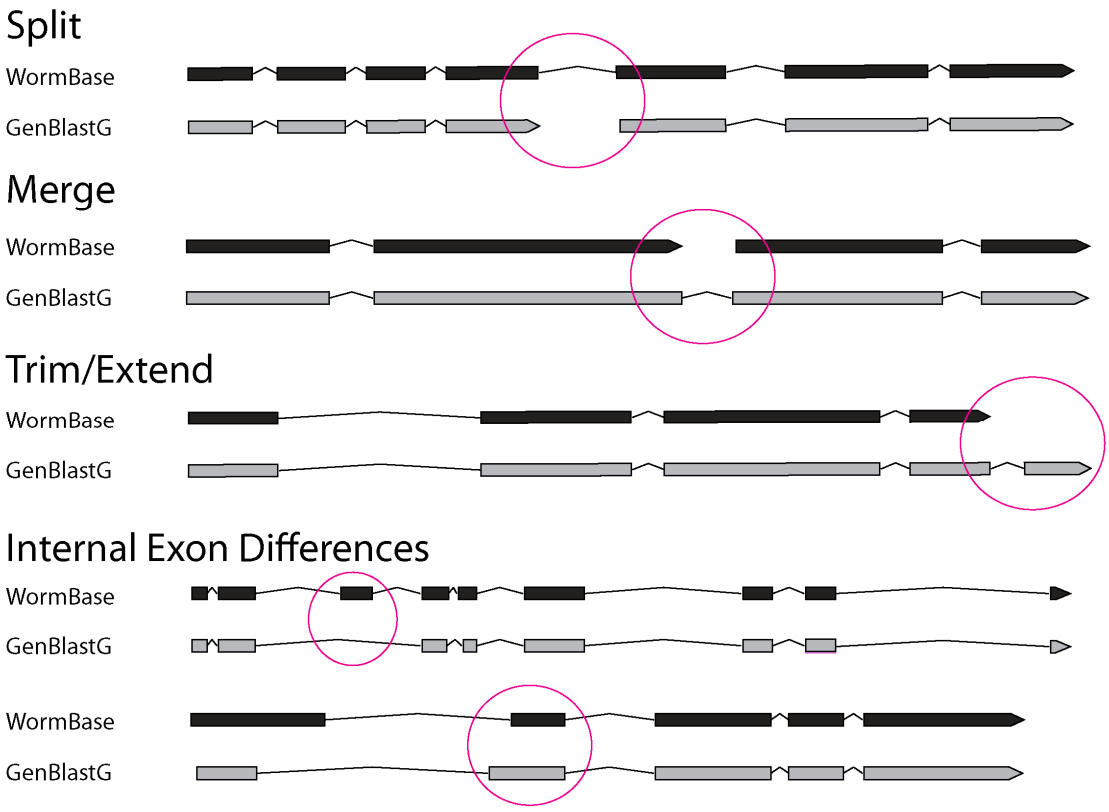
(1) gene model split: A current WormBase gene model is split into two or more gene models by genBlastG;

(2) gene model merge: Two or more current WormBase gene models are merged to form a single genBlastG gene model;

(3) gene model trimming/extension: The ends of a gene are trimmed or extended by genBlastG; and

(4) internal exon alteration: Internal exons are added/removed/revised by genBlastG.

Figure 26 Gene Model Differences



Additionally, genBlastG also discovered many novel gene models that show high similarity to known *C. elegans* genes. These are the genes located at the genomic regions where there is no gene models in the current WormBase annotation.

The summary of cases in each category is shown in Table 6. Each of these categories will be demonstrated in the following sections.

Table 6 Summary of 5 categories of gene revisions made by genBlastG (*C. briggsae* genome)

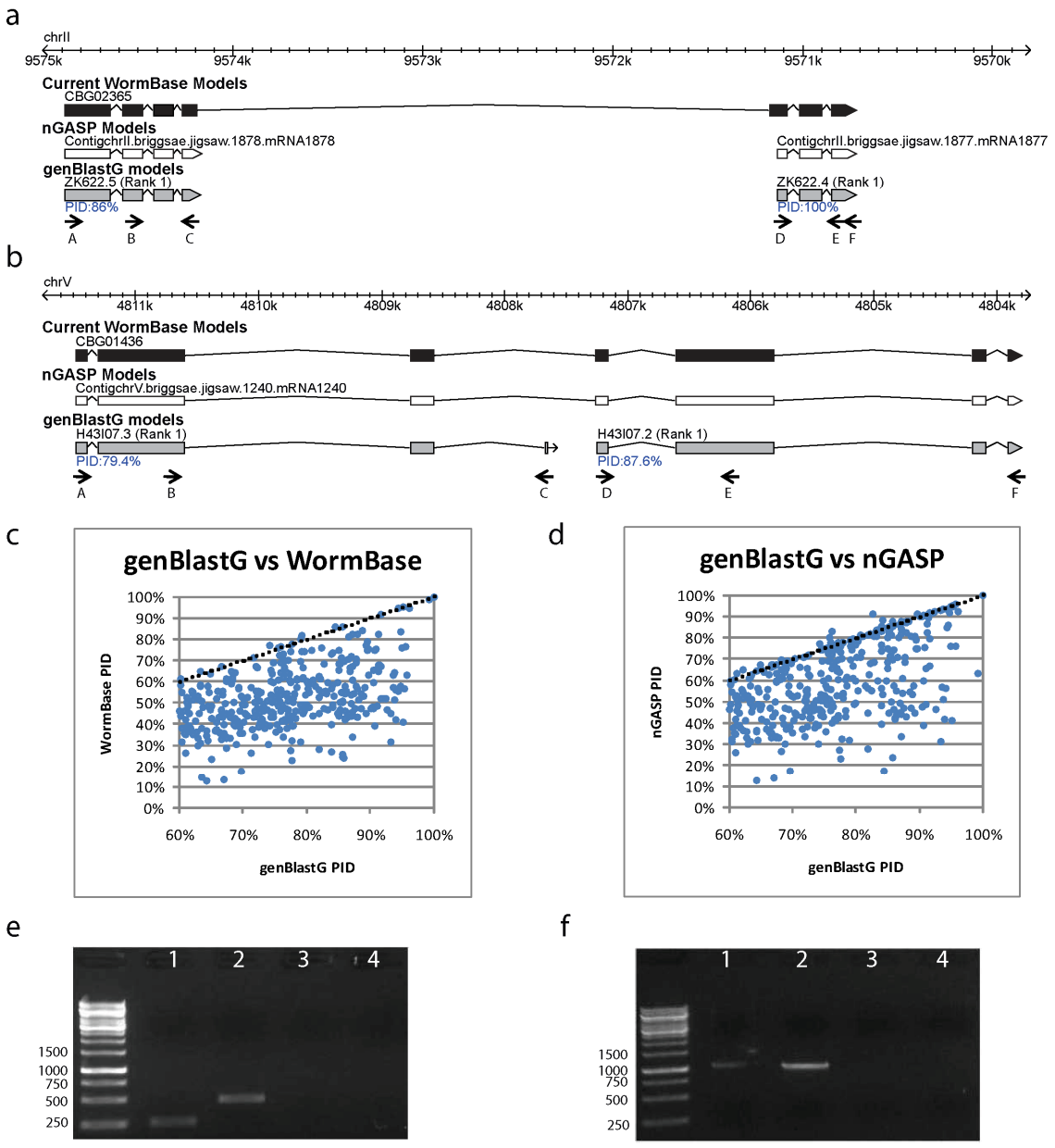
	Number of Cases	Number of cases with PID improvement by at least 10%	Avg. genBlastG PID (%)	Avg. Worm-Base PID (%)	Avg. nGASP PID (%)
Splits	398	340	77±10	52±14	60±18
Merges	239	179	79±10	58±15	60±16
Trims/Extends	3825	1032	78±10	70±15	69±16
Internal exon differences	4594	692	79±10	73±14	72±14
Novel	85		72±9		

4.8.4.1 Gene Model Split

Upon close examination, we found that many current WormBase models are in fact the false merge of separate adjacent gene models, and genBlastG is able to correctly identify them as individual models, as shown in Figure 27.

According to the homology to their corresponding *C. elegans* gene models, these gene models should be split to two or more gene models. Altogether, we

Figure 27 Gene Model Split Cases



have found 398 such cases. Some (158) of these cases have been fixed by the nGASP project (e.g. Figure 27a), while many (240) have not (e.g. Figure 27b). After the split, the new gene models predicted by genBlastG show significantly improved query alignment PIDs, as shown in Figure 27c and Figure 27d. The average PID of genBlastG models in these split cases is 77%, compared with

that of WormBase and nGASP models at 52% and 60%, respectively. The PCR verification for the two examples shows that individual gene models as predicted by genBlastG can be amplified, but the whole length gene model suggested by WormBase cannot be amplified (Figure 27e, f). These results suggest that the genBlastG predictions are true.

4.8.4.2 Gene Model Merge

A number of current WormBase gene models have been erroneously split into two or more unrelated gene models (Figure 28a, b). Based on the homology to their corresponding genes in *C. elegans* genome, these gene models should be merged to form a single gene model as predicted by genBlastG. We have found 239 merge cases in the current WormBase models. Again, some of these cases (40) have been fixed by the nGASP project, but many (199) have not. Nearly all merge cases show improved PID (Figure 28c, d). The PID of genBlastG models on average is 79%, whereas the average PIDs for WormBase and nGASP models are 58% and 60%, respectively. Verification by PCR indicates that many merges are real since the full length or the junction can be amplified from cDNA library (Figure 28e, f).

Many merge cases we observed are due to isoform predictions, where the same gene is alternatively spliced to code for different proteins. For example, an alternative splicing of a gene can skip an exon or retain an intron during splicing. In fact, alternative isoform prediction remains a challenge in gene prediction and many gene finders do not handle isoform prediction. Because genBlastG uses protein isoforms directly as queries, it is able to pick up the corresponding

isoform gene models and improve gene annotation quality. In the experiments on *C. briggsae* genome, predictions made by genBlast revealed that some gene models in the WormBase annotation may represent just one of the many isoforms and a nearby smaller (single exon) gene model is actually part of the same gene in another isoform. Figure 29 shows an example of how three small gene models (CBG26024, CBG20187, and CBG20185) are part of a larger model (CBG20190) in separate isoforms. This example shows that having isoform information will improve gene model prediction dramatically, especially for single exon genes.

Figure 28 Gene Model Merge Cases

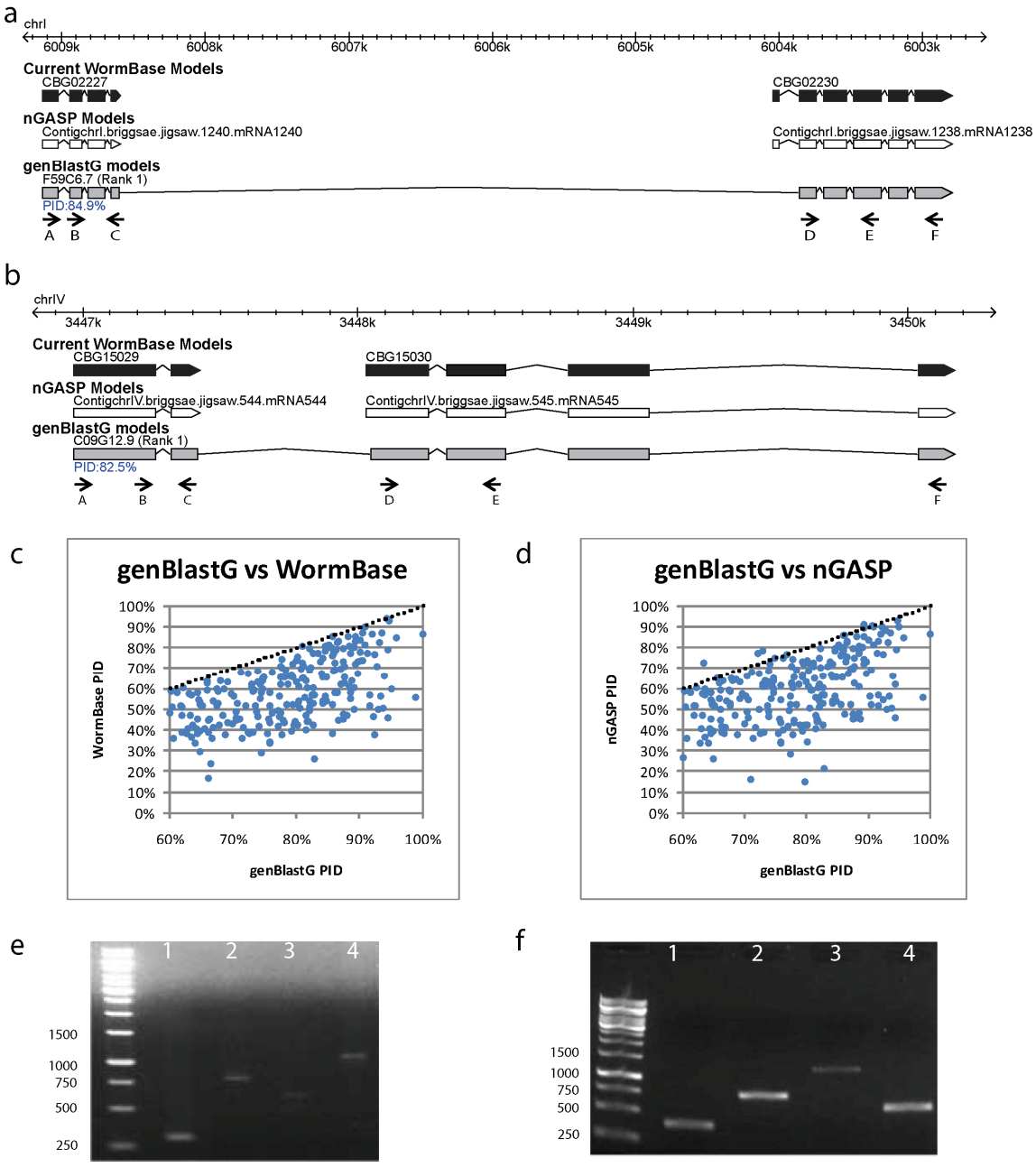
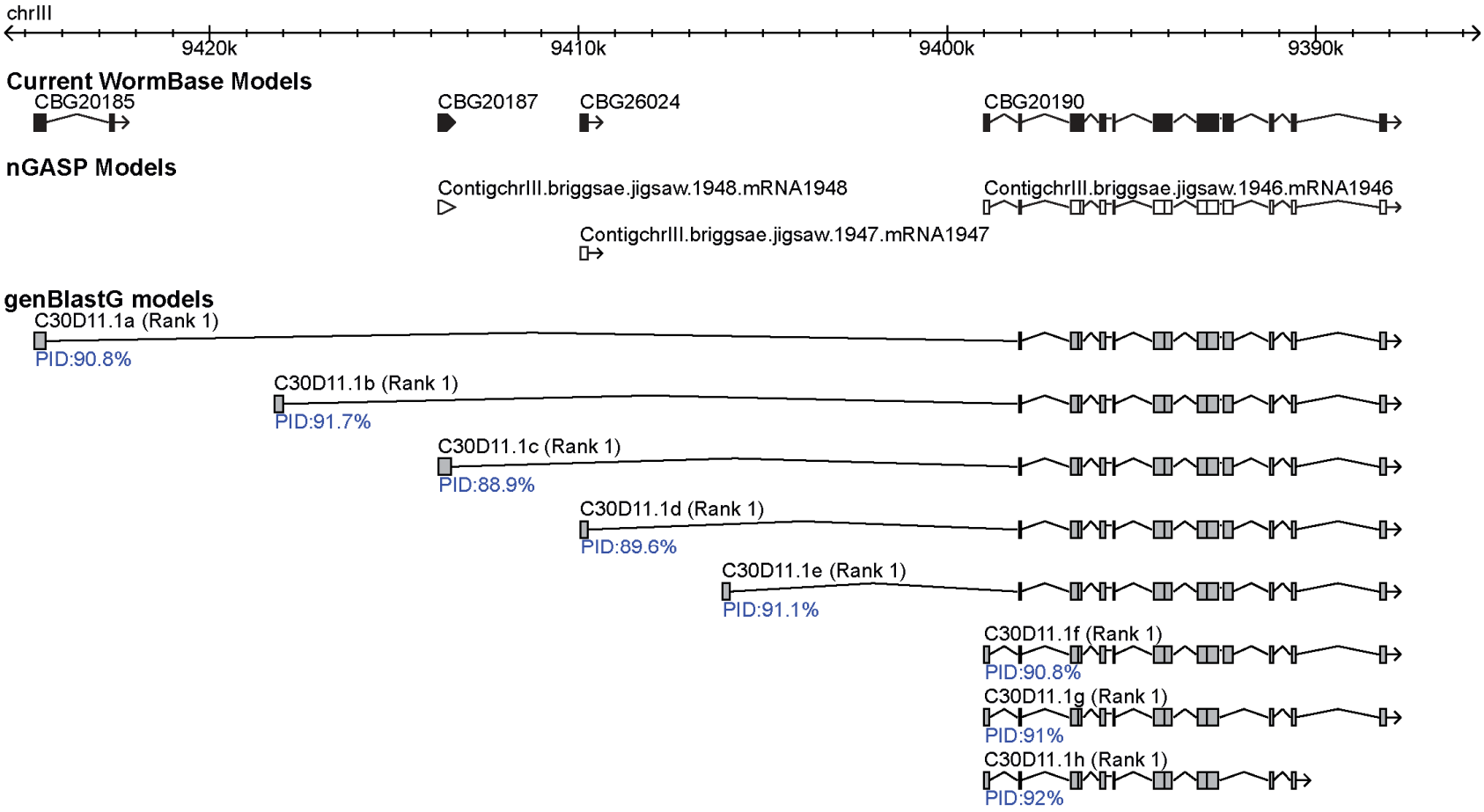


Figure 29 New Gene Models due to Isoforms

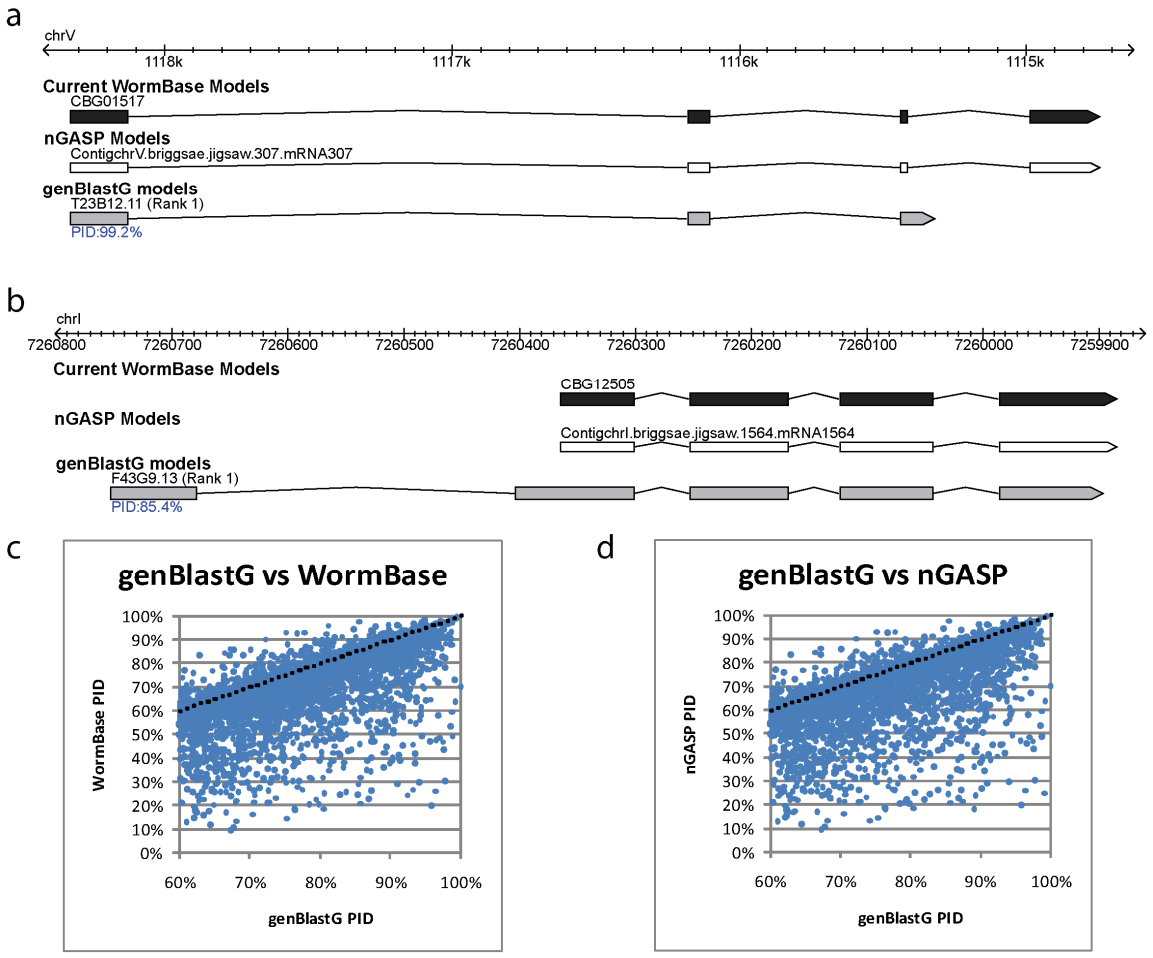


4.8.4.3 Gene Model Trimming/Extension

The first and last exons of many current WormBase gene models are defective, according to the homology to their corresponding *C. elegans* genes. In these cases, the gene models need to be trimmed or extended at either end. We found 3,825 cases, for which the average query alignment PID produced by genBlastG and WormBase is 78% and 70%, respectively. On the other hand, within these cases, 3,544 nGASP models show different start and end positions. The average PID for these nGASP models is 69%.

An example of gene trimming in shown in Figure 30a, and another example of gene extension is shown in Figure 30b. In both cases, genBlastG predictions improve the current WormBase models significantly. Overall, genBlastG models are found to be more similar to their *C. elegans* counterparts, as opposed to the WormBase models and nGASP models (Figure 30c, d). In some cases, changing the start or the end positions leads to a change of reading frame, creating an entirely different protein sequence. For example, in Figure 30b, the exon frames for the genBlastG model of F43G9.13 are 2, 3, 3, 1, and 3, whereas the exon frames for the corresponding WormBase model CBG12505 are 1, 1, 2, and 1. The change in reading frame produces an entirely different protein sequence that dramatically improved similarity to the *C. elegans* query protein.

Figure 30 Gene Model Trimming/Extension Cases



4.8.4.4 Internal Exon Alteration

These are the cases where the gene start and end are the same but the exons in between are either in different length, missing, or have extra exon(s). We found 4,594 cases where genBlastG has different internal exons from current WormBase *C. briggsae* models, for which the average query alignment PID of genBlastG and WormBase is 79% and 74%, respectively. Within these cases, 4,489 nGASP models contain differences in internal exons with an average PID of 73%.

Figure 31a shows an example where an extra exon is predicted by genBlastG, which improved its alignment PID with the *C. elegans* query. Overall, most genBlastG models are more similar to their *C. elegans* queries, as shown in Figure 31b and Figure 31c. Figure 31d shows a band of about 400 base pairs long, which represents the expected size of the new gene model that is revised by genBlastG. Similar PCR verifications were done for two other cases (CBG16922 and CBG06025), and both were found to support the gene models predicted by genBlastG.

4.8.4.5 Novel Genes

The experiments revealed 85 genBlastG models that do not overlap with current WormBase gene models (Figure 32), 9 of which were found independently by nGASP (Figure 32a). All of these genBlastG models show more than 60% of query alignment PID. The shortest model is 105 base pairs long and the longest is 1407 base pairs long. On average, these genBlastG models show 72% PID with average length of 375 base pairs. Figure 32 shows four such examples (Figure 32a, b, c, d), each of which is validated using PCR amplification from a cDNA library (Figure 33). These results indicate that there are still many gene models missing from previous annotations and the models predicted by genBlastG are likely to be real.

4.8.4.6 Summary

genBlastG was applied to the entire *C. briggsae* genome. Compared with current WormBase annotations, 1,805 genBlastG models showed significant

improvement as indicated by similarity to their orthologous genes in the *C. elegans* genome. The query alignment PIDs of these gene models are at least 10% higher than those of WormBase models. The comparison between genBlastG models and WormBase models revealed five categories of revisions: split, merged, trimmed/extended, different internal exons, and gene models that are missed entirely by WormBase. Experimental validation supports many gene models revised by genBlastG, demonstrating that many WormBase gene models are defective. These 1,805 genBlastG models can be used to replace the corresponding gene models in current WormBase annotations. Many additional gene models in the *C. briggsae* genome could be revised based on genBlastG predictions after more careful examination.

Figure 31 Internal Exon Alteration Cases

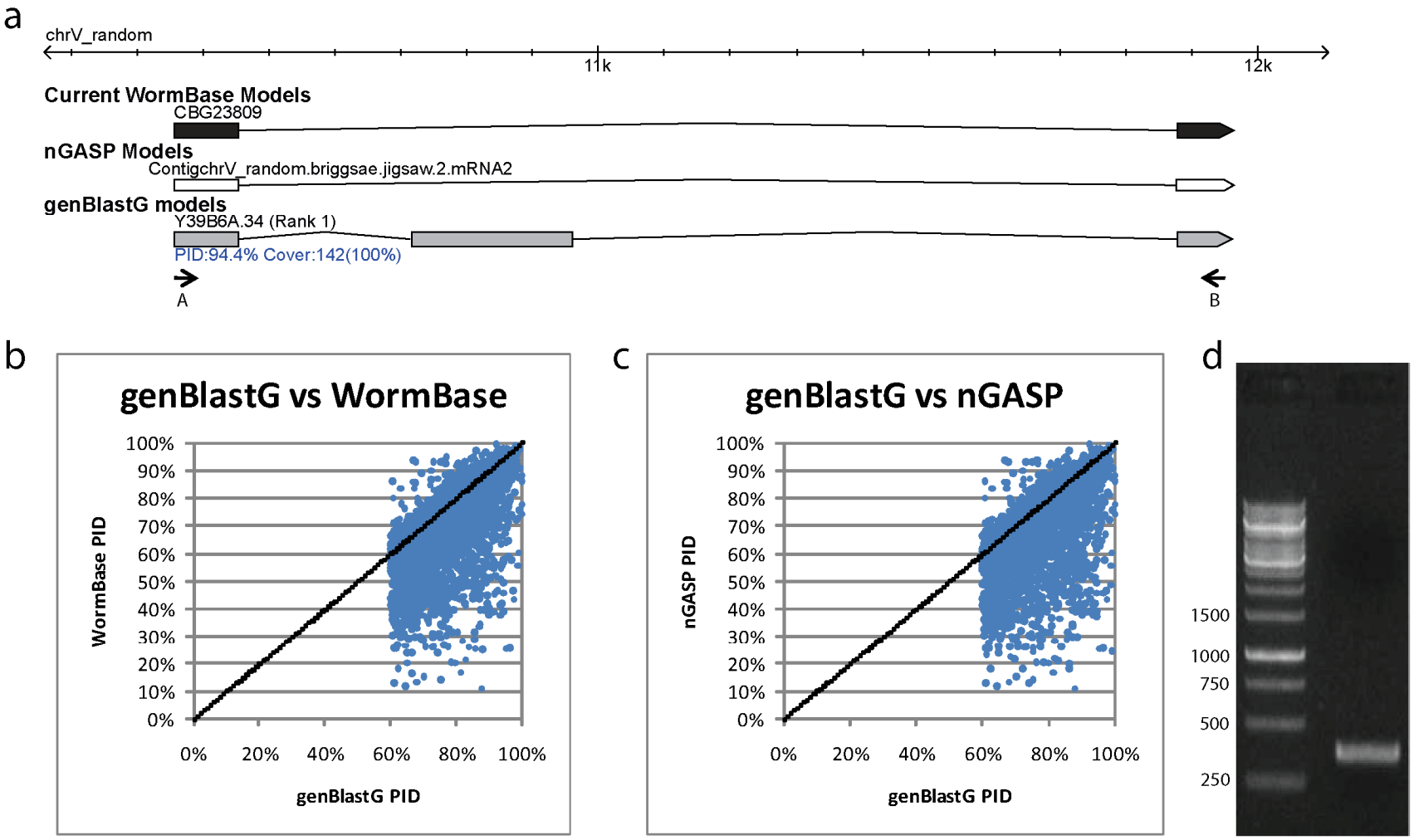
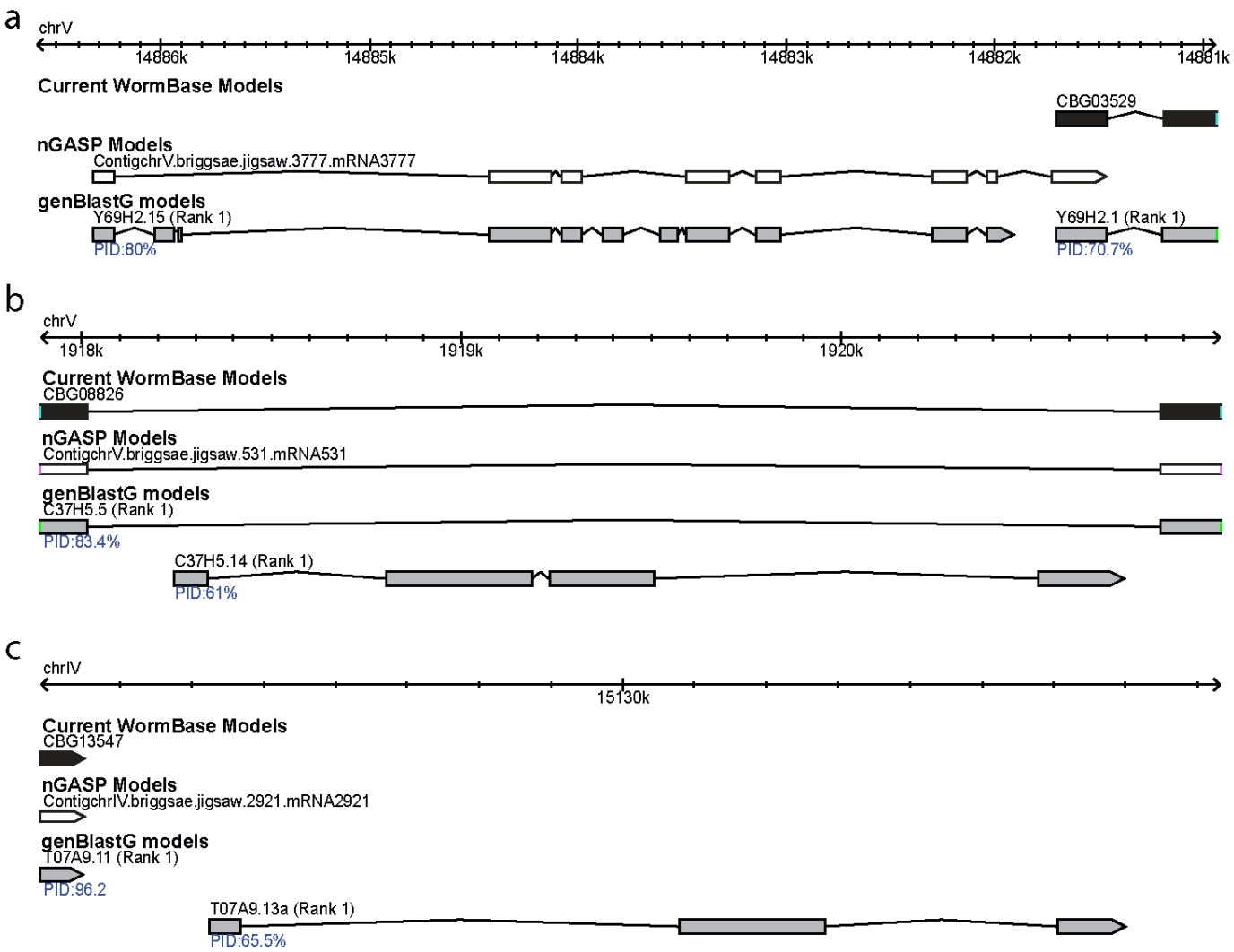


Figure 32 Novel Genes



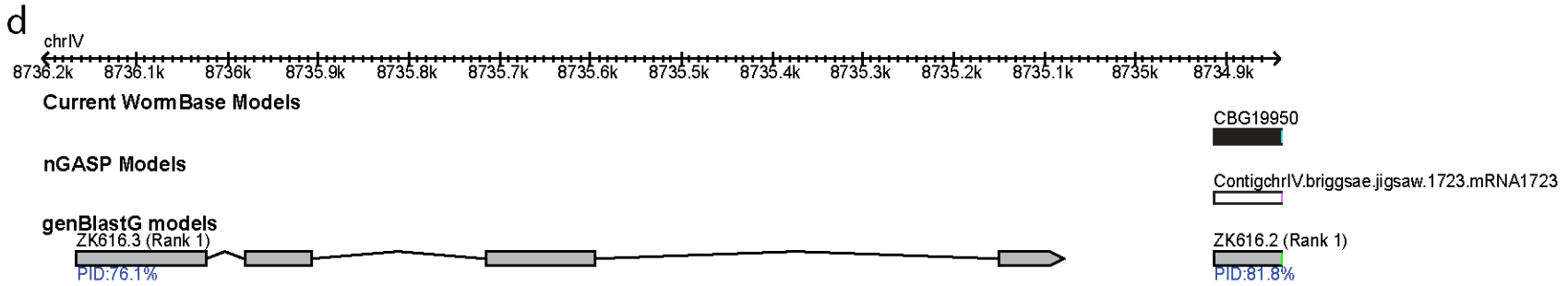
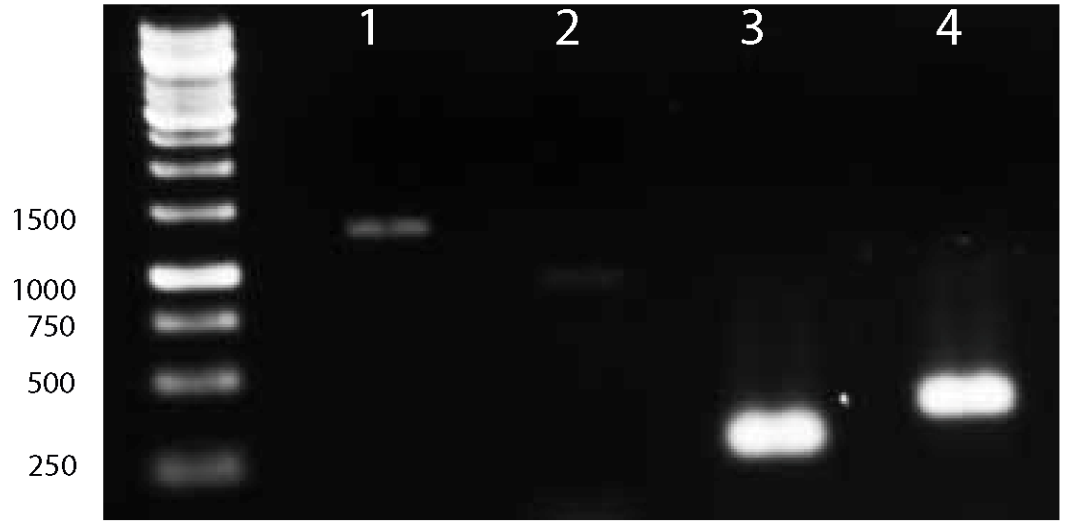


Figure 33 PCR Verification of Novel Genes



4.8.5 Results on the human genome

Both *C. elegans* and *C. briggsae* are relatively simple organisms that have small genomes, with about 100M base pairs. In order to test genBlastG on more complex organisms that have larger genomes, we evaluated its performance on the human genome that has over 3G base pairs. Researches on the human genome are generally of higher interest because of their direct impact on health-related studies, The performance of a gene prediction program on the human genome likely indicates the program's applicability and impact among all gene prediction tools.

We obtained all human peptide sequences from ENSEMBL database [2]. To compare genBlastG and GeneWise, we randomly selected 75 human peptide sequences as queries and run both genBlastG and GeneWise to find genes in the entire human genome. We also tested genBlastG on the human genome with all human peptide sequences as queries.

4.8.5.1 Speed Comparison

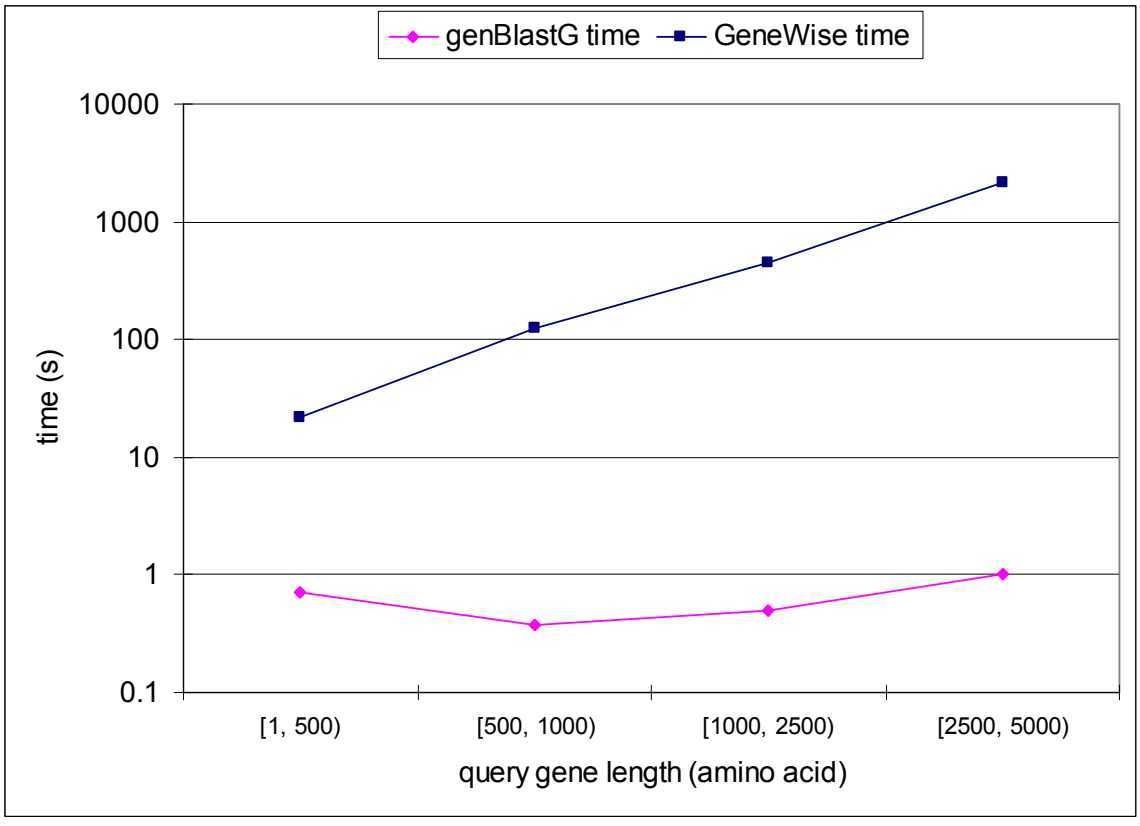
The 75 test genes were divided into four categories based on their lengths. Table 7 shows their length distributions. Figure 34 shows the average runtime of genBlastG and GeneWise in each category. It shows similar trends as in Figure 22 and Figure 23. GeneWise becomes slow for long query sequences, for which genBlastG is hundreds or even thousands of times faster than GeneWise. In general, the runtime of GeneWise on human genome is significantly slower than on smaller genomes such as *C. elegans* and *C. briggsae*. But genBlastG remains

to be fast on such large-scale genome. The total runtime of genBlastG is merely 44 seconds, compared with 2 hours and 47 minutes spent by GeneWise on 75 genes.

Table 7 Length Distribution of 75 test genes on the human genome

	Gene categories according to query length			
Query protein length	[1, 500)	[500, 1000)	[1000, 2500)	[2500, 5000)
# of genes	43	27	3	2

Figure 34 Running Time on the 75 human test genes



4.8.5.2 Comparison of Query Alignment PID

Figure 35 Query Alignment PID on the 75 human test genes

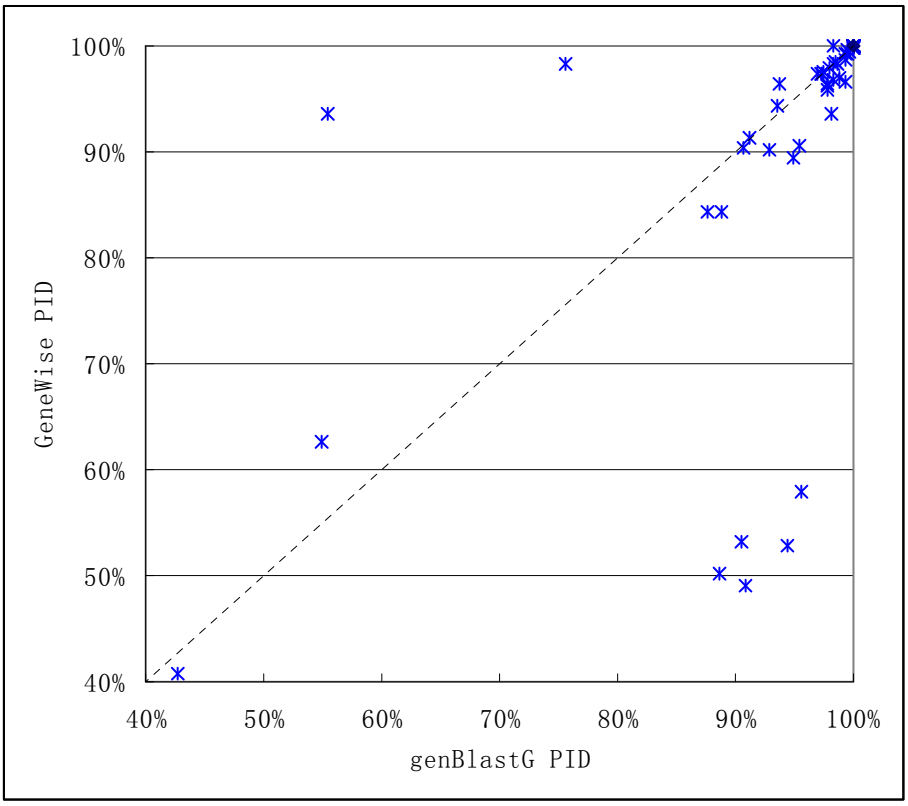


Figure 35 shows the query alignment PIDs of genBlastG and GeneWise models on the 75 test genes. Both genBlastG and GeneWise produced models with PIDs of more than 40% for all 75 genes. In most cases, genBlastG models show comparable PIDs to GeneWise models. The majority of genBlastG models (68 genes) have PIDs of more than 90%, whereas 65 GeneWise models have PIDs of more than 90%. The average PID of genBlastG models is 95.78%, and the average PID of GeneWise models is 93.60%. Out of the 75 genes, genBlastG models show higher PID than the corresponding GeneWise models on 28 genes, among which 5 genBlastG models show much higher PID (with more than 10% PID difference). On the other hand, 8 genBlastG models show

lower PIDs, among which 2 genBlastG models have more than 10% lower PIDs than the corresponding GeneWise models. These cases are clearly identified in Figure 35. It indicates that genBlastG is comparable to GeneWise in terms of overall prediction quality and the two algorithms can be used as complement to each other.

4.8.5.3 Running genBlastG on the entire human genome

We have also tested genBlastG on the entire human genome, using the entire set of 77,748 human protein sequences as queries. genBlastG produced gene models for 77,264 of these proteins, with some genes having alternative models that lead to the same alignment PID. On average, genBlastG gene models achieved alignment PID of 92.36%, with query coverage of 99.07%. This shows that these gene models are very similar to their corresponding queries. The average running time of genBlastG is less than 1 second, with the total time of just 12 hours.

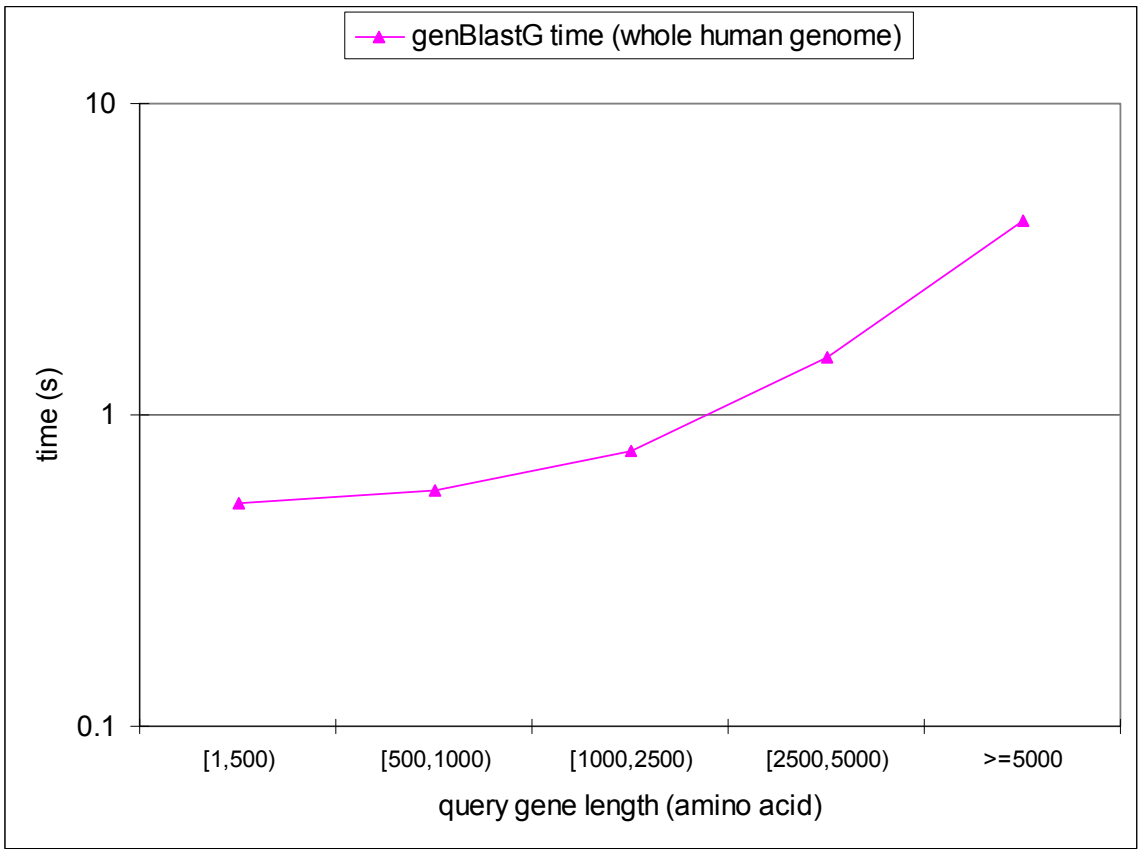
Table 8 Length Distribution of Genes on the entire human genome

	Gene categories according to query length				
Query protein length	[1, 500)	[500, 1000)	[1000, 2500)	[2500, 5000)	>=5000
# of genes	54913	15912	5864	499	76

Table 8 shows the length distribution of genes on the entire human genome. For each category, the average running time of genBlastG is shown in

Figure 36. It can be seen that the behaviour of genBlastG is persistant with its performance on the smaller genomes.

Figure 36 Average Running Time of genBlastG on the entire human genome



4.8.5.4 Summary

The experiments on the human genome show similar performances of genBlastG as in the *C. elegans* and *C. briggsae* experiments. In general, genBlastG is significantly faster than GeneWise while being competitive or slightly better in accuracy. Running genBlastG on the human genome confirmed the applicability of genBlastG on such large and complex genome, making genBlastG a highly valuable homology-based gene prediction tool.

5: CONCLUSIONS

5.1 Discussions

Gene prediction is a crucial aspect in the field of genomics. The scope of gene prediction discussed in this thesis is on predicting gene structures that consist of coding exons and introns. This thesis presents a novel homology-based gene prediction program, genBlast, which is able to quickly and effectively locate gene models homologous to a query protein. The genBlast program will be made publicly available upon publication [111].

Like GeneWise, genBlast takes a query protein sequence and predicts genes encoding the same or similar protein sequences in the target DNA sequence. The detection of gene homology is based on protein sequence similarity, therefore, it is able to detect genes with silent mutations that result in changes in DNA sequence while coding the same protein. Such methods are able to predict both genes and pseudogenes, using the same treatment based on sequence similarity. Pseudogenes are non-functional relatives of known genes that do not have protein-coding ability, resulting from various genetic disablements (stop codons, frameshifts, or a lack of transcription) [71]. On the other hand, being protein based, such programs obviously cannot predict UTRs (untranslated regions) [25, 115]. When used in predicting protein-coding genes, such methods are usually more accurate than *ab initio* methods and homology-based methods that do not use protein evidence, especially when the target gene

is relatively well conserved and carries sequence similarity of over 85% [20]. Therefore, the closer the species is, the better the prediction can be made by protein-homology-based methods. This is also demonstrated by our experiments on the *C. elegans* and *C. briggsae* genomes, where genBlast produced much higher accuracy than nGASP predictions on the *C. elegans* genome, while on the *C. briggsae* genome, the improvement in prediction quality is distinguishable but smaller.

However, unlike GeneWise, genBlast avoids the use of high-cost mathematical models. It makes direct use of HSPs reported by local alignment tools such as BLAST. Previous attempts that make use of BLAST-like tools only use HSPs to identify possible gene regions [44]. Once gene regions are identified, HSPs are no longer used in subsequent gene prediction. In genBlast, the sequence similarity information represented in HSPs is extensively exploited, not only in identifying gene regions, but also for gene structure prediction. By integrating HSPs as direct evidences of sequence similarity, genBlast is able to efficiently predict gene models that code for similar protein sequence.

The two major components of genBlast serve different purposes. genBlastA identifies regions of homologous genes, while genBlastG predicts gene structures given the output from genBlastA. genBlastA can also be used separately to help other gene finders such as GeneWise, providing flexibility in the gene prediction framework outline by genBlast.

genBlast relies on HSPs (local alignments) to find sequence similarities, thus the initial step of finding HSPs is critical to the quality of genBlast models.

The problem of local alignment is well defined with optimal solutions. Sequence alignment tools for analyzing biological sequences usually make use of heuristics due to speed considerations. They are able to consistently produce reliable HSPs. In our experiments, HSPs produced by default BLAST parameters are adequate for genBlast predictions.

The experiments on the *C. briggsae* genome using the *C. elegans* genes as queries have shown that the gene models detected by genBlast have considerable advantages over the current annotations in WormBase and nGASP, with some gene models experimentally validated by PCR amplification. Our re-annotated *C. briggsae* gene models will be made available via WormBase upon publication [72], providing a solid improvement over current WormBase annotation.

In addition, compared with the state-of-the-art homology-based gene finder, GeneWise, genBlast is orders of magnitude faster, due to its attractively simple modelling of the gene prediction problem. genBlast extensively utilizes alignment information embedded in HSPs, which can be obtained efficiently with the use of fast sequence similarity search tools. This speed advantage gives genBlast the capability to handle large genomes, as demonstrated in the human genome experiments. Meanwhile, genBlast is also highly competitive in accuracy and always predicts complete gene models, as indicated by our experiments on the two nematode and the human genomes. For many cases where GeneWise did not produce reliable gene models, genBlast is able to predict better models with much higher alignment PIDs, indicating the existence of homologous genes

that are not yet identified by previous approaches. Such performance improvements demonstrated the value of genBlast in annotating newly sequenced genomes, as a good alternative to GeneWise.

genBlast also provides a substantial extension to BLAST or other sequence similarity search tools, which are widely used to perform analysis on the genome sequences. The results of these tools are fragmented and often over-whelming even to experienced users. genBlastA organizes HSPs into ranked HSP groups, and genBlastG further provides gene structure predictions. This makes BLAST results more accessible and meaningful to experimental biologists. Thus genBlast extends the functionality of sequence similarity search tools by directly pinpointing the gene regions and even precise gene structures. The possibility of integrating genBlast into BLAST or WU-BLAST to enable them for gene prediction will be explored (N. Chen, personal communications) and if successful, it will have great impact in gene prediction community due to popularity of BLAST.

5.2 Future Work

The quest of gene prediction is still an on-going journey. The use of protein sequence is beneficial in achieving high accuracy in gene prediction. The merit of genBlast has been demonstrated by the experiments on the two nematodes and the human genomes. The overall performance of genBlast should be further evaluated by testing it on other species. In particular, genBlast should be assessed by using it on genomes with more diverse evolutionary

distances between the query and target genomes, in order to study the effect of genome distance on the accuracy of gene prediction.

A further extension to the current genBlast algorithm can be the incorporation of more evidences, both intrinsic and extrinsic. For example, one can incorporate advanced splice site finders that take into account the base dependency, length distributions and content biases between exons and introns. In addition, full-length cDNAs or ESTs could be used to refine gene prediction.

The study of genome variation is currently an active research topic [90]. Studies have shown that genomes of individual humans have numerous variations [67-69, 116], such as SNPs (single nucleotide polymorphism), insertions, deletions, inversions, transpositions, translocations, etc. Large-scale study of these genome variations will provide insight into the effect of genome variations on human health and lead to personalized medicine [142]. It is interesting to note that the design principle behind genBlastA is suitable for general-purpose homologous sequence assembly and can be useful for detecting genome variations. In fact, genBlastA is not limited to finding homologous sequences to proteins. It can be used to find homologous sequences to other biological sequences, such as DNA. For example, using a DNA sequence as the query and another DNA sequence as the target, genBlastA can be applied to assemble regions of query DNA that share sequence similarity to another region on the target DNA. By examining alignment between the two regions, one can identify various types of variations between the

two DNA sequences. It will be an interesting project to extend genBlastA in this direction.

BIBLIOGRAPHY

[1] ENSEMBL web site, human genome database, GRCh37 assembly, 2009. ftp://ftp.ensembl.org/pub/current_fasta/homo_sapiens/dna/Homo_sapiens.GRCh37.56.dna.toplevel.fa.gz.

[2] ENSEMBL web site, human genome peptide sequences, GRCh37 assembly, 2009. ftp://ftp.ensembl.org/pub/current_fasta/homo_sapiens/pep/Homo_sapiens.GRCh37.56.pep.all.fa.gz.

[3] WormBase web site, release WS200, 2009. http://www.wormbase.org/.

[4] nGASP predictions at WormBase ftp site. ftp://ftp.wormbase.org/pub/wormbase/nGASP_gene_predictions/predictions/.

[5] Adams, M. D., Celniker, S. E., Holt, R. A., Evans, C. A., Gocayne, J. D., Amanatides, P. G., et al. The Genome Sequence of Drosophila melanogaster. *Science,* 287(5461), 2185-2195. March 2000.

[6] Aizaki, H., Aoki, Y., Harada, T., Ishii, K., Suzuki, T., Nagamori, S., et al. Full-length complementary DNA of hepatitis C virus genome from an infectious blood sample. *Hepatology,* 27(2), 621-627. 1998.

[7] Allen, J. E., Majoros, W. H., Pertea, M. and Salzberg, S. L. JIGSAW, GeneZilla, and GlimmerHMM: puzzling out the features of human genes in the ENCODE regions. *Genome Biol.,* 7 Suppl 1, S9.1-13. 2006. DOI:10.1186/gb-2006-7-s1-s9.

[8] Allen, J. E. and Salzberg, S. L. JIGSAW: integration of multiple sources of evidence for gene prediction. *Bioinformatics,* 21(18), 3596-3603. September 2005.

[9] Altschul, S. F., Gish, W., Miller, W., Myers, E. W. and Lipman, D. J. Basic local alignment search tool. *J. Mol. Biol.,* 215(3), 403-410. October 1990.

[10] Altschul, S. F., Madden, T. L., Sch\affer, A. A., Zhang, J., Zhang, Z., Miller, W., et al. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.,* 25(17), 3389-3402. September 1997.

[11] Arabidopsis Genome Initiative. Analysis of the genome sequence of the flowering plant Arabidopsis thaliana. *Nature,* 408(6814), 796-815. December 2000.

[12] Ashburner, M. A biologist's view of the Drosophila genome annotation assessment project. *Genome Res.,* 10(4), 391-393. Apr 2000.

[13] Bayes, T. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London,* 53, 370-418. 1763.

[14]Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J. and Sayers, E. W. GenBank. *Nucleic Acids Res.,* 37(Database issue), D26-31. January 2009.

[15] Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J. and Wheeler, D. L. GenBank. *Nucleic Acids Res.,* 36(Database issue) January 2008.

[16] Bentley, D. R. Whole-genome re-sequencing. *Current opinion in genetics \& development,* 16(6), 545-552. December 2006.

[17] Bentley, D. R., Balasubramanian, S., Swerdlow, H. P., Smith, G. P., Milton, J., Brown, C. G., et al. Accurate whole human genome sequencing using reversible terminator chemistry. *Nature,* 456(7218), 53-59. November 2008.

[18] Bernal, A., Crammer, K., Hatzigeorgiou, A. and Pereira, F. Global discriminative learning for higher-accuracy computational gene prediction. *PLoS Comput. Biol.,* 3(3), e54. Mar 16 2007. DOI:10.1371/journal.pcbi.0030054.

[19] Birney, E. and Durbin, R. Using GeneWise in the Drosophila annotation experiment. *Genome Res.,* 10(4), 547-548. April 2000.

[20] Birney, E., Clamp, M. and Durbin, R. GeneWise and Genomewise. *Genome Res.,* 14(5), 988-995. May 2004.

[21] Black, D. L. Mechanisms of alternative pre-messenger RNA splicing. *Annu. Rev. Biochem.,* 72, 291-336. 2003.

[22] Blattner, F. R., Plunkett, G., Bloch, C. A., Perna, N. T., Burland, V., Riley, M., et al. The complete genome sequence of Escherichia coli K-12. *Science (New York, N.Y.),* 277(5331), 1453-1462. September 1997.

[23]Bolsover, S. R., Hyams, J. S., Shephard, E. A., White, H. A. and Weidemann, C. G. *Cell biology: a short course.* John Wiley & Sons, Hoboken, N.J., 2004.

[24] Brejova, B., Brown, D. G., Li, M. and Vinar, T. ExonHunter: a comprehensive approach to gene finding. *Bioinformatics,* 21 Suppl 1, i57-65. Jun 2005. DOI:10.1093/bioinformatics/bti1040.

[25] Brown, R. H., Gross, S. S. and Brent, M. R. Begin at the beginning: predicting genes with 5' UTRs. *Genome Res.,* 15(5), 742-747. May 2005.

[26] Brunak, S., Engelbrecht, J. and Knudsen, S. Prediction of human mRNA donor and acceptor sites from the DNA sequence. *J. Mol. Biol.,* 220(1), 49-65. Jul 5 1991.

[27] Bultrini, E. and Pizzi, E. A new parameter to study compositional properties of non-coding regions in eukaryotic genomes. *Gene,* 385, 75-82. Dec 30 2006. DOI:10.1016/j.gene.2006.05.030.

[28] Burge, C. and Karlin, S. Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol.,* 268(1), 78-94. April 1997.

[29] Burset, M. and Guig\'o R. Evaluation of gene structure prediction programs. *Genomics,* 34(3), 353-367. June 1996.

[30] Burset, M., Seledtsov, I. A. and Solovyev, V. V. Analysis of canonical and non-canonical splice sites in mammalian genomes. *Nucleic Acids Res.,* 28(21), 4364-4375. November 2000.

[31] C. elegans Sequencing Consortium. Genome sequence of the nematode C. elegans: a platform for investigating biology. *Science (New York, N.Y.),* 282(5396), 2012-2018. December 1998.

[32] Carrier, L., Bonne, G., B\ahrend, E., Yu, B., Richard, P., Niel, F., et al. Organization and sequence of human cardiac myosin binding protein C gene (MYBPC3) and identification of mutations predicted to produce truncated proteins in familial hypertrophic cardiomyopathy. *Circ. Res.,* 80(3), 427-434. March 1997.

[33] Chen, N., Harris, T. W., Antoshechkin, I., Bastiani, C., Bieri, T., Blasiar, D., et al. WormBase: a comprehensive data resource for Caenorhabditis biology and genomics. *Nucleic Acids Res.,* 33(Database issue) January 2005.

[34] Chen, N., Pai, S., Zhao, Z., Mah, A., Newbury, R., Johnsen, R. C., et al. Identification of a nematode chemosensory gene family. *Proc. Natl. Acad. Sci. U. S. A.,* 102(1), 146-151. January 2005.

[35] Chen, N. and Stein, L. D. Conservation and functional significance of gene topology in the genome of Caenorhabditis elegans. *Genome Res.,* 16(5), 606-617. May 2006.

[36] Chen, T. M., Lu, C. C. and Li, W. H. Prediction of splice sites with dependency graphs and their expanded bayesian networks. *Bioinformatics,* 21(4), 471-482. Feb 15 2005. DOI:10.1093/bioinformatics/bti025.

[37] Chimpanzee Sequencing Analysis Consortium. Initial sequence of the chimpanzee genome and comparison with the human genome. *Nature,* 437(7055), 69-87. September 2005.

[38] Coghlan, A., Fiedler, T., McKay, S., Flicek, P., Harris, T., Blasiar, D., et al. nGASP - the nematode genome annotation assessment project. *BMC Bioinformatics,* 9(1), 549. December 2008.

[39] Coghlan, A., Stajich, J. E. and Harris, T. W. Comparative genomics in C. elegans, C. briggsae, and other Caenorhabditis species. *Methods Mol. Biol.,* 351, 13-29. 2006.

[40] Coghlan, A. and Wolfe, K. H. Fourfold faster rate of genome rearrangement in nematodes than in Drosophila. *Genome Res.,* 12(6), 857-867. June 2002.

[41] Comeron, J. M. and Aguad\'e Montserrat. An Evaluation of Measures of Synonymous Codon Usage Bias. *J. Mol. Evol.,* 47(3), 268-274. September 1998.

[42] NCBI: The genetic codes. http://www.ncbi.nlm.nih.gov.proxy.lib.sfu.ca/Taxonomy/Utils/wprintgc.cgi.

[43] Cruveiller, S., Jabbari, K., Clay, O. and Bernardi, G. Compositional features of eukaryotic genomes for checking predicted genes. *Brief Bioinform,* 4(1), 43-52. January 2003.

[44] Cui, X., Vinar, T., Brejova, B., Shasha, D. and Li, M. Homology search for genes. *Bioinformatics,* 23(13), 97-103. July 2007.

[45] Curwen, V., Eyras, E., Andrews, T. D., Clarke, L., Mongin, E., Searle, S. M., et al. The Ensembl automatic gene annotation system. *Genome Res.,* 14(5), 942-950. May 2004.

[46] Deutsch, M. and Long, M. Intron-exon structures of eukaryotic model organisms. *Nucleic Acids Res.,* 27(15), 3219-3228. August 1999.

[47] Dewey, C., Wu, J. Q. Q., Cawley, S., Alexandersson, M., Gibbs, R. and Pachter, L. Accurate identification of novel human genes through simultaneous gene prediction in human, mouse, and rat. *Genome Res.,* 14(4), 661-664. April 2004.

[48] Djebali, S., Delaplace, F. and Crollius, H. R. Exogean: a framework for annotating protein-coding genes in eukaryotic genomic DNA. *Genome Biol.,* 7 Suppl 1, S7.1-10. 2006. DOI:10.1186/gb-2006-7-s1-s7.

[49] Do, J. H. and Choi, D. K. Computational approaches to gene prediction. *Journal of Microbiology,* 44(2), 137-144. 2006.

[50] Elsik, C. G., Mackey, A. J., Reese, J. T., Milshina, N. V., Roos, D. S. and Weinstock, G. M. Creating a honey bee consensus gene set. *Genome Biol.,* 8(1), R13. 2007. DOI:10.1186/gb-2007-8-1-r13.

[51] ENCODE Project Consortium. The ENCODE (ENCyclopedia Of DNA Elements) Project *Science,* 306(5696), 636-640. Oct 22 2004. DOI:10.1126/science.1105136.

[52] Fickett, J. W. and Tung, C. S. Assessment of protein coding measures. *Nucleic Acids Res.,* 20(24), 6441-6450. December 1992.

[53] Fleischmann, R. D., Adams, M. D., White, O., Clayton, R. A., Kirkness, E. F., Kerlavage, A. R., et al. Whole-genome random sequencing and assembly of Haemophilus influenzae Rd. *Science (New York, N.Y.),* 269(5223), 496-512. July 1995.

[54] Flicek, P., Aken, B. L., Ballester, B., Beal, K., Bragin, E., Brent, S., et al. Ensembl's 10th year. *Nucleic Acids Res.,* Nov 11 2009. DOI:10.1093/nar/gkp972.

[55] Florea, L., Hartzell, G., Zhang, Z., Rubin, G. M. and Miller, W. A computer program for aligning a cDNA sequence with a genomic DNA sequence. *Genome Res.,* 8(9), 967-974. September 1998.

[56] Gao, F. and Zhang, C. T. Comparison of various algorithms for recognizing short coding sequences of human genes. *Bioinformatics,* 20(5), 673-681. March 2004.

[57] Gross, S. S. and Brent, M. R. Using multiple alignments to improve gene prediction. *J. Comput. Biol.,* 13(2), 379-393. Mar 2006. DOI:10.1089/cmb.2006.13.379.

[58] Gross, S. S., Do, C. B., Sirota, M. and Batzoglou, S. CONTRAST: a discriminative, phylogeny-free approach to multiple informant de novo gene prediction. *Genome Biol.,* 8(12), R269. 2007. DOI:10.1186/gb-2007-8-12-r269.

[59] Guigó, R., Flicek, P., Abril, J. F., Reymond, A., Lagarde, J., Denoeud, F., et al. EGASP: the human ENCODE Genome Annotation Assessment Project. *Genome Biol.,* 7 Suppl 1 2006.

[60] Gupta, B. P., Johnsen, R. and Chen, N. Genomics and biology of the nematode Caenorhabditis briggsae. *WormBook : the online review of C.elegans biology,* , 1-16. 2007.

[61] Haas, B. J., Salzberg, S. L., Zhu, W., Pertea, M., Allen, J. E., Orvis, J., et al. Automated eukaryotic gene structure annotation using EVidenceModeler and the Program to Assemble Spliced Alignments. *Genome Biol.,* 9(1), R7. Jan 11 2008. DOI:10.1186/gb-2008-9-1-r7.

[62] Hardison, R. C. Comparative genomics. *PLoS biology,* 1(2), e58. November 2003.

[63] Harrow, J., Denoeud, F., Frankish, A., Reymond, A., Chen, C. K., Chrast, J., et al. GENCODE: producing a reference annotation for ENCODE *Genome Biol.,* 7 Suppl 1, S4.1-9. 2006. DOI:10.1186/gb-2006-7-s1-s4.

[64] Henikoff, S. and Henikoff, J. G. Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. U. S. A.,* 89(22), 10915-10919. November 1992.

[65] Henikoff, S., Keene, M. A., Fechtel, K. and Fristrom, J. W. Gene within a gene: nested Drosophila genes encode unrelated proteins on opposite DNA strands. *Cell,* 44(1), 33-42. January 1986.

[66]Hillier, L. W., Coulson, A., Murray, J. I., Bao, Z., Sulston, J. E. and Waterston, R. H. Genomics in C. elegans: So many genes, such a little worm. *Genome Res.,* 15(12), 1651-1660. December 2005.

[67] Human Genome Structural Variation Working Group, Eichler, E. E., Nickerson, D. A., Altshuler, D., Bowcock, A. M., Brooks, L. D., et al. Completing the map of human genetic variation *Nature,* 447(7141), 161-165. May 10 2007. DOI:10.1038/447161a.

[68] Iafrate, A. J., Feuk, L., Rivera, M. N., Listewnik, M. L., Donahoe, P. K., Qi, Y., et al. Detection of large-scale variation in the human genome *Nat. Genet.,* 36(9), 949-951. Sep 2004. DOI:10.1038/ng1416.

[69] International HapMap Consortium, Frazer, K. A., Ballinger, D. G., Cox, D. R., Hinds, D. A., Stuve, L. L., et al. A second generation human haplotype map of over 3.1 million SNPs *Nature,* 449(7164), 851-861. Oct 18 2007. DOI:10.1038/nature06258.

[70] International Human Genome Sequencing Consortium. Finishing the euchromatic sequence of the human genome. *Nature,* 431(7011), 931-945. October 2004.

[71] Jacq, C., Miller, J. R. and Brownlee, G. G. A pseudogene structure in 5S DNA of Xenopus laevis *Cell,* 12(1), 109-120. Sep 1977.

[72] Jeffrey SC Chu, Rong She, Jun Wang, Ke Wang and Nansheng Chen. Improving *Caenorhabditis briggsae* gene annotation using genBlastG. 2010. to be submitted.

[73]Johnson, J. M., Castle, J., Garrett-Engele, P., Kan, Z., Loerch, P. M., Armour, C. D., et al. Genome-wide survey of human alternative pre-mRNA splicing with exon junction microarrays. *Science (New York, N.Y.),* 302(5653), 2141-2144. December 2003.

[74]Karolchik, D., Baertsch, R., Diekhans, M., Furey, T. S., Hinrichs, A., Lu, Y. T., et al. The UCSC Genome Browser Database *Nucleic Acids Res.,* 31(1), 51-54. Jan 1 2003.

[75] Kent, W. J. BLAT--the BLAST-like alignment tool. *Genome Res.,* 12(4), 656-664. April 2002.

[76] Korf, I., Flicek, P., Duan, D. and Brent, M. R. Integrating genomic homology into gene structure prediction. *Bioinformatics,* 17 Suppl 1 2001.

[77] Krogh, A. 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.

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

[79]Lander, E. S., Linton, L. M., Birren, B., Nusbaum, C., Zody, M. C., Baldwin, J., et al. Initial sequencing and analysis of the human genome. *Nature,* 409(6822), 860-921. February 2001.

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

[81] Majoros, W. H., Pertea, M. and Salzberg, S. L. TigrScan and GlimmerHMM: two open source ab initio eukaryotic gene-finders. *Bioinformatics,* 20(16), 2878-2879. Nov 1 2004. DOI:10.1093/bioinformatics/bth315.

[82] Majoros, W. and Ohler, U. Advancing the State of the Art in Computational Gene Prediction. In Anonymous , 2007, 81-106.

[83] Makarov, V. Computer programs for eukaryotic gene prediction. *Brief Bioinform,* 3(2), 195-199. January 2002.

[84] Mardis, E. R. Next-generation DNA sequencing methods. *Annu. Rev. Genomics Hum. Genet.,* 9, 387-402. 2008. DOI:10.1146/annurev.genom.9.081307.164359.

[85] Mardis, E. R. The impact of next-generation sequencing technology on genetics. *Trends in genetics : TIG,* 24(3), 133-141. March 2008.

[86]Margulies, M., Egholm, M., Altman, W. E., Attiya, S., Bader, J. S., Bemben, L. A., et al. Genome sequencing in microfabricated high-density picolitre reactors. *Nature,* 437(7057), 376-380. September 2005.

[87] Mathé, C., Sagot, M. F., Schiex, T. and Rouzé, P. Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Res.,* 30(19), 4103-4117. October 2002.

[88] Matlin, A. J., Clark, F. and Smith, C. W. Understanding alternative splicing: towards a cellular code. *Nature reviews.Molecular cell biology,* 6(5), 386-398. May 2005.

[89] McLachlan, A. D., Staden, R. and Boswell, D. R. A method for measuring the non-random bias of a codon usage table. *Nucleic Acids Res.,* 12(24), 9567-9575. Dec 21 1984.

[90] Medvedev, P., Stanciu, M. and Brudno, M. Computational methods for discovering structural variation with next-generation sequencing *Nat. Methods,* 6(11 Suppl), S13-20. Nov 2009. DOI:10.1038/nmeth.1374.

[91] Merchant, S. S., Prochnik, S. E., Vallon, O., Harris, E. H., Karpowicz, S. J., Witman, G. B., et al. The Chlamydomonas Genome Reveals the Evolution of Key Animal and Plant Functions. *Science,* 318(5848), 245-250. October 2007.

[92] Meyer, I. M. and Durbin, R. Gene structure conservation aids similarity based gene prediction. *Nucleic Acids Res.,* 32(2), 776-783. 2004.

[93] Modrek, B. and Lee, C. A genomic view of alternative splicing. *Nat. Genet.,* 30(1), 13-19. January 2002.

[94] Mount, S. M., Burks, C., Hertz, G., Stormo, G. D., White, O. and Fields, C. Splicing signals in Drosophila: intron size, information content, and consensus sequences. *Nucleic Acids Res.,* 20(16), 4255-4262. August 1992.

[95]Mouse Genome Sequencing Consortium, Waterston, R. H., Lindblad-Toh, K., Birney, E., Rogers, J., Abril, J. F., et al. Initial sequencing and comparative analysis of the mouse genome. *Nature,* 420(6915), 520-562. December 2002.

[96] Mullis, K. B. The unusual origin of the polymerase chain reaction. *Sci. Am.,* 262(4) April 1990.

[97] Nagaraj, S. H., Gasser, R. B. and Ranganathan, S. A hitchhiker's guide to expressed sequence tag (EST) analysis. *Brief Bioinform,* 8(1), 6-21. January 2007.

[98]Nishio, H., Takeshima, Y., Narita, N., Yanagawa, H., Suzuki, Y., Ishikawa, Y., et al. Identification of a novel first exon in the human dystrophin gene and of a new promoter located more than 500 kb upstream of the nearest known promoter. *J. Clin. Invest.,* 94(3), 1037-1042. September 1994.

[99] Oliver, J. L. and Mar\'in, A. A relationship between GC content and coding-sequence length. *J. Mol. Evol.,* 43(3), 216-223. September 1996.

[100] O'Mahony, M. Sensory evaluation of food : statistical methods and procedures. In Anonymous New York : Dekker, , 1986, 487.

[101] Parra, G., Blanco, E. and Guigo, R. GeneID in Drosophila. *Genome Res.,* 10(4), 511-515. Apr 2000.

[102] Parra, G., Agarwal, P., Abril, J. F., Wiehe, T., Fickett, J. W. and Guig\'o Roderic. Comparative gene prediction in human and mouse. *Genome Res.,* 13(1), 108-117. January 2003.

[103] Pavlovic, V., Garg, A. and Kasif, S. A Bayesian framework for combining gene predictions. *Bioinformatics,* 18(1), 19-27. Jan 2002.

[104] Pearson, W. R. and Lipman, D. J. Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. U. S. A.,* 85(8), 2444-2448. April 1988.

[105] Quesada, V. OTC and AUL1, two convergent and overlapping genes in the nuclear genome of Arabidopsis thaliana. *FEBS Lett.,* 461(1-2), 101-106. November 1999.

[106] Reese, M. G., Eeckman, F. H., Kulp, D. and Haussler, D. Improved splice site detection in Genie. *J. Comput. Biol.,* 4(3), 311-323. Fall 1997.

[107] Reese, M. G., Kulp, D., Tammana, H. and Haussler, D. Genie--gene finding in Drosophila melanogaster. *Genome Res.,* 10(4), 529-538. Apr 2000.

[108] Robertson, H. M. and Thomas, J. H. The putative chemoreceptor families of C. elegans. *WormBook : the online review of C.elegans biology,* , 1-12. 2006.

[109] Rogers, A., Antoshechkin, I., Bieri, T., Blasiar, D., Bastiani, C., Canaran, P., et al. WormBase 2007. *Nucleic Acids Res.,* 36(Database issue), D612-7. Jan 2008. DOI:10.1093/nar/gkm975.

[110] Rogozin, I. B. and Milanesi, L. Analysis of donor splice sites in different eukaryotic organisms. *J. Mol. Evol.,* 45(1), 50-59. Jul 1997.

[111] Rong She, Jeffrey Shih-Chieh Chu, Ke Wang and Nansheng Chen. genBlastG: a fast homology-Based Gene Prediction Program. 2010. to be submitted.

[112] Salamov, A. A. and Solovyev, V. V. Ab initio gene finding in Drosophila genomic DNA. *Genome Res.,* 10(4), 516-522. Apr 2000.

[113] Sanger, F., Air, G. M., Barrell, B. G., Brown, N. L., Coulson, A. R., Fiddes, C. A., et al. Nucleotide sequence of bacteriophage phi X174 DNA. *Nature,* 265(5596), 687-695. February 1977.

[114] Schuster, S. C. Next-generation sequencing transforms today's biology. *Nature Methods,* 5(1), 16-18. December 2007.

[115] Schweikert, G., Zien, A., Zeller, G., Behr, J., Dieterich, C., Ong, C. S. S., et al. mGene: accurate SVM-based gene finding with an application to nematode genomes. *Genome Res.,* 19(11), 2133-2143. November 2009.

[116] Sebat, J., Lakshmi, B., Troge, J., Alexander, J., Young, J., Lundin, P., et al. Large-Scale Copy Number Polymorphism in the Human Genome. *Science,* 305(5683), 525-528. July 2004.

[117] She, R., Chu, J. S. S., Wang, K., Pei, J. and Chen, N. GenBlastA: enabling BLAST to identify homologous gene sequences. *Genome Res.,* 19(1), 143-149. January 2009.

[118] Shendure, J., Porreca, G. J., Reppas, N. B., Lin, X., McCutcheon, J. P., Rosenbaum, A. M., et al. Accurate multiplex polony sequencing of an evolved bacterial genome. *Science (New York, N.Y.),* 309(5741), 1728-1732. September 2005.

[119] Singh, G. B. Computational Approaches for Gene Identification. In Anonymous , 1999, 351-364.

[120] Slater, G. S. S. and Birney, E. Automated generation of heuristics for biological sequence comparison. *BMC Bioinformatics,* 6(1), 31. 2005.

[121] Smith, T. F. and Waterman, M. S. Identification of common molecular subsequences. *J. Mol. Biol.,* 147(1), 195-197. March 1981.

[122] Staden, R. Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids Res.,* 12(1 Pt 2), 505-519. Jan 11 1984.

[123] Stanke, M., Schoffmann, O., Morgenstern, B. and Waack, S. Gene prediction in eukaryotes with a generalized hidden Markov model that uses hints from external sources. *BMC Bioinformatics,* 7(1) 2006.

[124] Stanke, M., Tzvetkova, A. and Morgenstern, B. AUGUSTUS at EGASP: using EST, protein and genomic alignments for improved gene prediction in the human genome. *Genome Biol.,* 7 Suppl 1 2006.

[125] Stanke, M. and Waack, S. Gene prediction with a hidden Markov model and a new intron submodel. *Bioinformatics,* 19 Suppl 2 October 2003.

[126] Stein, L. Genome annotation: from sequence to biology. *Nature reviews.Genetics,* 2(7), 493-503. July 2001.

[127] Stein, L. D., Bao, Z., Blasiar, D., Blumenthal, T., Brent, M. R., Chen, N., et al. The genome sequence of Caenorhabditis briggsae: a platform for comparative genomics. *PLoS biology,* 1(2), e45. November 2003.

[128] Suzuki, H., Brown, C. J., Forney, L. J. and Top, E. M. Comparison of Correspondence Analysis Methods for Synonymous Codon Usage in Bacteria. *DNA Res.,* , dsn028. October 2008.

[129] Tech, M. and Merkl, R. YACOP: Enhanced gene prediction obtained by a combination of existing methods. *In. Silico Biol.,* 3(4), 441-451. 2003.

[130] Ter-Hovhannisyan, V., Lomsadze, A., Chernoff, Y. O. and Borodovsky, M. Gene prediction in novel fungal genomes using an ab initio algorithm with unsupervised training. *Genome Res.,* 18(12), 1979-1990. Dec 2008. DOI:10.1101/gr.081612.108.

[131] Trifonoy, E. N. and Sussman, J. L. The pitch of chromatin DNA is reflected in its nucleotide sequence. *Proceedings of the National Academy of Sciences USA,* 77(7), 3816-3820. July 1980.

[132] Udi Manber. *Introduction to Algorithms -- A Creative Approach.* Addison-Wesley, MA, USA, 1989.

[133] van Baren, M. J., Koebbe, B. C. and Brent, M. R. Using N-SCAN or TWINSCAN to predict gene structures in genomic DNA sequences. *Curr. Protoc. Bioinformatics,* Chapter 4, Unit 4.8. Dec 2007. DOI:10.1002/0471250953.bi0408s20.

[134] Venter, J. C., Adams, M. D., Myers, E. W., Li, P. W., Mural, R. J., Sutton, G. G., et al. The sequence of the human genome. *Science (New York, N.Y.),* 291(5507), 1304-1351. February 2001.

[135] Viterbi, A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory,* 13(2), 260-269. April 1967.

[136] Wang, J., Wang, W., Li, R., Li, Y., Tian, G., Goodman, L., et al. The diploid genome sequence of an Asian individual. *Nature,* 456(7218), 60-65. November 2008.

[137] Wang, Z., Chen, Y. and Li, Y. A brief review of computational gene prediction methods. *Genomics, proteomics \& bioinformatics / Beijing Genomics Institute,* 2(4), 216-221. November 2004.

[138] Watson, J. D. and Crick, F. Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid. *Nature,* 171(4356), 737-738. April 1953.

[139] Wei, C. and Brent, M. R. Using ESTs to improve the accuracy of de novo gene prediction. *BMC Bioinformatics,* 7, 327. Jul 3 2006. DOI:10.1186/1471-2105-7-327.

[140] Wheeler, D. A., Srinivasan, M., Egholm, M., Shen, Y., Chen, L., McGuire, A., et al. The complete genome of an individual by massively parallel DNA sequencing. *Nature,* 452(7189), 872-876. April 2008.

[141] Yin, C. and Yau, S. S. T. Prediction of protein coding regions by the 3-base periodicity analysis of a DNA sequence. *J. Theor. Biol.,* 247(4), 687-694. August 2007.

[142] Zhang, F., Gu, W., Hurles, M. E. and Lupski, J. R. Copy number variation in human health, disease, and evolution *Annu. Rev. Genomics Hum. Genet.,* 10, 451-481. 2009. DOI:10.1146/annurev.genom.9.081307.164217.

[143] Zhang, M. Q. and Marr, T. G. A weight array method for splicing signal analysis. *Comput. Appl. Biosci.,* 9(5), 499-509. October 1993.

[144] Zhang, M. Q. Computational prediction of eukaryotic protein-coding genes. *Nature reviews.Genetics,* 3(9), 698-709. September 2002.

[145] Zhou, Y., Liang, Y., Hu, C., Wang, L. and Shi, X. An artificial neural network method for combining gene prediction based on equitable weights. *Neurocomputing,* 71(4-6), 538-543. January 2008.

Appendix 1: genBlast Pseudocodes

Stage 1: BLAST
Input: a query protein sequence *q*, a target genome *T*
Output: HSPs

Run BLAST with query *q* and target *T*

Stage 2: genBlastA
Input: HSPs
Output: a ranked list of HSP groups

```
1.  For each DNA sequence t in T
2.      collect HSPs on t into set H
3.      initialize starting node σ and ending node τ
4.      construct all adjacency edges between HSP nodes in H
5.      construct all skip edges between HSP nodes in H
6.      construct separating edges from σ to HSP nodes
        without incoming edges
7.      construct separating edges from HSP nodes without
        outgoing edges to τ
8.      For each destination node of a separating edge
9.          compute single-source shortest extension path
10.     End for
11. End for
12. Rank all local shortest extension paths by length
13. Output groups in ranked order
```

Stage 3: genBlastG
Input: ranked HSP groups, HSPs, *q*, *T*
Output: gene models for all HSP groups

```
1.  For each HSP group in ranked order
2.      Find gene start (first in-frame start codon at the
        beginning or before first HSP)
3.      Find gene end (first in-frame stop codon at the end
        or after the last HSP)
4.      initial-gene-model = Compute-Exons(HSPs, gene-start,
        gene-end)
5.      final-gene-model = Repair-Gene-Model(initial-gene-
        model, HSPs, q, T)
```

```
6. End for
```

Compute-Exons(HSPs, gene-start, gene-end)

```
1. Find-intron-regions(HSPs)
2. For each intron region R in sequential order
3.     Find MAX_NUM_SPLICE_SITES of candidate donors around
       upstream border of R
4.     Find MAX_NUM_SPLICE_SITES of candidate acceptors
       around downstream border of R
5. End for
6. For each intron region in sequential order
7.     Find-best-donor-acceptor-pair(candidate-sites, HSPs,
       q, T)
8. End for
9. Return gene-model
```

Find-intron-regions(HSPs)

```
1. For each HSP in sequential order
2.     For each gap on HSP query segment with length >=
       MIN_INTRON_REGION_LEN
3.         Add the corresponding target (DNA) region to
           intron region
4.     End For
5.     If current HSP and next HSP have overlapping query
       segments
6.         Examine the overlapped query portion O in two
HSPs,
           identify the target region R that aligns with O
           with lower identity
7.         Add region R to intron region
8.     End if
9.     If the distance between current HSP and next HSP >=
       MIN_INTRON_REGION_LEN
10.        Add the DNA region between current and next HSP
to
           intron region
11.    End if
12. End for
```

Find-best-donor-acceptor-pair(candidate-sites, HSPs, *q*, *T*)

```
1. For each valid pair of donor d and acceptor a in
   candidate-sites (including no intron case)
```

```
2.     Get Sd, donor-side target segment for d
3.     Get Sa, acceptor-side target segment for a
4.     Get corresponding query segment
5.     Compute PID/score of alignment between spliced
       sequence Sd-Sa and the corresponding query segment,
       using HSP alignments
6.  End for

7.  If a single donor-acceptor pair d1-a1 has the highest
    PID
8.     return d1-a1
9.  Else
10.    If among the pairs with highest PID, a single pair
       d2-a2 has highest alignment score
11.       Return d2-a2
12.    Else
13.       For each pair of donor d and acceptor a among all
          pairs with highest-PID/score
14.          Compute optimal sequence alignment between
             spliced sequence and corresponding query
             segment
15.       End for
16.       Return the pair of d-a that has highest alignment
          PID/score
17.    End if
18. End if
```

Repair-Gene-Model(initial-gene-model, HSPs, *q*, *T*)

```
1.  If there is missing query coverge before first exon in
    the initial gene model
2.     find additional best local alignment in DNA region
       before the first exon, and add it to list of HSPs
3.     Find new gene start based on new alignment
4.     new-exons = Compute-Exons(HSPs, new-gene-start,
       first-exon-end)
5.     If the spliced alignment PID of new-exons is higher
       than that of initial first exon
6.        replace the initial first exon by new-exons
7.     End if
8.  End if
9.  For each exon in the current gene model
10.    If there is significant missing query coverage in
       current region (between exon-start of current exon E1
       and exon-end of next exon E2)
11.       Align corresponding DNA region and missing query
```

```
          region for best local alignment, add it to the
          list of original HSPs
12.       new-exons =Compute-Exons(HSPs, exon-E1-start,
          Exon-E2-end)
13.       If the spliced alignment PID of new-exons is
          Higher
14.          Replace E1 and E2 by new-exons
15.       End if
16.    End if
17. End for

18. If there is missing query coverage after the last exon
    in current gene model
19.    Find additional best local alignment in DNA region
       after the last exon, add it to list of HSPs
20.    Find new gene end based on new alignment
21.    new-exons = Compute-Exons(HSPs, last-exon-start, new-
       gene-end)
22.    If spliced alignment PID of new-exons is higher
23.       Replace the last exon by new-exons
24.    End if
25. End if
26. Return final-gene-model
```