GENE PREDICTION BY COMBINING OUTPUTS FROM EXONHUNTER AND SGP2

YUJING KUAI

Gene Prediction By Combining Outputs From ExonHunter and SGP2

by

© Yujing Kuai

A thesis submitted to the

School of Graduate Studies

in partial fulfilment of the

requirements for the degree of

Master of Science

Department of Computational Science

Memorial University of Newfoundland

February 2009

St. John's Newfoundland

Abstract

Recently gene prediction has become a critical research area in computational biology. This thesis introduces our research on predicting genes in human DNA sequences. We present two algorithms to predict human genes by combining two chosen gene finders. One gene finder uses combination methods and another applies cross-species comparative sequence analysis. Based on these algorithms, a client-friendly gene finder can be developed to accurately predict human genes and thus to help discover genetic reasons of incurable human diseases.

Combination methods and cross-species comparative sequence analysis are two methods which become increasingly helpful. This thesis first summarizes and classifies main algorithms applied in these two methods, respectively. To be specific, we study two gene finders using comparative sequence analysis and three gene finders applying combination methods. Their architectures and experiments are reviewed separately and overall comparisons are done. According to our survey, currently many gene finders can predict genes with an sophisticated accuracy, but either the methods that gene finders apply have limitations, or the application of these gene finders is difficult for biologists and researchers in medicine. Aiming at these two disadvantages, we develop two algorithms to combine outputs of gene finders using combination methods and cross-species comparative sequence analysis. By comparing the genomes of Mus musculus and Canis familiars, the algorithms are firstly tested on the HMR195 dataset and then on the sequence between the markers D3S1259 and D3S3659 on human chromosome 3p25. The results show that to some extent our algorithms improve the performance of the gene finder using either comparative

sequence analysis or combination methods, demonstrating their own advantages on predicting different genetic information. Additionally, our work shows an inspiring perspective of developing a gene finder with a more friendly interface.

Acknowledgements

Here, I would like to express my gratitude to many people who helped me make this thesis possible.

First of all, I greatly acknowledge my supervisor, Dr. Wolfgang Banzhaf, for his guidance, support and encouragement. Under his supervision, I have gotten to know this area, chosen the topic and started the thesis.

I would also like to thank the professors who gave me advices on this research and my thesis, especially Dr. Terry-Lynn Young, Dr. Brian E. Staveley and Dr. Todd Wareham. Dr. Young helped me on some knowledge of human chromosomes and the project of this work. Dr. Staveley was the instructor of the course BIOL 3530 by which I obtained some biology background for my research. Thank Dr. Wareham for having given me some suggestions of the research and answered my questions about bioinformatics.

I heartily thank Nornan White. Without his precious help, I would not be able to do this research and implement my algorithms. I present my appreciation to all my close friends. They stood by my side all the time. I could not go this further without their encouragement and caring. A special gratitude to my good friend Xiaonan Wu. Through the writing of the thesis, she gave me tons of enlightening suggestions and helped me on the corrections and revisions.

My last words go to my parents and my families. In all the years of studying at MUN, my parents have given me all their love, support and encouragement so that I have had a very happy and inspired life here. A great thanks to all.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Getting started

After human DNA sequences have been completely deciphered, the next step is to identify the gene content in them. However, gene prediction is considered to be one of the most difficult problems in computational biology. Gene prediction for prokaryotes, which are single cell organisms, could be efficiently solved by Interpolated Markov models (IMM) [10]. Prediction for eukaryotes, on the other hand, is a big challenge because they have a much larger range of genomes with lower coding density and more complicated structures [11].

In general, the approaches for predicting genes can be classified into two types: ab initio gene finding and sequence similarity. Ab initio gene finding is to predict genes by using known genome structures. Therefore this method relies on detecting signal sensors (like start codons, stop codons and splice sites) and content sensors like coding regions and non-coding regions. Sequence similarity detects genes based

on a comparison with known genes of related species. However, both methods have their limitations. It is hard for ab initio gene finding to be sufficiently accurate to predict signals and contents because the genome structures are so complicated that just small parts are known. Because sequence similarity depends on known genes and yet because known genes are limited, this method hardly discovers all unknown genes.

Due to these limitations, comparative analysis and combination methods have been proposed. Since an increasing number of genomes of some species have been annotated, it is promising to use these data for recognizing human genes using comparative sequence analysis. By comparing the genomes across species, conserved elements can be detected and genes are found to be dense in these areas. Researchers also combine ab initio gene finding approaches with sequence similarity approaches to obtain better results. The predicted evidences produced by different programs using diverse algorithms are integrated into a complete gene prediction model to obtain an optimal prediction of structures. Thus, the methods applied for combining the evidences and the choice of the evidences are issues which will decide the accuracy of this type of gene prediction.

By investigating previous work on combination methods, it was found that so far tools using combination methods rarely consider cross-species comparative sequence analysis as part of their approaches. Hence, the work presented in this thesis has developed two algorithms which combine an existing gene finder using combination methods and another gene finder using the comparative analysis. Our work applies the two algorithms on the dataset HMR195 and on the sequence from markers D3S1259 to D3S3659 on human chromosome 3 to find human genes related to sudden cardiac

death. In terms of evaluation of the accuracy of the predictions in this thesis, we look at the most widely used two measures: sensitivity (Sn) and specificity (Sp) [12]. We calculate Sn and Sp at three different levels: nucleotide, exon and gene.

This thesis is organized into five chapters. Chapter 2 is a review of gene prediction. It gives some biological background including an introduction to DNA, genes and the gene structures of prokaryotes and eukaryotes. Furthermore, this chapter introduces the two approaches for identifying genes: ab initio gene finding and sequence similarity. The main algorithms of each approach are introduced and the advantages and disadvantages of the two approaches are discussed.

Chapter 3 describes the cross-species comparative sequence analysis approach and two gene finders using this method, AGenDA and SGP2. As mentioned earlier, both major approaches discussed in Chapter 2 have their limitations, and so the comparative analysis method was considered. Comparing the sequences of different species could help researchers understand conserved elements and thus find genes. In this chapter, the architectures and experiments of the two gene finders are presented. Furthermore, they are compared based on their architectures, applications and performance.

Chapter 4 mainly presents the work on exploring three gene finders which use combination methods. The three gene finders are ExonHunter, GAZE and JIGSAW. First, the combination methods are described. The main algorithms applied in this type of methods are reviewed and classified. Next, the architecture and experiments of each gene finder are demonstrated. Moreover, a comparison research of the three gene finders is conducted. Their architectures, applications and performance are comprehensively compared.

In Chapter 5, the main work of the thesis is presented. The goal of our work was to detect genes and genetic information related to the disease of sudden cardiac death. First this disease is briefly introduced. According to our survey of the current research on gene prediction, we conclude that combination methods have the best prospect for this area. In general, however, the tools combining different evidence are not very client-friendly, especially for researchers who are not computer scientists. We find that it is critical how one combines various functional evidences effectively but conveniently. Therefore, we combine the results given by ExonHunter and SGP2 by two algorithms that are developed here. Both of these algorithms follow the And-based and Or-based rules which have been presented in Murakami et al.'s work [8]. The difference between the algorithms is that one is at the exon level whereas another is at the gene level. In addition, because Mus Musculus and Canis have been shown to be some of the closest species to humans, these two species have been selected to be used in SGP2. Thus, this chapter gives a brief review of the properties of the genomes of Homo sapiens, Mus musculus and Canis. It also introduces the properties of human chromosome 3, Mus musculus and Canis databases used. The two algorithms are applied to the dataset HMR195 and the results are analyzed. In the last section, these two algorithms are run on the sequence between markers D3S1259 and D3S3659 on chromosome 3. The results are presented and discussed.

Chapter 2

Gene Prediction

2.1 Review of Biological Background

2.1.1 DNA

DNA is the code of life. Its name stems from deoxyribonucleic acid which is a nucleic acid, that contains the genetic instructions for the developmental process of living organisms. Proteins, which are the basis of life, are coded from DNA by transcription and translation. The structure of DNA is a double helix where two strands wrap around each other.

Chemically, DNA is a macro molecule composed of a backbone made of sugars and phosphate groups and a long polymer of simple units called nucleotides each of which contains one of the four nitrogenous bases. The four bases are adenine (abbreviated A), cytosine (C), guanine (G) and thymine (T). Each sugar has one base attached to it. The bases on each strand are bonded to the bases on the other strand and that is the reason why DNA forms a double-stranded spiral. As the canonical Watson-Crick

base pairing describes, A is bonded to T and G is bonded to C. But Non-Watson-Crick base pairing also occurs. In addition, usually each strand has two different ends: 5'(five prime) end and 3'(three prime) end.

DNA resides in all living cells and controls the cellular activities, such as growth, division, movement, death and so forth. When a cell needs to produce a particular protein, a signal will stimulate an enzyme to unwind the DNA. Then by the two processes of transcription and translation, the DNA is encoded in to proteins. In transcription, the two strands of DNA are separated and the coding sequence on one strand is transcribed into a messenger RNA (mRNA) sequence, which will act as a template for further transformation. The only difference between mRNA and DNA is that mRNA has uracil (U) instead of thymine(T), with U bonding to A. After transcription, mRNA moves out from the nucleus. In translation, mRNA passes its genetic message to the outside of the nucleus by transfer RNA(tRNA). In this process, every three letter words (triplets), called codons, are translated into one amino acid. Eventually a protein is formed by linking some of the amino acids.

2.1.2 Genes

DNA is a very long molecule, however not all of the DNA encodes proteins. Genes with the genetic message in DNA determine what an organism looks like and how it acts. A gene is a locatable region of genomic sequence that corresponds to a unit of inheritance, which is associated with regulatory regions, transcribed regions and/or other functional sequence regions [13, 14]. Genes are stored in chromosomes and are responsible for the inherited characteristics that make individuals different.

In transcription, the genes presented in DNA are copied by RNA polymerase, which is an enzyme. The base sequence of each gene begins at a "promoter region", which is a very important signal in gene prediction. This "start" signal for a gene tells RNA polymerase that transcription starts from there. When RNA polymerase reaches the terminator region, the transcription of the gene stops. mRNA is produced and RNA polymerase falls off the DNA to be ready for the next round of transcription. Thus, in a cell there is a large collection of mRNAs because they are the copies of various genes. The next step is to translate the copies of DNA into proteins. Each amino acid is coded for by a sequence of three base pairs in DNA like ATC and CAA. Most of the amino acids are coded for by several codons; for example, CTT and CTA both code for the amino acid leucine. But there are two types of codons, start codons and stop codons that do not code for any amino acid. A gene has a start codon as the beginning and a stop codon as the end. When translation starts, the protein synthesis begins at the AUG RNA codon and the following amino acids are determined by the codons presented in the gene until the stop codon in mRNA is reached. Then the completed protein leaves the ribosome. The complete set of genes and non-coding sequences of DNA in an organism or a cell is known as the genome.

2.1.3 Gene Structures of Prokaryotes and Eukaryotes

In biology, there are two major groups of organisms: prokaryotes and eukaryotes. They both have DNA, ribosomes and a similar basic metabolism. However, a prokaryotic cell contains no nucleus, while a eukaryotic cell has a nucleus and other organelles. In prokaryotic cells, most of the DNA sequence codes for proteins, which means there

are no introns in the coding regions. Introns are regions of DNA that will not be translated into proteins. But eukaryotic cells have introns that split the coding regions into pieces. Hence, gene prediction for the two groups of cells must be different.

Prokaryotes are divided into the bacteria and archaea, and have relatively small genomes, with sizes ranging from 0.5 to 10 Mbp. The gene density in their genomes is very high; 90% of a genome sequence contains coding regions. In prokaryotes, DNA is just organized in a single loop. As it is shown in Figure 2.1, a typical prokaryotic gene has a consensus motif of AGGAGGT (Ribosome Binding Site), and a contiguous stretch of the coding region with a start codon and a stop codon. One unique feature

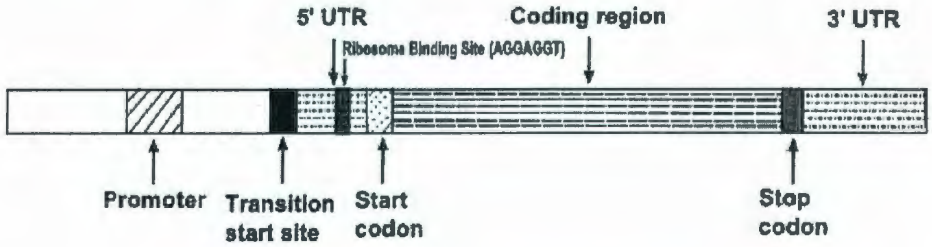

Figure 2.1: Gene Structure of Prokaryotes

of prokaryotes is the operon, which is a set of genes with similar functions [10].

Animals, plants, fungi, and protists are eukaryotes, however. The gene structure and expression of eukaryotic cells is much more complicated. A eukaryotic cell has a defined nucleus and is bound by a membrane in which chromosomes are present within DNA. Eukaryotic genomes are much larger than prokaryotic genomes. Their sizes range from 10Mbp to 670Gbp, but the gene density is generally very low. For example, only 3% of human genomes code for genes. Generally the regions of DNA for encoding proteins in eukaryotic cells are not continuous. These regions are composed

alternatively of exons and introns, as shown in Figure 2.3. Exons are the regions with

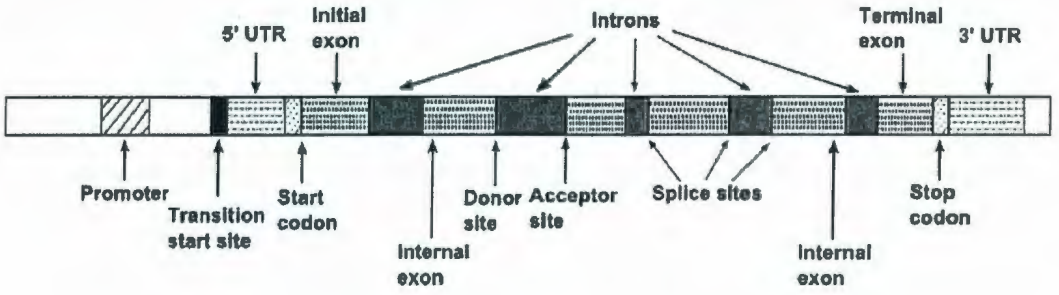

Figure 2.2: Gene Structure of Eukaryotes

codes that will be translated into proteins. Exons and introns are all transcribed into RNA in order. A splicing process will occur in which introns and exons are separated. Introns are discarded, while exons remain to be part of the mature RNA sequences. The sites at which the splicing occurs are called splice sites. The structure of a typical multi-exon gene at the DNA level begins at a promoter region followed by the 5' untranslated region (5' UTR), which is a transcribed non-coding region. The next segment is the initial exon with a start codon. Then internal exons and introns present alternatively and the terminal exon follows, containing a stop codon. A final non-coding region 3' UTR comes after the terminal exon. The end of the eukaryotic gene is a polyadenylation (polyA) signal.

2.2 Methods for Predicting Genes

Scientists are looking forward to discovering the secret of life and the modern DNA sequencing technology allows researchers to explore this area. Finding genes is one of the most important steps in understanding genomes. Gene prediction refers to

identifying the stretches of genomic sequences that carry genetic information in DNA. Prokaryotes are single-celled organisms. Compared to eukaryotes, finding genes in prokaryotes is easy since their gene structure is simpler. Usually Interpolated Markov models (IMM) are applied for prokaryotic gene prediction. However, eukaryotic gene prediction is far more difficult because of the low coding density of genomes and the presence of introns in RNA. Genes are represented by a collection of substrings, which are sections of contiguous triplets. This poses the problem of predicting the locations of genes in a genome given only the genomic DNA sequence [15]. Based on the difference between prokaryotes and eukaryotes, this review of gene prediction methods is only regarding the methods for finding eukaryotic genes.

To date, the methods for predicting genes mainly fall into two categories: ab initio gene finding and sequence similarity approaches. The first method predicts the gene structures by the knowledge of genes, while the other method looks for the similar sequences by searching the databases.

2.2.1 Ab initio Gene Prediction

The ab initio gene prediction approach uses the known gene structure as a template to detect genes in DNA sequences. This method usually makes use of two types of patterns: signal sensors and content sensors. Signal sensors attempt to identify the short sequence motifs such as promoters, splice sites, polyA signals, start codons and stop codons. Content sensors detect the content of the sequences like exons and CpG islands, in other words, they try to distinguish coding from non-coding regions. The most important and well-studied content sensor predicts coding regions [16]. To build

gene structure models, many algorithms are applied, such like dynamic programming, Hidden Markov models (HMMs), decision trees, neural networks and linear discriminant analysis (LDA). Dynamic programming and Hidden Markov models are the most widely used.

Dynamic Programming

Dynamic programming is the most fundamental method in bioinformatics. Sometimes a problem will be split into subproblems and the solutions of the subproblems will be combined to obtain the final solution. However, the subproblems may be very large and the same subproblem may have to be solved repeatedly, which increases the running time. Dynamic programming organizes the computations so as to avoid recomputation [15].

In the context of gene prediction, dynamic programming is always used to combine the submodels built for sequence features to produce an optimal solution. These submodels are constructed by other machine learning approaches. For example, many programs like GeneMark.hmm [17] using HMMs have the sub-HMMs for each type of signal and content like splice sites and donors. Then by being given the rules of the known gene structures, dynamic programming is applied to integrate these submodels into a model of the whole sequence and determine the highest scoring structure of the query sequence. Also, dynamic programming algorithms are combined with other methods such as neural network (GeneParser) and position weight arrays (PWAs). Presentative programs are GeneParser [18, 19] and GeneID [20, 21].

In gene prediction, dynamic programming is also the basic method to sequence

alignment which determines the optimal alignment by establishing a two-dimensional matrix. The matrix with each aligned sequence on one dimension is filled with the scoring scheme. By searching the highest score in this matrix, the dynamic programming algorithm finds the best alignment between the pair of sequences. Two examples are ClustalW [22] and SIM [23, 24]. The former produces both local and global multiple alignments, while the latter is a local pairwise alignment program.

Hidden Markov Models (HMMs)

Biological sequences can be modeled as the output of a stochastic process. In this process, the probability for a given nucleotide t occurring at position p depends on the nucleotide occupying k previous positions. This is called k-order Markov model [25]. A HMM is an extension of the Markov chain with the states of the model being hidden. Generally, a HMM is a statistical model whose input is a sequence of states and whose outputs are random sequences after a stochastic transition. Since only the output is shown but states in the process are invisible to an external observer, it is called "hidden". A HMM has a set of finite states X, initial state probabilities π associated with each state, transition probabilities A between states, output probabilities B of the observation symbols in states and a set of observable outputs Y.

There are three basic problems in HMMs. The first one is how to efficiently compute the observation probability P(O|λ), which O is an observation sequence and λ is the initial state probability distribution. The second one is how to choose an optimal corresponding state path $Q = q_1 q_2 \ldots q_T$ for the observation sequence. The third one is how to adjust the parameters of the model such that $P(O \mid \lambda)$ is the maximum.

Regarding these three problems, the forward algorithm, the Viterbi algorithm and the classic Baum-Welch algorithm are correspondingly applied to HMMs [26].

Although HMMs are popular in speech recognition, HMMs are now also a fundamental method for biological sequence analysis in molecular biology. For gene prediction, HMMs will be constructed based on the current knowledge of DNA sequences; this knowledge is used for estimating parameters of the models. Then by using training data, models are trained iteratively and parameters obtain new values each time. Models will be updated with new parameters until the likelihood of the training data is maximized. Afterwards, these models are used to find the optimal result for a query sequence, which means that the query sequence will be scored to show how well it matches these models. Each state of a HMM emits a set of elements and in gene prediction, the elements are the four bases A, T, G and C if the model is based on nucleotide sequences. When one state changes to another, there is a transition probability. The emission of each element in each state also has a emission probability (or output probability). In the final step, all of the transition and emission probabilities for each possible path are considered to calculate a total probability for the path. To build accurate models, submodels of distinct features are separately constructed by HMMs and integrated into one model of genes by statistical approaches. Currently one of the best programs is GenScan [27], which uses a semi-Markov HMM [11]. The HMM structure of GenScan (Figure 2.4) can explain the process of HMMS in gene prediction which has been given above. Other applications include HMMGene [28], GeneMark [29], Glimmer [30], GENIE [31], VEIL [32] and some other programs.

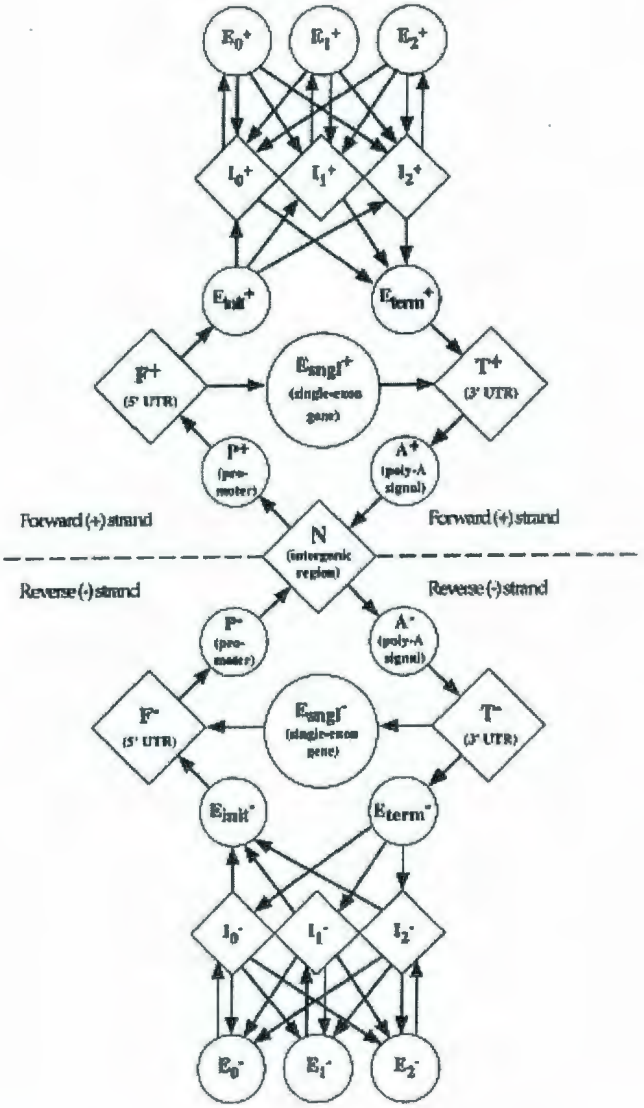

Figure 2.3: HMM Structure of GenScan [7]

Neural networks, Decision trees and Linear Discriminant Analysis (LDA)

GRAIL [33] is a gene prediction program that adopts the neural network. A neural network (or artificial neural network) [34, 10], which resembles the biological nervous system, is a machine learning method that is used in pattern recognition. It involves a large number of processing units, each of which has its own active rules and the rules act on the inputs to decide the outputs. There are certain weighted inconnections between these units that make all the units act as a network. A learning rule controls how to adjust the weights for the inputs/outputs from feedback in the training process. In general, neural networks have several layers, and all layers are hidden except the input and output layers. In gene prediction, usually the inputs are the genomic sequences with contents and signals and the outputs are the predicted exon or coding sequence structures. Neural networks will be trained by feeding the data of the known genomic sequences, such as the data of CpG islands, splice sites and promoter regions. In this process, the hidden layers will adjust their weights and the model can "learn" nucleotide patterns. When an unknown sequence is sent to the trained neural network, the model applies rules learned from the training data to predict the sequence.

A decision tree is introduced to solve the simpler problem of discriminating coding and non-coding DNA. The internal nodes of the tree are property values that are tested for each subsequence passed to the tree. The leaf nodes of the tree contain class labels to be finally associated with the subsequences. MORGAN [35] and OC1 [36] are programs that use decision trees.

LDA is a standard technique in multivariate analysis that can be used to linearly combine several measures in order to perform the best discrimination between two

functional classes of sequences. It can also serve to identify the most significant measure for a given discrimination problem. GeneFinder [37, 38] and FGENEH [39] use LDA.

2.2.2 Sequence Similarity

Researchers have already sequenced and annotated genomes of many species and thus a number of genome databases have been established. Accordingly, sequence similarity search is a method that compares query sequences with database entries to detect regions which are significantly same with or similar to the known sequences. Basically sequence similarity can be used in several ways [40]. First, comparing a genomic sequence with the databases of expressed sequence tags (ESTs) can identify the regions that may be transcribed to mRNA. BLASTN [41] is an example of a program used for this purpose. Second, the probable coding regions can be observed by comparing a genomic sequence with the databases of protein sequences. BLASTX [41] uses this approach. Third, alignment of the candidate coding regions of a genomic sequence with a homologous protein sequence may obtain more accurate results. PROCRUSTES [42] is this type of a program. Fourth, the peptides predicted by some programs can be compared with the protein sequence databases to verify these results. Fifth, comparison of the translated genomic sequences with databases of translated sequences or with cDNA databases can locate the similarities among the coding regions, like the program TBLASTX [42]. At last, genomic sequences can be aligned to homologous genomic sequence databases derived from closely related organisms. This can be very helpful to detect conserved regions which indicate impor-

tant coding functional elements in sequences. For example, human genome sequences can be compared to the mouse or dog annotated sequence databases. This method is called comparative analysis and is used by the programs such as CLUSTALW [22], AGenDA [43], SGP2 [1] and VISTA [44].

Conclusion

After the completion of the human genome project, people are eager to decipher the code of human life. Gene prediction is one of the most critical steps in achieving this goal. Ab initio approaches and sequence similarity are the two major methods in this research area. Ab initio approaches usually use machine learning algorithms to build models according to the sequences that have been annotated, and then search for the most probable genomic regions in unknown sequences. Sequence similarity approaches identify genes by aligning the unknown sequences to established databases with annotated data of genomic features. Each of these methods has its advantages and disadvantages. Ab initio approaches are helpful to find some genes that have never been detected before, while sequence similarity can assure with high probability the finding of known genes. Nevertheless, knowledge about the structure of DNA, genes and even of RNA is limited at present, and thus the models built by ab initio approaches have limitations, which may make these models miss some undiscovered but existing genes. For sequence similarity, the uncompleted databases restrict the accuracy of identifying genes as the databases depend on how much knowledge has been obtained already.

Chapter 3

Cross-species Comparative Sequence Analysis and Two Applications

With the availability of genome databases for a growing number of species, cross-species comparative sequence analysis has a strongly potential for helping predict genes. In order to find more unknown genes and genetic information in the human genome, our research considers a module applying cross-species comparative sequence analysis. In this chapter, we describe this method and outline the general structure of cross-species comparative gene finders. For further study of this method, we have chosen two cross-species comparative gene finders: AGenDA and SGP2, and compared them in terms of their architectures, application and performance. In this study, we have then selected SGP2 for the next step of our work.

3.1 Cross-species Comparative Sequence Analysis

In chapter 2, sequence similarity approaches were introduced. One of the ways to apply this class of approaches is comparative sequence analysis. The idea behind the cross-species comparative analysis method is that most of the functional elements of genomes are conserved during evolution, and thus the similarities of sequences from evolutionarily related species will indicate the biological function of genes [45]. Especially the regions of strong sequence conservation can be suspected to correspond to protein-coding exons. In addition, comparative sequence analysis is helpful to detect conserved non-coding regions that has become a hot research topic recently [46].

Cross-species comparative sequence analysis is a powerful method for predicting genes because functional sequences are biologically selected to remain conserved throughout evolution. The method compares sequences from different organisms to identify putative homologous genes and to generate a phylogenetic tree. By comparing genome sequences from different organisms, conserved sequences can be identified. Potential regulatory functions will be detected as well after analyzing the conserved sequences. Evolutionary distance is the scale of this method: the closer organisms are, the more similarities their sequences will have. Distantly related organisms like yeast and fish have less sequence similarity and show conservation in coding regions alone. Whereas more closely related organisms are likely to exhibit conserved coding regions and also other functional elements like regulatory sequences.

The gene finders predicting genes by cross-species comparative sequence analysis have a general architecture described as follows. First of all, the programs align the sequences from two or more species. For better accuracy, they usually consider

both, local alignment and global alignment. Global alignment gives an overview of the entire sequence while local alignment is more specific to highly conserved regions. Afterwards, an algorithm is applied to assemble the evidence obtained from alignment so that a gene model is constructed. The most used algorithm in this step is dynamic programming [47, 48, 49]. The dynamic programming algorithm is applied to find optimal chains of candidate genes with maximum scores. SGP2 however is an exception to the general structure. It combines the alignment of two sequences with the predictions from GeneID, which is an ab initio gene finder. Still sequence alignment is an essential step of cross-species comparative gene finders.

For predicting genes by cross-species comparative sequence analysis, an appropriate selection of species is critical. Comparison of two species that diverged 40-80 million years ago from a common ancestor reveals conservation in both coding regions and a significant number of non-coding regions [47]. Comparison of human with mouse and two species of fruitflies (Drosophila melanogaster with Drosophila pseudoobscura) are good examples. Furthermore, most protein coding sequences evolve slowly since they are responsible for retaining function. Therefore, the ability to detect conserved elements (both coding and non-coding) could be enhanced by comparing the orthologous DNA sequences of two species over a large phylogenetic distance, such as human and pufferfish that diverged 450 million years ago [50]. In addition, comparison of closely related species, such as human and chimpanzee, can not only identify conserved regions, but also find some unique genes related to certain species. This is beneficial for understanding the evolutionary process of these species. Zhang et al.'s paper [51] validates that evolutionary distance can decide the performance of gene prediction; a well-chosen distance can provide an optimal accuracy.

Their experiment also exhibits that evolutionary distance beyond that of human and mouse gives better performance even though the tested mouse sequence falls in the area with high accuracy.

3.2 AGenDA and SGP2

3.2.1 AGenDA

AGenDA (Alignment-based Gene-Detection Algorithm) [2], which is developed by Rinner and Morgenstern [43], is a tool using cross-species comparative sequence analysis. Based on pair-wise human/mouse alignments created by CHAOS [52] and DIALIGN [53], the gene finder searches for conserved splice sites around local sequence similarities to identify candidate exons and constructs a complete gene model from the candidate exons [45].

Architecture and Methods

AGenDA has five main steps as shown in Figure 3.1. In the first step, RepeatMasker [54] is applied to the input sequences to mask interspersed repeats and low complexity DNA sequences that are enriched for single amino acids. The second step is to obtain a chain of high-homology regions by using CHAOS. Local alignments from CHAOS will be used as anchor points to reduce the search space and execution time for the following steps and can fasten the global alignment considerately. Subsequently DIALIGN calculates an alignment of the input sequences based on anchor points created by CHAOS. This alignment program integrates local and global

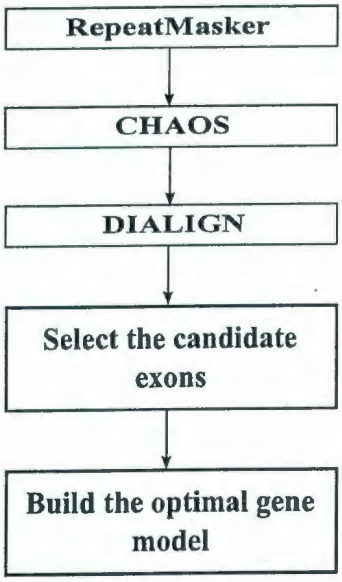

Figure 3.1: The Structure of AGenDA

features by assembling pair-wise and multiple alignments from gap-free local segment alignments (called "fragments") [43]. Each possible fragment is given a weight score [55] and the alignment, which has a collection of fragments with maximum total weight score, is chosen. Next, a list of candidate exons is obtained according to the fragments with highest scores identified by DIALIGN in the sequences. Each exon is exactly one Open Reading Frame (ORF). To find the optimal chain of candidate genes, each candidate exon is given a score defined as:

$$sc(E) = \frac{len_C - ND}{len_C} \sum_i w(f_i) + sc_{SP} \tag{3.1}$$

where len_C is the length of a cluster C of fragments $f_1, \ldots, f_k$ from which a candidate exon E is derived; ND is the length of the discrepancy regions between the cluster C and the exon E; $w(f_i)$ is the weight score of the fragment f_i; sc_{SP} is the score of the splice signals where E is bounded by.

At last, an optimal gene model with the maximum total score is created from the list of candidate exons by using the fragment-chaining algorithm [56]. This algorithm is a straight-forward dynamic programming procedure.

Experiments

To test AGenDA, a dataset with 117 pairs of genomic sequences is used [43]. It has been collected by Batzoglou et al. [49] with the sequences are from human and mouse. 12 sequences are used for training in order to optimize parameters of the gene finder and the remaining 105 sequences are used for testing. The performance of AGenDA on this dataset is compared with GenScan at the exon level. The sensitivity of AGenDA is 0.76 while GenScan's sensitivity is 0.82. The specificity of AGenDA is 0.78 and GenScan's specificity is 0.77. Aside from these two parameters, it is observed that AGenDA found an additional 12% of the annotated exons which were not identified by GenScan. But GenScan predicted 17% of unannotated exons which were not detected by AGenDA. In general, AGenDA has comparable performance to GenScan on predicting human and mouse sequences. However, the limitation of AGenDA is that its performance on predicting other species is not as good as the performance on human and mouse, because the parameters of AGenDA have been optimized for sequences of primates and rodents.

3.2.2 SGP2

SGP2 is a gene finder to predict genes by comparing the genomic sequences from two species. It integrates with the sequence similarity search program TBLASTX (WU-

BLAST) [57] and the ab initio gene finder GeneID [58]. In this gene finder, the query sequence is compared with a single sequence or a collection of sequences from other species and a final alignment is obtained. Then the alignment is applied to modify the scores of predicted exons from the ab initio gene finder. The Mouse Genome Sequencing Consortium 2002 has applied SGP2 to annotate the mouse genome and SGP2 also has been applied to identify previously unconfirmed genes [59].

Architecture and Methods

GeneID is one of the first programs predicting the full exonic structure of vertebrates. Its process is composed of the following steps. The first step predicts and scores splice sites, start and stop codons along the query sequence by applying position weight matrices (PWMs). The PWM is inferred from a training dataset for scoring the potential sites of the query sequence. In the second step, exons are built up by all the predicted sites. GeneID considers four types of exons: initial exon including a start codon and a donor site; internal exon including an acceptor site and a donor site; terminal exon including an acceptor site and a stop codon; single exon including a start codon and a stop codon. To score predicted exons accurately, a fifth-order Markov model is built by extracting the exon and intron sequences of the training set and then a log-likelihood ratio will be defined for coding sequences. Exons are therefore scored by summing up the scores of all sites and the log-likelihood ratio from the Markov model. The final step assembles all the exons into the whole gene structure by maximizing the sum of scores of exons in the whole set. A dynamic programming chaining algorithm [60] is used in GeneID to choose the gene structure

with maximum score $L_G(g)$ which is interpreted as:

$$L_G(g) = L_E^*(e_1) + L_E^*(e_2) + L_E^*(e_3) + \ldots + L_E^*(e_n) \tag{3.2}$$

where $L_E^*(e_i)$ is the score of the exon i.

By using TBLASTX, the query sequence is compared against one sequence or a collection of sequences from a distinct species to look for the counterpart homologous exon in the reference sequences at the amino acid level. TBLASTX provides an optimal alignment and gives a corresponding score. A substitution matrix, like BLOSUM matrix, is used to score the alignment. The score can be assumed to be a log-likelihood ratio. The optimal alignment is considered as the high-scoring segment pairs (HSP) with the maximal score. However, in the case of the collection of reference sequences which are fragmentary and irregular, the query exon sequence may have different optimal alignments corresponding to different sequences of the set. To solve this problem, SGP2 derives a score $s_t(e)$ from a set of HSPs covering each fraction of the candidate exon in the way described as follow. First, a set of HSPs $h_1 \ldots h_n$ found by TBLASTX are projected onto the query sequence and thus the query sequence is partitioned into pieces of segments as $x_1 \ldots x_n$ which lengths are exactly same with those of HSPs. The segments are scored as $s_p(x_1) \ldots s_p(x_n)$. For each predicted exon in the sequence e, a set of maximal scoring segments X_e overlapping the exon will be found. Hence the final score $s_t(e)$ is computed as:

$$s_t(e) = \sum_{x \in X_e} s_p(x) \frac{\|x \cap e\|}{\|x\|} \tag{3.3}$$

where $\|a\|$ denotes the length of the segment a. In this way, each predicted exon can include the maximally scoring HSPs which are from those HSPs overlapping the exon.

In the final step of SGP2, the final score of the exon s(e) is not just summing the scores up like $s_g(e) + s_t(e)$, where $s_g(e)$ is the score from GeneID. As $s_g(e)$ depends on the probability of sequence of e if it codes for the proteins, while $s_t(e)$ depends on the optimal alignment in which it is assumed that both query and reference sequences code for the proteins. The two parameters are not independent. Therefore, an "ad hoc" coefficient w is applied to weight the contribution of $s_t(e)$ and the final score of the exon is computed as $s_g(e) + ws_t(e)$.

Experiments

SGP2 has been tested on two data sets [1]. For optimizing some parameters of SGP2, the dataset is adapted from Jareborg et al.'s dataset [61]. 33 pairs of mouse and human sequences in the original dataset are collected. They code for single complete genes. In addition, six pairs of human and mouse sequences are added from the SWISSPROT database. So this training dataset has 39 pairs of sequences in total. Parts of the dataset are used for testing as well and this data set is called SCIMOG. Another testing dataset, which is called the MIT data set by SGP2's authors, is derived from the dataset used by AGenDA. It is constructed by Batzoglou et al. [49]. This dataset has 117 orthologous human and mouse genes. But the pairs of sequences with multiple genes have been discarded. Also, those pairs in which the coding regions start at position 1 in either sequence of the pairs have not been considered. Therefore, the final dataset used by SGP2 has 110 genes. Some overlap between the first and the second dataset have remained. Human and mouse databases are built from both datasets. SGP2 has been compared with GenScan, ROSETTA

and TBLASTX. Table 3.1 and 3.2 show the comparisons on the first and second datasets. They are compared at nucleotide and exon levels. For the first dataset, the nucleotide sensitivity is close to GenScan's sensitivity while its specificity reaches 0.97, which is higher than GenScan. Compared to TBLASTX, SGP2 has a significantly better performance. At exon level, the sensitivity is slightly lower than GenScan and TBLASTX, but the specificity is still better than the other two. As for the prediction on the second dataset, it has a better performance at nucleotide level than at exon level. At the nucleotide level its sensitivity is close to GenScan but it has a good

Table 3.1: Performance Comparison of the Performance on the SCIMOG Data Set [1]

	Nucleotide-Level		Exon-Level			
Programs	Sn	Sp	Sn	Sp	ME	WE
SGP2	0.94	0.97	0.80	0.87	0.10	0.02
GenScan	0.98	0.86	0.84	0.75	0.04	0.14
TBLASTX	0.89	0.76	0.82	-	0.19	0.11

specificity which is higher than GenScan. Plus, its performance is approaching that of ROSETTA. At exon level, the sensitivity of SGP2 is much lower than GenScan and the specificity is somewhat higher.

For both datasets, SGP2 has the lowest WE, but the ME is relatively high. All in all, the experiments on these two datasets present that SGP2 has a comparable performance with GenScan and ROSETTA.

Also, SGP2 has been tested on human chromosome 22 which is probably the best annotated human chromosome [1]. The database used in this test is the masked

Table 3.2: Performance Comparison of the Performance on the MIT Data Set [1]

	Nucleotide-Level		Exon-Level			
Programs	Sn	Sp	Sn	Sp	ME	WE
SGP2	0.96	0.97	0.71	0.79	0.12	0.03
GenScan	0.98	0.89	0.82	0.75	0.06	0.13
TBLASTX	0.94	0.79	-	-	0.13	0.13
ROSETTA	0.95	0.97	-	-	0.02	0.03

whole genome of the mouse genome (MGSCv3). Results of SGP2 are compared with GenScan , GenomeScan and GENEID at the nucleotide, exon and gene levels. The comparison is shown in Table 3.3. At both nucleotide and exon levels, GenomeScan

Table 3.3: Performance Comparison of the Performance on Human Chromosome 22 [1]

	Nucleotide-Level		Exon-Level				Gene-Level			
Programs	Sn	Sp	Sn	Sp	ME	WE	Sn	Sp	MG	WG
SGP2	0.83	0.67	0.68	0.56	0.16	0.31	0.13	0.10	0.36	1.14
GenScan	0.86	0.50	0.70	0.40	0.13	0.50	0.06	0.04	0.11	0.45
GenomeScan	0.87	0.44	0.72	0.36	0.10	0.55	0.11	0.06	0.12	0.52
GeneID	0.80	0.63	0.66	0.53	0.19	0.35	0.09	0.07	0.14	0.39

has the highest sensitivities and GenScan is inferior to it, while SGP2 is in between. But SGP2 is more specific than the other three programs and GenomeScan is the least specific. At gene level, the performance of SGP2 is outstanding. Whereas GenScan

is lowest on both sensitivity and specificity.

3.2.3 Comparison of AGenDA and SGP2

After having given an overview about the structures and experiments of AGenDA and SGP2 respectively, we comprehensively analyzed the two gene finders in detail, showing their differences and common features in terms of the architectures, applications and performance.

Comparison on the Architectures

The similarity between the architectures of AGenDA and SGP2 is that they both need alignments of sequences. However, AGenDA uses dynamic programming to calculate the maximum score of candidate exons from the alignment for obtaining an optimal chain of them, while SGP2 assembles the optimal alignment and the optimal gene structure from GeneID to attain the final prediction. Therefore, for AGenDA, its prediction mostly depends on the performance of the alignment, but SGP2's prediction of genes is based on both cross-species comparative sequence analysis and ab initio gene prediction. With respect to aligning sequences, AGenDA considers local and global alignments because of DIALIGN, while SGP2 just considers local alignment (TBLASTX is a local alignment program).

In other words, AGenDA has a general architecture for cross-species comparative analysis, and SGP2 has an architecture which uses combination methods.

Comparison on the Application and Performance

AGenDA is a web-based gene finder. After submitting sequence files of two species and selecting parameters, the prediction file will be generated. In the result files, the numbers of predicted genes and exons are given. Also information on predicted exons is listed, such as the position of beginning and ending, the score, the strand and the reading frame. As for SGP2, after running TBLASTX separately, the alignment is collected and the query sequence is fed to SGP2. SGP2 outputs each predicted gene and information on the predicted exons belonging to the gene. This information includes the types of exons (first, internal, terminal and single), the beginning and ending positions, the score, the strand and the reading frame. For application, the deficiency of AGenDA is that the size of the data sets are moderate because DIALIGN is slower than alternative programs for alignment. Only sequences up to 200kb in length are acceptable.

Both programs have run on the data set created by Batzoglou et al. [49]. In the testing of SGP2, this data set is called the MIT data set. Table 3.4 lists the sensitivity and specificity of the two programs at exon level. On this data set, AGenDA shows

Table 3.4: Performance Comparison on the MIT data set [1, 2]

Programs	SGP2		AGenDA	
Exon-	Sn	Sp	Sn	Sp
Level	0.72	0.79	0.76	0.78

a better sensitivity to exons, while SGP2 is slightly more specific for exons than AGenDA. Their performance on this data set is pretty close.

Discussion

AGenDA is a typical gene finder using cross-species comparative sequence analysis, whereas SGP2 combines the alignment of different species with the result of an ab initio gene finder. Ab initio approaches can provide prediction based on a whole gene structure. Therefore, SGP2 may take advantage of the ab initio gene finding methods to make up for what cross-species comparative analysis is not good at.

Regarding application and performance of the two gene finders, although AGenDA has a similar performance to SGP2 on that data set, AGenDA may have the best performance at predicting genes just depending on comparison of Homo sapiens and Mus musculus, because its parameters are optimized for human and mouse. Furthermore, the maximum length of input sequences is a limitation for AGenDA, that restricts AGenDA not to predict genes of long DNA sequences. If long sequences of DNA are split into smaller pieces, some critical genetic information may be missed. In comparing the result files of AGenDA and SGP2, SGP2 provides more detailed information than AGenDA. Additionally, SGP2 has been tested on human chromosome 22 and had an acceptable performance on this dataset. In conclusion, SGP2 can supply satisfying prediction of genes by comparing sequences of different species.

Conclusion

Our study aims at discovering some novel genes and genetic information of humans and the cross-species comparative sequence analysis shows the ability to explore some undisclosed areas of human DNA sequences. Because of evolution, present-day creatures are more and less genetically similar; important genetic information is con-

served between different species. Based on this theory, the cross-species comparative sequence analysis has been presented and applied.

The essence of the comparative sequence analysis is sequence alignment which includes local alignment and global alignment. Cross-species comparative gene finders usually first align sequences and subsequently apply an algorithm to predicts genes. The commonly used algorithm is dynamic programming. AGenDA has a structure like this and it makes use of both local alignment and global alignment. Whereas SGP2 considers both predictions from comparative sequence analysis and ab initio gene finder GeneID, and it uses TBLASTX, a local alignment sequence tool. We have conducted the study of AGenDA and SGP2. AGenDA's experiment on the MIT data set shows that its performance on this data set is close to GenScan. Experiments of SGP2 show that its predictions on the two data sets, SCIMOG and MIT, are competitive to the results from GenScan and better than the results from TBLASTX. It indicates that SGP2 reaches the performance of the ab initio gene finders and its performance is better than the gene finders that just use sequence alignment. We choose SGP2 as the cross-species comparative sequence analysis module of our work.

Chapter 4

Combination Methods and Three Applications

Since ab initio and sequence similarity approaches both have some disadvantages limiting detecting novel genetic information, combination methods have been brought out. In this chapter, we give an overview of combination methods. Two types of the methods are described by giving some examples. Then our work on three gene finders is presented. We choose GAZE [3], JIGSAW [5] and ExonHunter [6] because they are currently outstanding research works on combining a variety of evidence to predict overall gene structures. We introduce and compare these gene finders based on their architectures, applications and performance.

4.1 Overview of Combination Methods

Ab initio gene finding approaches are restricted by the knowledge of DNA and RNA sequences or genetic sequences, while sequence similarity approaches depend on genes

which have already been identified. Therefore, some unknown or novel genes would be missed when the two methods are applied independently. In addition, some of programs put more efforts in predicting certain genetic features, while other programs can provide more accurate predictions on other features. Therefore, combination methods have been proposed to integrate the evidence predicted by two or more programs so that more precise gene structures can be built and thus more accurate predictions can be generated.

In general, combination methods could be mainly classified into two types: one is to parse the outputs/evidence from different programs into a gene model by a statistic approach; another type is to just combine the outputs/evidence from different programs by applying the rules proposed by Katsuhiko Murakami and Toshihisa Takagi [8]. The first type usually builds a gene model by a statistic method and then the outputs/eviendence from some programs are complementary to the gene model. HMMs, dynamic programming, neural networks, decision trees, Bayesian networks or other algorithms is applied to integrate evidence. The work presented in Stanke et al.'s paper [62] uses a Generalized Hidden Markov Model (GHMM) to integrate the external evidence into a probabilistic model. Usually a GHMM for gene prediction defines a joint probability P(ϕ,s) for each pair of a DNA sequence s and a gene structure ϕ. In this paper, a piece of external evidence of the DNA sequence s, called *hint*, is taken into account to extend the GHMM. There are six types of hints, which stand for the translation start site, stop codon, the donor splice site, the acceptor splice site, the coding region and exon. Each hint is associated with each position i in the DNA sequence s. Thereby, the joint probability is extended to P(ϕ,s,h)which

is specified as:

$$P(\phi, s, h) = P(\phi, s) * \prod_{\substack{1 \le i \le |s| \\ t \in TYPE}} P(h_{i,t}|\phi, s) \tag{4.1}$$

$P(h_{i,t}|\phi, s)$ is a conditional probability and defined by three terms: $P(h_{i,t}|\phi, s)$ is $q^+(t, g)$ when the hint is compatible with the gene structure ϕ; the probability is $q^-(t, g)$ when the hint is compatible with the DNA sequence s but not the gene structure ϕ; the probability is zero when the hint is not compatible with both of them. The conditional probabilities $q^+(t, g)$ and $q^-(t, g)$ (s is the type of the hint and g is its grade) are estimated from the training data. The hints are collected from EST and protein databases and generated by the program AGRIPPA which extends the local alignment search tool WU-BLAST. It uses a protein sequence database and an EST database to infer hints about the coding regions in the input DNA sequence. Resembling to this work, some other works, such as a Bayesian network framework [63], ExonHunter [6], JIGSAW [9], GAZE [3] and EuGene [64], combine the evidence from other programs to build gene models.

Regarding the second type of combination methods, the programs parse and combine the output data from other programs by applying certain rules. Originally K. Murakami and T. Takagi proposed the following five methods shown in Figure 4.1:

1. AND-based method: only the overlapped exons, which are predicted by all the programs, are chosen.

2. OR-based method: the exons predicted by at least one program are chosen.

3. HIGHEST-based method: the exons having the highest probability scores among all the programs are considered.

4. RULE-based method: according to the performance of a data set, the exons,

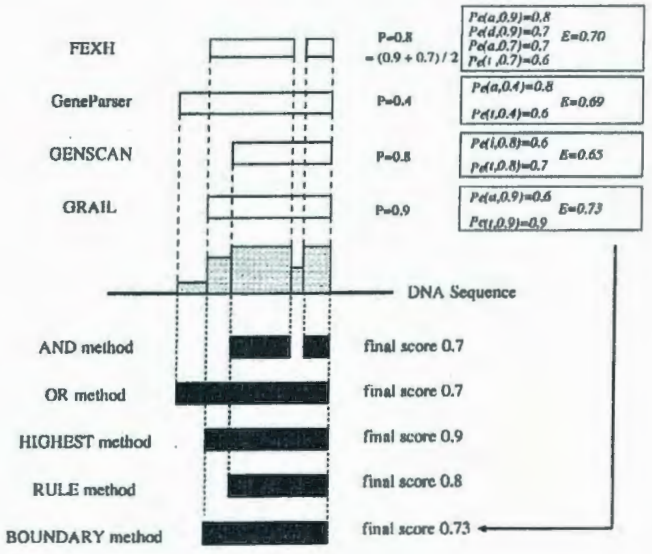

Figure 4.1: An Example of the Five Methods proposed by K. Murakami et al. [8]

which are predicted by the program with the best accuracy, have the highest priority among those predicted by other programs.

5. BOUNDARY method: for each predicted exon in the cluster(one or more exons predicted), a probability of each boundary will be derived from the training data and then a new score E of the exon will be calculated as the Nth root of the multiplication of all the probabilities of the boundaries. The exons with the highest new scores E will be the candidates. .

The gene finders EGPred [65] and GeneComber [66] are this type of programs. In EGPred, there are five steps. Initially query sequence is searched against the RefSeq protein database with an expectation value (E-value) < 1 by BLASTX. Then the second BLASTX search is to retrieve all the possible coding regions with relaxed parameters (E-value < 10) against the proteins found in the first BLASTX search. The third BLASTX search is to detect the intron regions of the query sequence against

the intron database [67]. All the obtained exons and introns are compared to filter the exons: the exons with 40% or more overlapping with the introns will be removed. Subsequently, a web-based tool NNSPLICE [68] is used to detect the start/stop codon and splice sites in the remaining candidate exons. Finally the prediction from GenScan or HMMgene are combined with the candidate exons from the similarity search. The exons from the ab initio program and the similarity search are classified into five groups for combination. The rule is that the boundaries of the exons predicted by the ab initio programs will be modified only when the evidence from BLASTX or NNSPLICE are higher than a cutoff threshold.

Another program, GeneComber, integrates two ab initio programs GenScan and HMMgene by using AND-based and OR-based rules. Three algorithms are developed: EUI (Exon Union-Intersection), GI (Gene Intersection) and EUI_frame (Exon Union-Intersection with Reading Frame Consistency). When the exon probability scores of GenScan and HMMgene are all higher than or equal to a threshold p_{th}, EUI considers the regions predicted by at least one program. Otherwise, EUI considers the overlapped regions when the probability scores are all lower than the threshold. The optimal value 0.775 is selected as the threshold by the experimental experience based on the data set in their study. Concerning GI, the candidate genes from both programs are all considered. The EUI method is used to filter the exons which belong to each GI gene. EUI_frame method is similar to the EUI method but also maintains reading frame consistency. Each gene is assigned a probability score which is the average of the scores for the exons belonging to the gene. The acceptor and donor sites are determined for each exon in the reading frame of the gene. If the genes from both programs are overlapped, the higher probability one is chosen to impose the

reading frame. EUI method is applied to filter the exons which are in the determined reading frame. This study is tested on the data set HMR195, the Burset/Guigo data set [12] and a Drosophila melanogaster Adh region [69]. Compared to the results of GenScan and HMMgene, the specificities of the three methods are improved more than the sensitivities at both nucleotide and exon levels. The test also demonstrates that EUI method would be best applied to short sequences, while GI and EUI_frame methods are more suitable for long sequences.

4.2 GAZE, JIGSAW and ExonHunter

4.2.1 Review of The Three Gene Finders

GAZE

GAZE provides a framework for the external evidence of signal sensors and content sensors. It does not work directly on gene sequences, but predicts a gene structure from input files with results being the scored signals and content. By using a dynamic programming algorithm, evidence is assembled and the highest score is obtained according to a configuration file. This file specifies how features (signal sensors) and segments (content sensors) are combined into a complete gene structure shown as Figure 4.2.

All input files of the evidence to GAZE have to be in the General Feature Format (GFF) which is an exchange format for feature description [70]. The configuration file, which is in XML format, directs how to read the lines from the GFF files and

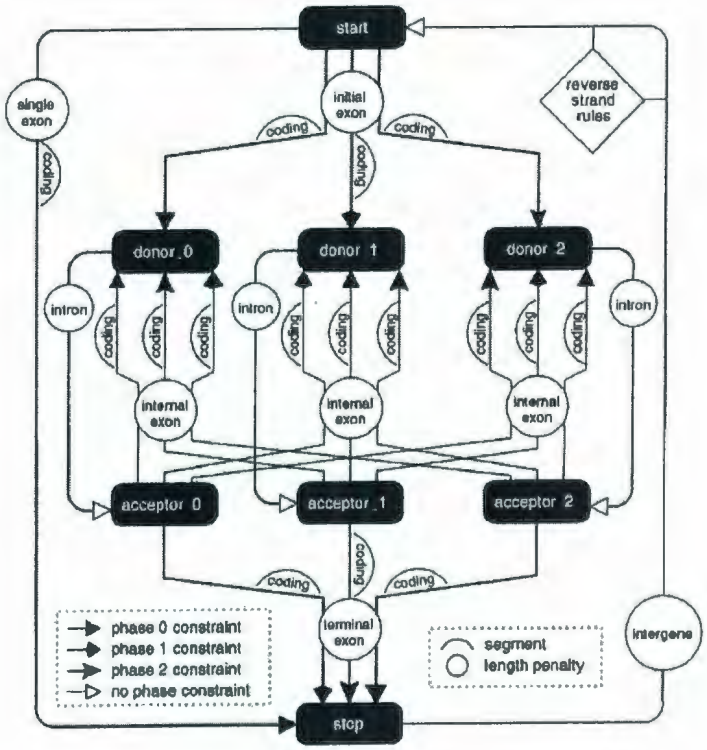

Figure 4.2: A GAZE-XML Gene Model for the Multiple Genes on Both Strands [3]

also defines the gene structures. In the configuration file, there are mainly four sections. The *gff2gaze* section declares how to obtain lists of segments and features from the input GFF lines. Each feature or segment is defined by "*source*" and "*target*", which indicate the start and end location of the feature or segment respectively. The *dna2gaze* section is to create features and segments from sequence motifs in input DNA sequences. Based on features and segments from evidence and input sequences, an overall gene structure can be scored with the sum of scores of the individual features and segments. Nonetheless, some constraints and rules have to be defined so that the gene structure will be as similar as the real gene. Therefore, in the *length-functions* section, length penalty functions are defined and segment qualifiers are given in the *model* section. Length penalty functions indicates that a given source and target pair is more likely at certain distance than any other distances. The functions are expressed in $< distance, penalty >$ pairs. Segment qualifiers denote the segments which act as supporting evidence of the region between a source and target pair. Moreover, the *model* section defines four constraints for calculating scores of qualifiers and some rules. This section tells GAZE how to assemble the information of features and segments obtained from the *gff2gaze* and *dnagaze¿* sections into a unit to construct a whole gene model.

According to the information from the configuration file, dynamic programming finds the highest scoring gene structure and the posterior probabilities is calculated to indicate how well the features fit to the gene structure. It uses the following recursion to obtain the optimal structure. Let $\phi_1 \ldots \phi_n$ be the complete list of features ordered by sequence position in which BEGIN and END are ϕ_0 and ϕ_{n+1}. Then the recursion for the maximal score is:

$$V(0) = 0, \tag{4.2}$$

$$V(i) = max_{j<i}[V(j) + Seg_{t(\phi_j)\rightarrow(\phi_i)}(l(\phi_j), l(\phi_i)) - Len_{t(\phi_j)\rightarrow(\phi_i)}(l(\phi_j), l(\phi_i)) + g(\phi_i)] \tag{4.3}$$

$t(\phi)$ is the type of the feature. $l(\phi)$ is the location. $g(\phi)$ is the score. $Seg_{src\rightarrow tgt}(x, y)$ is a sum of the scores for each segment type and $Len_{src\rightarrow tgt}(x, y)$ is the length penalty function. The maximal score is V(n+1) and the optimal structure is obtained by a traceback procedure which is storing the index j for the maximal score at each stage. Considering that commonly the gene structure with the highest score is not necessarily the correct one, GAZE calculates the posterior probability of features to measure how much the features match with the structure. The posterior probability is as following:

$$pf(\phi_i) = e^{F(i)+B(i)-F(n+1)} \tag{4.4}$$

which F(i) is a Forward vector and B(i) is a Backward vector both of which are in log-space. F(i) denotes the sum of the exponential scores of all the "upstream" partial gene structures till the feature ϕ_i. Whereas B(i) denotes the sum of the exponential scores of all the "downstream" partial gene structure beginning from the feature ϕ_i. the posterior probability applied by GAZE could be the indicator of reliability for the results.

GAZE has been applied on predicting genes of Caenorhabditis elegans and vertebrates. GAZE uses the evidence produced by the program Genefinder for C. elegans, and the program GENEID [21] for the vertebrates.

Since GAZE was originally anticipated to be a curation tool for C. elegans anno-

tation, it focused on finding genes in this species. Regarding the training data set, GAZE has adopted a set of 325 mRNA-confirmed gene structures from the WormBase WS52 database [71] which contains the curated gene structures associated with the "NDB_CDS" objects in the half of the C. elegans genome at the Sanger Institute. The 325 gene structures include 157 forward and 168 reverse strands. To construct an artificial test sequence, the genomic DNA underlying each gene structure is extracted. Half of the intergenic DNA to the nearest other curated gene in each upstream and downstream direction is chosen for each genomic DNA. The proportion of the protein-coding in this sequence is 0.24.

GAZE can not acquire the parameters of features, segments and length penalty functions by itself, so all the parameter analysis is obtained from Genefinder which is considered as one of the accurate gene prediction programs for worms. GAZE takes a set of frequency tables from Genefinder and a query DNA as inputs to predict the features and segments according to the frequency tables. The research work of GAZE has built three models. One is the standard model shown as Figure 4.2 which is referred as GAZE_std. The other two are GAZE_tsplice and GAZE_EST. GAZE_tsplice incorporates the standard model with trans-splicing. Taking trans-splicing into account is likely to avoid the mistake made by the confusion between the initial exon of a trans-spliced gene and an internal exon of a longer gene. Authors also integrated the EST matches into the model. The program EST_GENOME [72] has been applied to align the spliced cDNA to genomic DNA and thus produce a set of scored EST exons and their genomic positions.

In order to guarantee that the evidence of each type are appropriately weighted, As for GAZE_EST model, GAZE considers a method called Maximal Feature Dis-

crimination (MFD) [4] to estimate the optimal parameters and this is GAZE_EST model. For this model, authors divided the Wormseq into two sets. One is for training to get optimal parameters while the other is for testing. The results from the three models have been compared to the result from the gene prediction program Fgenesh [73] and the comparison is shown in Table 4.1.

Table 4.1: Performance Comparison for Wormseq (based on [3])

	Exon-Level		Gene-Level	
Programs	Sn	Sp	Sn	Sp
GAZE_std	0.84	0.77	0.35	0.35
GAZE_tsplice	0.86	0.80	0.47	0.42
GAZE_EST	0.90	0.84	0.59	0.53
Fgenesh	0.88	0.80	0.51	0.42

Table 4.2: Exon-Level Accuracy by Exon Types (based on [3])

	Initial		Internal		Terminal		Single	
Programs	Sn	Sp	Sn	Sp	Sn	Sp	Sn	Sp
GAZE_std	0.57	0.56	0.90	0.80	0.78	0.78	0.94	0.71
GAZE_tsplice	0.72	0.67	0.89	0.83	0.81	0.74	0.94	0.59
GAZE_EST	0.79	0.74	0.92	0.86	0.85	0.80	0.94	0.73
Fgenesh	0.75	0.61	0.92	0.87	0.84	0.68	0.63	0.77

Table 4.1 and 4.2 show the results. Apparently the accuracy of GAZE_EST at both

the exon and gene level is better than the other two models and Fgenesh as well. With respect to the sensitivity and specificity, GAZE_tsplice has been improved compared to GAZE_std, while GAZE_tsplice and GAZE_std are not better than Fgenesh. From each exon type at the exon level, Fgenesh is likely better on predicting internal exons than GAZE, but GAZE is better on single exons.

GAZE has also been tested on a vertebrate, Homo sapiens [4]. It has used the H176 data set for training and the HMR195 data set for testing. The H176 data set was constructed by Guigo et al. [74] which comprises only human genomic sequences. The entries of the data set were extracted from the EMBL nucleotide database version 50 (March 1997). The original data set has 178 sequences, but in this work the sequences HSADH6 and HSPVALB have been removed for their incorrectness. Instead of Genefinder, GENEID [21, 58] is used for Homo Sapiens to generate the parameters of signals, segments and length penalty functions. GAZE configures the model based on the results from GENEID. Its accuracy was improved four ways. One is to integrate the promoter prediction data from EPONINE_SCAN [75] into the model. Second is to adopt the exon length distributions from GENSCAN to the GAZE model. The third one is that the C+G% parameters from GENEID have been introduced into the GAZE model. The last one considers all three types of evidence (promoter prediction data, exon length distributions and C+G content). All models have been optimized by MFD. The comparison of these models are shown in the following table. $GAZE_{tss}^{MFD-10}$ stands for the GAZE model combining the promoter prediction data and having 10 parameters to optimize. $GAZE_{exo}^{MFD-4}$ stands for the model combining the exon length distribution and having 4 parameters to optimize. $GAZE_{GC}^{MFD}$ stands for the model combining the C+G content. $GAZE_{all}^{MFD}$ stands

for the model with all the three types of evidence.

Table 4.3: Performance Comparison for Homo Sapiens (based on [4])

	H176				HMR195			
	Exon-Level		Gene-Level		Exon-Level		Gene-Level	
Programs	Sn	Sp	Sn	Sp	Sn	Sp	Sn	Sp
$GAZE_{MFD}$	0.69	0.70	0.29	0.23	0.64	0.69	0.27	0.22
$GAZE_{tss}^{MFD-10}$	0.72	0.73	0.32	0.27	0.65	0.71	0.27	0.23
$GAZE_{exo}^{MFD-4}$	0.73	0.74	0.34	0.28	0.67	0.72	0.29	0.24
$GAZE_{GC}^{MFD}$	0.72	0.73	0.32	0.26	0.70	0.74	0.27	0.22
$GAZE_{all}^{MFD}$	0.76	0.76	0.37	0.31	0.70	0.74	0.33	0.27
GenScan	0.82	0.75	0.40	0.35	0.77	0.73	0.37	0.33
Fgenesh	0.64	0.60	0.30	0.25	0.64	0.63	0.32	0.29

In Table 4.3, the results show that accuracy of the model with all the evidence is the best among all the other implemented GAZE models. Compared with GenScan and Fgenesh, GAZE models work better than or equally as good as Fgenesh. However, the best GAZE model does not outperform GenScan.

JIGSAW

JIGSAW [5] uses dynamic programming to combine diverse models and evidence from other gene prediction programs into a frame-consistent gene prediction.

Essentially, a generalized Hidden Markov Model framework is constructed to predict an overall gene structure. Its structure is exhibited in Figure 4.3. In Section

4.1, we have briefly introduced how GHMMs are applied for predicting genes. A

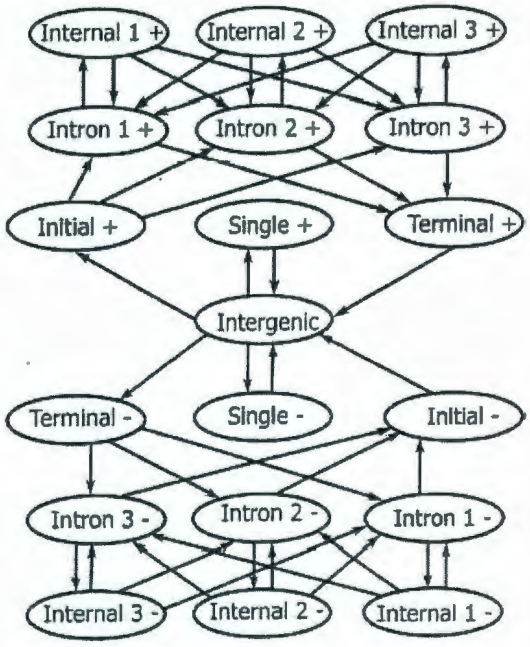

Figure 4.3: The Generalized Hidden Markov Model of JIGSAW [9]

GHMM defines a joint probability of a sequence parse t and a genome sequence S: P(t,S). However, JIGSAW allows the external evidence provide the prediction of up to six features: start codon, stop codon, intron, protein coding nucleotide, donor and acceptor sites. Therefore, the joint probability of the sequence parse t becomes the probability between t, sequence S and evidence E:

$$P(t, S, E) = P(E|t, S) \times P(S|t) \times P(t) \tag{4.5}$$

JIGSAW's purpose is to find the optimal parse maximizing the joint probability. For computing the optimally scoring parse, JIGSAW uses dynamic programming matrix D(j,q) (j is the position and q is the state) to store the highest scoring sets of states

which are from the position i to j.

Predictions of JIGSAW are based on an independence assumption, which is that predictions from each program are independent. However, even if JIGSAW follows this assumption, the number of parameters will grow exponentially as the number of programs applied goes up. To avoid this, the assumption is justified by the fact that the collection of programs used to predict each feature type are assumed to be independent. Therefore, an independent conditional probability, $P(type|v_{type})$, is defined for each six gene features. v_{type} is a feature vector of one type. A function is defined to check if the feature would occur at the position k in the state q:

$$h(q, k, type, S[i, j], B_k) = \begin{cases} P(type|v_{type}) \\ 1 - P(type|v_{type}) \end{cases} \tag{4.6}$$

B_k is the set of feature vector on position k. Then the dynamic programming matrix is:

$$D(j, q) = max_{i,q'} P(q|g_{i,j}, S[i, j]) * P(q|q') * D(i, q') \tag{4.7}$$

$P(q|g_{i,j}, S[i, j])$ is the probability of state k aligning to the sequence (from the position i to j) S[i,j] given that all the feature vectors $g_{i,j}$ is between position i and j:

$$P(q|g_{i,j}, S[i, j]) = \prod_{k=i}^{j} \prod_{type} h(q, k, type, S[i, j], B_k) \tag{4.8}$$

For presenting the features of the evidence in JIGSAW, first feature vectors are built for all the features at each position k presented in the sequence S. To improve the flexibility of the type and amount of evidence, the independent conditional probability $P(type|v_{type})$ for each feature is defined. This probability is estimated from the training set. The training is to count the the number of times that the evidence of each features occurs in the training set and subsequently a percentage of cases is

obtained for predicting the location of the feature. In this procedure, to avoid the problem of large sample space, a decision tree algorithm is applied to partition the feature vector space into subregions so as to distinguish the accurate set from the inaccurate set of the feature vectors.

This research work used two test sets to evaluate JIGSAW on human gene prediction [5]. One data set is 1563 random genes selected from a set of 17477 non-redundant RefSeq genes [76]. 2/3 of the data is used for training and 1/3 is for testing. The second data set is the annotations from the Havana group [77] in the 44 ENCODE regions. This data set does not overlap with the 1563-gene data set and includes some known alternatively spliced genes. As to the evidence, JIGSAW used an annotation database which was downloaded from the UCSC genome annotation database [78]. The collection of evidence includes cDNA from human genes, RefSeq genes from non-human species, predictions from ab initio gene finders such as Genscan, Geneid, GeneZilla and GlimmerHMM, the TIGR Gene Index [79] and others. The results of the first data set are shown in Table 4.4. With respect to gene sensitivity, JIGSAW's predictions using three different combinations of evidence are not as good as the predictions from Ensembl and the cDNA alignment. But other results are better.

Table 4.5 presents the performance on the ENCODE data set. The results from JIGSAW with four distinct combinations of evidence are compared with those from Ensembl, cDNA alignments and two comparative analysis tools: Twinscan [80] and SGP. cDNA alignments to Swiss-Prot/TrEMBL proteins performed outstandingly on the sensitivity at the gene, exon and nucleotide levels. Whereas JIGSAW using human expression evidence as input(the first two rows) has the nearly best performance on

Table 4.4: Performance Comparison on 1563 Test Genes (based on [5])

	Nucleotide-Level		Exon-Level		Gene-Level	
Programs	Sn	Sp	Sn	Sp	Sn	Sp
JIGSAW	0.90	0.98	0.87	0.89	0.59	0.66
JIGSAW(no Ensembl)	0.90	0.98	0.86	0.88	0.55	0.62
JIGSAW(all)	0.90	0.98	0.86	0.89	0.58	0.63
Ensembl	0.85	0.95	0.85	0.80	0.62	0.50
cDNA alignments	0.82	0.93	0.84	0.77	0.65	0.38

Table 4.5: Performance Comparison on ENCODE Data set (based on [5])

	Nucleotide-Level		Exon-Level		Gene-Level	
Programs	Sn	Sp	Sn	Sp	Sn	Sp
JIGSAW	0.92	0.94	0.88	0.93	0.43	0.72
JIGSAW(no Ensembl)	0.92	0.95	0.87	0.93	0.42	0.72
JIGSAW(no curated genes)	0.89	0.93	0.85	0.89	0.31	0.54
JIGSAW(no expression data)	0.85	0.85	0.73	0.76	0.15	0.24
Ensembl	0.93	0.89	0.86	0.88	0.56	0.55
cDNA alignments	0.94	0.91	0.88	0.92	0.63	0.48
Twinscan	0.78	0.86	0.70	0.65	0.12	0.20
SGP	0.82	0.83	0.69	0.67	0.11	0.15

specificity at the three levels. They are much better than the other two JIGSAW performances(the fifth and sixth rows). The one with no curated genes which means no Ensembl and no Swiss-Prot/TrEMBL alignment input and another has no gene expression evidence. Ensembl and cDNA alignments have better performances than these two JIGSAWs. From all of the above results, in this work JIGSAW chiefly keeps the accuracy of sensitivity and specificity in a balance. According to the tests of the ENCODE Gene Prediction Workshop (EGASP) [81], on human gene prediction JIGSAW in most cases outperformed both ab initio methods and other combination methods using homology.

ExonHunter

ExonHunter exploits the information from proteins, ESTs, genome-genome comparisons and sequence repeats and combines them into a probabilistic framework, which is based on a HMM. The HMM first models a basic gene structure consisting of signals and contents sensors, such as exons, introns, and intergenic regions. Thus a conditional probability distribution $Pr(A|sequence)$ over all possible annotations A for a query sequence is defined. Next the external evidence will yield one or more advisors which will be combined into a superadvisor to define a probability distribution $Pr(A|evidence)$ over all the annotations. Finally the two probability contributions $Pr(A|sequence)$ and $Pr(A|evidence)$ are integrated by the Bayesian Theorem and the annotation A^* will be found to maximise $Pr(A^*|sequence, evidence)$. A sketch of the architecture of ExonHunter is shown in Figure 4.4.

The HMM is a generalized HMM similar to the ones of GENSCAN or AUGUS-

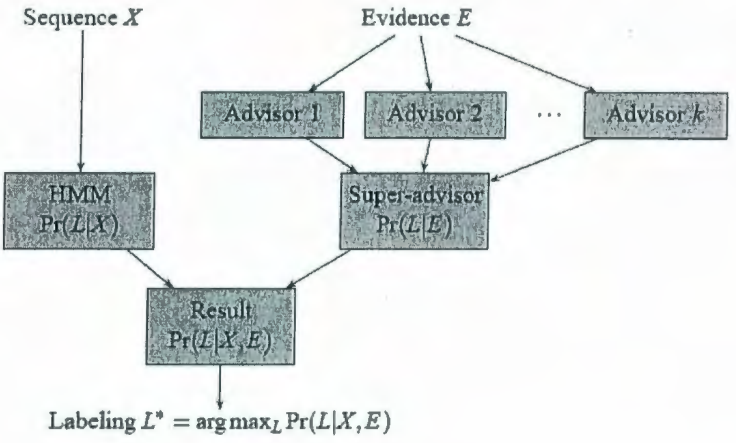

Figure 4.4: ExonHunter's Brief Architecture [6]

TUS [82], but it has three principal extensions. First, the transition and emission probabilities of ExonHunter's HMM are determined by GC content level which is estimated from a 1000bp window around the current position. There are four GC content levels each of which covers 25% of the whole sequence. Whereas the HMMs of some other gene finders decide the two probabilities based on GC content of the whole sequence. ExonHunter makes this change because the GC content level could have significant differences between coding and non-coding regions even within a single gene. The second extension is that ExonHunter uses higher order trees (HOT) models [83] to model donor and acceptor site signals. The HOT models capture the important non-adjacent intrasignal dependencies which provide more accurate estimates of probabilities. The third difference of ExonHunter is the length distribution [84]. Length distribution is a disadvantage for HMMs since a geometric probability distribution will rise on the length of the region of a state. But the geometric distribution can not accurately estimate the length distributions of the elements. Therefore, ExonHunter

proposes a technique which is the geometric-tail distribution [84] to solve the problem. In this method, the length distribution is decomposed into two parts: a head with arbitrary distribution and a geometrically decaying tail. This technique helps model accurate exon and intron non-geometric length distributions. The HMM is the basis of the framework and thus ExonHunter is an ab initio gene finder when there is no external evidence input.

The supplementary evidence is presented as advisors. In ExonHunter, there are four types of advisors: the advisors for protein alignments, EST alignments, genome alignments and repeats. For the query sequence, an advisor specifies a probability distribution over a set of annotation labels which correspond to signals and contents. As it is hard to estimate a complete probability distribution from some sources of information, an advisor is allowed to provide partial information. For example, the advisor predicting donor signals can not be used for estimating the probability of a position which is inside an intron. Accordingly, an overall probability distribution for each position will be created to coordinate the different advisors. It is called the superadvisor which is a probability distribution $x^* = (x_1, \ldots, x_n)$ over all labels at a particular position; x_i is the probability of the ith label from a set of labels. The problem in combining the advisors is that different advisors produce different partitions. Suppose that π_a is a partition of the set of labels and S is the set of the partition elements. If all the advisors do not conflict with each other, then the sum of the probabilities in x^* for all labels in S should be equal to the probability distribution over all the partition elements for the advisor a, $p_a(S)$. However, actually the advisors are always in conflicts. Thus, the probability distribution x^* should maximumly satisfy the constraints under which advisors are compatible with each

other. To find this probability distribution, ExonHunter chooses to minimize the sum of the weighted distances between the advices and x^*. It is defined as follows:

$$dist_a(x^*) = \sum_{S \in \pi_a} \frac{1}{prior(S)} \cdot \left(p_a(S) - \sum_{j \in S} x_j \right)^2 \tag{4.9}$$

where

$$prior(S) = \sum_{j \subset S} prior(j) \tag{4.10}$$

and prior(j) is the prior probability of label j, which is estimated as the proportion of the genome annotated with a given label. Then the advisors are combined by a convex quadratic programming [85]:

$$minimize \sum_a w_a \cdot dist_a(x^*) \tag{4.11}$$

$$subject\ to \sum_{j \in \sum} x_j = 1, x_i \geq 0 for\ all\ the\ labels\ j \in \sum \tag{4.12}$$

where w_a is the positive weight of advisor a and $\sum$ is the set of labels.

The last step in ExonHunter combines the superadvisor and the HMM. The final result is to find the most probable annotation given the sequence and the external evidence. It uses the Bayes' rule to calculate the probability:

$$Pr(A|seq, ev) = \frac{Pr(seq, ev|A) \cdot Pr(A)}{Pr(seq, ev)} \tag{4.13}$$

This formula is based on an assumption that is the DNA sequence and the external evidence are conditionally independent given the annotation A. Therefore, according to the formula

$$Pr(seq, ev|A) = Pr(seq|A) \cdot Pr(ev|A) \tag{4.14}$$

and Bayes Theorem, Equation 4.13 can be simplified to:

$$Pr(A|seq, ev) \propto Pr(A|seq) \cdot \frac{Pr(A|ev)}{Pr(A)} \tag{4.15}$$

Here Pr(A) is the prior probability of the annotation A, which is a sequence of labels $l_1 l_2 \ldots l_n$:

$$Pr(A) = \prod_{i=1}^{n} prior(l_i) \tag{4.16}$$

Nevertheless, in practice the independence assumption is not true. Hence, a parameter α is introduced to the equation to reduce the effect from the superadvisor:

$$Pr(A|seq, ev) \propto Pr(A|seq) \cdot \frac{Pr(A|ev)^{\alpha}}{Pr(A)^{\alpha}} \tag{4.17}$$

where $\alpha < 1$.

ExonHunter used the data set from ENCODE project [86]. The data set is split into the testing set and training set. The research work also used the ROSETTA set of 117 human single-gene sequences [87] collected by Batzoglou et al. [49] for the testing. For the training, a set of 1284 human single-gene sequences created by Stank [88] is adopted to train the HMM and all the parameters of advisors are trained on the ENCODE training data set. But 81 sequences from the Stank data set have been removed due to the significant similarities to the ROSETTA set used for testing. The training set included a set of 15,263 human splice sites from SpliceDB [89] for training the models of splice site signals. Additionally intergenic region lengths have been trained on a part of the annotated human chromosome 22 from the RefSeq [76].

ExonHunter compared its performance with other gene finders like GENSCAN, ROSETTA, SLAM and some others. The following Table 4.6 is the comparison with ExonHunter and other programs on the ROSETTA data set. From the table, it is significant that the sensitivities and specificities at all three levels from ExonHunter are higher or equal to those of other programs. Compared with ExonHunter including supplement evidence (EH column), ExonHunter without additional evidence (EH-nh

column) has lower performance, but is still better than most of other programs.

Table 4.6: Performance Comparison on the ROSETTA Data set (based on [6])

	Nucleotide-Level		Exon-Level		Gene-Level	
Programs	Sn	Sp	Sn	Sp	Sn	Sp
GenScan	0.98	0.88	0.82	0.73	0.44	0.41
ROSETTA	0.94	0.98	0.83	0.83	-	-
SLAM	0.95	0.98	0.78	0.76	-	-
Twinscan	0.98	0.89	0.84	0.77	-	-
SGP1	0.94	0.96	0.70	0.76	-	-
EH	0.99	0.93	0.91	0.83	0.74	0.66
EH-nh	0.99	0.93	0.89	0.81	0.68	0.62

Next table (Table 4.7) shows the comparison of performance on the ENCODE data set. In the first block of the table, it is the comparison among the ab initio gene finders in which ExonHunter has no additional evidence. The second block contains the gene finder TwinScan using genomic alignments and in the last block they are all the gene finders choosing various supplemental evidence. Compared with other gene finders which either use ab initio methods or combination methods, the performance of ExonHunter is in between. Its performance on the sensitivity at the nucleotide level is relatively better than the other sensitivities and specificities at the nucleotide and exon levels. It is analyzed that the reason for the lower exon sensitivity is probably the simple methods for aligning the protein database to the query sequence. Concerning the lower specificity of ExonHunter, it is because there is

Table 4.7: Performance Comparison on the ENCODE Data set (based on [6])

	Nucleotide-Level		Exon-Level	
Programs	Sn	Sp	Sn	Sp
GenScan	0.85	0.44	0.59	0.37
ExonHunter(no adv.)	0.79	0.52	0.55	0.39
GeneID	0.74	0.78	0.47	0.59
TwinScan	0.77	0.83	0.52	0.65
JIGSAW	0.95	0.92	0.80	0.89
TwinScan(ESTs)	0.87	0.92	0.76	0.88
Fgenesh	0.91	0.77	0.75	0.69
ExonHunter	0.93	0.67	0.69	0.51
RefSeq	0.86	0.98	0.64	0.83

no penalty for the unsupported genes in ExonHunter. Nevertheless, according to the evaluation results, sensitivity and specificity of ExonHunter at the gene level are 29% and 10%, compared to the 17% gene sensitivity and 6% gene specificity performed by GenScan. It can be demonstrated that ExonHunter is engaged in finding as many possible genes as the query sequence probably has.

4.2.2 Comparison of the Three Gene Finders

Comparison on the Architectures

Among the present gene finders, the three gene finders comprehensively combine the external evidence that includes predictions of both signal and content sensors. Firstly they all construct basic gene structures. Then the evidence is applied as a complement so that more accurate gene structures could be obtained. GAZE uses an XML file to describe a gene model and to assemble the evidence into a model. Subsequently, a dynamic programming algorithm calculates the highest score of the model. In JIGSAW, a generalized HMM (GHMM) is applied to build an overall gene structure. Evidence is combined into this GHMM by a dynamic programming algorithm. ExonHunter also has a GHMM for building a whole gene model. The supplemental evidence is integrated into the gene model by dynamic programming and the help of the Baysian Theorem. A summary for the comparison of the architectures is listed in Table 4.8.

Regarding the frameworks of the three gene finders, both JIGSAW and ExonHunter build basic gene models by GHMMs. The structures of GHMMS are similar,

Table 4.8: Summary for the Comparison on the Architectures

Programs	Method for Building a Gene Model	Algorithm for Combining Evidence	Evidence from Cross-Species Comparative Sequence Analysing
GAZE	XML	Dynamic Programming	No
JIGSAW	GHMM	Dynamic Programming	Yes
ExonHunter	GHMM*	Baysian Theorem and Dynamic Programming	Yes

* It has some extensions on GC content, signal models and length distribution

but ExonHunter's HMM has some extensions for GC content, signal models and length distribution. Moreover, there is an essential difference between the GHMMs of JIGSAW and ExonHunter: in JIGSAW the evidence is a part of the GHMM for building a whole gene structure, while the GHMM of ExonHunter can construct a gene structure without any evidence. In other words, ExonHunter is an ab initio gene finder when no external evidence is input, but JIGSAW is not. Unlike JIGSAW and ExonHunter applying HMMs, GAZE defines a gene structure by the configuration files written in XML and then sets the parameters of the model by the predictions from other gene finders. This is a novel method to build gene models.

After having fundamental frameworks, the programs will coordinate the incorperation of evidence into their frameworks. GAZE and JIGSAW apply dynamic programmings to integrate evidence. Besides defining gene models, the XML file of GAZE configures how to read lines of input evidence files so that the parameters of the model are determined. Then a dynamic programming algorithm decides the best gene structure by calculating the highest score. JIGSAW assembles evidence into its framework by adding an independent conditional probability to the GHMM. Similarly, JIGSAW predicts the best gene structure by dynamic programming. Whereas

ExonHunter initially coordinates different evidence into the superadvisor that is an overall probability distribution for each position. Afterwards, it adopts the Bayesian Theorem to join the superadvisor and the predictions from its HMM.

As to the choice of the evidence, the three gene finders all combine the predictions of up to six features: start/stop codon, donor/acceptor site, coding regions and introns. JIGSAW and ExonHunter exploit the predictions from different gene finders, while GAZE considers the predictions produced only by Genefinder for C. elegans and by GENEID for vertebrates. Therefore, compared to JIGSAW and ExonHunter, the limitation of GAZE is that GAZE cannot consider predictions from comparative sequence analysis. Nevertheless, comparative sequence analysis approaches are very helpful for predicting genes now.

Comparison on the Applications and Performance

Aside from a comparison on the architectures of the three programs, we also tested these gene finders and compared their performance in term of the sensitivity and specificity. Since evidence for each program is different, we firstly compare the evidence they adopt and the formats of the evidence files defined by the programs. Next, we discuss their performance according to the experiments mentioned in the section 4.2.1. Also we compare the performance of GAZE and ExonHunter on the hmr195 data set.

The three gene finders all have their own specific requirements for the formats of the evidence files. For input files of evidence, GAZE requires files in the GFF format. JIGSAW reads several file formats: "btab" and "gff" in general, and "glimmerm"

from GlimmerM or GlimmerHMM, "phat" from PHAT, "fgenesh" from Fgenesh, "genemarkhmm" from GeneMark, "genscan" from GenScan and "snap" from Snap. Before execution of both GAZE and JIGSAW, evidence should be collected and saved in specific directories by running gene finders corresponding to the evidence. ExonHunter has its default configuration for these programs. It supports evidence of repeat masking, ESTs and protein from RepeatMasker, SIM4 and WUBLAST, and blastx, respectively. These programs are executed to obtain evidence when ExonHunter is running. If users prefer to use other evidence to ExonHunter, they need to create their own tasks, which means the parameter files have to be created by users. In summary, currently evidence cannot be flexibly and easily chosen and integrated into the frameworks of combination-method gene finders. Either evidence is restricted by file formats required, or it is too complicated for users to select and apply their own evidence.

Based on the discussion of the experiments on the three gene finders, we can draw general conclusions about their performance. Overall, the three programs all show a better performance than ab initio gene finders. Also, they are better than gene finders that only apply comparative sequence analysis. JIGSAW may have the best performance among the three programs. In terms of sensitivity and specificity at all levels, nucleotide, exon and gene, JIGSAW and GAZE can keep a balance, whereas ExonHunter shows that its sensitivity is always better than its specificity. In the review section about ExonHunter, we presented that JIGSAW and ExonHunter were both tested on the ENCODE data set. The results are listed in Table 4.7. JIGSAW achieves higher sensitivity and specificity than ExonHunter does at both nucleotide and exon levels. Especially at exon level JIGSAW is much more outstanding than

ExonHunter.

Since this thesis uses the HMR195 data set as a test data set, we have compared the performance of ExonHunter and GAZE on this data set. GAZE was trained on the H176 data set and the evidence produced by GeneID was applied. Its parameters have been optimized by MFD. ExonHunter trained its HMM on a set of 1284 human single-gene sequence created by Stanke and Waack. It adopted the following external evidence: repeats, human ESTs alignments, mouse ESTs alignments and human protein alignments. Table 4.9 presents the evaluation results. Although optimized GAZE is better than one without any optimization,

Table 4.9: Performance Comparison of GAZE and ExonHunter on HMR195

	Exon-Level		Gene-Level	
Programs	Sn	Sp	Sn	Sp
GAZE_{GeneID} [4]	0.61	0.65	0.13	0.12
$\text{GAZE}_{GeneID}(ML)$ [4]	0.56	0.68	0.23	0.19
$\text{GAZE}_{GeneID}(MFD)$ [4]	0.64	0.69	0.27	0.22
ExonHunter	0.88	0.35	0.56	0.19

its sensitivities are lower than those of ExonHunter at both exon and gene levels. Nevertheless, ExonHunter's specificities are not as good as those of GAZE, especially at exon level. Accordingly, ExonHunter is more sensitive to genes and exons than GAZE, but GAZE is able to identify genes and exons more correctly. This result of the comparison is not unexpected in that the purpose of ExonHunter is finding as much genetic information as it possibly can. Therefore, ExonHunter predicts more

possible genes and exons, but the correctness is somehow lowered. Although GAZE is not prominent, its sensitivities and specificities at both levels are close. It has an average performance which shows that GAZE can provide good specificities to genes and exons and does not lower its sensitivities.

Discussion

In conclusion, JIGSAW and ExonHunter adopt a wider range of evidence that can include predictions from programs using comparative sequence analysis, whereas GAZE cannot. Although ExonHunter's outstanding sensitivity indicates that it is good at predicting as many genes as possible, JIGSAW and GAZE show that they could not only guarantee the correctness of predictions but also keep sensitive to genes and exons. As to the usage of these programs, all have their advantages and disadvantages. To run GAZE, an XML file needs to be created for each species on which the prediction will be made. With respect to JIGSAW, though it has better performance than the other two, the collection of evidence is not a pleasant work. In addition, JIGSAW cannot run multiple-sequence files, which are universally used now. Compared with JIGSAW and GAZE, ExonHunter does not have better performance, but it is easy to use and to obtain result files relatively fast. Additionally, ExonHunter can also be used as an ab initio gene finder.

All in all, the three gene finders demonstrate that combination methods truly improve the accuracy of gene prediction. This class of gene finders is limited by the incompleteness and availability of evidence. As more knowledge about genomes of further species is obtained. evidence will become more and more refined. We think

that how to extend the types of evidence and how to conveniently integrate new evidence into existing frameworks is an important issue. It is crucial to make gene finders simple and friendly for researchers who are from biology, pharmacy and other research areas instead of computer science.

Conclusion

Up to now, the studies done on the combination methods have shown that the accuracy of computational gene prediction could be improved by exploring the existing methods and programs. The gene finders applying only either ab initio algorithms or similarity search approaches cannot meet the demand for today's accuracy. Combination methods are in development and promise to take advantage of the currently available programs to improve the performance. We reviewed and summarized the present research work employing combination methods. In this chapter, we categorized them into two types. The first type of gene finders builds a gene model by statistical algorithms and evidence from other programs is complementary to the gene model in order to improve the accuracy of the gene model. The second type only considers outputs from different programs and integrates the outputs using certain rules presented by K. Murakami and T. Takagi. We found that the first type of gene finders is more precise and flexible to make use of evidence, while the second type is simpler for users to integrate other evidence.

By working on the three gene finders, we have a sense that ExonHunter is easier for using and meanwhile it can have reasonable results for predicting genes. This is a critical qualification of a gene finder. Therefore, this part of our work inspires us to

develop a framework that has the ability to easily combine evidence and also to give out reliable prediction results.

Chapter 5

Two Algorithms to Combine ExonHunter and SGP2

After surveying work done on predicting genes, we found that the tendency in gene prediction is to use combination methods. Current gene finders using combination methods have improved the accuracy on predicting genes. In addition, some of these gene finders are flexible for external evidence; users can choose predictions from some other programs to integrate into the gene finders. However, we also recognize that it is a bit hard for users to integrate additional evidence into whole systems. There are two main reasons. First, the formats of external evidence are variable and thus there is a heavy workload for converting formats. Second, there are some limitations on evidence; either certain types of evidence are allowed to use, or certain source programs of evidence are used. Inspired by GeneComber [66]'s work, we have a thought of a system, which can eliminate the two main limitations: users can combine evidence from any programs they prefer and the combination process is simplified. To achieve

this, our work described in this thesis adopts a gene finder using combination methods and another gene finder using comparative sequence analysis, and then combines their predictions.

According to our work described in chapter 4, we choose ExonHunter as the basic framework of our work from the three gene finders, because ExonHunter predicts as many possible genes as it can. This is useful for detecting novel genes. Also, the usage of ExonHunter is simpler than the other two and it can give good predictions. However, the methods applied by ExonHunter for aligning the protein database to a query sequence are relatively simple which may affect its exon sensitivity. In order to improve the cross-species comparative sequence analysis module of ExonHunter's framework, the gene finder SGP2 has been considered based on our work introduced in chapter 3.

At the beginning of this chapter, the motivation of the thesis, which is the heart disease Sudden Cardiac Death, is reviewed. Next, we demonstrate the two algorithms we developed. In the experimental section, first the genomic properties of Homo sapiens, Mus musculus and Canis familiaris are reviewed because we predict genes on human chromosome 3 by using the mouse and dog databases. We also briefly introduce human chromosome 3 and the mouse and dog databases we use. Then the results of our algorithms running on the HMR195 data set are discussed. Finally, we give our predictions of the sequence between the markers D3S1259 and D3S3659 on human chromosome 3 and analyse the results.

5.1 Overview of Sudden Cardiac Death

It is known that Sudden Cardiac Death (SCD) is the most mysterious and deadly of the cardiac diseases. Each year the number of Canadians dying of SCD is between 35,000 to 45,000 and there are about 325,000 adult deaths in the United States. More people died from this disease than from AIDS, breast cancer, and lung cancer together [90]. SCD is an unexpected death due to cardiac causes involving an abrupt loss of consciousness owing to the disruption in cerebral blood flow [91]. It may occur in the patients who may or may not have any symptoms of heart diseases.

However, with the exception of an implantable defibrillator, there are few effective approaches to prevent SCD and even fewer clues with respect to patient phenotypes predisposed to life-threatening arrhythmias [92]. Therefore, it is necessary to clarify the genetic risk and to identify genetic factors that can advance the study of SCD. It is increasingly clear that genetic factors may play a role in the development of myocardial substrate associated with a predisposition to thrombosis and arrhythmia [91]. There are three main lines of evidence [92] which indicate that the inherited sequence variation provides a potential risk for SCD. Firstly, inherited mutations in coding sequences in at least seven cardiac sarcolemmal, Na^+, K^+ and Ca^+ ion channel subunit genes result in increased propensity to SCD. The second evidence is the interaction between some channel proteins and a number of "proarrhythmic" drugs and environmental agents. The third one is family history. This evidence comes from two large retrospective epidemiological case-control studies conducted in broad community-based populations. Thus it is critical to study the families with the history of SCD and identify the genes related to the disease. In recent years scientists

discovered some genes and new genetic pathways in SCD. For example, researchers at the University of California, San Diego (UCSD) Institute of Molecular Medicine (IMM) found that the gene KChIP2 is related to SCD [93, 94] and later discovered a new genetic pathway playing a pivotal role [95]. But genetic variants associated with instances of sudden cardiac death are far more prevalent and diverse than first thought, especially among minorities. Idiopathic long QT syndrome is a rare familial disorder for heart diseases and more than 200 mutations in the 7 genes related to long QT syndrome have been found [96]. One of the 7 genes is SCN5A and its mutation on chromosome 3 is one of the most common ones. Also, in 1992 [97] described the Brugada syndrome. In their study, a mutation in SCN5A on chromosome 3 is detected in 25-30% of the patients.

To summarize, chromosome 3 is potentially related to SCD and there are still many unknown genes on this chromosome. Therefore, our study chooses to focus on human chromosome 3p25, especially the sequence between the markers D3S1259 and D3S3659. It is expected that some unknown genes and novel coding sequences related to this disease will be predicted by our work.

5.2 Algorithms

Two algorithms have been developed in this thesis to combine gene finder ExonHunter with cross-species comparative gene finder SGP2. The fundamental idea of the two algorithms comes from research done by Murakami et al. [8] and Rogic et al.'s work GeneComber [66]. The algorithms follow the And-based and Or-based rules discussed in [8] to combine results from two gene finders. They follow different considerations

regarding priorities of exon and gene prediction, respectively.

5.2.1 And-based and Or-based Rules at Exon Level

And-based and Or-based rules have been introduced in Chapter 4. This algorithm basically follows the two rules: for the And-based rule, it considers the overlapping exons and genes predicted by both ExonHunter **and** SGP2. While for the Or-based rule, it considers all the predicted exons and genes predicted by at least one of the two gene finders: ExonHunter **or** SGP2.

To decide whether And-based or Or-based rule will be chosen, a threshold is calculated. Since ExonHunter does not provide any probability score in the predicted results, only probability scores of SGP2 results are considered and a threshold is determined by these scores. As shown in Figure 5.1, the algorithm fundamentally considers four cases:

1. If the predicted exons given by ExonHunter and SGP2 overlap:
    - If the scores of SGP2 are higher than the threshold, then only the overlapping regions of ExonHunter and SGP2 are considered;
    - Otherwise, all exons from ExonHunter including the overlapping regions with SGP2 are considered.
2. If the exons given by ExonHunter and SGP2 have no overlap:
    - If the scores of SGP2 are higher than the threshold, all the predicted results given by both ExonHunter and SGP2 are considered;
    - Otherwise, only the exons from ExonHunter are considered.

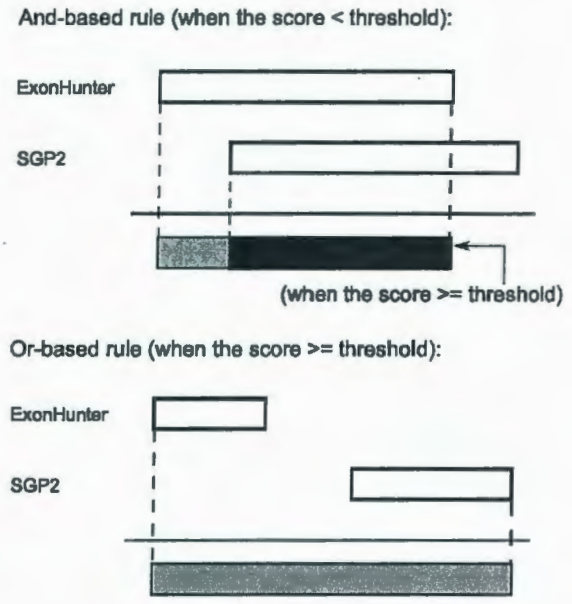

Figure 5.1: Graphical explanation to the first algorithm (at exon level). The boxes represent exons.

How to find the threshold is an important issue for the algorithm. Here we determine the threshold as the average of all scores given by SGP2: not only the average of values, but also the central point of the distribution of all scores. Since the scores of SGP2 are randomly distributed in a wide range, the purpose of the threshold will be to ensure that the number of scores which are higher than or equal to the threshold is nearly equal to the number of scores lower than the threshold. Correspondingly, the value of the threshold should be close to the medium of all scores.

We observed that the range of probability scores given by SGP2 is from -50 to $+\infty$, but most of the scores fall between 0 and 100, some portion of the scores are between -50 to 0 and some between 100 to 1000, and a very small portion of them are significantly higher than 1000. According to this observation, we decided to only

consider scores between -50 to 1000 and ignore the minority of scores beyond 1000. First, the scores in this range are converted the values ranging for 0 to 1. Subsequently, the maximum and minimum scores in the range of -50 to 1000 are detected. For the consideration of the frequencies of the scores in different ranges, the weighted mean is applied to determine the threshold. It is calculated as:

$$p_{thr} = \frac{\sum_{i=1}^{n} \omega_i * p_i}{\sum_{i=1}^{n} \omega_i} \tag{5.1}$$

where ω_i is the number of scores falling in the i-th range belonging to the whole range (0,1), p_i is the mean of the scores in the i-th range, and n is the total number of segments. The division of the whole range and the number of segments are determined by the distribution of all scores. To ensure that the average can evenly distributed all scores into two parts, we divide the whole range (0,1) into eight segments by experiments.

This algorithm integrates ExonHunter and SGP2 at the exon level. It considers all candidate exons given by ExonHunter and SGP2 and the scores are probability scores of exons.

5.2.2 And-based and Or-based Rules at Gene Level

This second algorithm is similar to the first one, but it first looks at gene level and then exon level. It considers all candidate genes given by ExonHunter and SGP2. But only genes predicted by both gene finders are considered. Then for each candidate gene, the first algorithm is applied to decide the exons belonging to the genes. Figure 5.2 illustrates how this algorithm selects the predicted genes and exons.

First, the algorithm separates the predicted genes of each gene finders into two

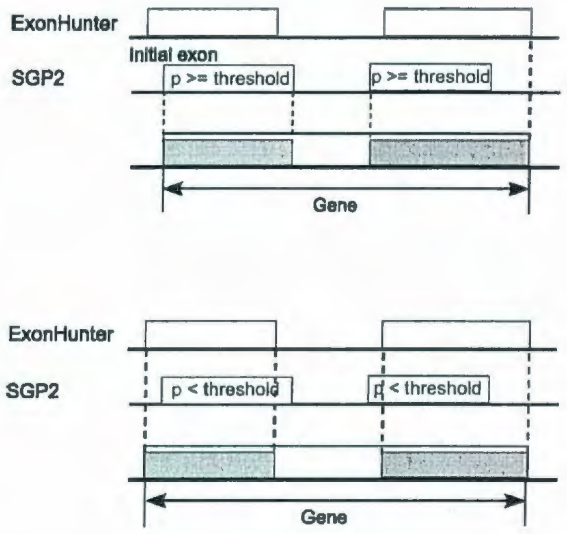

Figure 5.2: Graphical explanation to the second algorithm (at gene level). The boxes with shadows represent exons.

groups: one is for the genes on the "-" strand and another one is for the genes on the "+" strand. Then the algorithm compares one predicted gene from ExonHunter and another from SGP. If the start positions of the initial exons and the last positions of the last exons in the genes meet the overlapping criteria, the overlapping region of the two genes will be considered. Subsequently, the first algorithm will be used to the exons belong to the overlapping of the genes.

5.3 Experiments and Discussion

Here, our algorithms are applied on the data sets HMR195 and chromosome 3. The HMR195 dataset is a benchmark data set so that we can analyse what improvements our work generates to the ExonHunter framework. The segment from D3S1259 to D3S3659 on human chromosome 3p25 is the motivation and goal of this work. Our

purpose is to detect genes and genetic information contained in this part of DNA. By comparing the result with the annotation of this segment, new genetic information that is probably related to sudden cardiac death could be discovered.

For the evaluation, we have chosen the evaluation tool Eval [98] developed by the Computational Genomic Laboratory of Washington University. This software provides a set of statistics to show the similarities and differences between the standard annotations and the predictions. It reports the sensitivities and specificities of gene, transcription, exon and nucleotide. Also, it reveals the statistical information about the main signals and certain characteristics of genes, exons and so on. Moreover, it can produce two types of plots: one is the histogram showing the distribution of the statistics and another plots the categories of exons or genes by their length or GC content. The input annotation and prediction files are required to be GTF file format [99] by Eval.

5.3.1 Introduction to Data Sets

Genomic Properties of Three Species(Homo sapiens, Mus musculus and Canis Familiaris)

The Human Genome Project(HGP) began in 1990 and was completed in 2003. But it is just the first step in understanding human genomes. Human beings individually have 24 chromosomes (22 autosomal + X + Y). In human DNA there are a total of approximately 3 billion DNA base pairs, estimated to contain 20,000-25,000 genes (International Human Genome Sequencing Consortium 2004). Surprisingly, this number

of human genes is similar to or lower than the number of genes of simpler organisms such as Arabidopsis thaliana(26,000 genes) and pufferfish(33,000 genes). However, the human proteome has more complex architecture than invertebrates have due to alternative splicing. It is confirmed that alternative splicing is critical in creating the proteome diversity. Also it is estimated that the frequency of alternative splicing per gene ranges from 35% to 60% [100]. More than 98% of the human genome does not code for genes [101]; many portions of the human genome consist of repetitive DNA elements such as long interspersed elements, short interspersed elements and long-terminal-repeat retrotransposons.

The house mouse (Mus Musculus) genome has had a huge impact on biological and medical studies. It is considered to be a sufficiently stable genome sequence. Many researchers have worked on the comparison of human and mouse genomes. Sequencing of the whole Mus musculus genome was completed in 2000. It has approximately 3 billion base pairs and is estimated to have at least 30,000 genes [102]. Although the mouse genome is 14% smaller than the human genome, approximately 99% of mouse genes can find counterparts in human DNA. At the nucleotide level, approximately 40% of the human genome can be aligned to the mouse genome [103]. Because human and mouse evolved from a common ancestor about 75 million years ago and both species have inherited the ancestor's genes, they have many common genetic elements. So far, 1200 new genes in humans have been identified by comparing with the mouse genome [104]; a significant number of these genes are likely to be involved in cancer and other diseases. Therefore, the mouse genome accelerates the speed of finding genes in the human genome and thus helps to better understand human diseases.

In 2005 the Broad Institute of MIT and Harvard announced the publication of the genome sequence of the dog (Canis familiaris) [105]. The genome of the domestic dog, which has a close evolutionary relation with human, is a new powerful tool for understanding the human genome. The origin of the domestic dog is the grey wolf in East Asia, which can be tracked back to at least 15,000 or probably as far as 100,000 years ago [106, 107, 108, 109]. The dog genome is similar in size to the genomes of humans and other mammals. It contains approximately 2.5 billion DNA base pairs. A survey paper [110], done on the dog genome sequence, demonstrates that more than 650 million base pairs (> 25%) of dog sequence align uniquely to the human genome, including fragments of putative orthologs for 18,473 of 24,567 annotated human genes. In addition, dogs are the most intensely studied animal in medical practice except for human [105]. In summary, dogs play an important role in the studies of the mammalian genomes and evolution and are helpful in discovering human diseases.

Human Chromosome 3, Mus musculus Database and Canis Database

Chromosome 3 is one of the largest human chromosomes [111] and it is composed of four contigs, one of which represents the longest unbroken stretch of finished DNA sequence so far. Chromosome 3 spans almost 200 million base pairs and represents about 6.5% of the total DNA in the cells. It pressumably has 1,000 to 1,500 genes.

In [111], 1,585 gene loci have been annotated. Among them, there are 1,425 known coding genes, 8 novel genes, 27 novel transcripts, 3 putative genes and 122 pseudogenes. In this chromosome there are two gene clusters on the p-arm which

contain 18.9 and 21.1 genes per Mb falling into the base coordinates from 10 to 17 Mb and from 41 to 55 Mb respectively. These two regions represent 26% of the genes on this chromosome. The sequence we work on is a part of p-arm. In addition, chromosome 3 is remarkable for having lowest rate of segment duplication in the genome. It only has 1.7% of its bases composed of duplicated sequences compared to the whole genome average of 5.3% which is the lowest percentage for any other human chromosomes.

This paper also presents some evolution studies that have been done on chromosome 3. It has been compared with chicken, African apes, chimpanzee, gorilla, rheus macaque and more. It is found that a large-scale pericentric inversion occurred in the ancestor of the African apes, chimpanzees and gorilla also is present in human chromosome 3 as well. Additionally, two scaffolds from the study of rheus macaque $Mmul_0$.1 assembly were found to span both breakpoints of the human inversion [111] and the inversion regions on chromosome 3 are characterized by segmental duplications [112]. The inv(3)(p25:q21) pericentric inversions may be the most interesting because they exist along with other accompanying chromosomal abnormalities which cause severe developmental abnormalities [113, 114]. Regions of segmental duplications involved in evolutionary rearrangements can be included in the rearrangements related to human disease as well. Furthermore, chromosome 3 includes a chemokine receptor gene cluster as well as numerous loci involved in multiple human cancers. At least 505 disease loci have been mapped to this chromosome. A large number of cancer lesions have also been mapped to it and cancer breakpoints likely correlate with the four known breakable sites on the chromosome. The paper concludes that chromosome 3 is a rich resource for the study of evolution histories and for understanding

of human variations and diseases.

In our experiments, the mouse chromosome 9 and the Mouse genome (mm9) [115] have been used as the databases for predicting the genes of HMR195 and human chromosome 3, respectively. The mouse genome has been assembled by NCBI and the sequence has been masked by RepeatMasker. We also tested our algorithms on a dog genome database. The database we selected is Canis_familiaris.BROADD2.48 with DNA sequences in fasta format. This database was updated on November 30th, 2007 from ENSEMBL's ftp website [116]. It contains the full sequence of the assembly of Canis familiaris genome. The sequence is the masked genomic DNA; interspersed repeats and low complexity regions are detected and masked.

5.3.2 Results on HMR195

In this experiment, the data set is HMR195 which has been concisely introduced in Chapter 4. We choose this data set because this benchmark is widely tested by other programs including ExonHunter. We evaluated the results by comparing with the annotation file of HMR195, which can be downloaded from the author's website [117]. Since ExonHunter performed better than SGP2, we compared the results of our algorithms with the result of ExonHunter.

Results on the Mouse Database

To run ExonHunter combining the external evidences, some parameter files of the species are needed. These parameter files can be obtained from the ExonHunter's website or from one of the authors Tomas Vinar [118]. The parameter file package

used in our work is written for human. We apply all default external evidences: repeat, alignment of human ESTs database, alignment of mouse ESTs database and alignment of human protein database. For masking repeat sequences, RepeatMasker is applied. WUBLAST and SIM4 are used for aligning the human and mouse ESTs databases. BLASTX is selected for aligning the human protein database. Those databases have been downloaded from the NCBI ftp website [119]. First, tBLAST included in WUBLAST is used to obtain the alignment of HMR195 sequences and our mouse database. Then SGP2 predicts genes based on the HMR195 sequence file and the alignment.

After running the first algorithm on ExonHunter and SGP2 result files, it predicts 212 genes, 1398 exons (208 initial exons, 1228 internal exons and 212 terminal exons) and 1171 introns. The results are compared with the ExonHunter's results and the comparison is presented in Table 5.1. ExonHunter predicts 225 genes, 1044 exons (177 initial exons, 642 internal exons and 177 terminal exons) and 819 introns. Except the sensitivity and specificity on nucleotide level, other measurements of the first algorithm are slightly lower than those of ExonHunter. Our nucleotide sensitivity reaches 100%, which means this algorithm is very sensitive to the nucleotides.

The second algorithm is the one at the gene level. It predicts 225 genes, 1041 exons (224 initial exons, 818 internal exons and 224 terminal exons). Table 5.1 presents the comparison of the performance of our algorithm and ExonHunter.

This algorithm performs almost as well as ExonHunter. The measurements on nucleotide level and the sensitivity on transcript level are higher than those of ExonHunter. Whereas all the other measurements are close to ExonHunter's. Table 5.2 presents the statistics of the predicted signals. "Pred" means the number of predicted

Table 5.1: Performance Comparison of Our Two Algorithms and ExonHunter on HMR195 (Mouse Database)

	Nucleotide-Level		Transcript-Level		Exon-Level		Gene-Level	
	Sn	Sp	Sn	Sp	Sn	Sp	Sn	Sp
1st Algorithm (Exon Level)	100.00%	49.63%	39.19%	13.74%	86.09%	25.73%	39.19%	13.74%
2nd Algorithm (Gene Level)	99.96%	64.47%	55.41%	18.22%	88.73%	35.54%	55.41%	18.22%
ExonHunter	98.37%	37.89%	51.03%	19.11%	88.81%	35.73%	56.58%	19.11%

Table 5.2: Comparison of the Two Algorithms and ExonHunter on Signals of HMR195 (mouse)

		ExonHunter	1st Algorithm	2nd Algorithm
Splice Acceptor	Pred	819	1276	876
	Correct	326	401	355
	Sn	94.48%	94.46%	94.73%
	Sp	39.80%	31.43%	40.53%
Splice Donor	Pred	819	1377	1031
	Correct	330	415	417
	Sn	95.93%	95.63%	96.79%
	Sp	40.29%	30.14%	40.45%
Start Codon	Pred	225	211	225
	Correct	60	57	58
	Sn	80.87%	74.32%	78.38%
	Sp	26.67%	27.01%	27.11%
Stop Codon	Pred	225	211	225
	Correct	67	52	66
	Sn	91.34%	70.27%	87.84%
	Sp	29.38%	24.64%	29.33%

signals and "Correct" stands for the correctly predicted signals. "Sn" and "Sp" indicate the correct sensitivity and specificity of the predicted signals. As we can see, our two algorithms predict more correct splice acceptor and donor sites than ExonHunter does. But because the first algorithm predicts quite more splice sites than ExonHunter and the second algorithm do, its correct sensitivities and specificities are lowered. However, the second algorithm has higher sensitivity and specificity on the splice sites than ExonHunter. Other measurements are pretty close to those of ExonHunter except the specificity of start codons.

Results on the Dog Database

The HMR195 data set is aligned with this Canis database by tBLAST. Then SGP2 predicts genes based on the alignment and the HMR195 sequences. The results from ExonHunter are not changed, since the parameter files for the alignment in ExonHunter is just for mouse. For the first algorithm, it predicts 212 genes, 1385 exons (207 initial exons, 1191 internal exons and 212 terminal exons) and 1107 introns. Table 5.3 shows the comparison of the results from the first algorithm using dog database and from ExonHunter. As we can see, like the algorithm using the mouse database, the algorithm on Canis database still has better performance on nucleotide level than ExonHunter. However, other measurements are lower than those of ExonHunter.

The second algorithm predicts 231 genes, 1046 exons (224 initial exons, 825 internal exons and 224 terminal exons) and 815 introns. Its comparison with ExonHunter is shown in Table 5.3.

Table 5.3: Performance Comparison of Our Two Algorithms and ExonHunter on HMR195 (Dog Database)

	Nucleotide-Level		Transcript-Level		Exon-Level		Gene-Level	
	Sn	Sp	Sn	Sp	Sn	Sp	Sn	Sp
1st Algorithm (Exon Level)	100.00%	61.00%	28.38%	9.91%	82.25%	24.77%	28.38%	9.91%
2nd Algorithm (Gene Level)	99.96%	64.29%	54.05%	17.32%	88.73%	35.37%	54.05%	17.32%
ExonHunter	98.37%	37.89%	51.03%	19.11%	88.81%	35.73%	56.58%	19.11%

Still, the performance of this algorithm is very close to ExonHunter's performance except that the measurements on nucleotide level and the sensitivity on transcript level are higher than those of ExonHunter. In Table 5.4, the statistics and measurements of the predicted signals are given. The statement of each item is same with Table 5.2. Our two algorithms predict more splice sites and more correct ones than ExonHunter does. The first algorithm has a better sensitivity on splice acceptor than ExonHunter. The second one shows better performance on splice acceptor than ExonHunter. Moreover, it is more sensitive on detecting splice donor than ExonHunter, but not as specific as ExonHunter is. As for the start and stop codons, our algorithms are not as good as ExonHunter. However, the second algorithm is slightly more specific for start codons than ExonHunter, because it predicts two more start codons than ExonHunter does.

Table 5.4: Performance Comparison of the Two Algorithms and ExonHunter on Signals of HMR195 (dog)

		ExonHunter	1st Algorithm	2nd Algorithm
Splice Acceptor	Pred	819	1279	886
	Correct	326	392	361
	Sn	94.48%	94.75%	94.75%
	Sp	39.80%	30.65%	40.74%
Splice Donor	Pred	819	1326	1033
	Correct	330	396	416
	Sn	95.93%	94.17%	96.79%
	Sp	40.29%	29.86%	40.27%
Start Codon	Pred	225	212	231
	Correct	60	53	62
	Sn	80.87%	68.92%	78.38%
	Sp	26.67%	25.00%	26.84%
Stop Codon	Pred	225	212	231
	Correct	67	38	63
	Sn	91.34%	51.35%	83.78%
	Sp	29.38%	17.92%	27.27%

5.3.3 Results on Chromosome 3 of Homo Sapiens (D3S1259-D3S3659)

The motivation of the thesis was to predict genes and discover some novel genetic information related to sudden cardiac death, on the sequence from D3S1259 to D3S3659 on chromosome 3p25 of Homo Sapiens. The length of the sequence is about 10Mbp (12073681-22914093). In this section, we will present the predictions obtained by running our algorithms and compare the results with those predicted by SGP2. Because SGP2's prediction is better than ExonHunter's prediction. Also, SGP2 has predicted the genes on chromosome 3 and these predictions can be downloaded from a website [120].

In this experiment, we did a slight change for the first algorithm. We have considered a parameter from the result file of ExonHunter: the percentage of the predicted coding region that is supported by the evidence. To calculate this percentage, "CDS_support_abs" and "transcript_length", which are provided in the result file of ExonHunter, are applied. "CDS_support_abs" refers to the length of the predicted exon that is supported by evidence. "transcript_length" refers to the length of a predicted gene. We divided "CDS_support_abs" by "transcript_length". Then a scale, which is in the range of [0,1], has been obtained. It is used to separate exons into two groups. One group contains the exons whose percentages are higher than or equal to 0.5, while another group includes the exons whose percentages are lower than 0.5. When predictions are overlapping, the former group will be combined with the SGP2 results that are higher than SGP2's threshold, whereas the latter group will be combined with the SGP2 results that are lower than the threshold; otherwise,

the algorithm considers both predictions from the two gene finders when percentages of exons from ExonHunter are higher than 0.5 and SGP2's scores are higher than the threshold. This is shown in Figure 5.3. The second algorithm is same, but only uses

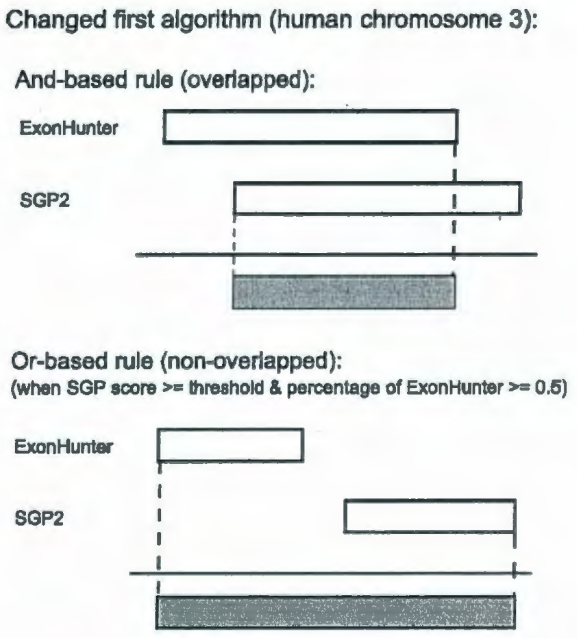

Figure 5.3: Graphical explanation to the changed first algorithm (at exon level). The boxes with shadows represent the predicted exons.

the changed first algorithm for predicting exons.

We have evaluated our results at nucleotide and exon levels by comparing the latest annotations of the sequence D3S1259-D3S3659 on chromosome 3. The annotation file in GTF format can be obtained from Ensemble's website [121]. It includes 78 genes, 628 exons (65 initial exons, 500 internal exons and 62 terminal exons) and 568 introns. Furthermore, with respect to the signals, it contains 562 splice acceptor sites, 565 splice donor sites, 98 start codons and 102 stop codons. We compared the results

of our algorithms with the result of SGP2 because SGP2 had better performance on the sequence than ExonHunter did.

Results on the Mouse Database

To run ExonHunter on this sequence of chromosome 3, the same parameter file package for human has been used as the one used for predicting the HMR195 data set. Also, external evidence is the same: alignments of human ESTs database, mouse ESTs database and human protein database. To get the predictions from SGP2, the mouse genome (mm9) has been aligned with the sequence of chromosome 3 by tBLAST.

Table 5.5: Performance Comparison of Our Two Algorithms and SGP2 on Chromosome 3 (Mouse Database)

	Nucleotide-Level		Exon-Level		Gene-Level	
	Sn	Sp	Sn	Sp	Sn	Sp
1st Algorithm (Exon Level)	83.19%	78.74%	63.22%	64.45%	-	-
2nd Algorithm (Gene Level)	85.07%	77.25%	64.65%	61.52%	-	-
SGP2	44.14%	39.71%	34.24%	32.28%	-	-

* "-" stands for no statistics for this subject.

Table 5.5 gives the comparison on the performance of the first and second algorithms and SGP2 at nucleotide and exon levels. Compared with SGP2, our two

algorithms have a large improvement on the accuracy. The performance of our algorithms are pretty close. But the second algorithm is more sensitive than the first algorithm, while the first one is more specific than the second one.

In Table 5.6, the prediction on signals of SGP2, the first algorithm and the second algorithm is listed and compared. Apparently, our two algorithms predict more splice

Table 5.6: Comparison of the Two Algorithms and SGP2 on Signals of Chromosome 3 (mouse)

		SGP2	1st Algorithm	2nd Algorithm
Splice Acceptor	Pred	665	614	622
	Correct	228	461	464
	Sn	39.15%	78.05%	80.78%
	Sp	34.29%	75.08%	74.60%
Splice Donor	Pred	665	616	611
	Correct	231	422	432
	Sn	38.41%	76.81%	78.41%
	Sp	34.74%	68.51%	70.70%
Start Codon	Pred	1	2	102
	Correct	0	0	31
	Sn	0%	0%	42.86%
	Sp	0%	0%	30.39%
Stop Codon	Pred	1	1	102
	Correct	0	0	0
	Sn	0%	0%	0%
	Sp	0%	0%	0%

sites and more correct one than SGP2 does. Their sensitivities and specificities are all better than those of SGP2. Except for the second algorithm on start codons, none of them gives an acceptable performance on predicting start codons and stop codons. Moreover, the second algorithm predicts signals better than the first algorithm does

and it is the only one which discovers some start codons.

Results on the Dog Database

The steps to apply the dog database are the same as those of the mouse database. The same result file from ExonHunter has been used. The dog database has been aligned with the sequence of human chromosome 3 to predict genes by SGP2. Then the result file from ExonHunter has been combined with the result file from SGP2.

Table 5.7 lists the comparison on the performance of the first and second algorithms and SGP2. The performance of our algorithms is better than SGP2, except for the nucleotide specificity. As for the first algorithm, the specificities and sensitivities at both levels are not in a balance; the specificities are slightly higher than those of the second algorithm, while the sensitivities are much lower than those measured on the predictions from the mouse database. Regarding the second algorithm, it bal-

Table 5.7: Performance Comparison of Our Two Algorithms and SGP2 on Chromosome 3 (Dog Database)

	Nucleotide-Level		Exon-Level		Gene-Level	
	Sn	Sp	Sn	Sp	Sn	Sp
1st Algorithm (Exon Level)	85.11%	34.67%	64.33%	39.84%	-	-
2nd Algorithm (Gene Level)	84.19%	77.62%	64.17%	62.48%	-	-
SGP2	44.14%	39.71%	34.24%	32.28%	-	-

* "-" stands for no statistics for this subject.

ances the sensitivity and specificity. It is much more specific than the first algorithm at either of the levels.

We compared the results of SGP2, the first algorithm and second algorithm on predicting signals. Table 5.8 shows the statistics. Still, our two algorithms have better

Table 5.8: Comparison of the Two Algorithms and SGP2 on Signals of Chromosome 3 (dog)

		SGP2	1st Algorithm	2nd Algorithm
Splice Acceptor	Pred	665	1013	607
	Correct	228	474	459
	Sn	39.15%	80.78%	79.89%
	Sp	34.29%	46.79%	75.62%
Splice Donor	Pred	665	1014	601
	Correct	231	430	429
	Sn	38.41%	78.05%	77.88%
	Sp	34.74%	42.41%	71.38%
Start Codon	Pred	1	2	96
	Correct	0	0	31
	Sn	0%	0%	42.86%
	Sp	0%	0%	32.29%
Stop Codon	Pred	1	2	96
	Correct	0	0	0
	Sn	0%	0%	0%
	Sp	0%	0%	0%

performance than SGP2 has and all of them do not predict start codons and stop codons well except for the second algorithm on start codons. On the database of the Canis familiaris, the first algorithm predicts more splice sites and correct ones than the second algorithms. In addition, the first algorithm works better on this database than on the mouse database.

In addition to the evaluation of our predictions on the mouse and dog database, we analyze our predictions of chromosome 3 by using UCSC genome browser [122]. This analysis can illustrate if our algorithms likely find some novel genes and exons. We

Table 5.9: Candidate Predictions and Annotated Predictions from 10 Samples

	Candidates		In the Annotations	
	Genes	Exons	Genes	Exons
Predictions (Mouse Database)	4	37	19	161
Predictions (Dog Database)	3	90	16	233

choose 10 different regions and compare the annotations and our predictions falling in the 10 regions in UCSC genome browser. Totally, in the 10 regions the annotations include 11 genes and 224 exons, the predictions on the mouse database contain 23 genes and 198 exons, and the predictions on the dog database have 19 genes and

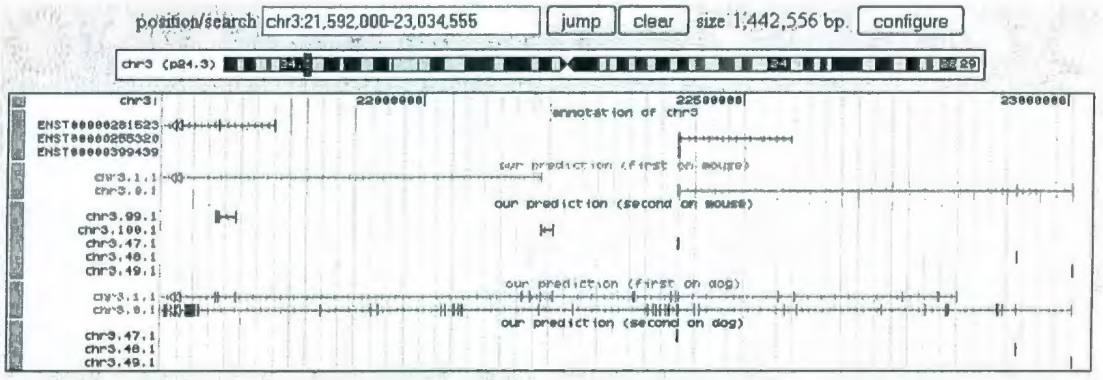

Figure 5.4: Tracks in UCSC Genome Browser on Human Chr3 (21,282,630-23,083,624)

323 exons. Table 5.9 shows the numbers of predicted genes and exons which are in the annotations and are candidates. We present one sample run in UCSC genome browser in Figure 5.4.

Also, we compared the results obtained by applying the mouse database and the canis database. Combination of the two gene finders greatly improved the accuracy at nucleotide and exon levels and thus is helpful for finding novel genetic information on chromosome 3. In general, the mouse database performs better than the canis database for finding genes on this sequence. By comparing the mouse genome, more exons and nucleotides can be detected and the correctness of the predictions is reliable as well. In our experiments, the canis genome also advanced the performance on predicting genes on the sequence, but the specificities are not satisfying on the first algorithm. For the first algorithm, the canis genome made a bit more predictions than the mouse genome. For the second algorithm, the genomes of the two species achieved similar performance.

5.3.4 Discussion

With regard to the experiments on the HMR195 data set, the performance of our algorithms using both databases is quite close to that of ExonHunter. Briefly, the algorithm on the gene level works better than on the exon level. Plus, the results on the mouse database are more accurate than those on dog database.

The first algorithm works not as well as ExonHunter except for the performance at nucleotide level - its nucleotide sensitivity reaches 100%. We analyze that it is because the purpose of this algorithm is to consider exons rather than genes and thus

more exons have been integrated from the predictions of SGP2. Since our algorithm fundamentally considers all the candidates from ExonHunter and extra exons given by SGP2 are combined only if their probability scores are higher than the threshold, the specificities are possibly decreased but the sensitivities for exons and the accuracy for detecting nucleotides would be increased. The second algorithm works better overall than the first one. It is more sensitive than ExonHunter at nucleotide and transcript levels. In addition, it can discover more correct nucleotides than ExonHunter and the first algorithm do. This is because this algorithm considers genes predicted by both programs and also counts the exons belonging to the candidate genes as much as it can. Therefore it can guarantee the accuracy of genes and meanwhile have much more information about exons and nucleotides.

Concerning the results on the mouse database and dog database respectively, the predictions from the comparison with mouse database likely provide more accurate external evidence. But the evidence acquired by comparing with dog database help predict more genes and exons. This may be caused by the threshold defined in the algorithms. However, it is promising that the dog database probably supplies some novel genetic information.

The experiments of the two algorithms on the HMR195 dataset proved that our algorithms has increased the accuracies on nucleotide and transcript levels by integrating the predictions from the comparison with mouse and dog into ExonHunter. The first algorithm can detect more genetic information than ExonHunter, whereas the second one partially has a higher accuracy than ExonHunter does. This indicates that our algorithms could be helpful for discovering novel genes and genetic information on the human chromosome 3, which are related to sudden cardiac death.

In the experiments on the sequence of chromosome 3, our algorithms have been compared with the predictions published on SGP2's website. At nucleotide and exon levels, our algorithms have a significantly better performance than SGP2 does.

When the mouse database is applied, the two algorithms exhibit similar accuracy. The first algorithm is moderately more specific than the second one, but the second algorithm displays higher sensitivities. Concerning the dog database, the first algorithm has an abrupt decline on the specificities on contrast with the specificities on the mouse database, though its specificity of exon level is higher than SGP2's specificity. Despite of this, the second algorithm keeps a good performance on the dog database. In the experiment on the dog database, conversely, the first algorithm is more sensitive than the second one while the second one has much higher specificities than the first algorithm has. Theoretically, the first algorithm should be able to detect more coding sequences than the second one does. However, we have considered the percentages of the predictions matching the evidence from the ExonHunter's result file. Thus these parameters could restrict the number of considered exons and coding regions from ExonHunter, which is good at detecting as much genetic information as it can. Regarding the first algorithm's low specificities on the dog database, the algorithm predicts much more coding sequences than the annotations have; it includes 1014 exons, 1011 introns, 1013 splice acceptor sites and 1014 splice donor sites. The large quantities affect the quality of the prediction, which may lower the specificity. In spite of that, because the second algorithm firstly considers genes predicted by both gene finders and then the exons belonging to the candidate genes, it can keep a good sensitivity and specificity.

Also, we compared the results on the mouse database and the dog database.

Overall, the mouse database performs better than the dog database does for finding genes on this sequence. The predictions made on the mouse database provide more exons and nucleotides and have a satisfying correctness as well. Compared with the measurements of the results from SGP2, the application of the dog genome also helps predict genes on the sequence. But the specificities of the first algorithm are low. With regard to the first algorithm, a bit more predictions are made by using the dog genome than the mouse genome. Our study proves that the comparative results from both the mouse genome and the dog genome are crucial to disclose more genetic information for the sequence of chromosome 3. The mouse genome likely performs better than the dog genome. This may because the study of the mouse genome is more mature than the study of the dog genome. Moreover, it is possible that this segment on chromosome 3 has less common genetic information with dogs than other parts of human chromosomes.

Conclusion

This chapter introduced the algorithms we developed to combine the outputs from ExonHunter and SGP2. Furthermore, it presented experiments on the HMR195 data set and on the sequence between the markers D3S1259 and D3S3659 on human chromosome 3. Based on our studies on the methods for predicting genes, we narrowed down our work to combination methods and cross-species comparative analysis. After further studies on some gene finding programs using either of the two types of methods, we sought to develop a system for three purposes. First, our work aimed at exploiting the advantages of cross-species comparative analysis by combining it

with other evidence obtained. Second, we attempted to implement a system so that researchers could easily work with this system and flexibly combine evidence from the programs they choose. Last, despite simple usage, the system should provide plenty of beneficial predictions of the genetic information.

We developed two algorithms: one only considers exons and another considers both genes and exons. Both algorithms follow And-based and Or-based rules to combine the evidence depending on if the probability scores of the predictions from SGP2 are higher or lower than the threshold. As for the cross-species comparative analysis, we chose the databases of Mus musculus and Canis familiaris, because many studies discovered that there are many conserved elements in both the genomes of Homo sapiens and Mus musculus. The genome sequence of Canis familiaris has been published recently and some researchers presented that the database of Canis familiaris would be a powerful tool for discovering human genome since this species is quite close to humans. By testing our algorithms on the data set HMR195, it was observed that the second algorithm has better predictions than the first algorithm on both databases according evaluation of the results. Because the first algorithm is supposed to provide more genetic information than the second one, it has a better performance at nucleotide level but lowers down the accuracy at gene and transcript levels. Regarding the performance on the mouse database and dog database, the measurements of the mouse database show that the mouse genome makes more precise gene prediction on humans than the dog genome does. But the evidence predicted by comparing with the dog database supplies more possible genes and exons. Since we chose the mouse chromosome 9 as the database and it appears to have pretty many homologies with human genomes, it may make the database have better performance.

Moreover, the data set HMR195 includes some sequences of mouse and thus it could be another reason that the mouse database is more outstanding than the dog database. However, our work could give evidence that the dog database may be helpful to detect some novel genetic information hidden in the human genome. The performance on the HMR195 data set is on the track which we supposed it should be and therefore it is proved that our designed algorithms can implement our purpose for detecting the genetic information on human chromosome 3: discovering more genetic information but also keeping a reasonable correctness. Then we ran our algorithms on the sequence of chromosome 3 to find more genetic information that is related to sudden cardiac death. In this testing, we used the mouse genome (mm9) as the mouse database. In this experiment, we considered a parameter for the ExonHunter's predictions in the first alsorithm: the percentage of the exons supported by evidence. By comparing the predictions of chromosome 3 provided on SGP2's website, our algorithms reveal a higher accuracy. As we expected, the second algorithm worked better than the first algorithm, though their performance on both databases had no significant difference. The second algorithm keeps a good balance on the sensitivity and specificity and also has the close performance on both databases. The first algorithm performs as well as the second one on the mouse database, but its specificities are much lower on the dog database. Like the experiments on the HMR195 data set, the mouse database performs better than the dog database. We analyze that it could because that the mouse genome has been known further than the dog genome. Another possibility is that this sequence on chromosome 3 has less genetic information conserved in the dog genome than the other sections of human chromosomes.

Through this work, we prove that a system with an interface for combining distinc-

tive evidence can be developed. The interface will eliminate the difficulties from different formats of prediction files and simplify the combination process. Furthermore, our study shows comparing the human genome with different species can advance the performance of the present gene finding programs. The cross-species comparative analysis is a powerful tool which is worth being considered. For the future work, we will endeavor to combine more gene finding programs with diverse advantages on predicting genes. How to find a more decent method to obtain a more effective threshold is critical for next step of the future work. We consider that genetic algorithms are commendable for determining a more accurate threshold. Also, the committee decision maker presented in Peter J. Bentley's work [123] could be an interesting solution. In a word, the future work for the system will be more concentrated on how to develop a better method to decide the threshold without the probability scores provided and the rules to combine more evidence.

Chapter 6

Conclusion

This thesis has presented a study for improving the accuracy of finding human genes from the DNA sequences and an attempt for simplify the process of combination in gene finders. Two algorithms, considering both combination methods and cross-species comparative sequence analysis, have been proposed based on the study of the gene finders using combination methods and cross-species comparative sequence analysis. Consequently, the algorithms have been applied to a sequence on human chromosome 3 for detecting genes and genetic information.

Since the two basic approaches for predicting genes had fallen behind the requirement of disclosing more novel genes and genetic information, researchers have found that the combination of the approaches or gene finders could provide some innovational genetic information. Three prominent gene finders, GAZE, JIGSAW and ExonHunter, have been studied. They combine variable evidence. By comparing with some outstanding gene finders which use the two basic approaches, the three gene finders have shown some encouraging advantages and as well as some improve-

ments. Furthermore, cross-species comparative sequence analysis has been paid more and more attention because of the increasing understanding of more species. In addition, by the study of gene finders using combination methods, it has been noticed that comparative sequence analysis has not been efficiently combined into these gene finders. Therefore, we investigated the research on the gene finders applying comparative sequence analysis. Our study has confirmed that the genomes of Mus musculus and Canis familiaris have the potential to discovering some novel human genes and genetic information.

Besides improving the accuracy of gene finders, our study also initiates us consider exploring a way to make combining diverse evidence into a gene finder simpler. This thesis has described two algorithms developed in this research work. It has demonstrated how the two algorithms combined the gene finders which applied combination methods and comparative sequence analysis, and what type of genetic information they were good at predicting respectively. The algorithms have been run on the HMR195 dataset and a sequence on human chromosome 3. The evaluations of results have been also given so that their performance has been available for being compared and discussed.

Our work has achieved three results. First, our work confirms that the performance of ExonHunter can be enhanced by integrating predictions from SGP2. It indicates that evidence obtained by cross-species comparative sequence analysis is helpful for improving the accuracy of current gene finders using combination methods. Furthermore, comparing Homo sapiens with other species is a good idea to enhance the performance on predicting human genes. Second, we combine different evidence in a simple way, but also keep the accuracy of predictions satisfying. It

proves that there is a promising possibility to develop a system which has a more friendly interface than current gene finders to combine various evidence. Third, our work predicts more correct predicted exons and genes on human chromosome 3 than the predictions published by SGP2. In addition, our analysis done in UCSC genome browser indicates that our algorithms likely discover some new genes and exons on chromosome 3. This would provide researchers wider but more accurate range to discover the genes that cause sudden cardiac death.

Our work has shown a more significant improvement at the nucleotide and the transcript level than at the gene and the exon levels. We analyzed it is probably because our algorithms have increased the amount of predicted genes, exons and signals. In addition, as ExonHunter does not provide probability scores that could be essential for determining the threshold, our work could have only considered the scores from SGP2 and this might restrict the performance of our algorithms. Therefore, for the future work, a nature next step is to ask how to attain a more effective threshold. Since ExonHunter provides the numeric information about how much the predictions match the evidence, this information is possibly useful and could be coordinated with the probability scores of SGP2 to decide the threshold. Plus, for the determination of the threshold, genetic algorithms or the committee decision maker presented in Peter J. Bentley's paper [123] are interesting to be considered. Moreover, the next step of the research could be endeavored to combine more gene finding programs with diverse advantages on predicting genes. Also, other species can be compared in the module of comparative sequence analysis. In a word, the future work for the system will be more concentrated on how to develop a better method to decide the threshold without probability scores provided and to produce more powerful rules to flexibly

combine more evidence.

Bibliography

[1] Genis Parra, Pankaj Agarwal, Josep F. Abril, Thomas Wiehe, James W. Fickett, and Roderic Guigó. Comparative Gene Prediction in Human and Mouse. *Genome Res.*, 13:108–117, 2003.

[2] http://bibiserv.techfak.uni-bielefeld.de/agenda.

[3] Kevin L. Howe, Tom Chothia, and Richard Durbin. GAZE: A Generic Framework for the Integration of Gene-Prediction Data by Dynamic Programming. *Genome Res.*, 12:1418–1427, 2002.

[4] Kevin Howe. *Gene prediction using a configurable system for the integration of data by Dynamic Programming.* PhD thesis, University of Cambridge, February 2003.

[5] Jonathan E. Allen and Steven L. Salzberg. JIGSAW: Integration of Multiple Sources of Evidence for Gene Prediction. *Bioinformatics*, 21:3596–3603, 2005.

[6] Bronislava Brejovà. *Evidence Combination in Hidden Markov Models for Gene Prediction.* PhD thesis, the University of Waterloo, 2005.

[7] Chris Burge and Samuel Karlin. Prediction of complete gene structures in human genomic DNA. *Journal of Molecular Biology*, 268:78–94, 1997.

[8] Katsuhiko Murakami and Toshihisa Takagi. Gene recognition by combination of several gene-finding programs. *Bioinformatics*, 14:665–675, 1998.

[9] Jonathan Edward Allen. *Predicting Gene Structure In Eukaryotic Genomes.* PhD thesis, The Johns Hopkins University, 2006.

[10] Jin Xiong. *Essential Bioinformatics.* Cambridge University Press, second edition, 2006.

[11] Steven L. Salzberg. Gene Discovery in DNA Sequences. *IEEE Intelligent Systems*, 14(6):44–48, 1999.

[12] M. Burset and R. Guigo. Evaluation of gene structure prediction programs. *Genomics*, 34:353–367, 1996.

[13] Pearson H. Genetics: What is a Gene? *Nature*, 441:398–401, 2006.

[14] Elizabeth Pennisi. DNA Study Forces Rethink of What It Means to Be a Gene. *Science*, 316:1556–1557, 2007.

[15] Neil C. Jones and Pavel A. Pevzner. *An Introduction to Bioinformatics Algorithms.* MIT Press, 2004.

[16] David Haussler. Computational Genefinding. *Trends in Biochemical Sciences and Supplementary Guide to Bioinformatics*, pages 12-15, 1998.

[17] A.V. Lukashin and M. Borodovsky. GeneMark.hmm: New solutions for gene-finding. *Nucleic Acids Res.*, 26:1107–1115, 1999.

[18] E.E. Snyder and G.D. Stormo. Identification of coding regions in genomic DNA sequences: an application of dynamic programming and neural networks. *Nucleic Acids Res.*, 21:607–613, 1993.

[19] E.E. Synder and G.D. Stormo. Identification of protein coding regions in genomic DNA. *J. Mol. Biol.*, 248:1–18, 1995.

[20] R. Guigó. Computational gene identification: an open problem. *Computers and Chemistry*, 21:215–222, 1997.

[21] Genis Parra, Enrique Blanco, and Roderic Guigó. GeneID in Drosophila. *Genome Research*, 10:511–515, 2000.

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

[23] X. Huang, R.C. Hardison, and W. Miller. A Space-Efficient Algorithm for Local Similarities. *Computer Applications in the Biosciences*, 6:373 381, 1990.

[24] X. Huang and W. Miller. A Time-Efficient and Linear-Space Local Similarity Algorithm. *Advances in Applied Mathematics*, 12:337–357, 1991.

[25] Jean-Michel Claverie. Computational Methods for the Identification of Genes in Vertebrate Genomic Sequences. *Human Molecular Genetics*, 6(10):1735–1744, 1997.

[26] Lawrence R. Rabiner. *Fundamentals of Speech Recognition.* PTR Prentice Hall, 1993.

[27] C. Burge and S. Karlin. Prediction of Complete Gene Structures in Human Genomic DNA. *J. Molecular Biology*, 268:78–94, 1997.

[28] Anders Krogh. Two methods for improving performance of an HMM and their application for gene-finding. In *Proceedings of the Fifth International Conference on Intelligent Systems for Molecular Biology*, pages 179–186, 1997.

[29] M. Borodovsky and J. McIninch. GENMARK: Parallel Gene Recognition for Both DNA Strands. *Comput. Chem.*, 17:123–133, 1993.

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

[31] Kulp D, Haussler D, Reese MG, and Eeckman FH. A generalized hidden Markov model for the recognition of human genes in DNA. In *Proc Int Conf Intell Syst Mol Biol*, pages 134–142, 1996.

[32] J. Henderson, S. Salzberg, and K. Fasman. Finding genes in DNA with a hidden Markov model. *J. Comput. Biol.*, 4:127–142, 1997.

[33] Ying Xu, R.J. Mural, J.R.Einstein, M.B. Shah, and E.C. Uberbacher. GRAIL: a multi-agent neural network system for gene identification. In *Proceedings of the IEEE*, pages 1544–1552, 1996.

[34] Christopher M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, USA, 1996.

[35] S. Salzberg, A. Delcher, K. Fasman, and J. Henderson. A Decision Tree System for Finding Genes in DNA. *Journal of Computational Biology*, 5:667–680, 1998.

[36] Sreerama Murthy, Simon Kasif, Steven Salzberg, and Richard Beigel. OC1: Randomized Induction of Oblique Decision Trees. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 322–327, 1993.

[37] http://www.biostat.jhsph.edu/~wmchen/gf.html.

[38] Liang KY, Chiu YF, and Beaty TH. A robust identity-by-descent procedure using affected sib-pairs: multipoint mapping for complex diseases. *Hum Hered*, 51:64–78, 2001.

[39] Solovyev VV, Salamov AA, and Lawrence CB. Identification of human gene structure using linear discriminant functions and dynamic programming. In *Proc Int Conf Intell Syst Mol Biol*, pages 367–375, 1995.

[40] Christopher B. Burge and Samuel Karlin. Finding the Genes in Genomic DNA. *Structural Biology*, 8:346–354, 1998.

[41] Altschul S F, Madden T L, Schaffer 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 Research*, 25:3389–3402, 1997.

[42] Gelfand MS, Mironov AA, and Pevzner PA. Gene Recognition via Spliced Sequence Alignment. *Proc Natl Acad Sci USA*, 93:9061–9066, 1996.

[43] Oliver Rinner and Burkhard Morgenstern. AGenDA: Gene Prediction by Comparative Sequence Analysis. *Silico Biology*, 2:4673–4680, 2002.

[44] Gabriela G. Loots, Ivan Ovcharenko, Lior Pachter, Inna Dubchak, and Edward M. Rubin. rVista for Comparative Sequence-Based Discovery of Functional Transcription Factor Binding Sites. *Genome Res.*, 12:832 839, 2002.

[45] L. Taher, O. Rinner, S. Garg, A. Sczyrba, and B. Morgenstern. AGenDA: gene prediction by cross-species sequence comparison. *Nucleic Acids Res.*, 32:305–308, 2004.

[46] Catherine MatheÂ, Marie-France Sagot, Thomas Schiex, and Pierre Rouze. Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Research*, 30:4103–4117, 2002.

[47] Kelly A. Frazer, Laura Elnitski, Deanna M. Church, Inna Dubchak, and Ross C. Hardison. Cross-Species Sequence Comparisons: A Review of Methods and Available Resources. *Genome Res.*, 13:1–12, 2003.

[48] Rong Chen and Hesham H. Ali. A New Approach for Gene Prediction Using Comparative Sequence Analysis. *ACM Symposium on Applied Computing*, pages 177 – 184, 2005.

[49] S. Batzoglou, L. Pachter, J.P.Mesirov, B. Berger, and E.S. Lander. Human and mouse gene structure: comparative analysis and application to exon prediction. *Genome Research*, 10(7):950–958, 2000.

[50] Aparicio S, Chapman J, Stupka E, Putnam N, Chia JM, Dehal P, et al. Whole-Genome Shotgun Assembly and Analysis of the Genome of Fugu rubripes. *Science*, 297:1283–1285, 2002.

[51] Lingang Zhang, Vladimir Pavlovic, Charles R. Cantor, and Simon Kasif. Human-Mouse Gene Identification by Comparative Evidence Integration and Evolutionary Analysis. *Genome Res.*, 13:1190–1202, 2003.

[52] M. Brudno, M. Chapman, B. Gottgens, S. Batzoglou, and B. Morgenstern. Fast and sensitive multiple alignment of large genomic sequences. *BMC Bioinformatics*, 4:66, 2003.

[53] B. Morgenstern. DIALIGN 2: improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics*, 15:211–218, 1999.

[54] http://ftp.genome.washington.edu/.

[55] B. Morgenstern, A. Dress, and T. Werner. Multiple DNA and Protein Sequence Alignment Based On Segment-to-segment Comparison. *Proc. Natl. Acad. Sci.*, USA 93:12098–12103, 1996.

[56] Dan Gusfield. *Algorithms on Strings and Trees and and Sequences: Computer Science and Computational Biology.* Cambridge University Press, second edition, 1997.

[57] W. Gish. (1996-2004) http://blast.wustl.edu.

[58] R. Guigó, S. Knudsen, N. Drake, and T.F. Smith. Prediction of gene structure. *J. Mol. Biol.*, 226:141–157, 1992.

[59] R. Guigó, E.T. Dermitzakis, P. Agarwal, Chris P. Ponting, Gens Parra, Alexandre Reymond, et al. Comparison of mouse and human genomes followed by experimental verification yields an estimated 1,019 additional genes. *Proc. Natl. Acad. Sci.*, 100:1140–1145, 2003.

[60] R. Guigó. Assembling genes from predicted exons in linear time with dynamic programming. *J. Comp. Biol.*, 5:681–702, 1998.

[61] N. Jareborg, E. Birney, and R. Durbin. Comparative analysis of noncoding regions of 77 orthologous mouse and human gene pairs. *Genome Res.*, 9:815–824, 1999.

[62] Mario Stanke, Oliver Schöffmann, Burkhard Morgenstern, and Stephan Waack. Gene prediction in eukaryotes with a generalized hidden Markov model that uses hints from external sources. *BMC Bioinformatics*, 7, 2006.

[63] Vladimir Pavlovic, Ashutosh Garg, and Simon Kasif. A Bayesian framework for combining gene predictions. *Bioinformatics*, 18:19–27, 2002.

[64] Thomas Schiex, Annick Moisan, and Pierre Rouze. EuGene: An Eukaryotic Gene Finder that Combines Several Sources of Evidence. In *Computational Biology*, pages 111–125, 2001.

[65] Biju Issac and Gajendra Pal Singh Raghava. EGPred: Prediction of Eukaryotic Genes Using Ab Initio Methods After Combining with Sequence Similarity Approaches. *Genome Research*, 14:1756–1766, 2004.

[66] Sanja Rogic, B.F. Francis Quellette, and Alan K. Mackworth. Improving gene recognition accuracy by combining predictions from two gene-finding programs. *Bioinformatics*, 18:1034–1045, 2002.

[67] Sakharkar M, Passetti F, de Souza J E, Long M, and de Souza S J. ExInt: An exon intron database. *Nucleic Acids Res.*, 30:191–194, 2002.

[68] M.G. Reese, F.H. Eeckman, D. Kulp, and D. Haussler. Improved splice site detection in Genie. *J. Comput. Biol.*, 4:311–323, 1997.

[69] M. Ashburner. A biologist's view of the Drosophila genome annotation assessment. *Genome Res.*, 10:391–393, 2000.

[70] http://www.sanger.ac.uk/Software/formats/GFF/GFF_Spec.shtml.

[71] L. Stein, P. Sternberg, R. Durbin, J. Thierry-Mieg, and J. Spieth. WormBase: network access to the genome and biology of Caenorhabditis elegans. *Nucleic Acids Res.*, 29:82–86, 2001.

[72] R. Mott. EST GENOME: A program to align spliced DNA sequences to unspliced genomic DNA. *Comput. Appl. Biosci.*, 13:477–478, 1997.

[73] A.A. Salamov and V.V. Solovyev. Ab initio gene finding in Drosophila genomic DNA. *Genome Res.*, 10:516–522, 2000.

[74] R. Guigó, P. Agarwal, J.F. Abril, M. Burset, and J.W. Fickett. An assessment of gene prediction accuracy in large DNA sequences. *Genome Res.*, 10:1631–1642, 2000.

[75] T. Down and T. Hubbard. Computational detection and location of transcription start sites in mammalian genomic DNA. *Genome Res.*, 12:458–461, 2002.

[76] K.D. Pruitt, T. Tatusova, and D.R. Maglott. NCBI reference sequence (RefSeq): a curated non-redundant sequence database of genomes and transcripts and proteins. *Nucleic Acids Research*, 33:D501–504, 2005.

[77] J. L. Ashurst, C.-K. Chen, J. G. R. Gilbert, K. Jekosch, S. Keenan, P. Meidl, et al. The Vertebrate genome annotation(Vega) database. *Nucleic Acids Res.*, 33:459–465, 2005.

[78] http://hgdownload.cse.ucsc.edu/goldenPath/hg17/database.

[79] Y. Lee, J. Tsai, S. Sunkara, S. Karamycheva, G. Pertea, R. Sultana, et al. The TIGR gene indices: clustering and assembling EST and known genes and integration with eukaryotic genomes. *Nucleic Acids Res.*, 33:71–74, 2005.

[80] Paul Flicek, Evan Keibler, Ping Hu, Ian Korf, and Michael R. Brent. Leveraging the mouse genome for gene prediction in human: from whole-genome shotgun reads to a global synteny map. *Genome Res.*, 13:46–54, 2003.

[81] http://genome.imim.es/gencode/workshop2005.html.

[82] M. Stanke and S. Waack. Gene prediction with a hidden Markov model and a new intron submodel. *Bioinformatics*, 19:II215–II225, 2003.

[83] B. Brejova, D.G. Brown, and T. Vinar. Optimal DNA signal recognition models with a fixed amount of intrasignal dependency. *Algorithms and Bioinformatics: 3rd International Workshop(WABI)*, 2812 of LNBI:78–94, 2003.

[84] B. Brejova and T. Vinar. A better method for length distribution modeling in HMMs and its application to gene finding. *Combinatorial Pattern Matching(CPM)*, 2373 of LNCS:190–202, 2002.

[85] Roger Fletcher. *Practical Methods of Optimization.* Wiley, second edition, 2000.

[86] ENCODE Project Consortium. The ENCODE (ENCyclopedia Of DNA Elements) project. *Science*, 306(5696):636–640, 2004.

[87] Broňa Brejovà, Daniel G. Brown, Ming Li, and Tomàš Vinař. ExonHunter: a comprehensive approach to gene finding. *Bioinformatics*, 21 Suppl. 1:i57–i65, 2005.

[88] M. Stank and S. Waack. Gene prediction with a hidden Markov model and a new intron submodel. *Bioinformatics*, 19(Suppl. 2):II215–II225, 2003.

[89] M. Burset, I.A. Seledtsov, and V.V. Solovyev. SpliceDB: database of canonical and noncanonical mammalian splice sites. *Nucleic Acids Research*, 29(1):255–259, 2001.

[90] Sudden cardiac death still a mystery - Heart Disease. USA Today (Society for the Advancement of Education), February 2004. http://findarticles.com/p/articles/mi_m1272/is_2705_132/ai_113302305.

[91] Douglas P Zipes MD. Epidemiology and mechanisms of sudden cardiac death. *Can J Cardiol*, 21(suppl A):37A–40A, 2005.

[92] Dan E. Arking, Sumeet S. Chugh, Aravinda Chakravarti, and Peter M. Spooner. Genomics in Sudden Cardiac Death. *Circulation Research*, 94:712–723, 2004.

[93] Gene Linked to Sudden Cardiac Death Identified by UCSD School of Medicine Researchers. Website, December 2001. http://ucsdnews.ucsd.edu/newsrel/health/12%2005%20chien.htm.

[94] Hai-Chien Kuo, Ching-Feng Cheng, Robert B. Clark, Jim J. C. Lin, Jenny L. C. Lin, Masahiko Hoshijima, et al. A Defect in the Kv Channel-Interacting Protein 2(KChIP2) Gene Leads to a Complete Loss of I_{to} and Confers Susceptibility to Ventricular Tachycardia. *Cell*, 107:801-813, 2001.

[95] Nguyen-Tran V.T.B., Kubalak S.W., Minamisawa S., Fiset C., Wollert K.C., Brown A.B., et al. A Novel Genetic Pathway for Sudden Cardiac Death via Defects in the Transition between Ventricular and Conduction System Cell Lineages. *Cell*, 102:671–682, 2000.

[96] Ali A Sovari. Sudden Cardiac Death, July 17 2006. http://www.emedicine.com/med/TOPIC276.HTM.

[97] P. Brugada and J. Brugada. Right bundle branch block, persistent ST segment elevation and sudden cardiac death: a distinct clinical and electrocardiographic syndrome. A multicenter report. *J Am Coll Cardiol*, 20:1391 1396, 1992.

[98] E. Keibler and M.R. Brent. Eval: A software package for analysis of genome annotations. *BMC Bioinformatics*, 4:50, 2003.

[99] http://mblab.wustl.edu/GTF22.html.

[100] J.M. Johnson, J. Castle, and P. Garret-Engele et al. Genome-wide survey of human alternative pre-mRNA splicing with exon junction microarrays. *Science*, 302:2141–2144, 2003.

[101] Jonathan Pevsner. *Bioinformatics and Functional Genomics*. John Wiley & Sons, first edition, 2003.

[102] International team of researchers assembles draft sequence of mouse genome, May 2002. http://www.genome.gov/10002983.

[103] Mouse Genome Sequencing Consortium, Waterston RH, Lindblad-Toh K, Birney E, Rogers J, Abril JF, et al. Initial sequencing and comparative analysis of the mouse genome. *Nature*, 420:520–562, 2002.

[104] Susan Mayor. Mouse genome shows many disease genes shared with humans. *BMJ*, 325(7376):1319, December, 2002.

[105] Kerstin Lindblad-Toh, Claire M Wade, Tarjei S. Mikkelsen, Elinor K. Karlsson, David B. Jaffe, Michael Kamal, et al. Genome sequence, comparative analysis

and haplotype structure of the domestic dog. *Nature*, 438:803–819, December 8, 2005.

[106] Robert K. Wayne, Eli Geffen, Derek J. Girman, Klaus P. Koepfli, Lisa M. Lau, and Charles R. Marshall. Molecular systematics of the Canidae. *Syst. Biol.*, 46:622–653, 1997.

[107] Caries Vila, Peter Savolainen, Jesus E. Maldonado, Isabel R. Amorim, John E. Rice, Rodney L. Honeycutt, et al. Multiple and ancient origins of the domestic dog. *Science*, 276:1687–1689, 1997.

[108] C. Bardeleben, R.L. Moore, and R.K. Wayne. Isolation and molecular evolution of the selenocysteine tRNA (Cf TRSP) and RNase P RNA (Cf RPPH1) genes in the dog family and Canidae. *Mol. Biol. Evol.*, 22:347–359, 2005.

[109] P. Savolainen, Y.P. Zhang, J. Luo, J. Lundeberg, and T. Leitner. Genetic evidence for an East Asian origin of domestic dogs. *Science*, 298:1610–1613, 2002.

[110] Ewen F. Kirkness, Vineet Bafna, Aaron L. Halpern, Samuel Levy, Karin Remington, Douglas B. Rusch, et al. The Dog Genome: Survey Sequencing and Comparative Analysis. *Science*, 301:1898–1903, 2003.

[111] Donna M. Muzny, Steven E. Scherer, Rajinder Kaul, Jing Wang, Jun Yu, Ralf Sudbrak, et al. The DNA sequence and annotation and analysis of human chromosome 3. *Nature*, 440:1194–1198, 2006.

[112] E.E. Eicher and D. Sankoff. Structural dynamics of eukaryotic chromosome evolution. *Science*, 301:793–797, 2003.

[113] P.W. Allderdice, N. Browne, and D.P. Murphy. Chromosome 3 duplication q21 leads to qter deletion p25 leads to pter syndrome in children of carriers of a pericentric inversion inv(3)(p25q21). *Am. J. Hum. Genet.*, 27:699–718, 1975.

[114] S.B. Stine and C.E. Clark et al. Ullrich-Turner syndrome (45,X/46,X,i[Xq]) in a child with a familial inversion of chromosome 3. *Am. J. Med. Genet.*, 12:57–62, 1982.

[115] http://hgdownload.cse.ucsc.edu/goldenPath/mm9/bigZips/.

[116] ftp://ftp.ensembl.org/pub current_fasta/canis_familiaris/dna/.

[117] http:// www.cs.ubc.ca/~rogic/evaluation/dataset.html.

[118] http://compgen.bscb.cornell.edu/~tvinar/.

[119] ftp://ftp.ncbi.nih.gov/blast/db/FASTA.

[120] http://genome.imim.es/genepredictions/index.php#HSAPsgpmouse.

[121] ftp://ftp.ensembl.org/pub/current_gtf/homo_sapiens/.

[122] http://genome.ucsc.edu/cgi-bin/hgGateway.

[123] Peter J. Bentley. "Evolutionary and my dear Watson" Investigating Committee-based Evolution of Fuzzy Rules for the Detection of Suspicious Insurance Claims. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, pages 702 709, 2000.

APPENDIX A

Algorithm 1: The main function: ehsgp.pl

Data: The result files from ExonHunter and SGP2
Result: Predictions from combining the two result files
initialization;
Check the two input files;
if *The files are right* **then**
  Create SGP object and ExonHunter object for the input files, respectively;
end
if *Choose run the first algorithm* **then**
  Run the first algorithm;
end
if *Choose run the second algorithm* **then**
  Run the second algorithm;
end

Algorithm 2: The subfunction: _parse_prediction (in SGP.pm), which parses the data from the SGP file

```
Data: The SGP file
Result: A SGP object
initialization;
while not at the end of the SGP file do
    Check if the line is from the SGP file;
    Split the predictions into fields;
    Grep the internal exon and give it an id;
    Grep the gene number;
    if It is a new gene then
        Create a new gene object;
    end
    Create an exon object;
    Add the exon to the gene;
end
for j ← 0 to The end of the exon array do
    Convert the score of the exon into the percentage format;
    if score <0.001 then
        Count the number;
    end
    if 0.001 ≤ score<0.00175then
        Count the number;
    end
    if 0.00175 ≤ score<0.0045then
        Count the number;
    end
    if 0.0045 ≤ score<0.0062then
        Count the number;
    end
    if 0.0062 ≤ score<0.0077then
        Count the number;
    end
    if 0.0077 ≤ score<0.0179then
        Count the number;
    end
    if 0.0179 ≤ score<0.05then
        Count the number;
    end
    if 0.05 ≤ score ≤ 1.0then
        Count the number;
    end
    end
    Calculate the weight mean to get the threshold p_thr;
```

Algorithm 3: The subfunction: _parse_prediction (in EH.pm), which parses the data from the ExonHunter file

Data: The ExonHunter file
Result: An ExonHunter object
initialization;
while *not at the end of the ExonHunter file* **do**
    Check if the line is from the ExonHunter file;
    Split the predictions into fields;
    Grep the internal exon and give it an id;
    Grep the gene number;
    Grep the number of transcript bases supported by the evidence;
    Grep the number of exon bases supported by the evidence;
    Calculate the percentage of the supported transcripts to get the score;
    if *It is a new gene* **then**
        Create a new gene object;
    end
    Create an exon object;
    Add the exon to the gene;
end

Algorithm 4: The subfunction: first (in ES.pm), which executes the first algorithm

Data: The object for the ExonHunter file and the object for the SGP2 file
Result: The results from combining the two result files by the first algorithm
initialization;
if *the SGP and ExonHunter objects have been defined* **then**
    if *SGP and ExonHunter have at least one prediction* **then**
        Create an object for the first algorithm;
    end
end

Algorithm 5: The subfunction: first (in ES.pm), which executes the first algorithm

Data: The ExonHunter and SGP2 objects
Result: The results from combining the two files by the first algorithm
initialization;
Check if the SGP and ExonHunter objects have been defined and have at least one prediction;
while *not at the end of the SGP file* **do**
  if *The exon belongs to the same gene as the one in the last loop* **then**
    if *Its strand is minus* **then**
      if *Its score is lower than the threshold* **then**
        Push it into the array of exons (<the threshold '-' strand);
      else
        Push it into the array of exons (>the threshold '-' strand);
      end
    else
      if *Its score is lower than the threshold* **then**
        Push it into the array of exons (<the threshold '+' strand);
      else
        Push it into the array of exons (>the threshold '+' strand);
      end
    end
  end
end
while *not at the end of the ExonHunter file* **do**
  if *The exon belongs to the same gene* **then**
    if *Its strand is minus* **then**
      if *Its score is lower than 0.5* **then**
        Push it into the array of exons (<0.5 '-' strand);
      else
        Push it into the array of exons (>0.5 '-' strand);
      end
    else
      if *Its score is lower than 0.5* **then**
        Push it into the array of exons (<0.5 '+' strand);
      else
        Push it into the array of exons (>0.5 '+' strand);
      end
    end
  end
end
Execute the subfunction ***fir_union*** on the exons higher than threshold;
Execute the subfunction ***fir_intersec*** on the exons lower than threshold;

Algorithm 6: The subfunction: fir_union (in Alg.pm), which units the exons

```
Data: The exons which scores are higher than the thresholds from
      ExonHunter and SGP2
Result: List of computed exons
initialization;
Check if there is any predictions in the SGP and ExonHunter objects;
while not at the end of the both sets of exon arrays do
    if The two exons from SGP and ExonHunter are overlapping then
        if The two exons on the same strand then
            Execute the subfunction union to extract the overlapping part;
        end
        Add the new exon into the predicted exon list and define the last exon
        of the list;
        if The start position of ExonHunter exon < The start position of SGP
        exon then
            Increase the index of ExonHunter's list;
        else
            if The start position of ExonHunter exon > The start position of
            SGP exon then
                Increase the index of ExonHunter's list;
            else
                Increase both indexes;
            end
        end
    else
        Mark the non_overlapping exons;
    end
end
```

Algorithm 7: The subfunction: fir_intersec (in Alg.pm), which intersects the exons

Data: The exons which scores are lower than the thresholds from ExonHunter and SGP2
Result: List of computed exons
initialization;
Check if there is any predictions in the SGP and ExonHunter objects;
while *not at the end of the both sets of exon arrays* **do**
  if *The two exons from SGP and ExonHunter are overlapping* **then**
    Execute the subfunction ***intersec*** to extract the overlapping part;
  else
    Set the exon from SGP as the predicted exon;
  end
  Add the new exon into the predicted exon list and define the last exon of the list;
  Increase the indexes;
end

Algorithm 8: The subfunction: union (in Alg.pm), which units the exons

Data: The exons from ExonHunter and SGP2
Result: The union of the two exons
initialization;
if *The start position of SGP exon* $<$ *The start position of ExonHunter exon* **then**
  The start position of SGP exon is the start position of the new exon;
else
  The start position of ExonHunter exon is the start position of the new exon;
end
if *The end position of SGP exon* $>$ *The end position of ExonHunter exon* **then**
  The end position of SGP exon is the end position of the new exon;
else
  The end position of ExonHunter exon is the end position of the new exon;
end
Set the exon type;

Algorithm 9: The subfunction: intersec (in Alg.pm), which intersects the exons

Data: The exons from ExonHunter and SGP2
Result: The intersection of the two exons
initialization;
if *The start position of SGP exon > The start position of ExonHunter exon* **then**
| The start position of SGP exon is the start position of the new exon;
else
| The start position of ExonHunter exon is the start position of the new exon;
end
if *The end position of SGP exon The end position of ExonHunter exon* **then**
| The end position of SGP exon is the end position of the new exon;
else
| The end position of ExonHunter exon is the end position of the new exon;
end
Set the exon type;

Algorithm 10: The subfunction: second (in ES.pm), which executes the second algorithm

Data: The object for the ExonHunter file and the object for the SGP2 file
Result: The results from combining the two result files by the second algorithm
initialization;
if *the SGP and ExonHunter objects have been defined* **then**
| **if** *SGP and ExonHunter have at least one prediction* **then**
| | Create an object for the second algorithm;
| **end**
end

Algorithm 11: The subfunction: _parse_prediction (in SEC.pm), which executes the second algorithm

Data: The object for the ExonHunter file and the object for the SGP2 file
Result: The results from combining the two result files by the second algorithm

```
initialization;
Check if the SGP and ExonHunter objects have been defined, and that SGP
and ExonHunter have at least one prediction;
if The files are right then
    while not at the end of the SGP file do
        if The exon belongs to the same gene as the one in the last loop then
            if Its strand is minus then
                Set the exon belong to the gene and push the gene into the array
                of genes that are on the minus strand;
            else
                Set the exon belong to the gene and push the gene into the array
                of genes that are on the plus strand;
            end
        end
    end
    while not at the end of the ExonHunter file do
        if The exon belongs to the same gene then
            if Its strand is minus then
                Set the exon belong to the gene and push the gene into the array
                of genes that are on the minus strand;
            else
                Set the exon belong to the gene and push the gene into the array
                of genes that are on the plus strand;
            end
        end
    end
    Execute the subfunction sec on ExonHunter and SGP2 genes which are on
    the plus strand;
    Execute the subfunction sec on ExonHunter and SGP2 genes which are on
    the minus strand;
    Add the predicted gene into the predictions;
end
```

Algorithm 12: The subfunction: sec (in SEC.pm), which intersects the overlapping genes

Data: Two arrays of genes
Result: An array of genes
initialization;
while *not at the end of the two arrays of genes* **do**
  if *two genes are overlapping* **then**
    Execute the subfunction ***gene_intersec***;
  end
end

Algorithm 13: The subfunction: gene_intersec (in SEC.pm), which intersects the overlapping genes

Data: Two genes from ExonHunter and SGP2
Result: A gene object
initialization;
Set the start position and the end position of the overlapping region;
for *each SGP exon belonging to the overlapping region* **do**
  if *its score the threshold* **then**
    Push it into the array in which all the exons have scores lower than the threshold;
  else
    Push it into the array in which all the exons have scores higher than the threshold;
  end
end

for *each ExonHunter exon belonging to the overlapping region* **do**
  if *its score the threshold* **then**
    Push it into the array in which all the exons have scores lower than the threshold;
  else
    Push it into the array in which all the exons have scores higher than the threshold;
  end
end
Execute ***fir_union*** to the ExonHunter and SGP exons, which scores are higher than the thresholds;
Execute ***fir_intersec*** to the ExonHunter and SGP exons, which scores are lower than the thresholds;
Add the predicted exons to the predicted gene object;