379
N816
No. 4605

EXON/INTRON DISCRIMINATION USING THE
FINITE INDUCTION PATTERN
MATCHING TECHNIQUE

DISSERTATION

Presented to the Graduate Council of the
University of North Texas in Partial
Fulfillment of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

By

Pamela A. Taylor, B.S., M.S.

Denton, Texas

December, 1997

Taylor, Pamela A., Exon/Intron Discrimination Using the Finite Induction Pattern Matching Technique. Doctor of Philosophy (Computer Science), December, 1997, 102 pp., 8 tables, 69 illustrations, references, 35 titles.

DNA sequence analysis involves precise discrimination of two of the sequence's most important components: exons and introns. Exons encode the proteins that are responsible for almost all the functions in a living organism. Introns interrupt the sequence coding for a protein and must be removed from primary RNA transcripts before translation to protein can occur.

A pattern recognition technique called Finite Induction (FI) is utilized to study the language of exons and introns. FI is especially suited for analyzing and classifying large amounts of data representing sequences of interest. It requires no biological information and employs no statistical functions. Finite Induction is applied to the exon and intron components of DNA by building a collection of rules based upon what it finds in the sequences it examines. It then attempts to match the known rule patterns with new rules formed as a result of analyzing a new sequence. A high number of matches predict a probable close relationship between the two sequences; a low number of matches signifies a large amount of difference between the two. This research demonstrates FI to be a viable tool for measurement when known patterns are available for the formation of rule sets.

379
N816
No. 4605

EXON/INTRON DISCRIMINATION USING THE
FINITE INDUCTION PATTERN
MATCHING TECHNIQUE

DISSERTATION

Presented to the Graduate Council of the
University of North Texas in Partial
Fulfillment of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

By

Pamela A. Taylor, B.S., M.S.

Denton, Texas

December, 1997

ACKNOWLEDGMENT

Special appreciation is expressed to Dr. Cathy Wu at the University of Texas Health Center at Tyler for supplying the original exon and intron data sets used to construct the filter and unknown test sets necessary for this project. The books, articles, and expertise contributed by Dr. Wu were also essential to this work and greatly appreciated.

TABLE OF CONTENTS

	Page
ACKNOWLEDGMENT	iii
LIST OF TABLES	vi
LIST OF ILLUSTRATIONS	vii
 Chapter	
1. INTRODUCTION	1
1.0 Introductory Statement	1
1.1 The Importance of DNA	1
1.1.1 The Relationship Between DNA and Protein	3
1.1.2 Protein Composition	5
1.1.3 DNA Signal Components	6
1.2 The Problem Statement	9
1.3 Structure for the Remainder of the Paper	11
2. PREVIOUS WORK IN EXON/INTRON RECOGNITION	12
2.0 Gene Searching Methods	12
2.1 Search by Content	14
2.2 Search by Signal	16
2.3 Integrated Systems	18
2.4 Internal Exon Predictors	26
2.5 Summary of Current Research Problems	28
3. FINITE INDUCTION	30
3.0 Introduction	30
3.1 FI Definition	33
3.1.1 Example for FI: FACTORING	34
3.1.2 Example for FI: FOLLOWING	38
3.1.2.1 EXACT FOLLOWING	38
3.1.2.2 BLIND FOLLOWING	38

3.1.2.3 REPLACEMENT FOLLOWING	39
3.2 Additional FI Considerations	39
4. MATERIALS AND METHODS	43
4.0 The Data Sets	43
4.1 RULING Construction and Behavior	44
4.2 Filter Characteristics	49
4.3 The FOLLOWINGs	51
4.3.1 Findings for EXACT FOLLOWING	52
4.3.1.1 RESIDUAL Length	54
4.3.1.2 RESIDUAL Content	56
4.3.1.3 Distance Between RESIDUAL Components	59
4.3.2 Findings for BLIND FOLLOWING	60
4.3.3 Findings for REPLACEMENT FOLLOWING	62
4.3.4 Additional Techniques	64
4.3.4.1 By 2 Overlapping Procedure	64
4.3.4.2 By 3 Overlapping Procedure	65
4.3.4.3 By 3 Non-overlapping Procedure	66
4.3.4.4 Codon Procedure	68
4.3.4.5 Passive FOLLOWING	70
4.3.4.5.1 Human, Homo Sapiens, and Gorilla Data Runs	71
4.3.4.5.2 Codon Triplet Runs	73
4.3.4.5.3 Splice Site Runs	73
5. SUMMARY	77
6. CONCLUSION	83
6.0 Conclusion	83
6.1 Recommendations	86
APPENDIX	88
BIBLIOGRAPHY	99

LIST OF TABLES

Table	Page
1. Matrix representing the signal TATAAT.	13
2. Accuracy of correct exon prediction for integrated gene search systems	26
3. Accuracy of correct internal exon prediction for gene search systems	28
4. Effect on level count by limiting sequence size	47
5. The percentage of sequences in which each rule is seen	74
6. Highest percentages of correct predictions of exons and introns for each group of splice site rules.	75
7. Accuracy of correct exon prediction for FI as compared to integrated gene search systems	84
8. The comparison of FI with internal exon prediction systems	84

LIST OF ILLUSTRATIONS

Figure	Page
1. The composition of a human genome	2
2. Conceptualization of the relationship between DNA and protein	3
3. The relational components from proteins to nucleotides.	5
4. The conceptualization of the signals of a DNA sequence	7
5. Architecture of GeneModeler	19
6. GRAIL architecture	21
7. The architecture of GenParser	24
8. FI process representation	30
9. FOLLOWING as applied to input to produce a RESIDUAL using a RULING	31
10. RULING containing implicants from three objects.....	40
11. Data set sequence sizes.....	43
12. Levels generated by test sets for all IB	45
13. Levels generated by test sets for IB 5 – N	46
14. Level comparison for EXON (S), IB 2 – IB 4	46
15. Level comparison for EXON (S), IB 5 – IB N	46
16. Level comparison for INTRON (S), IB 2 – IB 4 Appendix	89
17. Level comparison for INTRON (S), IB 5 – IB N Appendix	89
18. Level 1 average implicants for the four data sets	48

19. EXON (S) sequence size versus levels		48
20. INTRON (S) sequence size as related to levels	Appendix	90
21. EXON (S) IB 5 – IB N sequence size as related to levels	Appendix	90
22. Level comparison of exon/intron sequences		49
23. Comparison of total, filter, and unique rules for small exon data set		50
24. Unique rule composition of filter for all four data sets		50
25. Data sets for unknown sequences		51
26. Comparison of exon filters		53
27. Residual results for inductive base 5 using the large exon filter		54
28. Residual for inductive base 2 using exon filter	Appendix	91
29. Residual for inductive base 3 using exon filter	Appendix	91
30. Residual for inductive base 4 using exon filter	Appendix	88
31. Residual for inductive base 6 using exon filter	Appendix	92
32. Residual for inductive base 7 using exon filter	Appendix	92
33. Residual for inductive base 8 using exon filter	Appendix	92
34. Residual for inductive base 11 using exon filter	Appendix	93
35. Residual for inductive base 15 using exon filter	Appendix	93
36. Residual for inductive base 20 using exon filter	Appendix	93
37. Residual for inductive base M using exon filter	Appendix	94
38. Residual for inductive base N using exon filter	Appendix	94
39. Residual for inductive base 2 using intron filter	Appendix	95
40. Residual for inductive base 3 using intron filter	Appendix	95

41. Residual for inductive base 4 using intron filter	Appendix	95
42. Residual for inductive base 5 using intron filter	Appendix	96
43. Residual for inductive base 6 using intron filter	Appendix	96
44. Residual for inductive base 7 using intron filter	Appendix	96
45. Residual for inductive base 8 using intron filter	Appendix	97
46. Residual for inductive base 11 using intron filter	Appendix	97
47. Residual for inductive base 15 using intron filter	Appendix	97
48. Residual for inductive base 20 using intron filter	Appendix	98
49. Residual for inductive base M using intron filter	Appendix	98
50. Residual for inductive base N using intron filter	Appendix	98
51. Percentage of rules too large to be included in Level 1 for all IBs		55
52. Residual comparison for inductive bases 7 through 20 at Level 1 demonstrates the similarity in the results produced by these filters		56
53. Filters for inductive bases 8 through 20 are restrictive as to the number of levels provided the unknown sequences		57
54. Comparing the content of the residual for IB 5 using the large exon filter provides no discrimination between exons and introns		57
55. Residual content A< C, G< or T demonstrates little difference between exon and intron unknowns		58
56. The distance between residual components did not exhibit a difference between exon and intron unknown sequences		59
57. RESIDUAL, RESIDUAL content, and distance comparison for EXACT FOLLOWING versus BLIND FOLLOWING		61
58. RESIDUAL, RESIDUAL content, and distance comparison for EXACT FOLLOWING versus BLIND FOLLOWING versus REPLACEMENT FOLLOWING		63

59. Transform count comparison for exons versus introns	64
60. Comparison of total results for exons and introns through alphabet extension taking nucleotides 2 at a time	65
61. Comparison of total results for exons and introns through alphabet extension taking nucleotides 2 at a time	66
62. Exon – intron results of taking bases three at a time, non-overlapping, beginning at positions 1, 2, and 3	67
63. A demonstration of the dramatic difference in overlapping and not overlapping nucleotide bases three at a time	68
64. Comparing the By 3 Overlapping results with those of the By 3 Non- Overlapping and Codon procedures	69
65. Rule comparison for the special codon file of exons	70
66. Residual comparison using three unique filters built from single exon sequences and used to filter related exon and intron sequences	72
67. A comparison of the residual produced by both exon unknown sequences and intron unknown sequences as filtered by the Passive FOLLOWING as defined by codon triplets	73
68. The predictive accuracy of the four sets of selected rules is shown to be very close	76
69. The randomly generated strings “Fake Good” blend in with the actual DNA sequences. The antecedents of the randomly generated sequences “Fake Bad” are represented as an (almost) straight line.	86

CHAPTER 1

INTRODUCTION

1.0 Introductory Statement

The Human Genome Project is producing roughly a chromosome per year (Lander, Landridge, and Saccocio, 1991). It is predicted that by 1999, the amount of data sequenced will yield a full genome each year. One of the major challenges of this endeavor is developing the technology to recognize and interpret the information contained in genomic DNA sequence data. The more traditional approaches to gene isolation, including identification of conserved sequences and direct screening of DNA libraries, are effective but very laborious and expensive (Zhang, 1997). It is the purpose of this research to consider an approach to the problem of identifying genetic elements found in unknown DNA sequences. In order to formulate the problem, the following section and its three subsections are provided to aid the reader in understanding the background so a more specific problem statement can be presented.

1.1 The Importance of DNA

The human genome is the totality of genetic information in each of us. Figure 1 demonstrates that this genetic information is distributed in the genome among twenty-three pairs of separate duplexes of DNA known as chromosomes and are considered to be the bearers of our heredity (Watson, Tooze, and Kurtz, 1983). The twenty-three

chromosomes are composed of some 50,000 to 100,000 genes that encode all the properties of the human organism. Each gene is made up of DNA (deoxyribonucleic acid). DNA is a very large molecule containing up to hundreds of millions of nucleotides (Gall, 1981). The four nucleotides A (adenine), C (cytosine), T (thymine), and G (guanine) are the building blocks, most often referred to as the base elements, of DNA.

Figure 1. The composition of a human genome.

DNA is an extremely important substance. It carries within its structure the hereditary information that determines the structures of proteins. Biochemists discovered that a very broad class of molecules called proteins is responsible for almost all the functions in a living organism. To better understand the relationship between DNA and protein, figure 2 will be discussed in detail.

Figure 2. Conceptualization of the relationship between DNA and protein.

1.1.1 The Relationship between DNA and Protein.

The portion of figure 2 labeled “DNA SEQUENCE”, represents a linear sequence of the nucleotides A, C, T, and G. DNA is actually a double helix. This means that the DNA sequence is two chains of nucleotides that are oriented in such a way that each

forms hydrogen bonds to bases on the opposing chain. The nucleotide A always bonds to T and likewise, C always bonds to G. Therefore, the area occupied by the paired bases is always the same throughout the length of the DNA molecule. More importantly, each of these base pairs possesses a symmetry which, given the sequence of one chain, (for example, TCGCAT) the sequence of its partner (AGCGTA) can easily be determined.

The solid black area in the DNA sequence simply represents a portion of interest that will now be discussed. Notice that the DNA has components referred to as exons and introns. Only a small percentage of the nucleotides in a genome forms exons. Exons are the components of a DNA sequence that encode RNA (discussed in section 1.1.3) (Konopka and Owens, 1990). Exons are expressed as all or part of the polypeptide (a molecular chain of amino acids) chain of a protein. Bell (1990) estimates that the total exon coding DNA would be about three percent of the total genome. In the case of exons the encoding scheme and the genetic code itself are known. In the case of introns they are not known.

Approximately thirty percent of the remaining genome is found in segments of DNA that interrupt the coding sequences (exons) of genes. These are known as introns. An intron is a polynucleotide sequence within a gene that does not code information for the protein amino acid sequences. The function of introns is controversial (Wills, 1991). Genes contain roughly five to ten times more DNA in introns than in coding sequences. The introns interrupt the nucleotide sequence coding for a protein and must be removed from primary RNA transcripts before translation can occur. Upon the removal of the intron, its two adjacent exons must be spliced together. Once all exons are concatenated,

the resulting sequence may be translated into a particular protein. The protein may then be studied as to its functionality and appropriately classified.

1.1.2 Protein Composition

Figure 3 denotes the relational components from the protein down to the nucleotides. A protein is built up from amino acids linked to each other by peptide bonds to form linear polypeptide chains. There are only twenty different amino acids. However, a protein chain may contain any combination of from 50 to over 2000 of these amino acids giving a virtually unlimited number of possible different proteins.

Figure 3. The relational components from proteins to nucleotides.

The building blocks of coding sequences are codons. All codons contain three nucleotides (a triplet). Four possible nucleotides taken three at a time allows for sixty-four possible codons. Sixty-one of the sixty-four codons code for specific amino acids, with between one and six code words for each amino acid. The three remaining codons (TAA, TAG, and TGA) code for stop signals that indicate chain termination. Stop signals will be discussed in the next section.

1.1.3 DNA Signal Components.

Examples of other components of a DNA sequence that must be understood are identified in figure 4 on the next page. The nucleotide sequences shown for the components are reasonably universal, however, there are many other variations.

Whereas the nucleotide compositions of exons and introns are referred to as the “content” of the DNA sequence, the nucleotide representations of the other components are known as the “signals” of a sequence (Stormo, 1987 and Staden, 1990). A common approach to locating the signals of a sequence is to search for similarities to “consensus sequences”. Individual signal sites might not be identical to the consensus and different positions within the site may vary in their importance within the consensus.

The genetic information within a DNA molecule (in the cell nucleus) usually serves as the template for a large number of shorter RNA molecules (in the cell cytoplasm), most of which in turn serve as templates for the synthesis of specific polypeptide chains of protein. Transcription is the process of constructing an RNA molecule using a DNA molecule as a template with the resulting transfer of genetic information to the RNA. Promoters (the first signal shown in figure 4) precede

initial or initiating exon; the last in the series is referred to as the terminal or terminating exon. All exons between the initiating and terminating exons are noted as internal exons.

Most initiating exons begin the amino acid chain with the amino acid methionine, represented by the codon ATG. This sequence is referred to as the “start codon”.

However, this codon is also utilized internally when methionine is required elsewhere in the polypeptide chain. The terminal exon encodes the “stop codon.” As mentioned earlier, a stop codon is one of the three combinations of nucleotides that do not specify any amino acid. Rather, they indicate amino acid chain termination.

At the ends of each intron can be observed two other conserved sequences. These important nucleotide pairs are known by specific names related to their functions. The pair GT is known as the 5' splice or donor site. The pair AG is known as the 3' splice or acceptor site. Although the GT and AG pairs are used to delimit introns, they are also found quite frequently within the sequence of exons as well as introns. At the end of most eukaryotic (multi-cell) genes is the sequence AATAAA called the poly-adenylation signal. It is not, however, a transcription-termination signal. It serves as a processing signal for subsequent nuclease (any of various enzymes that promote hydrolysis of nucleic acids) to clip the RNA chain at a specific site some 10 to 15 bases further downstream where a second enzyme adds a 100-to-200-base poly-A tail.

Two predominant methods are employed in computerized searching of DNA to determine coding regions using the components discussed above. These two methods are known as search by content and search by signal and are described in detail in the next chapter.

1.2 The Problem Statement

All properties of an organism are encoded in its genome. The genome can be represented as a sequence of four symbols and then simply treated as a text carrying many (generally unknown) messages written in many (also unknown) languages. Experimental molecular biology provides evidence that the genome is non-uniform in the sense that its different regions are involved in different biological processes. Only a small percentage of all nucleotides in these genomes form exons that encode proteins. (Konopka and Owens, 1990)

The problem lies in accurately extracting the exons from a very complex text string composed of a very small alphabet and doing this in a timely manner. Nucleotide sequence data banks such as the GenBank, DNA Data Bank of Japan, and EMBL Data Library contain millions of nucleotides in thousands of sequences. Currently, the data banks enter and contain more experimentally determined (or confirmed) annotation than annotations derived solely from pattern searches and comparisons (Burks, 1990). The thrust at present is to use an automated approach to generate sequences that contain significant regions for which biological function(s) have not yet been experimentally identified or confirmed in order to greatly improve the efficiency of gene identification.

Various mathematical, statistical, and learning algorithms have been suggested for the automated evaluation of DNA sequences for the purpose of identifying the various components of the sequence in an effort to predict probable genes. There are two important measures used to gauge the success of the various algorithms. These measures are known as “sensitivity” and “specificity.” Sensitivity is defined and calculated as

$$\text{true positives} / \text{true positives} + \text{false negatives}.$$

True positives are sequences predicted to be exons and also observed as exons. False negatives are sequences predicted to be introns but are observed as exons. Specificity is defined and calculated as

$$\text{true positives} / \text{true positives} + \text{false positives}$$

where false positives are sequences predicted to be exons but are actually introns. These four data values are used in a correlation coefficient to measure overall accuracy (discussed in chapter 2). None of the techniques to date have been shown to be 100% sensitive and most have low specificity.

The purpose of this research study is to apply the technique of Finite Induction (FI) pattern recognition to the exon and intron components of DNA to determine its computational ability for distinguishing between the two. This will facilitate the automated extraction of exons from large DNA sequences for purposes of gene identification. FI builds a collection of rules based upon what it finds in the sequences it examines. It then attempts to match the known rule patterns with new rules formed as a result of analyzing a new sequence. A high number of matches predict a probable close relationship between the two sequences; a low number of matches is equated with a large amount of difference between the two.

The FI system provides the following capabilities:

- ◆ a method for representing the known objects in a concise form;

- ◆ a recognition speed $O(n)$ where n is the number of symbols representing the input object, and not dependent upon the size of the known object space (Das, Fisher, and Zhang, 1993);
- ◆ a concise pair of basic algorithms making up FI;
- ◆ a technique that is easily executed on a PC, even for complex strings such as those found in music (Fisher, Novaes, and Mobus, 1990), speech recognition, and the genome sequence.

The FI technique is described in detail in chapter 3.

1.3 Structure for the Remainder of the Paper

Chapter 2 investigates various methods utilized to discriminate exons from introns. It includes the results of these previous studies. Chapter 3 is an in-depth discussion of the Finite Induction pattern recognition technique. Definitions are given and multiple examples are demonstrated. Chapter 4 delineates the algorithms used and results obtained from a plethora of exercises conducted for this research effort. Chapter 5 evaluates these efforts in summary form. A conclusion follows in Chapter 6.

CHAPTER 2

PREVIOUS WORK IN EXON/INTRON RECOGNITION

2.0 Gene Searching Methods

Computational biology was first used successfully, by a cancer research group in 1984, to compare a newly discovered oncogene (a tumor gene) to all known genes. Since that time many methods and algorithms have been suggested for computerized searching of genes. These can be generally classified into two different approaches (reviewed by Stormo, 1987 and Staden, 1990).

The first method concentrates on the signature of recognizable coding sequences. This is known as search by content. Various coding techniques and statistical means are employed to determine the protein-coding potential of a given sequence. Search by content collects a specified number of nucleotides in a sequence referred to most often as a “window.” The window is analyzed using one or more of the three levels of composition namely the amino acids, the codons, or the base elements which are the nucleotides. From chapter 1, it was noted that protein chains are composed of from 50 to over 2000 amino acids, each amino acid is represented by from 1 to 6 codons, and codons are composed of three nucleotides. The base elements are the individual nucleotides.

Proteins share a typical amino acid composition, which can be used to rate a sequence on its likelihood of being translated into protein. Codon composition can also

be used to identify coding regions, because while a sequence window may have similar amino acid compositions and base compositions, they may have very different codon compositions. Base composition can also be used as a determinate.

The second method, known as search by signal, attempts to delimit coding regions by recognizing signal sequences such as promoters, splice sites, translation initiating and terminating sites and poly-adenylation signals (defined and illustrated in chapter 1). Matrices which contain an element for each possible base at every position within a signal site are used in this process. For example, if the signal to be recognized is TATAAT, then a matrix representing this signal might appear as given below.

Table 1. Matrix representing the signal TATAAT.

BASE ELEMENT	T	A	T	A	A	T
A	-38	19	1	12	10	-48
C	-15	-38	-8	-10	-3	-32
G	-13	-48	-6	-7	-10	-48
T	17	-32	8	-9	-6	19

The six columns represent the six positions of the signal. The four rows are in the base order of A, C, G, T from top to bottom. The element weights are assigned based on the likelihood of each base appearing at a particular position. Given the unknown sequence, ACTATAATCG, and starting with the second base element, C, the resulting score would be -93 $[(-15)+(-32)+(1)+(-9)+(10)+(-48)]$; however, beginning with the third base, the score would be 85 $[17+19+8+12+10+19]$ (Stormo, 1987). As can be seen, the matrix

also allows for the signal TAAAAT. Nevertheless, this signal would not be as strong a pronouncement as the TATAAT signal [$17+19+1+12+10+19=78$].

2.1 Search by Content

Fickett and Tung (1992) surveyed twenty-one coding measures utilized by computer search-by-content algorithms. Coding measures are functions that produce, given any sample window of a sequence, a number or vector intended to measure the degree to which a sample sequence resembles a window of exonic DNA. The study concluded that fifteen of the twenty-one measures were simply functions of some other measure. Two of the remaining six demonstrated poor performance. The four remaining measures, Dicodon/Hexamer-n, ORF, Autocorrelation/Fourier, and Run, will be discussed briefly.

The Dicodon/Hexamer-n is the measure upon which the majority of other measures are based. Therefore, it is the most fundamental and most widely used. A Dicodon/Hexamer is six nucleotides. The Hexamer-n Measure is the count of all hexamers offset by n from the starting base of a test-codon. Test codons are the successive nonoverlapping trinucleotides of an arbitrary sample window of sequence data. The Hexamer-n calculates the probability of each window based on each of four separate Markov models (three exon, one intron) using Bayes theorem. The Markov model step lengths tested were 1, 2, 3, and 5.

Whereas the Dicodon/Hexamer-n is basically a codon measure, the ORF (Open Reading Frame) analyzes the amino acid usage and base compositional bias. It also looks for the presence or absence of in-frame stop codons (TAA, TAG, or TGA). The Open

Reading Frame Measure is defined as the length of the longest stretch of test-codons, not containing a stop codon. Both the Dicodon/Hexamer-n and ORF Measures assume the DNA sequence under observation comes from a collection of genes with common features.

The Autocorrelation/Fourier Measure studies the imperfect periodicity in base occurrences. The Autocorrelation Measure is defined as follows. Let $\text{auto}(b, i)$ be the number of pairs of base b with i intervening bases. The number of such pairs expected is corrected on the basis of base composition. Thus evolves a matrix:

$$[\text{auto}(b, i) / (\text{window_length} - i - 1) (\text{frequency_of_}b)^2]$$

where $b = A, C, T, G$ and $i = 0, \dots, 9$. Whereas, the Autocorrelation Measure depends upon the types of bases, the Fourier transform depends on the overall patterns of periodicity in the sequence and not on which bases are involved in the periodic patterns. A detailed explanation of the Fourier transform is related in (Silverman and Linkster, 1986).

The Run measures the number of appearances of each of the nontrivial subsets of the nucleotides. Let $S_1, S_2, \dots, S_{14}$ be the nontrivial subsets of the set $\{A, C, G, T\}$. For each S_i construct a new sequence by replacing each base in S_i with 1 and replacing each base not in S_i with 0. Using this sequence, define r_{ij} to be the number of runs of i of length j , for $j=1, 2, 3, 4, 5$ and let r_{i6} be the number of runs of i of length greater than 5. The Run Measure will be the set of values $[r_{ij}]$.

The technology having the highest success rate to date for search by content is the use of a neural net. Uberbacher and Mural (1991), combine a set of sensor algorithms, all

in the Dicondon/Hexamer-n Measures, and a back propagation neural network to localize the coding regions. A 99 base sequence window of human DNA was tested. Signals were ignored. A 90% success rate was reported for exons of base length greater than 100. Base lengths less than 100 recorded a 47% success rate. False positives (incorrectly predicted exons), were noted at a rate of 8%.

A higher success rate of 99.4% was reported by (Farber, Lapedes, and Sirotkin, 1992). Again a back propagation neural net was used. The neural net was trained on the frequency of dicodons and combined with a suitable measure for comparing 2 probability distributions called the Kullback divergence measure. ORFs of 60 codons (180 bases) were used. Their accuracy was defined as the average of the exon and intron sensitivities (true positives or actual exons and introns discovered). Specificity (false positives/true negatives) was not addressed. Once again, the problem encountered was that of detecting exons with short base length.

2.2 Search by Signal

Algorithms exercising search by signal experience the problem of low specificity. False positives are abundant for the following reasons. The donor site, GT, that announces an intron, constitutes roughly 5% of the dinucleotides in human DNA. Of the DNA, only 1.5% are actual splicing donor sites. This allows the remaining 3.5% to be interpreted as false positives. The acceptor site, AG, that terminates an intron, constitutes roughly 7.5% of the dinucleotides in human DNA. Only 1% of the DNA is actual splicing acceptor sites. This allows for 6.5% to be interpreted as false positives. Despite the problem, some researchers using this technique have experienced moderate success.

A general method based on linear discriminant analysis used information on consensus sequence patterns around splice junctions (Nakata, Kanehisa, and DeLisi, 1985). Statistics were also gathered on base pairing and periodic appearance of specific bases. The method correctly identified 80% of the true splice junctions.

Lapedes, et al (1990) applied a neural net constructing training and prediction sets for acceptor and donor sites embedded in bases of length 11, 21, and 41. An accuracy of 91.2% was obtained for intron/exon splice junctions and 94.5% for exon/intron splice junctions. A high number of false positives were recognized. However, one of the seven false positives was later identified to actually be coding.

Brunak, Engelbrecht, and Knudsen (1991), using a neural net with window lengths of 11 to 401 bases, obtained 87.4% accuracy for intron/exon splice junctions and 94.1% accuracy for exon/intron splice junctions. Again, the problem of false positives was experienced. An average of $1\frac{1}{2}$ false donor sites per true donor site and 6 false acceptor sites per true acceptor site were reported. The number of false positives was reduced by a factor between 2 and 30 by incorporating a larger network trained to recognize coding regions (search by content) as well as recognize the signals (search by signal).

A scanning model for acceptor site recognition was simulated by the AMELIE system (Vignal, et al, 1996). Sequences were scanned for the existence (or non-existence) of the dinucleotides AG upstream (to the left of) the AG acceptor site. The absence of the AG pattern in certain stretches was found to be a general property of

introns. Other content related tests were conducted. Detection of 84% of true sites was achieved, while 3.5 false sites per 1000 bases was reported.

2.3 Integrated Systems

As with splice junction detection, gene structure prediction is improved when both the techniques of search by content and search by signal are combined. The two methods best known for integrating these search techniques are the rule-based approach and the connectionist AI (artificial intelligence) technique. The rule-based paradigm involves chained rules that are derived automatically rather than being derived by an expert. The advantage of the rule-based approach is that the logic involved in arriving at a conclusion is obvious, whereas, for a neural net, the logic is encoded in the network architecture and the network weights. This can sometimes make biological interpretation difficult.

One example of a rule-based system is GeneModeler (Fields and Soderlund, 1990). Figure 5 is a diagram illustrating the GeneModeler architecture. The function of each module is described below.

- ◆ The ORF module returns the beginning and ending coordinates of all ORFs and assigns frame labels to each ORF.
- ◆ The WINDOW module accepts as input a window width, a window overlap and a C+G frequency cut-off. It identifies potential coding regions by executing a C+G frequency analysis.

Figure 5. Architecture of GeneModeler.

- ◆ Matrix identifies 5' and 3' splice sites by accepting as input a consensus matrix and returning the coordinates of all sites that match the consensus matrix.
- ◆ The FREQ (frequency) module accepts as input a code, either AT or CG and determines its frequency.
- ◆ The DIFREQ (difrequency) module returns a ratio $\{AT\}/\{CG\}$, calculated as the number of occurrences of the dinucleotides AA, AT, TT or TA divided by the number of occurrences of the dinucleotides CC, CG, GG or GC.
- ◆ CODON accepts as input a table of codon asymmetry pairs. It returns an asymmetry score calculated as the number of pairs in which the specified asymmetry is satisfied minus the number for which it is not satisfied.

- ◆ TRANSLATE accepts as input a set of exon start and end coordinates. It returns a continuous amino acid translation spanning the specified exons producing a file of predicted amino acid sequences.

GeneModeler uses a log-likelihood method to score splice junctions and measures AT versus GC richness in and around ORFs to predict gene assemblies. A rule-based system designed from a known gene organization is then used to recognize when all of the component patterns have occurred. Ultimately, the user must determine whether some, all, or none of the predicted candidate genes is an actual gene. Although no actual accuracy figure was noted, it appears that exons were predicted within 83%.

GeneID (Guigo and Knudsen, 1992) was constructed similar to GeneModeler. The component algorithms recognize the major features of genes: initiation codons, stop codons, acceptor sites, and donor sites. Then the first, internal and terminal exons are predicted. The predicted exons are filtered by applying a set of rules. The filtered exons are then grouped into equivalence classes by means of constructing rooted trees to represent gene models. Predicted genes for a DNA sequence are obtained as linear arrangements of these exon classes. Finally, a function of the values attached to each of the component exon classes is assigned to the potential gene. This score is derived to establish a ranking. GeneID performed with 88% sensitivity in predicting a coding region to actually be coding and 80% specificity in predicting the coding region accurately.

GRAIL (Uberbacher and Mural, 1991) is an instance of a connectionist AI technique. It identifies exons by combining information from numerous content statistics

and then weighting these scores using a neural network. The seven sensor algorithms responsible for gathering the statistics are illustrated in figure 6 and then briefly explained below.

Figure 6. GRAIL architecture.

- ◆ BIAS is the frame bias matrix. The basis of this method is the nonrandom frequency with which each of the four bases occupies each of the three positions within codons. If a region codes for protein, then one reading frame should have a significantly better correlation to the bias matrix than the other two possible reading frames (by shifting right twice).
- ◆ PERIOD is an algorithm developed by (Fickett, 1982) which examines the 3-periodicity of each of the four bases and compares them to the periodic properties of coding DNA. It also compares the overall base composition of the test DNA with the known composition for coding and noncoding DNA.

- ◆ DINUC represents an algorithm which studies the dinucleotides. Dinucleotide occurrence is not random. For example, AA and TC are quite common while CG is rare. By examining the transitions of sequential dinucleotides, it is possible to view a DNA sequence as a dynamic function. These transitions appear to be different in coding and noncoding regions.
- ◆ CODING is a sensor module which evaluates 6-tuple word preferences. A “word” is a string of 6 nucleotides. Exons and introns have different distributions of word occurrences. Each word’s preference value is calculated as the logarithmic ratio of its normalized frequency of occurrence in coding vs. noncoding human DNA, and the sum of preference values in the window provides the coding indicator.
- ◆ IN-FRAME is similar to the previous sensor except that the observed 6-tuples in the test DNA are compared with the preference values of inframe 6-tuples compiled from coding DNA. The total preference is computed three times for the analysis window, once for each reading frame. The predicted reading frame is taken to be the one that provides the best 6 tuple in-frame coding vs. noncoding preference, and the sensor value corresponds to the total preference for this frame.
- ◆ COMMON studies word commonality. The overall frequency of occurrence or commonality of a given 6-tuple in bulk DNA is related to its context. For example, introns use extremely common words and exons relatively rare words.
- ◆ REPETITIVE compares the test sequence with 6-tuple statistics for several classes of repetitive DNAs in a manner similar to that used in the previous three algorithms, and the largest total preference in the window is used.

As can be seen, all seven of these sensor algorithms fall under of the Dicodon/Hexamer-n Measure that was described in section 2.1. GRAIL's output shows only the positions of candidate exons on a linear sequence and does not attempt to produce assembled genes.

GeneParser (Snyder and Stormo, 1993) produces a better ranking by recursively optimizing a solution based on the raw input scores and the current weight values. Weights are adjusted for errors. This process is repeated until the weights accomplish the mapping to the desired degree of accuracy. The architecture can be seen in figure 7 on the next page.

The input values to the nodes of the hidden layer are the differences between the correct and incorrect solutions for each statistical test (exons and introns) and the difference in the number of predicted exons and introns. The nodes in the hidden layer then calculate the sum of the score-difference inputs and the number-difference inputs times the biases. These sums are multiplied by weights to give the input to the output node. The output node calculates the logistic function on its input.

The five statistical functions are as follows.

- ◆ CODON uses a table of in-frame 6-tuples generated from the database of human genes used by (Hutchinson and Hayden, 1992). The unknown sequence is compared to this standard and an estimate is calculated.
- ◆ LOCAL compositional complexity is used to quantify the property that large amounts of repetitive DNA segments are found in noncoding regions, whereas, coding regions tend to be informationally rich. Eight nucleotides are used and averaged over the length of the unknown sequence.

Figure 7. The architecture of GeneParser.

- ◆ LENGTH distributions are characteristic in exons and introns. Frequency histograms of exons and introns were utilized by normalizing the values such that the most frequent sequence length of each type received a score of 1.0.
- ◆ K-TUPLE frequencies of 6-tuple words were compiled for both introns and exons. As in the codon usage statistic, the log-likelihood ratio for a given subsequence being either an exon or intron can be calculated from their respective 6-tuple frequency tables.
- ◆ SPLICE SIGNALS use weight matrices calculated for the donor and acceptor sequences from the collection of splice junction sequences of (Shapiro and

Senapathy, 1987). Constraints are enforced such that introns must be bounded at the 5' end by a donor site and at the 3' end by an acceptor site. Similarly, internal exons must be bounded by a 5' acceptor site and a 3' donor site.

A prediction score of 67% was achieved on exons with bases less than 60 in length. It identified 74% of the internal exons with a specificity of 62 %. However, the structure of only 17% test genes was exactly predicted. The problem exists in the fact that terminal exons dramatically decrease accurate prediction. Difficulty with long introns was also admitted yielding 38%.

Table 2 represents the results of a test of the performance of the gene prediction programs presented above (Solovyev, Salamov, and Lawrence, 1994). The performance testing method uses a correlation coefficient (Matthews, 1975 and Brunak, Engelbrecht, and Knudsen, 1991) shown as follows:

$$C(X) = \frac{(P_x N_x) - (N_x^f P_x^f)}{\sqrt{(N_x + N_x^f)(N_x + P_x^f)(P_x + N_x^f)(P_x + P_x^f)}}$$

where:

X = exon or intron class

P_x = correctly predicted positives

N_x = correctly predicted negatives

P_x^f = incorrectly predicted positives

N_x^f = incorrectly predicted negatives

C(X) = 1; perfect prediction

Table 2. Accuracy of correct exon prediction for integrated gene search systems.

INTEGRATED SEARCH SYSTEM	C(X)
GeneModeler	0.02
GeneID	0.33-0.42
GRAIL	0.31-0.52
GeneParser	0.47

2.4 Internal Exon Predictors

Accurate prediction of internal exons is very important, because a cloned genomic DNA fragment rarely contains the entire sequence of a gene. Three internal exon prediction programs not included above are worthy of mention.

SORFIND searches a sequence for splice ORFs (SORFs) and analyzes the region for codon usage (Hutchinson and Hayden, 1992). As with GRAIL, SORFIND reports only the position of suspected internal exons. An accuracy of 87% was reported.

Solovyev, Salamov, and Lawrence (1994) developed a new method, FGENTH, using a linear discriminant function that combines information about significant triplet frequencies of various functional parts of splice site regions and preferences of oligonucleotides (a chain of 2 to 10 nucleotides) in protein coding and intron regions. This technique of discriminant analysis was applied to relate a given sequence which has particular values of the p measures $x_1, \dots, x_p$ to one of two alternative classes; class 1 (sites or exons) or class 2 (pseudosites or pseudoexons) (Cinkosky, et al, 1991). The term

‘pseudo’ is applied to all GT and AG dinucleotides, which are not splice sites.

Analogously, all open reading frame fragments flanked by GT and AG dinucleotides which are not real exons are referred to as pseudoexons. The procedure of linear discriminant analysis is to find a linear combination of the measures (called the linear discriminant function or LDF), that provides maximum discrimination between real and pseudosites (or pseudoexons). One advantage of the approach is that the tables of triplets are recalculated to obtain increasingly reliable statistics as the size of the sequence data base increases. The eleven discriminant function equations can be studied in (Solovyev, Salamov, and Lawrence, 1994). The recognition quality computed at the level of individual nucleotides is 89% for exon sequences and 98% for intron sequences. The accuracy of precise internal exon recognition was 77% with a specificity of 79%. The precision of this approach was shown to be better than other methods to date and was tested on a larger data set.

A third method, MZEF (Zhang, 1997), uses an extension of the classical linear discriminate analysis (LDA) called quadratic discriminant analysis (QDA). QDA provides a more effective curved boundary between two populations that have different shapes and orientations than LDA, which provides only a straight-line boundary. A test set of 473 genes containing 1509 internal coding exons of known organizations was used to compare the results for the GRAIL integrated system and the FGENEH system (both previously discussed) with MZEF. Table 3, on the next page, presents the sensitivity measure (S_n) determined for the three.

Table 3. Accuracy of correct internal exon prediction for gene search systems.

INTERNAL EXON PREDICTORS	Sn
GRAIL2	0.53
FGENEH	0.73
MZEF	0.78

2.5 Summary of Current Research Problems

A summary of the problems found in current research dealing with exon/intron discrimination follows:

1. Exons with short bases are difficult to recognize.
2. Terminal exons dramatically decrease accurate prediction.
3. Accurate prediction of internal exons is important because DNA fragments rarely contain the entire sequence of a gene.
4. The abundance of false positives when searching by signal.

These problems are possibly brought about by the statistical and mathematical methods employed. Statistical methods might not deal effectively with the following issues:

1. Too much data might be discarded.
2. General patterns might be used rather than specific instances.
3. Noise promotes generalization and therefore, unspecificity.

Mathematical methods, too, might suffer from the following:

1. The method might possibly involve too much data that can hinder the analysis.
2. It might be difficult to distinguish between essential data and noise.

The method utilized in this research, Finite Induction (FI), is designed to counteract these shortcomings.

1. FI keeps enough data, but not all.
2. It discards some data, but none of potential importance.

CHAPTER 3

FINITE INDUCTION

3.0 Introduction

The Finite Induction (FI) process involves two major algorithms, FACTORING and FOLLOWING. Figure 8 below features the two algorithms and their relationship.

Figure 8. FI process representation.

Associated with these two algorithms are two data structures: RULING and RESIDUAL. The RULING results from the application of FACTORING to the input string representing the sequences of particular interest. The RESIDUAL results when the algorithm, FOLLOWING, is applied to the string representing the unknown sequence

using the RULING. The RULING can be computed and used, or the RULING can be stored on the disk and used by reference. These stored patterns can represent known or suspected sequences representing features of significance.

Figure 9 portrays only this second process of FOLLOWING creating a RESIDUAL from an unknown input sequence of symbols. It depicts the relationship between the unknown input string and the output or RESIDUAL. Here, the RULING is shown as a multilevel structure, and RESIDUAL elements come from the input sequence under some ruling guide. It will be clarified later in this chapter how this process operates. In a sense the RULING behaves as a filter applied to the input string, and the RESIDUAL is those symbols in the input sequence which are not captured by the RULING.

Figure 9. FOLLOWING as applied to input to produce a RESIDUAL using a RULING.

The following facts can be asserted about input strings and RESIDUALS:

- ◆ If the RESIDUAL and the input string are identical, then the RULING contains no rules of the sequence represented by the input string;
- ◆ If the RESIDUAL is empty, then the RULING contains an exact representation, or all the rules (superset) of the sequence represented by the input string;
- ◆ A non-empty RESIDUAL is always a subset of the input sequence.

A measure of closeness between the input string and the information contained in the RULING is given by the number (count) of symbols contained in the RESIDUAL. In another sense, the RULING acts as a filter, and the configuration of the filter can be specified to simply output those symbols that do not fit the known configuration represented by the rules contained in the RULING. The method of filtering, (i.e., the method of applying the RULING during FOLLOWING) can be specified so that input symbols are filtered in a very loose way or a very exact way or one of several ways in between. Further, the RULING can be used to predict, in the absence of input symbols, those symbols which would be likely to occur given that some input symbols have provided the beginning steps of FOLLOWING. Given a few of the first input symbols of an unknown string and the beginning steps of a FOLLOWING, an appropriate RULING can be selected from a collection of RULINGS.

These points are clarified by providing a simple sequence together with its RULING in a subsequent section. An illustration of how an unknown sequence can be processed is also demonstrated. However, first are presented a few definitions to formalize the FI environment (Case and Fisher, 1984).

3.1. FI Definition

The following definitions provide the formalization of the FI process.

Definition 1. Let A be a finite set of symbols called the alphabet, let S be a sequence (called a string) of these symbols which may be finite or infinite. The sequence S is Finitely Inductive (FI) if the choice of a letter at any particular position depends (immediately) only upon the choices of letters at the previous n positions. The greatest such n is called the inductive base of the sequence S .

Definition 2. Let S be an FI sequence, and let the pair (w,p) consist of a word w over the alphabet A and the letter p be in the alphabet such that:

- ◆ w occurs at least once as a substring (subsequence of contiguous entries) of the sequence; and
 - ◆ wherever w occurs as a substring there is a succeeding entry and it is p .
- The pair (w,p) is normally written $w \rightarrow p$ and is called an implicant; w is called the antecedent, and p is called the consequent.

Definition 3. An implicant is in reduced form if its antecedent contains no proper terminal segment that is an antecedent of another implicant of the sequence.

These definitions are immediately generalizable to families of sequences. The pair (w,p) is an implicant of the family if its antecedent occurs at least once as a substring of one of the sequences and whenever w occurs as a substring of one of the sequences there is a succeeding entry that is p .

The following observations can be easily noted:

- ◆ Any finite sequence is FI;
- ◆ For any finitely inductive sequence, the inductive base is the maximal length of the antecedents in its reduced form implicants;
- ◆ Any periodic FI sequence has a period less than or equal to k^n , where k is the number of letters in the alphabet and n is the inductive base;

- ◆ If an FI sequence has an inductive base n and an alphabet of k letters, then k^n is an upper bound for the number of its reduced form implicants.

Definition 4. A table of implicants is a finite table of pairs of the form $w \rightarrow p$ where w is a word over a given alphabet and p is a letter in the same alphabet.

A table of implicants satisfies the following conditions: (i) no antecedent of one entry in the table is a terminal segment of the antecedent of any other entry in the table, and (ii) without losing generality we can suppose that every sequence has a special symbol called the start symbol, then the only way the start symbol may occur in the table is as the first symbol in an antecedent. The inductive base for such a table is the maximal length of the antecedents.

Definition 5. Given an FI sequence S , there is a finite sequence of tables of implicants T_i called a RULING such that for each i from 1 to L , T_i is called the table of implicants for level i .

To aid in understanding the above formal definitions, an example of an FI sequence and the associated RULING and construction of the sequence of tables will be demonstrated.

3.1.1 Example for FI: FACTORING

Consider the following assumed partial intron sequence formulated from the four symbols, A, C, G, T:

G T G A G T G G T C T T A G

The first step is to determine the implicants or rules as given in Definition 2. The initial set of implicants is called the prime implicants, and is determined by starting with the first symbol G. Assuming the sequence to have a start symbol, S, begin with this start symbol to determine the symbol or group of symbols which uniquely identifies the first

symbol in the string, G. Thus it can be written $S \rightarrow G$. This means that every time an S is found there will be a G following.

Rules can be generalized to make them non-deterministic or probabilistic simply by changing the formulation of the implicant to allow multiple consequents (with probability). For example, $GT \rightarrow G|C$, would be interpreted that when the group GT is found, a G or a C will follow, and the likelihood of it being either the G or the C is unknown.

The prime implicants for the above string are:

$S \rightarrow G$	$SGTG \rightarrow A$	$AGT \rightarrow G$	$GGT \rightarrow C$	$TT \rightarrow A$
$SG \rightarrow T$	$A \rightarrow G$	$AGTG \rightarrow G$	$C \rightarrow T$	$A \rightarrow G$
$SGT \rightarrow G$	$AG \rightarrow T$	$GG \rightarrow T$	$CT \rightarrow T$	

For this set of implicants, the inductive base for the string is four; that is, the longest antecedent for the collection of implicants is four.

There is a FI theorem which states:

If the inductive base, n , of a set of prime implicants is larger than some implicant in the prime set, then the inductive base can be changed to any value less than n .

This theorem implies that the inductive base of any sequence can be changed to any value less than the size of the original inductive base size; and this value is the determining number of precursor elements that are needed to determine the next element in the sequence.

Change the inductive base from four to two, for example. If the above implicants are considered, then nine of the implicants are already appropriate - that is, their antecedents are already less than or equal to two. For the remainder, if the consequent

symbol from the sequence can be deleted, then that implicant would disappear. This is accomplished as follows:

				G	A			G	G		C					
S	G	T	G	A	G	T	G	G	T	C	T	T	A	G		

The upper sequence represents the symbols that if deleted, leave only the implicants suitable for an inductive base of two. The upper level (RESIDUAL) can now be treated as a new sequence and the FACTORING process can be applied to determine the new implicants. These would be, again starting with the first symbol:

S --> G	A --> G	GG --> C
SG --> A	AG --> G	

It can be seen from this set of implicants, that all of the implicants are satisfactory.

Therefore, there is no residual and the process terminates with this level.

A RULING is now formed whose inductive base is two, and the number of levels is two. Duplicate implicants are deleted at each level. The RULING is:

<u>LEVEL 1</u>		<u>LEVEL 2</u>
S --> G	CT -> T	S --> G
SG --> T	TT -> A	SG --> A
A --> G		A --> G
AG -> T		AG --> G
GG -> T		GG --> C
C -> T		

The inductive base for this RULING would be formally stated as (2,2), that is the inductive base for each level is two, but since two is the largest value, the inductive base can just be stated as two.

The RULING above represents a collection of deterministic RULES obtained from the input string. Please note also:

- ◆ The RULING can exactly regenerate the original sequence. The technique used is not immediate, and it is not suggested here, as regeneration is not important.
- ◆ The RULING that is derived is a filter, and can be used to filter unknown strings. This is called FOLLOWING.
- ◆ Only the amount of data required is kept in the RULING. Notice that the two level RULING needed no more rules than the number of rules in the prime implicant set. Oftentimes fewer rules are needed to represent the sequence.
- ◆ Consider the number of rules possible for an inductive base four system as compared to an inductive base two system. In the first, there are a possible 4^4 number of rules, and in the second, there are a possible 4^2 number of rules times the number of levels required to express the sequence. Generally, the number of levels is around ten, and the inductive base is between three and six for complex strings.
- ◆ Only a single string has been treated for input, but from the definitions, it is possible to treat many sequences simultaneously. The possible use of this would be to treat the known variety of occurrences for a particular substring as simultaneous strings to build a RULING. The number of such simultaneous strings can be arbitrarily large.
- ◆ The last point, the inductive base of real-life sequences such as speech, music, and other phenomena have inductive bases perhaps into the hundreds of symbols from a very rich alphabet. The number of symbols and the inductive base size are simply irrelevant. Each sequence that matches the criteria in the theorem is factorable as shown.

3.1.2 Example for FI: FOLLOWING

Using the RULING derived in the previous section, it is now possible to process any string to determine how that string matches the sequence encapsulated within the RULING. Applying the new sequence string:

G T G A G T C A G T G C A G

and the two levels of rules determined from the first string given above, three different methods of FOLLOWING will be demonstrated.

3.1.2.1 EXACT FOLLOWING

EXACT FOLLOWING deletes consequents if the antecedent and the consequent match, applying the rules in each level sequentially. The process is as follows:

2:		C A	G C A	
1: S	G A	C A	G C A	
0: S	G T	G A	G T	C A G T G C A G

Each level in the RULING is applied to produce the corresponding levels 1 and 2. The proper symbols are deleted, and the next level treats the RESIDUAL.

3.1.2.2 BLIND FOLLOWING

In this situation, the consequent is deleted if and when the antecedent matches, irrespective of the consequent match. The justification for this approach is that the consequent may be in error due to noise or experimental error. The results are:

2:			G C	
1: S	G A	C	G C	
0: S	G T	G A	G T	C A G T G C A G

Note that the RESIDUAL in the BLIND FOLLOWING is less than the RESIDUAL in the EXACT FOLLOWING. In fact, the RESIDUAL of BLIND FOLLOWING is always a subset of EXACT FOLLOWING.

3.1.2.3 REPLACEMENT FOLLOWING

In this approach, instead of deleting the consequent as was done in BLIND FOLLOWING, the symbol is actually changed, (noted by lower case below) and the string is processed with the change. The justification for this approach is not only the possible error, but also the idea that the error should not impact the neighboring symbols. The number of such changes are noted and supplied with the RESIDUAL. The results are:

2:		g		g	t		G	C	t
1:	S		G	A		C	t		G
0:	S	G	T	G	A	G	T	C	A

The number of transforms must be taken into consideration. There are six transforms in this example. The number of transforms can also provide a measure of closeness to the string encapsulated within the RULING.

3.2 Additional FI Considerations

Character sequence strings representing music, speech, DNA, etc., have different characteristics. A FOLLOWING that works well for one sequence might not work well for another.

There is another consideration. If the RULING is formed from several strings factored simultaneously, or from several classes where each class is represented by

several instances, then one can apply FOLLOWING in whatever manner is desired, and the identification can be obtained as shown in Figure 10.

Figure 10. RULING containing implicants from three objects.

Within the RULING, the implicants are annotated to include the sequence or class identifier. When an annotated rule is used, it has a counter that is incremented by one. Common areas shown in Figure 5 might indicate some manner of non-deterministic result. This figure represents three classes of sequences, and the rules found in the RULING are grouped by classes, but the classes are not distinct -- thus the overlapping groups. It is clear that a rule 'firing', or match, would increment only one counter, and when the string is processed, the counter totals would represent the characterization of the unknown string. Any of the FOLLOWING techniques described above could be used in conjunction with this organization.

It has not been explained how probabilistic or non-deterministic FACTORING and FOLLOWING work. However, it should be clear how the extension to this domain would occur. Any rule in a RULING can have multiple consequences, and each of the possible consequences can have a probability associated with it according to the frequency of occurrence found in the strings which build the RULING. This multiple consequence idea does not allow regeneration of the input strings from the RULING, but it does compress the RULING as contradictions are not pushed out, but are simply included as another consequence in the particular implicant.

Matching strings can be a difficult process, but the idea of a RESIDUAL is simply the amount of difference between the RULING and the unknown string. The length, the transform count, and the content of the RESIDUAL are all indicators of match. For example, if two dissimilar strings are processed by a RULING and the RESIDUALS after FOLLOWING turn out to be similar or identical, then the two strings can be said to differ from the known string in the RULING by the same amount. This does not imply that the strings are identical, but it does suggest that they are different from the known string in ways that are similar. This is a very important concept. The idea is: strings can be compared by the amount of difference to a common string that is the string or strings encapsulated within the RULING. Thus, the comparison is by difference, not by similarity.

A second point to make suggests that the RULING actually is a filter as already suggested, and then, the RESIDUAL is the amount of surprise the unknown string brings to the filter. Strings with lots of surprise have long RESIDUALS and are thus not similar

to the encapsulated string, but they may be similar to already established collections of RESIDUALS and therefore similar to known sequences. This similarity can be measured by forming RULINGS from the RESIDUALS and applying the FOLLOWING algorithm again. Thus, two stages of RULINGS may be used to find matches.

The rules in RULINGS are likewise important. For example, one can count the number of times an identical rule is found, and then FACTOR two unknown strings into two separate RULINGS. The RULINGS can then be compared level by level and a measure or figure of similarity can be stated. This would provide a building or structure map for sequence identification.

It is also interesting to note that the symbols associated together at levels other than the lowest are not adjacent to one another. However, this more distant association may likewise have some consistent behavior in particular sequences when noise or other symbols are present.

As indicated in chapter 2, the two approaches used for sequence identification focus on search by content or search by signal. FI can do both simultaneously. That is, the RULING pattern can include sequences that represent instances from both types of structure and then using the RULING, the search is not focused on either technique.

This is an approach whose complexity is not impacted by the size of the RULING, only by the length of the unknown string to be processed. This is quite different from the linear programming, neural network, or other similar approaches.

CHAPTER 4

MATERIALS AND METHODS

4.0 The Data Sets

The Non-Redundant Functionally Equivalent Sequences (NRFES) database, compiled and created by A.K. Konopka, was the source for all data sets used for this project (Konopka, 1993). One data set of individual primate exons and one of introns was extracted. The two data sets were scanned and sequences containing non-ACTG base entries and/or containing nucleotide base lengths in excess of 3500 were excluded. The number of sequences contained in the exon data set numbered four greater than that in the intron set. Sequence lengths ranged from 101 to 3339 bases. Less than 10% of the sequences consisted of base lengths greater than 1000. Figure 11 compares half (for legibility) of the exon file with half of the intron file as to sequence lengths.

Figure 11. Data set sequence sizes.

4.1 RULING Construction and Behavior

The RULINGS were produced, during the FACTORING phase, using subsets of the exon and intron data sets. One exon ruling and one intron ruling were created using approximately 100 sequences evenly distributed throughout the original files. The first two small data sets will be referred to as EXON (S) and INTRON (S), respectively.

Additional exon/intron rulings were created using data sets containing over four times the number of sequences than the first set. The second two large data sets will be referred to as EXON (L) and INTRON (L), respectively. Sequences not participating in the data sets used for RULINGS were engaged as unknown sequences to be filtered during the FOLLOWING phase. The unknown sequence data sets will be described in section 4.3.

RULINGS were constructed for inductive bases (IBs) 2, 3, 4, 5, 6, 7, 8, 11, 15, and 20. In addition, an inductive base referred to as “M” was realized by searching the set of implicants for the greatest number of rules having antecedents of a particular length. That “maximum” length was selected to become the IB for the next level. Observing that lengths just beyond the maximum length regularly demonstrated a large number of rules, yet another base was computed by searching beyond the maximum until the number of rules became less than one-half the number of rules for the previous antecedent length. This IB is referred to as “N.”

As the definitions for Finite Induction (FI) indicate, the smaller the inductive base, the fewer the number of rules produced and the greater the number of levels. Also evidenced for sequence data is: the greater the number of sequences involved, the greater the number of rules produced. It is also observed that, generally, the longer the sequence,

the greater the number of levels. These statistics are demonstrated for the four data sets used in the FACTORING phase in the figures below.

Figure 12. Levels generated by test sets for all IB.

In figure 12, note the dramatic decline in the number of levels in the short span of three inductive bases, IB 2 to IB 4. Figure 13, on the next page, reveals what is otherwise hidden in Figure 12. The maximum number of levels taper off from 10 (IB 5) to 1 (IB 20). IB M and IB N maximum levels fall within the range of those for IB 5 and IB 6

Figure 14 and figure 15, also on the next page, are more realistic in that they present the average number of levels together with the minimum and maximum for each IB of EXON (S). The high number for the maximum level count is produced by the long sequences, 1000 – 3000, that rarely occur. If the factoring phase were limited to sequences of length less than 1000, the EXON (S) IB maximum levels would decrease according to table 4, page 47.

Figure 13. Levels generated by test sets for IB 5 – IB N.

Figure 14. Level comparison for EXON (S), IB 2 – IB 4.

Figure 15. Level comparison for EXON (S), IB 5 – IB N.

Table 4. Effect on level count by limiting sequence size.

Inductive Base	Maximum Level for Sequence Size ≥ 1000	Maximum Level for Size < 1000	Percentage Decrease in Number of Levels
2	1234	417	66%
3	525	101	81%
4	55	11	80%
5	8	5	38%
6	4	3	25%
7	3	3	0%
8	2	2	0%
11	2	2	0%
15	1	1	0%
20	1	1	0%
M	6	6	0%
N	5	3	40%

The INTRON (S) data set produced similar results to that of EXON (S) shown in figures 14 and 15 and can be viewed in appendix A, page 87, figures 16 and 17. It follows that EXON (L) and INTRON (L) would present similar charts to that of the small test sets.

Figure 18, on the next page, displays the increase in the average number of Level 1 rules as the IB increases for each of the four test sets. Beginning at IB 8 through IB 20, the number of Level 1 rules generally equals the number of bases.

As the sequence length increases, so does the number of levels for most all IBs. This is demonstrated in figure 19 using the EXON (S) data file. The introns behave similarly and can be viewed in appendix A, page 88, figure 20. Figure 21, on the same

page in the appendix, demonstrates the level activity for IB 5 – IB N which is not visible in figure 17. Level increases are less extreme for IBs 5 and above.

Figure 18. Level 1 average implicants for the four data sets.

Figure 19. EXON (S) sequence size versus levels.

Twenty-two exon and twenty-two intron sequences possessing the same base lengths were extracted from the data for comparison purposes. Figure 22, below,

indicates that for sequence sizes less than 215, the only activity noticeable is for IB 2.

Note, once again, the introns behave in accord with the exons.

Figure 22. Level comparison of exon/intron sequences.

4.2 Filter Characteristics

Filters were constructed by collapsing the collection of RULINGS in each of the four data sets. Each rule in each level was passed through the aggregation of all rules for all sequences of that level searching for duplicates and deleting them. Figure 23 on the next page, illustrates for each IB of EXON (S), the total number of rules for Level 1 (TOTAL), the number of rules remaining after duplicates were removed (FILTER), and the number of rules in the filter that were seen just once in the collection of RULINGS (UNIQUE). Note that IB M and IB N behave in a similar manner to IB 5 and IB 6, respectively, as witnessed previously.

Figure 23. Comparison of total, filter, and unique rules for small exon data set.

It is reasonable to assume that the fewer number of unique rules in a filter, the better the filter represents its collection of sequences. The percentage of the rules in each filter that were unique was calculated. These percentages are graphed in Figure 24. IB 4 and IB 5 demonstrate the lowest percentage in the majority of cases.

Figure 24. Unique rule composition of filter for all four data sets.

The number of sequences utilized to produce a filter fluctuates between being a negative influence or a positive influence. EXON (S) appears with fewer unique rules than does EXON (L) by 8% for IB 2, while INTRON (L) betters INTRON (S) by 25% for IB 6.

4.3 The FOLLOWINGs

Two data sets were constructed containing sequences not used during the FACTORING phase that created the filters. One set of 95 exons, evenly distributed as possible, was extracted from the original exon file. Likewise, 95 introns, having sequence lengths identical to or within close proximity to the sequence lengths of the exons, were collected into a data set of “unknowns.” Figure 25 depicts these two data sets.

Figure 25. Data sets for unknown sequences.

The two data sets of unknowns were passed through the filters previously described using an array of methodologies. First, the results for the EXACT

FOLLOWING, BLIND FOLLOWING, and REPLACEMENT FOLLOWING will be discussed. Variations of these procedures were then implemented and executed. These procedures are referred to as By 2 Non-Overlapping, By 3 Non-Overlapping, By 3 Overlapping, and Codon. The findings of these implementations will be reported.

4.3.1 Findings for EXACT FOLLOWING

As described previously in chapter 3, during the process of EXACT FOLLOWING, each unknown sequence is factored to determine its set of prime implicants. The Level 1 ruling for the unknown is then matched against the Level 1 ruling of some filter. If a rule in the unknown ruling exactly matches a rule in the filter, then that nucleotide base is excused. If no rule exists in the filter, then the base is placed in the RESIDUAL. Once the RESIDUAL is completed, it is then factored, generating a Level 2 ruling which is compared to the Level 2 rules of the filter producing yet another residual string. This process is repeated until there are no more levels provided by the filter or the residual string is empty.

Each of the two unknown files was passed through each of the four filters, EXON (S), EXON (L), INTRON (S), and INTRON (L), for each of the twelve IBs producing a total of 96 runs. Statistics gathered for each level during a particular run included information concerning the RESIDUAL length, the RESIDUAL content, and the distance between nucleotides whose rules found no match.

For purposes of increased graph legibility, the following approach was necessary. Due to the large number of levels assembled by the smaller IBs, only those levels

producing meaningful results were plotted. In the majority of instances, three levels were sufficient to portray the behavior of the various IBs.

Figure 26 illustrates the barely distinguishable difference using the small exon filter versus using the larger exon filter. EXON (L) does indicate a slightly more pronounced improvement, as the sequence length breaks past the 1000 mark. Also, as observed previously in figures 12 and 13, the larger filters provide a higher plateau of levels within which the unknown sequences may work.

Figure 26. Comparison of exon filters.

The graphs presented in the body of the paper for EXACT FOLLOWING will represent EXON (L) IB 5 and INTRON (L) IB 5, which demonstrated the most promise as a filter based on the least number of unique rules. The other IBs may be viewed in appendix A when noted.

4.3.1.1 RESIDUAL Length

Figure 27 illustrates the results using EXON (L) at IB 5 to filter both the exon unknowns and the intron unknowns. Observe that the sequence length plays a role in determining the residual length. Residual length generally increases as the sequence length increases.

All IBs reacted in like manner. The smaller IBs, IB 2 and IB 3, produced residuals that reached up to 99.6% of the original string at Level 1. The few rules that did match included those having the start symbol as a component.

Figure 27. Residual results for inductive base 5 using the large exon filter.

Due to the large number of levels for IB 2 (over 1000), only Levels 1, 55, and 110 were plotted, as seen in appendix A, figure 28, to demonstrate the similarity for both exons and introns. Likewise, Levels 1, 7, and 13 were extracted for IB 3. Appendix A, pages 89 - 92, figures 28 through 38 contain graphs for all IBs other than IB 5 shown

above using the large exon filter. Figures 39 through 50, pages 93-96, appendix A, depict the same results for all IBs using the large intron filter.

Note that for IBs 11, 15, and 20, the residuals remain fairly consistent at all levels causing a melding of the levels. Not only are the residuals conforming between levels, but they are also congruous among the 3 IBs. This can be explained by the fact that from IB 11 through IB 20, the number of rules forbidden entry into Level 1 due to their antecedent lengths being greater than the IB is less than 0% (.0007%). Figure 51 demonstrates this characteristic for all IBs. Once again IB M and IB N behave in a similar manner to IB 5 and IB 6 respectively.

Figure 51. Percentage of rules too large to be included in Level 1 for all IBs.

Figure 52 below provides further evidence of the similarity of IBs 7 through 20. The residuals for each sequence in the exon file being filtered are virtually the same at Level 1. The residual produced at levels above that of Level 1 are brought about by sequences exhibiting lengths in excess of 500 nucleotides.

Figure 52. Residual comparison for inductive bases 7 through 20 at Level 1 demonstrates the similarity in the results produced by these filters.

Another interesting observation is the number of unknown sequences requiring more levels than provided by the filter for each inductive base (figure 53). Inductive bases 8 through 20 produced filters with the majority of rules being represented in Level 1. Subsequent levels, if they exist, have very few rules. Therefore, even though these inductive bases show a high percentage of matches, any residual that results produces levels that do not exist in the filter.

4.3.1.2 RESIDUAL Content

FI has the capability of discovering differences between sequences based on the content of the residual produced. The residual content for exon and intron sequences is concerned with the amount of A, C, G, and T nucleotides of which the residual is composed. Figure 54 on the next page demonstrates there is no significant difference in the content of the residual produced by the exon versus the intron unknown sequence files using the large exon filter.

Figure 53. Filters for inductive bases 8 through 20 are restrictive as to the number of levels provided the unknown sequences.

Figure 54. Comparing the content of the residual for IB 5 using the large exon filter provides no discrimination between exons and introns.

To clarify these results, the As, Cs, Gs, and Ts shown in figure 54 above are graphed separately in figure 55 on the next page. The graphs for all other inductive bases for the exon filters as well as the intron filters are not presented as the results were exactly the same. No distinction was found.

Figure 55. Residual content A, C, G, or T demonstrates little difference between exon and intron unknowns.

4.3.1.3 Distance Between Residual Components

The distance between residual components can also be a determinant in sequence discrimination using finite induction. The distance for the exon/intron sequence processing was computed as follows. Upon discovering a match for a residual component's rule, a counter was incremented by one. When a nucleotide's rule did not find a match, the contents of the counter was first displayed, then cleared. For graphing purposes, an average was determined when the end of the sequence was reached. The averages among the unknown sequences fluctuated widely as seen below in figure 56. However, the distance averages remained fairly consistent among exon and intron sequences of a given size. The results were the same for all inductive bases for both the exon and the intron filters.

Figure 56. The distance between residual components did not exhibit a difference between exon and intron unknown sequences.

4.3.2 Findings for BLIND FOLLOWING

The processing of sequences using BLIND FOLLOWING is identical to that of EXACT FOLLOWING with the following exception. During the rule-matching phase, only the antecedent is required to match. If the antecedents match, but there is no matching consequent, the consequent of the rule for the unknown sequence is excused from entering the RESIDUAL. In EXACT FOLLOWING, it would have been required to become a part of the RESIDUAL.

Even though BLIND FOLLOWING is a more forgiving approach, it produced very little difference in results from those of the EXACT FOLLOWING. Rather than exhibiting the graphs for all IBs and all filters to demonstrate this truth, the following three graphs compare the results of executing the BLIND FOLLOWING versus that of the EXACT FOLLOWING. For consistency, the unknown exon file was run using the large exon filter and IB 5. Figure 57 is a comparison between EXACT and BLIND of the RESIDUAL, the RESIDUAL content, and the distance. Only the first 24 sequences of Level 1 are displayed for visual clarity. Note that the RESIDUAL of BLIND FOLLOWING is always a subset of EXACT FOLLOWING as discussed in chapter 3.

The RESIDUAL content graph demonstrates that there is even less variation in the number of As, Cs, Gs, and Ts produced by BLIND FOLLOWING than by EXACT FOLLOWING. The assumption that BLIND FOLLOWING's pardoning nature should promote greater distances between RESIDUAL components whose rules have no match is borne out in the distance comparison of figure 57.

Figure 57. RESIDUAL, RESIDUAL content, and distance comparison for EXACT FOLLOWING versus BLIND FOLLOWING.

4.3.3 Findings for REPLACEMENT FOLLOWING

REPLACEMENT FOLLOWING proceeds the same as EXACT and BLIND FOLLOWING with one exception. During the second pass of the rule matching phase, if the antecedents match but the consequents do not match, then the consequent of the rule in the filter is placed in the RESIDUAL (unlike BLIND FOLLOWING) rather than the consequent of the unknown sequence's rule (as done for EXACT FOLLOWING). Not only is there forgiveness, but an attempt to correct the fault is involved. To illustrate the similarity between the results for the three methods, three graphs will be offered as in the previous section. Figure 58 compares the RESIDUAL, the RESIDUAL content, and the distance for the three FOLLOWINGS. Note that the distances are identical for BLIND FOLLOWING and REPLACEMENT FOLLOWING. This is always the case at Level 1 for these two procedures.

Another key measure provided by REPLACEMENT FOLLOWING is that of the transform count. Rather than deleting the consequent as was done in BLIND FOLLOWING, the symbol is actually changed. The string is then processed with that change. This allows for possible error and does not impact the neighboring symbols. The number of times a filter's consequent is placed in the RESIDUAL may be an indicator of difference between diverse sequences. Figure 59, page 64, illustrates that there is erratic difference between exons and introns as to the transform count. The majority of the introns exhibit a higher transform count. However, this is not consistent, as the exons for six of the sequences had higher transform counts.

Figure 58. RESIDUAL, RESIDUAL content, and distance comparison for EXACT FOLLOWING versus BLIND FOLLOWING versus REPLACEMENT FOLLOWING.

Figure 59. Transform count comparison for exons versus introns.

4.3.4 Additional Techniques

Variations of the EXACT, BLIND, and REPLACEMENT FOLLOWINGS were enacted to seek more positive results. In that the DNA alphabet is exceptionally small, different techniques were implemented whereby the alphabet was increased in size. These procedures described below are the By 2 Overlapping, By 3 Overlapping, and By 3 Non-Overlapping. Observing that the sequences representing the exons did not always begin on codon boundaries, a special data set was created which insured that the included exon sequences were aligned on and contained codons. This is described below as the Codon Alignment procedure.

4.3.4.1 By 2 Overlapping Procedure

Four nucleotides taken two at a time present sixteen combinations to which unique characters, A through P, of the alphabet may be assigned. This increases the alphabet fourfold. During both the FACTORING and FOLLOWING phases, the sequences were first converted to their new alphabets by looking at two nucleotides,

assigning them a unique character, then shifting right by one and repeating the process. The new sequence was then factored in the normal fashion. Figure 60 simplifies the results by representing the total number of matching rules (MATCH), total number of rules that could find no match (NOMATCH), and total number of transforms (TRANSFORMS).

Figure 60. Comparison of total results for exons and introns through alphabet extension taking nucleotides 2 at a time.

Figure 60 indicates that extending the alphabet by four reveals little divergence between exons and introns. Would further extension make a difference? Once again the alphabet size was increased.

4.3.4.2 By 3 Overlapping Procedure

The alphabet size was increased by sixteen (4^3) by taking the nucleotides 3 at a time and assigning a unique character. The identical procedure was followed as

discussed in the section 4.3.4.1 with the exception of working with three bases rather than two. The composite results are demonstrated in Figure 61.

Figure 61. Comparison of total results for exons and introns through alphabet extension taking nucleotides 3 at a time.

Extending the alphabet sixteen times had no affect on discriminating between the exons and introns. Would there be an advantage to not shifting one base at a time, but rather shifting three?

4.3.4.3 By 3 Non-overlapping Procedure

The same steps were followed as those presented in section 4.3.4.2 with the exception that the shift to the right was made three symbols at a time. This essentially captures possible codons assigning each a unique character. The exon sequences are composed of these codons. But close observation of the exon sequences reveals that their lengths are not always evenly divisible by three. This indicates that the exon might or might not begin on a codon boundary. This phenomenon is due to the fact that introns

might be positioned within codons as well as between them (Long, deSouza and Gilbert, 1995 and Tomita, Shimizu, and Brutlag, 1996). Therefore, three separate runs were conducted. The first run began at the starting position 1 (P1) of the sequence, the second began at position 2 (P2), and the third at position 3 (P3). Figure 62 compares the exon and intron results for the three executions.

Figure 62. Exon – intron results of taking bases three at a time, non-overlapping, beginning at positions 1, 2, and 3

As expected, the By 3 Non-Overlapping cut the total number of rules by two-thirds. The surprising phenomenon was that the number of matches was dramatically reduced from 74% to 15%. The transforms reacted in a reverse manner. They increased from 13% to 66%. The rules not finding a match (other than those transformed) remained fairly stable increasing only 6%. (See figure 63.) The significance of this occurrence is that the alphabet and rules representing the possible codons introduces constraints not evidenced in the previous runs.

Figure 63. A demonstration of the dramatic difference in overlapping and not overlapping nucleotide bases three at a time.

4.3.4.4 Codon Procedure

As mentioned above, the exon sequences in the database do not necessarily begin on codon boundaries, nor do they necessarily contain complete codons. Thus the amino acid chain might be incomplete. A program was implemented to search each exon for codon completeness. The procedure is similar to that used to discover ORFs. Three passes were conducted through each sequence. The first pass began with the first nucleotide base, the second began with the second base, and the third began with the third base. Each pass had the objective of identifying three consecutive bases representing one of the stop codons. If a stop codon was discovered at any position other than the end of the exon sequence, the sequence beginning at that particular positional base was deemed unacceptable. Normally, there was only one acceptable pass. If there were two, then the sequence was excluded. The length of the sequence representing the acceptable pass was

divided by three to test for a complete group of codons. Sequences unevenly divisible by three were excluded. Several of the sequences were compared with their amino acid counterparts in GenBank via the internet. This was an extremely tedious process and required thorough checking (Korning, et al, 1995), but served as verification for the procedure. Figure 64 compares the results with those of By 3 Overlapping and By 3 Non-Overlapping.

Figure 64. Comparing the By 3 Overlapping results with those of the By 3 Non-Overlapping and Codon procedures.

Despite the dramatic effect observed when assigning symbols to groups of three and not overlapping the bases, there was no impact on the way the exons were interpreted versus the introns. Figure 65 compares the ruling results in percentages because the intron file was larger than the specially constructed exon file.

Figure 65. Rule comparison for the special codon file of exons.

4.3.4.5 Passive FOLLOWING

The previous FOLLOWING algorithms are what might be referred to as “active” FOLLOWINGS in the sense that each unknown sequence played an important role in determining its matching capacity. The number of levels and rules for the inductive base of interest were first computed for an unknown sequence before that sequence was passed through the filter. Therefore the attributes of the unknown can be considered as constraints placed on the filtering process.

It was decided to remove these constraints and construct a Passive FOLLOWING. Rather than searching the filter for a rule that matched each rule constructed for the unknown sequence, each rule of the filter was passed across upon the unknown sequence in an attempt to find a match. With this procedure, the unknown sequence became an inactive player. This idea of the unknown sequence being a passive participant was incorporated into the algorithm for the EXACT FOLLOWING. The results are now best

expressed in terms of residual as there are no longer rules representing the unknown to be matched with rules in the filter. The residual will be given as a percentage of the total sequence length. Various data and procedures were executed to determine the ability of the new method for discriminating between exons and introns. The first process looks at a select set of individual exons and introns and is described in section 4.3.4.5.1. The second method makes another attempt at expanding the alphabet and is illustrated in section 4.3.4.5.2. Section 4.3.4.5.3 reports on the use of a filter composed of rules created using the acceptor and donor sites of the intron sequences.

4.3.4.5.1 Human, Homo Sapiens, and Gorilla Data Runs

Unique data sets were selected for testing this new algorithm. The majority of the primate sequences are prefixed “HUM” (human) in their identifying headings. The next larger group is prefixed “HS” (Homo sapiens). A very small group includes “GOR” (gorilla) plus other primates. One exon from each of these groups was chosen and a filter for each was constructed. Then one exon (different from the above) and one intron was also selected from each group (6 total) to act as the unknown sequences. Figure 66 gives some interesting results using the three filters.

In that the filters were constructed using exons, the logical expectation would be that the respective exon unknowns would exhibit less residual (more matches) than their intron counterparts. Only in the instance of the gorilla filter was this expectation realized and then only by less than 1%.

Figure 66. Residual comparison using three unique filters built from single exon sequences and used to filter related exon and intron sequences.

4.3.4.5.2 Codon Triplet Run

The new Passive FOLLOWING algorithm was modified to handle three nucleotides at a time, non-overlapping. As before, this provides a symbol for each triplet and expands the alphabet to sixty-four symbols. The filter and unknown sequence data sets used were the same as those especially created for the codon procedure of section 4.3.4.4. The residual for the closely matched (in length) exon and intron unknown sequences is compared in figure 67.

Figure 67. A comparison of the residual produced by both exon unknown sequences and intron unknown sequences as filtered by the Passive FOLLOWING as defined by codon triplets.

4.3.4.5.3 Splice Site Runs

As shown in chapter 1, section 1.1.3, introns possess the conserved sequences “GT” and “AG” at the beginning and ending, respectively, of their DNA stretch of nucleotides. Rules were constructed at each splice site that involved the donor site (GT) plus three additional nucleotides to its immediate right and the acceptor site (AG) and the

three nucleotides to its immediate left. The intron file in its entirety was involved in gathering the statistics presented in table 5.

Table 5. The percentage of sequences in which each rule is seen.

NUMBER OF RULES	PERCENTAGE OF SEQUENCES IN WHICH RULE IS SEEN
1	35
1	23
1	14
2	10
2	9
2	6
1	5
3	4
4	3
11	2
21	1
51	Below 1%

As can be seen, there is no set of rules which predominates. Only three rules are seen in over 10% of the intron sequences. Six groups of rules were assembled starting with those rules found in the most sequences, namely the three rules found in over 10% of the sequences. The next level of rules seen less often (5, 6, 9, and 10%) was added to the previous set for a second filter. This grouping continued until all rules seen at least once were included for a sixth set of 100 rules. The same procedure was repeated for nucleotides of length two (GTXX, XXAG) and length four (GTXXXX, XXXXAG) to the right and left of the donor and acceptor sites, respectively.

Several runs were conducted using one intron file and one exon file as the unknown sequences. Each file contained approximately one hundred sequences. The six sets of rules compiled for each of the two, three, and four extensions to the splice sites were applied to the exon file then the intron file. In addition to these variables, a break point was utilized as a discriminator. For example, a break point value of five would allow saying that all sequences having greater than five matches are introns, while all those having less than or equal to five are exons. The break point values ranged from one to sixteen. From the multitude of values representing correct predictions of exons and introns, the following percentages were selected (Table 6).

Table 6. Highest percentages of correct predictions of exons and introns for each group of splice site rules.

EXTENSION LENGTH	GROUP 1	GROUP 2	GROUP 3	GROUP 4	GROUP 5	GROUP 6
Len. 2 EXONS	50	56	68	59	62	66
Len. 2 INTRONS	73	79	58	66	66	62
Len. 3 EXONS	55	63	70	72	68	64
Len. 3 INTRONS	88	76	76	72	63	65
Len. 4 EXONS	81	55	72	82	76	66
Len. 4 INTRONS	54	83	78	74	73	62

The criteria for the selection was based upon the highest percentage of correctly predicted exons as well as introns. It was consistently observed that as the break point value increased, the percentage of correctly predicted introns decreased and the percentage of correctly predicted exons increased. Therefore, the difference between the two percentages representing the exons and introns was a factor in the decision as to which to accept as most satisfactory. If the differences were identical between two sets

of percentages for a particular group of rules, the sets that favored the introns were chosen.

Of the percentages represented in table 6, those compiled for rules with extensions of length two can be discounted. The percentages and their corresponding break point values selected for lengths three and four were those for Group 3 and Group 4. These four sets of rules were used to filter the complete exon file (2,838 exons) and the complete intron file (740 introns). The results are demonstrated in figure 68.

Figure 68. The predictive accuracy of the four sets of selected rules is shown to be very close.

The four sets of rules resulted in accuracy predictions in the range between 65% and 75%. Group 4 of the four base extension out performed Group 3 of the 3 base extension by only 1%.

CHAPTER 5

SUMMARY

The finite induction pattern matching technique was implemented to discover its potential for discriminating between the exon and intron regions of a DNA sequence. The data sets for the project were compiled from a database containing “cleaned” samples extracted from GenBank release 78. Data sets of various sizes were used to build sets of rules referred to as filters. Filters were constructed for both exons and introns. Data sets not containing any of the sequences involved in building the filters were utilized as “unknown” sequences. Multiple FACTORING algorithms were prepared for the development of various filters and multiple related FOLLOWINGS were created to study the reaction of the unknown sequences as they were passed through a given filter. A wide range of inductive bases, which restricted the length of the rules, was tested.

The first study involved that of the EXACT FOLLOWING. The rules derived for the unknowns were required to find an exact match as they were filtered. The residual, representing those base elements whose rules found no match; the content of the residual, the number of bases A, C, T, and G; and the distance between residual components were used as measurements. Regardless of the size of the filter or the size of the inductive base, none of the above measurements presented a difference between the exons and introns.

BLIND FOLLOWING was exercised for the second study. This algorithm is more forgiving in that it requires only the antecedent part of the rules to match. The various sized filters for both the exons and introns were run with the twelve inductive bases. Again the measurements of residual length, content, and distance were observed. There were no distinguishing marks between the behavior of the exons versus that of the introns.

The REPLACEMENT FOLLOWING is not only forgiving, but it also attempts to be a correcting algorithm. The rules, for which only the antecedent finds a match but not the consequent, have their consequent replaced by that of the filter rule. This new consequent becomes a component of the resulting residual that is then sent through the filtering and following processes again. Regardless of the filter size, the filter type, or the inductive base, the exons behaved similarly to the introns.

It was thought that perhaps increasing the alphabet size of the DNA sequences might reveal some divergent characteristics between the exons and introns. The sixteen unique pairs of nucleotides (AA, AC, ..., TG, TT) were assigned a letter A through P. The sequences were first converted to their new symbolic representation by taking two nucleotide bases and assigning them their appropriate symbol. A one base right shift was made and the next two bases were assigned their respective symbol. New filters were constructed for the newly formed sequences. As each unknown was converted, it was passed through the filter. The results presented no indication of variance between the exons and introns.

Once again the alphabet was increased. This time the DNA sequences were represented by sixty-four symbols. The same processing was conducted for this extended alphabet as for the sixteen-symbol alphabet. No improvement was shown.

The next exercise attempted to capture the nature of the codons (triplets). By incorporating the extended alphabet, which produced a symbol for each triplet, with a right shift of three bases rather than one, it was thought that the coding ability of the exons might be distinguished from the non-coding of the introns. There was a dramatic drop in the number of matches (74% down to 15%), however, this occurred for both the introns as well as the exons.

Individual exon sequences (non-concatenated) do not necessarily begin on codon boundaries, nor do they necessarily contain complete codons at their boundaries. A file of exon sequences containing complete codons was painstakingly compiled. A filter was constructed from a portion of those sequences. The remainder of the “pure codon” sequences was passed through the filter. The results were compared with the results of passing an intron file of sequences through the same filter. No distinguishing factor was revealed. In fact, the “pure codon” sequences performed very similarly to those sequences that had not been altered.

The FOLLOWING was again revised. Previously, the rules of each unknown sequence were first determined then matched to those of the filter. This feature was removed from the algorithm. Now, each rule of the filter was passed across the unknown sequence in an attempt to match some linear sequence of bases. Three renderings of this technique were implemented.

The first rendition involved data sets and runs similar to those of EXACT FOLLOWING. The difference, of course, was that the unknown sequences were not factored into their respective rule sets. The exon sequences could not be distinguished from the introns when the runs were compared. The second trial involved modifying the FOLLOWING to handle the sequences in the “pure codon” file. Nucleotides were taken three at a time, non-overlapping. Again there was no difference shown between exons and introns.

It was decided to look exclusively at the signals provided by the data sets. Rather than construct RULINGS based on both the content and the signals, as done for all the previous experiments, the filter would contain only those rules reflecting the behavior of the 5' and 3' splice sites. These were the only obvious signals available in the data sets. Although the start and stop codons could be observed in certain exons, they did not present a strong enough presence due to the inclusion of internal as well as initial and terminating exons. Also, because of the division of the data into separate exon and intron sets, only the intron side of the splice sites could be analyzed.

Multiple sets of rules of lengths four, five, and six were established as filters. These rules were constructed by looking to the right of the 5' splice site a given number of bases (2, 3, and 4) and, likewise, to the left of the 3' splice site. Six groups of the rules were constructed depending upon the probability of their being seen. The first group contained only those rules seen most often. The sixth group contained all the rules.

From the onset, there was an obvious difference between the exons and introns as to the number of patterns in the unknown sequences that matched the rules in the filter.

This difference allowed for the establishment of a break point. The breakpoint would then relate accuracy statistics for the unknown sequences as they were run. Multiple breakpoints could be exercised for the various sets of rules. The sets of splice site rules and corresponding break points that resulted in the highest percentage of correct predictions for both exons and introns were selected. There were six sets representing both a three base extension and a four base extension that produced remarkably close results. These six sets were executed using the EXACT FOLLOWING with the complete exon and intron data files as the unknown sequences. The highest percentages obtained were a correct identification of 75% of 2838 exons and 67% of 740 introns. The RULING (possibly of some biological interest) used to obtain these percentages is shown below with the first column representing the donor sites and the second, the acceptor sites.

GTAAG -> T	CTGCA ->G
GTGAG -> T	CCACA -> G
GTAAG -> G	TTTCA -> G
GTGAG -> C	CCCCA -> G
GTGAG -> G	CCGCA -> G
GTAAG -> A	CCTCA -> G
GTAAG -> C	CTCCA -> G
GTATG -> T	TCCCA -> G
GTGAG -> A	TTCCA -> G
GTAGG -> T	TTGCA -> G
GTATG -> G	CTACA -> G
GTGGG -> T	CTCTA -> G
	CTTCA -> G
	TCACA -> G
	TCCTA -> G
	TTACA -> G
	TTCTA -> G
	TTTTA -> G

As demonstrated in the conclusion, these are remarkably high accuracy figures considering that other than the conserved GT and AG dinucleotides, no other biological information is supplied to the algorithm. All other algorithms noted in the literature contain statistical and mathematical analysis of the content and signals of the unknown sequences in comparison to that of known sequences. They also take advantage of methods of learning whereby the information contained in the algorithm is improved each time additional data is made available. The finite pattern matching technique currently does not avail itself of any of these features. Given that opportunity, its predictive accuracy should become even greater.

CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

6.0 Conclusion

As noted in the summary, the experiment involving the splice sites produced the most positive results. A sensitivity of 75% (correctly predicted positives) was found for the exons and 67% (correctly predicted negatives) for the introns. Conversely, the specificity was 25% (incorrectly predicted positives) and 33% (incorrectly predicted negatives), respectively. Applying the formula for the correlation coefficient described in section 2.3, a value of 0.36 was calculated.

Table 7 compares FI with the integrated systems described in section 2.3. The range of values indicated for GenelD and GRAIL represent various data sets. This is a problem with representing accurate comparisons. Many times comparisons are made when the systems are utilizing different data sets albeit the data sets have their origin in GenBank.

FI presents itself well in table 7 when one takes into consideration that it is not an integrated system as are the others. This means that it does not incorporate the learning algorithms present in most integrated systems. These learning algorithms glean information from the multitude of measurements presented in section 2.1 such as codon usage, local compositional complexity, 6-tuple frequency, length distribution and

Table 7. Accuracy of correct exon prediction for FI as compared to integrated gene search systems.

INTEGRATED SEARCH SYSTEM	C(X)
GeneModeler	0.02
GeneID	0.33-0.42
GRAIL	0.31-0.52
FI	0.36
GeneParser	0.47

periodic asymmetry, just to name a few. Neither does FI utilize statistical practices such as linear or quadratic discriminant analysis.

Table 8 compares FI with those systems dedicated to recognizing internal exons. Here again, these systems not only depend upon splice site recognition, but also content measurements specifically tailored for internal exons. In addition, the data sets for the internal exon predictors included internal exons only. The data sets for FI included initial and terminal exons as well as internal exons.

Table 8. The comparison of FI with internal exon prediction systems.

INTERNAL EXON PREDICTORS	Sn
FI	0.36
GRAIL2	0.53
FGENEH	0.73
MZEF	0.78

Two problems concerning FI's amenability to DNA sequences are to be noted. First of all, a sequence of symbols is best served by FI when it possesses a large alphabet such as those available in music and digital imaging. The alphabet for DNA is simply the four nucleotides. However, increasing the alphabet to sixty-four symbols to represent the codons still had no discriminating effect. This indifference is possibly attributable to the seemingly randomness of the DNA sequences.

An experiment was performed to compare actual DNA sequences with randomly generated sequences. Five random sequences of length 100 containing the four bases A, C, G, and T were generated (Fake Good). Then five random sequences of length 100 containing the nucleotide C with only 1% or 2% each of A, G, and T nucleotides were created (Fake Bad). All strings were factored with an unrestrained inductive base. Statistics were gathered on the antecedent lengths. The results are represented as percentages in figure 69.

This research suggests that FI would be a competitive measure for exon/intron discrimination if combined with the knowledge available concerning the content of these sequences. Knowing the patterns for the splice sites, resulted in immediate positive results. If coupled with information that cannot be represented as rules, FI should perform with even greater precision.

6.1 Recommendations

In that the FI pattern matching technique has demonstrated merit in regard to differentiating between exons and introns, the following recommendations are made for

Figure 69. The randomly generated strings (Fake Good) blend in with the actual DNA sequences. The antecedents of the randomly generated sequences “Fake Bad” are represented as an (almost) straight line.

future exploration. First, FI should be conducted on strings of greater complexity. The exon and intron data sets involved in this project were a first step and did not incorporate other known consensus sequences such as the promoter sites (although highly varied) and poly-adenylation signals. Strings that include both initial and terminal exons would grant the capability of constructing FI rules to represent the start and stop codons. FI rules could also be determined for the exon side of all 5' and 3' splice sites given strings in which exons and introns appear collectively.

A second area to explore should be that of the FI pattern matching technique in collaboration with various combinations of the content measurements as given in chapter 2. The measures based on DNA content traditionally involve repeating sequences of nucleotide bases of various length, diverse codon concentrations, and amino acid compositions. Weightings can be also be determined by studying the absence, rather than

the presence, of certain nucleotide sequences. It is suggested that a cooperative effort between FI and this additional information would increase the accuracy of prediction for the exon and intron data sets used in this project.

It is further recommended that a third experimental effort be conducted combining those FI rules constructed from the signals of recommendation number one with the measurements determined best suited as a result of following through with recommendation number two. Such exploration could possibly result in a system with the ability to locate and verify the existence of exon and intron sequences within larger DNA sequences.

APPENDIX
SUPPLEMENTARY CHARTS
FOR CHAPTER 4

Figure 16. Level comparison for INTRON (S), IB 2 – IB 4.

Figure 17. Level comparison for INTRON (S), IB 5 – IB N.

Figure 20. INTRON (S) sequence size as related to levels.

Figure 21. EXON (S) IB 5 – IB N sequence size as related to levels.

Figure 28. Residual for inductive base 2 using exon filter.

Figure 29. Residual for inductive base 3 using exon filter.

Figure 30. Residual for inductive base 4 using exon filter.

Figure 31. Residual for inductive base 6 using exon filter.

Figure 32. Residual for inductive base 7 using exon filter.

Figure 33. Residual for inductive base 8 using exon filter.

Figure 34. Residual for inductive base 11 using exon filter.

Figure 35. Residual for inductive base 15 using exon filter.

Figure 36. Residual for inductive base 20 using exon filter.

Figure 37. Residual for inductive base M using exon filter.

Figure 38. Residual for inductive base N using exon filter.

Figure 39. Residual for inductive base 2 using intron filter.

Figure 40. Residual for inductive base 3 using intron filter.

Figure 41. Residual for inductive base 4 using intron filter.

Figure 42. Residual for inductive base 5 using intron filter.

Figure 43. Residual for inductive base 6 using intron filter.

Figure 44. Residual for inductive base 7 using intron filter.

Figure 45. Residual for inductive base 8 using intron filter.

Figure 46. Residual for inductive base 11 using intron filter.

Figure 47. Residual for inductive base 15 using intron filter.

Figure 48. Residual for inductive base 20 using intron filter.

Figure 49. Residual for inductive base M using intron filter.

Figure 50. Residual for inductive base N using intron filter.

REFERENCES

- Bell, George I. 1990. The Human Genome: An Introduction. In *Computers and DNA: Proceedings of the Interface Between Computation Science and Nucleic Acid Sequencing Workshop held in Santa Fe, New Mexico 12-16 December 1988*, ed. George I. Bell and Thomas G. Marr, VII: 3-12. New York: Addison-Wesley.
- Brunak, Soren, Jacob Engelbrecht, and Steen Knudsen. 1991. Prediction of Human mRNA Donor and Acceptor Sites from the DNA Sequence. *Journal of Molecular Biology* 220: 49-65.
- Bucher, Philipp, and Edward N. Trifonov. 1986. Compilation and Analysis of Eukaryotic POL II Promoter Sequences. *Nucleic Acids Research* 14, no. 24: 10009-10026.
- Burks, Christian. 1990. The Flow of Nucleotide Sequence Data into Data Banks: Role and Impact of Large-Scale Sequencing Projects. In *Computers and DNA: Proceedings of the Interface Between Computation Science and Nucleic Acid Sequencing Workshop held in Santa Fe, New Mexico 12-16 December 1988*, ed. George I. Bell and Thomas G. Marr, VII: 35-45. New York: Addison-Wesley.
- Case, James H. and Paul S. Fisher. 1984. Long Term Memory Modules. *Bulletin Mathematical Biology* 46, no. 2: 295-326.
- Cinkosky, Michael J., James W. Fickett, Paul Gilna, and Christian Burks. 1991. Electronic Data Publishing and GenBank. *Science* 252 (May) no. 5010: 1273-1277.
- Das, Sajal K., Paul S. Fisher, and Hua Zhang. 1993. Efficient Parallel Algorithms for Pattern Recognition. In *Parallel Algorithms and Applications: Proceedings of Twenty-sixth Hawaii International Conference in Systems Sciences*, 2:81-98. Gordon and Breach Science Publishers S.A.
- Farber, Robert, Alan Lapedes, and Karl Sirotkin. 1992. Determination of Eukaryotic Protein Coding Regions Using Neural Networks and Information Theory. *Journal of Molecular Biology* 226: 471-479.
- Fickett, James W. 1982. Recognition of Protein Coding Regions in DNA Sequences. *Nucleic Acids Research* 10, no. 17: 5303-5318.

- Fickett, James W., and Chang-Shung Tung. 1992. Assessment of Protein Coding Measures. *Nucleic Acids Research* 20, no. 24: 6441-6450.
- Fields, C.A. and C.A. Soderlund. 1990. GM: a Practical Tool for Automating DNA Sequence Analysis. *CABIOS* 6, no. 3: 263-270.
- Fisher, Paul S., M.N. Novaes, and G.E. Mobus. 1990. Pattern Recognition and Generation in Music. In *Proceedings of the Music Workshop of the 1990 European Conference in Artificial Intelligence, Stockholm, Sweden*.
- Gall, Joseph. G. 1981. "Chromosome Structure and the C-Value Paradox. *Journal of Cell Biology* 91, no. 3: 3s-14s.
- Guigo, Roderic, Steen Knudsen, Neil Drake, and Temple Smith. 1992. Prediction of Gene Structure. *Journal of Molecular Biology* 226: 141-157.
- Hutchinson, Gordon B. and Michael R. Hayden. 1992. The Prediction of Exons Through an Analysis of Spliceable Open Reading Frames. *Nucleic Acids Research* 20, no. 13: 3453-3462.
- Konopka, Andrzej K. 1993. Plausible Classification Codes and Local Compositional Complexity of Nucleotide Sequences. In *Second International Conference on Bioinformatics, Supercomputing and Complex Genome Analysis*. ed. H. A. Lim, J. W. Fickett, C. R. Cantor, R.J. Robbins. World Scientific, Singapore. 69-87.
- Konopka, Andrzej K. and John Owens. 1990. Non-Continuous Patterns and Compositional Complexity of Nucleic Sequences. In *Computers and DNA: Proceedings of the Interface Between Computation Science and Nucleic Acid Sequencing Workshop held in Santa Fe, New Mexico 12-16 December 1988*, ed. George I. Bell and Thomas G. Marr, VII: 147-155. New York: Addison-Wesley.
- Korning, Peter G., Stefan M. Hebsgaard, Pierre Rouze, and Soren Brunakl 1996. Cleaning the GenBank *Arabidopsis thaliana* data set. *Nucleic Acids Research* 24, no. 2: 316-320.
- Lander, E.S., R. Langridge, and D. Saccocio. 1991. Mapping and Interpreting Biological Information. In *Communications of the ACM* 34, no. 11 (November): 33-39.
- Lapedes, A., C. Barnes, C. Burks, R. Farber, and K. Sirotkin. 1990. Application of Neural Networks and Other Machine Learning Algorithms to DNA Sequence Analysis In *Computers and DNA: Proceedings of the Interface Between Computation Science and Nucleic Acid Sequencing Workshop held in Santa Fe, New Mexico 12-16 December 1988*, ed. George I. Bell and Thomas G. Marr, VII: 157-182. New York: Addison-Wesley.

- Long, Manyuan, Sandro Jose de Souza and Walter Gilbert. 1995. Evolution of the Intron-Exon Structure of Eukaryotic Genes. *Current Opinion in Genetics & Development* 5: 774-778.
- Matthews, B. W. 1975. Comparison of the Predicted and Observed Secondary Structure of T4 Phage Lysozyme. *Biochimica et Biophysica Acta* 405:442-451.
- Nakata, Kotoko, Minoru Kanehisa and Charles DeLisi. 1985. Prediction of Splice Junctions in mRNA Sequences *Nucleic Acids Research* 13, no. 14: 5327-5340.
- Shapiro, Marvin B., and Periannan Senapathy. 1987. RNA Splice Junctions of Different Classes of Eukaryotes: Sequence Statistics and Functional Implications in Gene Expression *Nucleic Acid Research* 15, no. 17: 7155-7174.
- Silverman, B.D. and R. Linsker. 1986. A Measure of DNA Periodicity. *Journal of Theoretical Biology* 118: 295-300.
- Snyder, Eric E. and Gary D. Stormo. 1993. Identification of Coding Regions in Genomic DNA Sequences: and Application of Dynamic Programming and Neural Networks. *Nucleic Acids Research* 21, no. 3: 607-613.
- Solovyev, Victor V., Asaf A. Salamov, and Charles B. Lawrence. 1994. Predicting Internal Exons by Oligonucleotide Composition and Discriminant Analysis of Spliceable Open Reading Frames. *Nucleic Acids Research* 22, no. 24: 5156-5163.
- Stormo, G.D. 1987. Identifying Coding Sequences. Chapter 10 of *Nucleic Acid and Protein Sequence Analysis* ed. M.J. Bishop and C.J. Rawlings 231-258. Oxford: IRL Press.
- Staden, R. 1990. Finding Protein Coding Regions in Genomic Sequences. *Methods of Enzymology*, ed. R.F.Doolittle, 183:163-180. Academic Press, Inc.
- Tomita, Masaru, Nobuyoshi Shimizu, and Douglas L. Brutlag. 1996. Introns and Reading Frames: Correlation Between Splicing Sites and Their Codon Positions. *Molecular Biology Evolution* 13, no. 9: 1219-1223.
- Uberbacher, Edward C. and Richard J. Mural. 1991. Locating Protein-Coding Regions in Human DNA Sequences by a Multiple Sensor-Neural Network Approach. In *Proceedings of the National Academy of Science USA* 88: 11261-11265.
- Vignal, L., Y. d'Aubenton-Carafa, F. Lisacek, E. Mephu Nguifo, P. Rouze, J. Quinqueton, C. Thermes. 1996. Exon Prediction in Eucaryotic Genomes. *Biochimie*. 78: 327-334.

- Watson, James D., John Tooze, and David T. Kurtz. 1983. *Recombinant DNA: A Short Course*. Scientific American Books.
- Wills, Christopher, 1991. *Exons, Introns, and Talking Genes*. New York: Harper Collins.
- Zhang, Michael Q. 1997. Identification of Protein Coding Regions in the Human Genome by Quadratic Discriminant Analysis. In *Proceedings of the National Academy of Sciences, USA* 94: 565-568.