Algorithms for Molecular Biology **Fall Semester, 2000**

Lecture 7: December 11, 2000

Lecturer: Ron Shamir *Scribe: Michal Ozery-Flato and Gil Shklarski*

7.1 Gene Finding

7.1.1 Motivation

GenBank is huge. There are more than 3 billions bases of human DNA sequences and complete DNA sequences for dozens of species available in GenBank. Not all the sequences are *coding*, namely are a template for a protein. In the human genome only 3%-5% of the sequences are coding. Due to the size of the database, manual searching of genes who do code for proteins is not practical. We need to find a way for automatic finding of genes.

7.1.2 Biological Background

Gene expression is the biological process by which a DNA sequence generates a protein. It involves two steps: *transcription* and *translation*. Transcription produces a mRNA (messenger RNA) sequence using the DNA sequence as a template. The mRNA sequence produced is complementary to the DNA strand which was used as template. The subsequent process, called translation, synthesizes the protein according to information coded in the mRNA. This process is performed by sub cellular elements called *ribosomes* (Figure 7.1).

The transcription is carried out from the 5' end to the 3' end of the copied DNA strand (from 3' to 5' of the complementary, template strand). This direction along the strand is called *downstream* while the opposite direction is called *upstream*. The enzyme performing the transcription, RNA polymerase, starts transcription a few bases upstream of the region that actually codes for a protein, and terminates a few bases after the end of that coding region. The regions in both ends of the DNA coding region which are transcribed into mRNA, but do not code the protein are called *untranslated regions (UTR)* (see Figures 7.3 and 7.8). RNA polymerase molecules start transcription by recognizing and binding to *promoter regions* upstream of the desired transcription start sites. These promoter regions control the rate of gene expression.

Proteins are composed of amino acids. The ribosomes produce sequences of amino acids by translating the information coded in the mRNA sequences. Each triplet of bases in the mRNA is a command for the ribosomes, called *codon*. There are 64 different possible codons

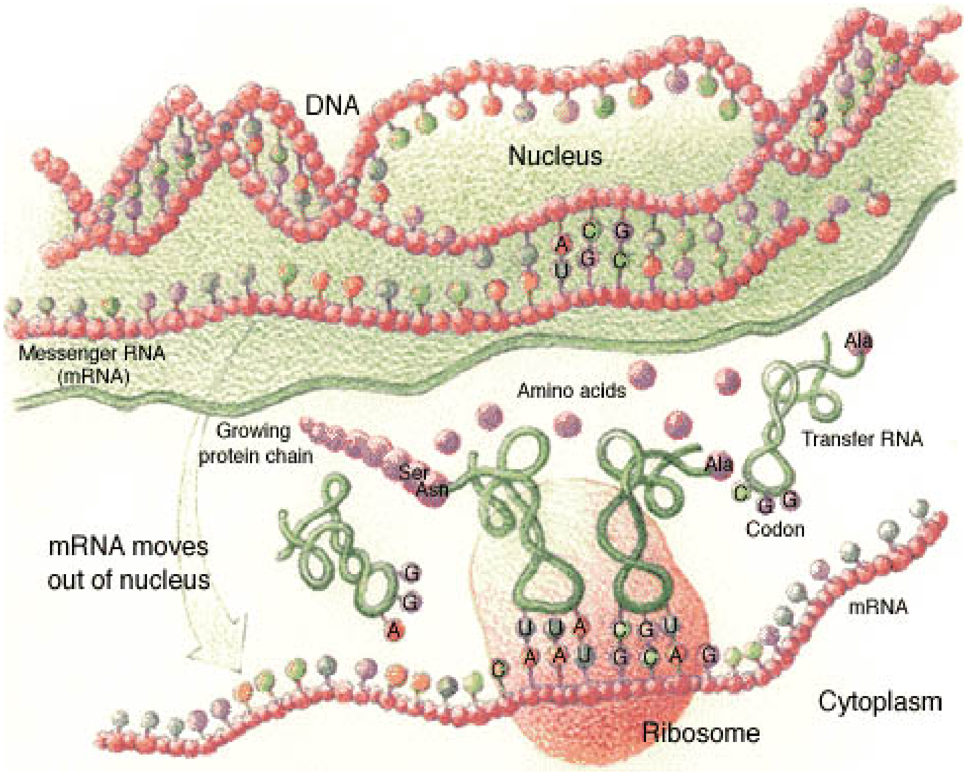


Figure 7.1: DNA → RNA → Protein [2].

and only 20 amino acids, thus multiple codons represent the same amino acid. Mapping from codons to amino acids, called the genetic code, is shown in Figure 7.2. One of the codons, called *start codon*, indicates the beginning of translation (as well as coding for the amino acid Methonine), and three, called *stop codons*, indicate end of translation. The ribosome scans the mRNA molecule, sliding along it from its 5' end to its 3' end. Upon detecting a start codon the ribosome starts generating an amino acid sequence coded by the mRNA. The process stops when that ribosome detects a stop codon.

7.2 Finding Genes in Prokaryotes

7.2.1 Prokaryotes

A *prokaryotic cell* is a cell which doesn't have a nucleus, while *eukaryotic* cells have nuclei. There are some differences between prokaryotic cells and eukaryotic cells which are relevant to gene recognition. In prokaryotic cells most of the DNA sequence is coding for protein. For example, almost 70% of the genome of the bacterium *H.influenzea* is coding. Also, each

The Genetic Code

	U	C	A	G	
	Phe	Ser	Tyr	Cys	U
U	Phe	Ser	Tyr	Cys	C
	Leu	Ser	STOP	STOP	A
	Leu	Ser	STOP	Trp	G
	Leu	Pro	His	Arg	U
C	Leu	Pro	His	Arg	C
	Leu	Pro	Gln	Arg	A
	Leu	Pro	Gln	Arg	G
	Ile	Thr	Asn	Ser	U
A	Ile	Thr	Asn	Ser	C
	Ile	Thr	Lys	Arg	A
	Met(start)	Thr	Lys	Arg	G
	Val	Ala	Asp	Gly	U
G	Val	Ala	Asp	Gly	C
	Val	Ala	Glu	Gly	A
	Val	Ala	Glu	Gly	G

Figure 7.2: The genetic code. AUG us the start codon, while UAA, UAG and UGA are the stop codons [6].

gene is one continuous stretch of bases. That is, there are no introns in the coding region.

7.2.2 Long ORFs

Open Reading Frames (ORFs)

Any given nucleotide sequence (single DNA strand or mRNA) can be interpreted in three possible ways, depending on where the coding starts. For example, the mRNA sequence $ACCUUAGCGUA$ can be translated into Threonine-Leucine-Alanine ($\underbrace{ACC}\underbrace{UUA}\underbrace{GCG}UA$), Proline-Stop-Arginine ($A\underbrace{CCU}\underbrace{UAG}\underbrace{CGU}A$) or Leucine-Serine-Valine ($AC\underbrace{CUU}\underbrace{AGC}\underbrace{GUA}$). These three ways are called *reading frames*. An *open reading frame (ORF)* is a sequence of codons with no stop codon.

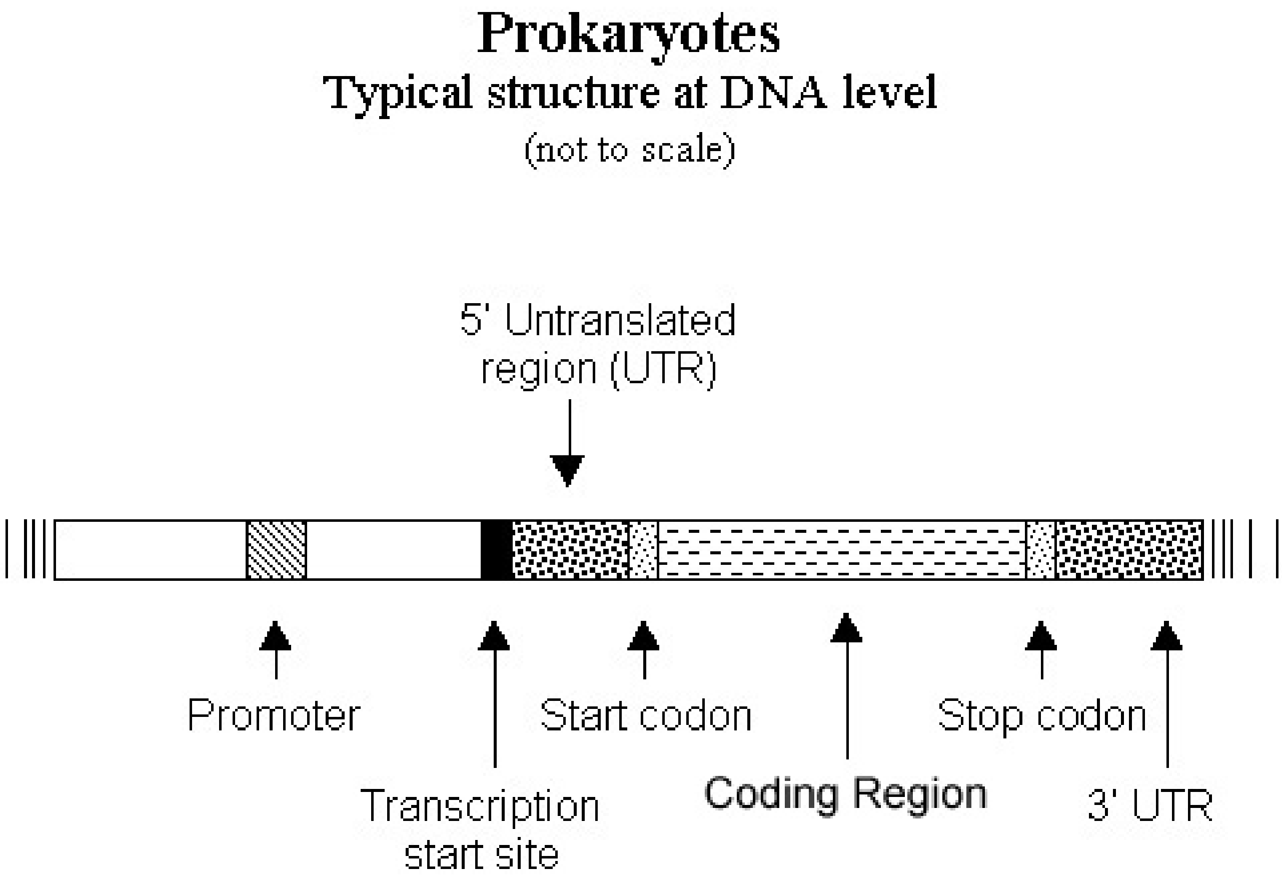


Figure 7.3: Typical prokaryotic gene structure at DNA level (not to scale) [?].

Finding long ORFs

One way to distinguish coding regions from non-coding regions, is to examine the frequencies of stop codons. Assuming a uniform random distribution, a stop codon is expected to be observed every $64/3 \approx 21$ codons (since there are 3 stop codons). Average proteins are much longer, being coded by about 1000bp (base pairs). Each coding region has only one stop codon, which terminates the region. Therefore, one way to detect the coding regions, is to look for long sequences of codons, without any stop codon. The algorithm that uses the above idea scans the DNA sequence, looking for long ORFs in all three reading frames. Upon detecting a stop codon, the algorithm scans backward, searching for a start codon. This algorithm will fail to detect very short genes, as well as overlapping long ORFs on opposite strands. Moreover, there are a lot more ORFs than genes. For example, we can find 6500 ORFs in the DNA of the bacterium *E.coli* while there are only 4400 genes.

7.2.3 Detection of Coding Regions

Codon frequencies in coding regions

A more informative method to determine coding regions, takes advantage of the frequencies in which the various codons occur in coding regions. For example, the amino acids Leucine, Alanine and Tryptophan are coded by 6,4 and 1 different codons respectively. In a translation of a uniformly random DNA sequence, these amino acids should occur in the ratio 6:4:1, but in a protein they occur in a different ratio - 6.9:6.5:1. Therefore coding DNA is not random. Another example of the non-uniformity of coding DNA is the fact that A or T occurs in the 3^{rd} position of a codon in a rate over 90% (these statistics vary for different species).

ORFs as Markov chains

Assuming we found all ORFs in a sequence, we can use codon frequencies to find which ORFs are coding and which are *non coding open reading frames (NORFs)*. We translate each ORF into a codon sequence and get 64-state Markov chain. We use a state for each codon rather than a state for each amino acid, because codons are more informative than their translations. (There might be a preference for a specific codon in gene expression over other codons that encode the same amino acid). The transition probabilities are the probabilities for each codon to follow any other codon in a coding region. Using this model, we can compute the probability that a given ORF is really a coding region. Figure 7.4 shows the results of such method.

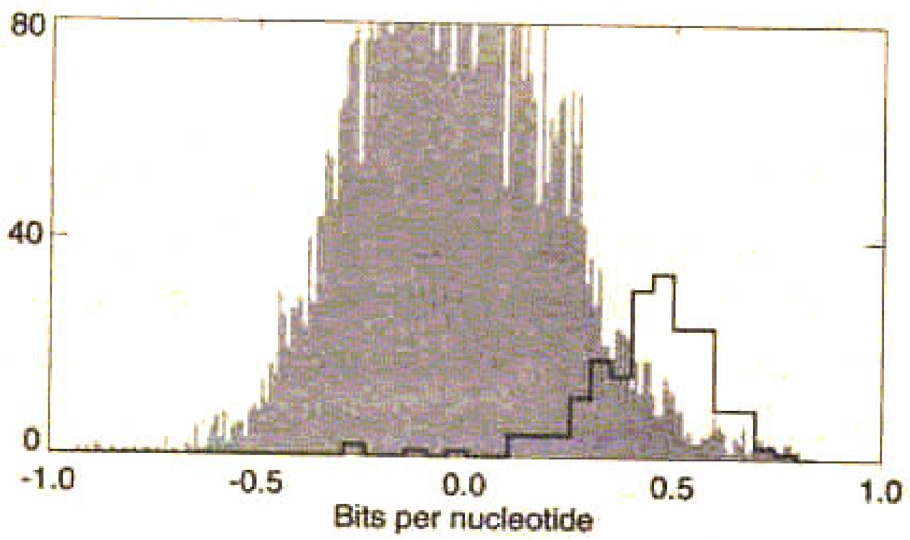


Figure 7.4: Coding regions recognition using 64 state markov model [?].

Using codon frequencies

In the model described above the probability of a codon occurrence depends on the preceding codon. We now consider a simpler model in which successive codons are independent. Let f_{abc} denote the frequency with which the codon abc occurs in a coding region. Given a coding sequence $a_1, b_1, c_1, a_2, b_2, c_2, \ldots, a_{n+1}, b_{n+1}$ with an unknown reading frame, the probability of observing the sequence of n codons appearing in the reading frame starting with $a_1b_1c_1$ is

$$p_1 = f_{a_1b_1c_1} \times f_{a_2b_2c_2} \times \ldots \times f_{a_nb_nc_n}$$

Similarly, the probability of observing the n codons in the second and third coding frames are:

$$\begin{aligned} p_2 &= f_{b_1c_1a_2} \times f_{b_2c_2a_3} \times \ldots \times f_{b_nc_na_{n+1}} \\ p_3 &= f_{c_1a_2b_2} \times f_{c_2a_3a_3} \times \ldots \times f_{c_na_{n+1}b_{n+1}} \end{aligned}$$

Let P_i denote the probability of the ith reading frame being the coding reading frame (assuming the region is coding). P_i can be calculated as follows:

$$P_i = \frac{p_i}{p_1 + p_2 + p_3}$$

The above computation can be used in a search algorithm as follows: Slide a window of size n along the sequence, and compute P_i for each start position of the window. The *Codon Preference* program, which is part of the *GCG* library, implements this method.

Figure 7.5 shows a the plot of $log(P/1 - P)$, which is the log likelihood, for the three reading frames. Each point represents the score for a 25 codon window around it. The actual genes are plotted as rectangles at the bottom. We can see that in the reading frame matching the upper plot, the genes are clearly recognized.

Figure 7.6 shows the plot of a program using only the 3^{rd} position bias information. These methods depend on the accuracy of the codon frequency statistics of already found genes. The algorithm will also have difficulty in detecting horizontal gene transfer and other causes of heterogeneity.

7.2.4 Detection of Promoter Regions

RNA Transcription

Not all open reading frames are transcribed into genes. The transcription depends on regulatory regions that control the transcription rate. In the transcription process, an RNA polymerase binds tightly to the promoter. The promoter is an "anchor" point, it pinpoints where RNA transcription should begin. At the stop signal the polymerase releases the RNA and detaches itself from the DNA.

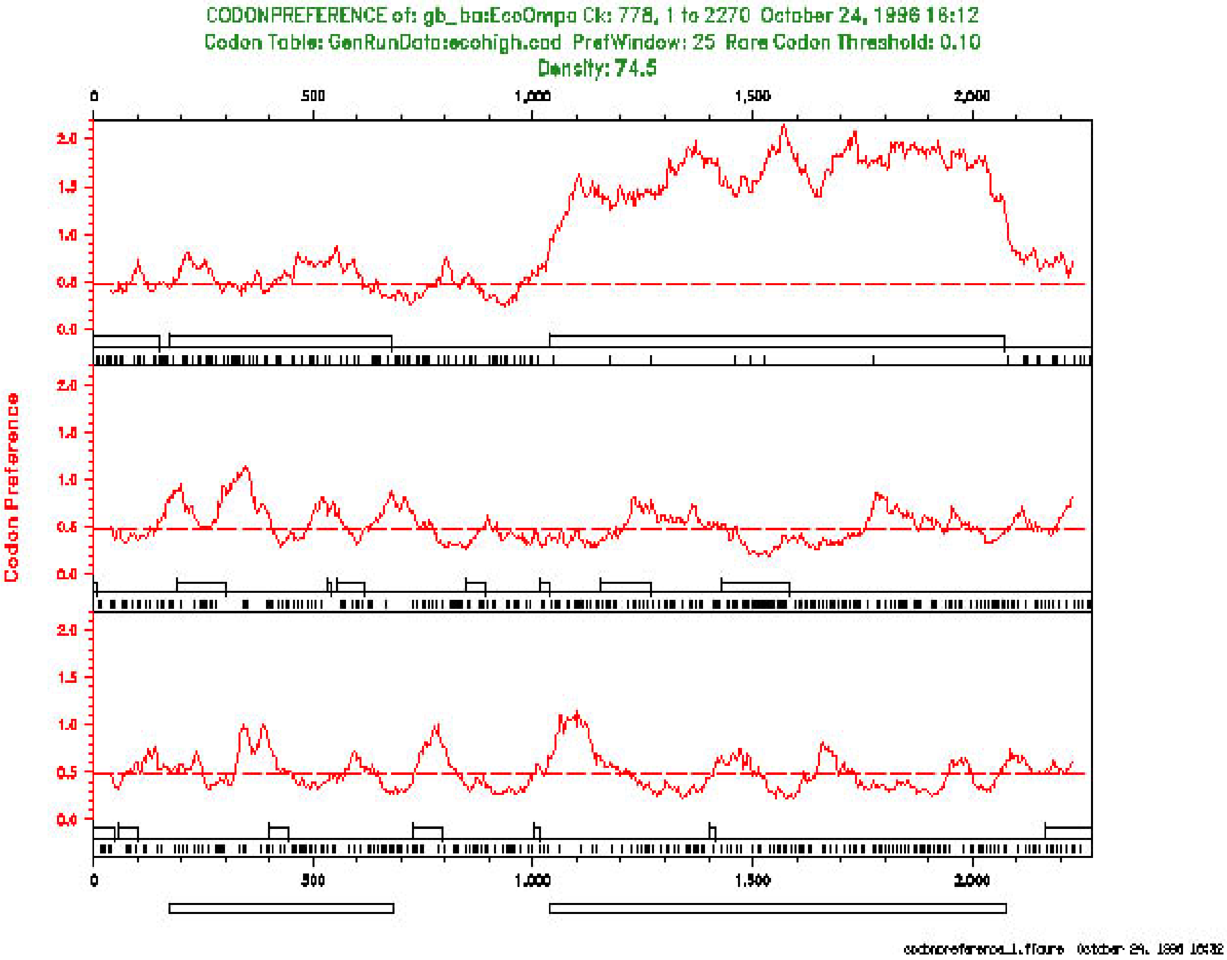


Figure 7.5: Results of codon preference program [1].

E.coly **promoters**

We will use *E.coly* as an example. In *E.coly* we can find the following consensus sequence around RNA transcription start point: $nnnTTGACAnnnnnnnnnnnnnnnnnTATAATnnnnnnNnnn$. N is the transcription start point. $TTGACA$ appears 35 bases before N, and $TATAAT$ (also known as *TATA box* or Pribnow box) appear 12 bases before N. We have here 2 anchor points for the polymerase. These sequences are short but the frequency of their occurrence is high enough to stand out. There are other common features that can be used to recognize promoter regions which are beyond the scope of this lecture.

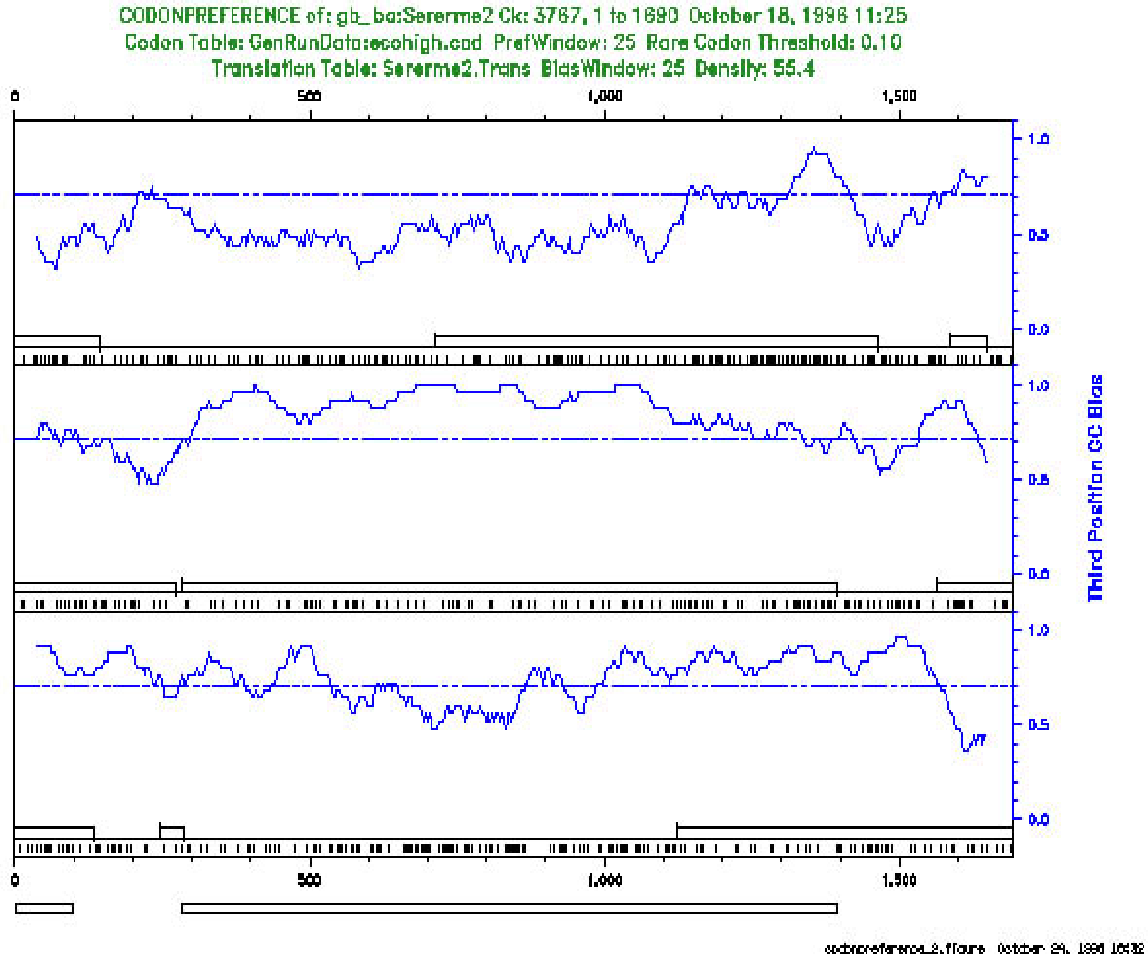


Figure 7.6: Results of codon preference program - 3^{rd} position bias [1].

Positional Weight Matrix

Due to the variability of binding site sequences, exact methods cannot be used for identifying promoter regions by the TATA box. Instead, we use a pattern search method based on frequencies. We construct a table of statistics, $f_{b,i}$, where $f_{b,i}$ is the frequency of the base b in position i of the known promoter region suffixes. We assume positions are independent.

Scoring Function

Let f_b denote the expected frequency of the base b in the genome (the background frequency). We calculate the likelihood of a given sequence being a TATA-box. For a sequence $S =$

$B_1 B_2 \dots B_6$ the likelihood of it being a TATA-box is:

$$P(S|S \text{ is a TATA-box}) = \prod_{i=1}^{6} f_{B_i,i}$$

Similarly, the likelihood of observing it, given it is a "non-promoter" is:

$$P(S|S \text{ is not a TATA-box}) \approx P(S) = \prod_{i=1}^{6} f_{B_i}$$

The log-likelihood ratio is therefore:

$$\log\left(\frac{P(S|\text{promoter})}{P(S|\text{non-promoter})}\right) = \log\left(\frac{\prod_{i=1}^{6} f_{B_i,i}}{\prod_{i=1}^{6} f_{B_i}}\right) = \sum_{i=1}^{6} \log\left(\frac{f_{B_i,i}}{f_{B_i}}\right)$$

This model has the disadvantage that it doesn't exploit all of the known information (i.e. dependencies between bases occurring in the promoter regions etc.) The f_{B_i} are given in Figure 7.7.

pos:	1	2	3	4	5	6
A	2	95	26	59	51	1
C	9	2	14	13	20	3
G	10	1	16	15	13	0
T	79	3	44	13	17	96

Figure 7.7: Positional weight matrix for TATA box [**?**].

Promoter variation

Why aren't promoters precise like the stop codons etc.? A possible answer is that nature uses the variation in promoters to control expression levels of various genes. That is, the rate of the gene expression process depends on the conservation of the promoter region. This hypothesis is supported by results from chemistry. Experiments show that when a RNA polymerase molecule gets bounded to the promoter region in order to initial transcription, there is an 80% correlation between the weight matrix score of the region and the binding energy. This means that if the promoter region is very conserved, i.e., very similar to the consensus sequence, then the binding energy barrier is low and thus the protein production rate is higher (because the RNA polymerase can easily bind to the protein coding region). When the difference from the consensus sequence is bigger, the energy barrier is higher, and the protein production is slower. One consequence of this insight is that finding regulatory sequences is an inherently stochastic problem.

7.3 Gene Finding in Eukaryotes

7.3.1 Eukaryote gene structure

The gene structure and the gene expression mechanism in eukaryotes are far more complicated than in prokaryotes. In typical eukaryotes, the region of the DNA coding for a protein is usually not continuous. This region is composed of alternating stretches of *exons* and *introns*. During transcription, both exons and introns are transcribed onto the RNA, in their linear order. Thereafter, a process called *splicing* takes place, in which the intron sequences are excised and discarded from the RNA sequence. The remaining RNA segments, the ones corresponding to the exons, are ligated to form the mature RNA strand. A typical multi-exon gene has the following structure (as illustrated in Figure 7.8). It starts with the promoter region, which is followed by a transcribed but non-coding region called *5' untranslated region (5' UTR)*. Then follows the initial exon which contains the start codon. Following the initial exon, there is an alternating series of introns and internal exons, followed by the terminating exon, which contains the stop codon. It is followed by another non-coding region called the *3' UTR*. Ending the eukaryotic gene, there is a polyadenylation (polyA) signal: the nucleotide Adenine repeating several times. The exon-intron boundaries (i.e., the splice sites) are signaled by specific short (2bp long) sequences. The 5'(3') end of an intron (exon) is called the *donor* site, and the 3'(5') end of an intron (exon) is called the *acceptor* site.

7.3.2 Typical figures: vertebrates

We'll use the example of vertebrates. On average, a vertebrate gene is around 30Kb long, out of which the coding region is only about 1-2Kb long. The average coding region consists of 6 exons, each about 150bp long. The promoter is about 6bp long and appears about 30bp upstream of the *transcription start site (TSS)*. Huge deviations from the average are observed. For example, the gene called *dystrophin* is 2.4MB long. *Blood coagulation-factor VIII* has 26 exons whose size varies from 69bp to 3106bp. Intron number 22 produces 2 transcripts unrelated to this gene, one for each strand. Other typical figures are transcription rate of less than 50b/sec and splicing process taking several minutes. Figure 7.9 shows typical length distribution of introns and exons in human genes.

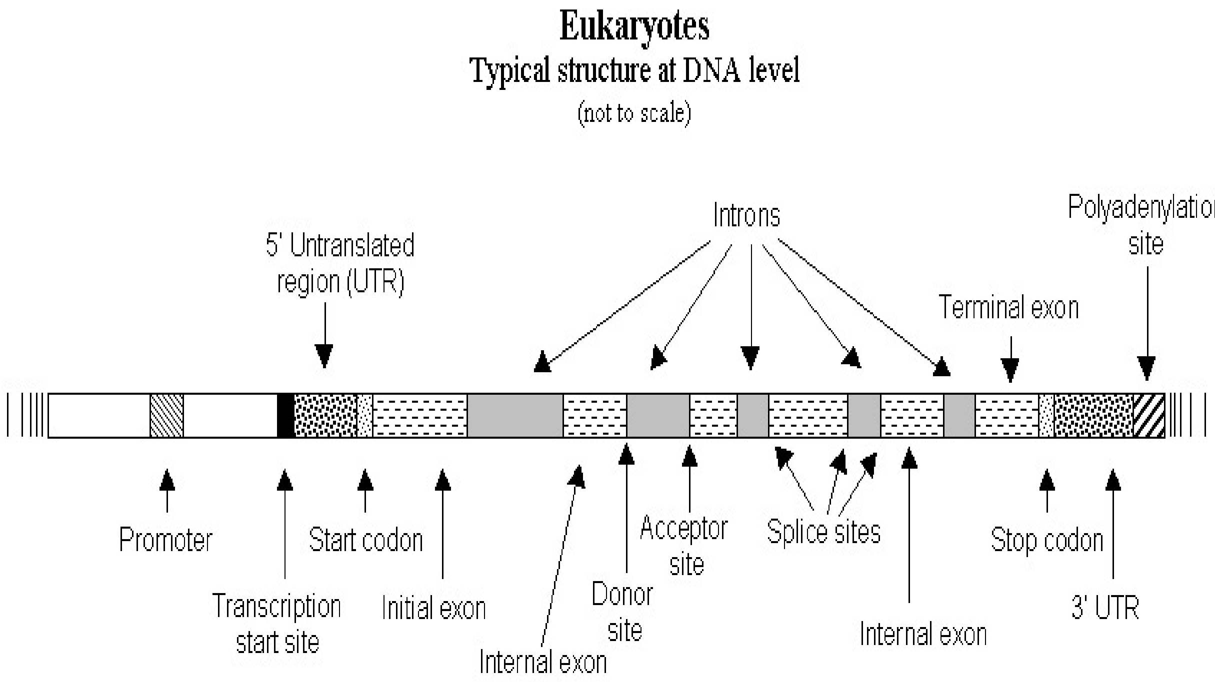


Figure 7.8: Typical eukaryote gene structure [?].

7.3.3 Markov Sequence Models

There are some models for distinguishing coding regions from non-coding regions that use Markov chains. These models are based on statistical differences between coding and non-coding regions.
A popular model is based on examining windows of 6 consecutive bases in the DNA sequence. This is a 5^{th} order Markov model. We'll prepare in advance two probability tables one for coding regions and one for non-coding regions. Each table will be of size 4^6. For each 6-tuple of bases the table will register the probability of observing the 6^{th} base, given the 5 preceding bases appeared in our window. Given a sequence we'll estimate the likelihood of it being coding using those 2 tables.
This model does not take into account any reading frame information. It is therefore called a*homogeneous* model. An *non-homogeneous* model is a model that has different tables for the 3 possible reading frames. The problem with such models when dealing with eukaryote genome is that sometimes the exons are too short and that it is hard to detect *splice junctions* (*donor* and *acceptor* sites).

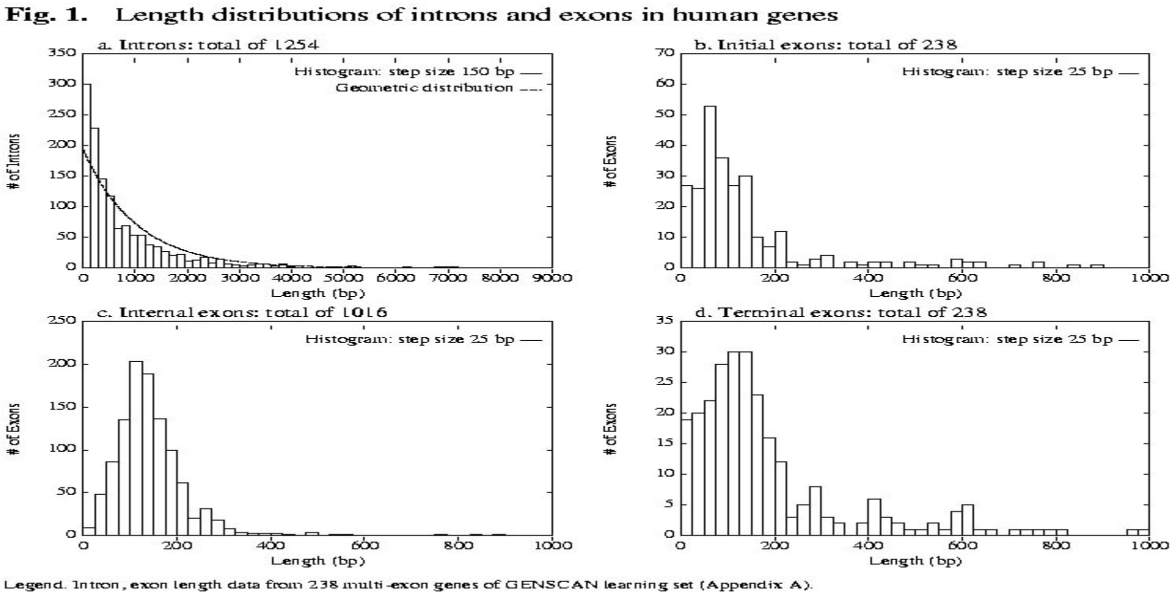


Figure 7.9: Distribution of intron and exon lengths [?].

7.3.4 Splice junctions

Figure 7.10 shows a consensus sequence for a typical eukaryote gene. We can see that in the exon-intron junctions there is great similarity to the consensus sequence (i.e. the frequencies there are close to 100%).

The figure shows an anchor point in the intron called *branch point* that appears frequently. Another statistical characteristic is a pyrimidine (bases C,T) rich area that appears between the branchpoint and the acceptor site. This naturally leads to using algorithms based on position specific weight matrices. This idea does not exploit all the information (reading frames, intron/exon states, etc) and is not suitable for short genes. Consequently, we will look for integrated approaches.

7.3.5 Length Distribution

A possible Hidden Markov Model (HMM)

Figure 7.11 shows a simple hidden Markov model for identifying a gene. Assuming probability p for staying in an exon state, the exon state length has a geometric distribution:

$$P(\text{exon of length } k) = p^k \times (1-p)$$

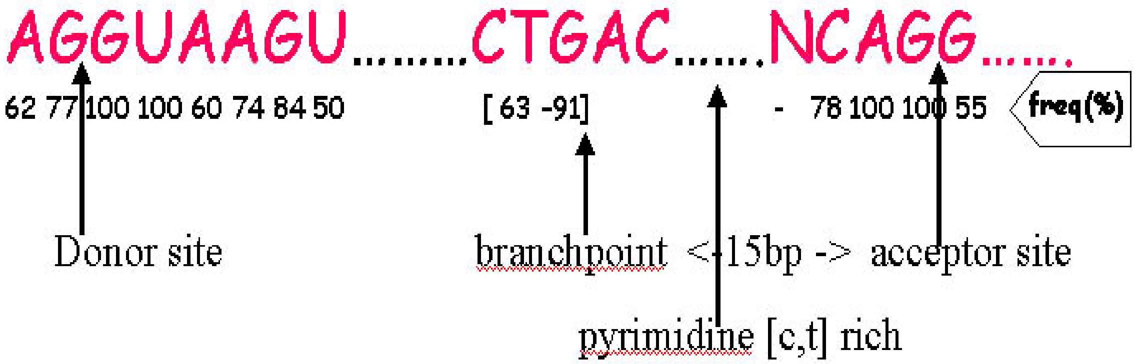


Figure 7.10: Intron-Exon splice junctions [?].

Since an HMM is a memory-less process, the only length distribution that can be modeled is geometric.

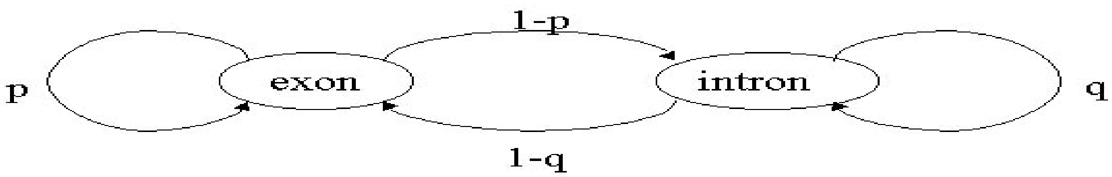


Figure 7.11: Simple HMM for eukaryotic genes [?].

Exon Length Distribution

Unfortunately, exon length does not have a geometric distribution. The length seems to have a functional role on the splicing itself. Typically, exons that are too short (under 50bp) leave no room for the *spliceosomes* (enzyms that perform the splicing) to operate and exons that are too long (above 300bp) are being difficult to locate. This leads us to search for another model for exon length.

7.3.6 Generalized HMM

Introduction to GHMM

As we have seen, a Hidden Markov Model (HMM) is a Markov chain in which the states are not directly observable. Instead, the output of the current state is observable. The output symbol for each state is randomly chosen from a finite output alphabet according to some probability distribution.

A *Generalized Hidden Markov Model* (GHMM) generalizes the HMM as follows: in a GHMM, the output of a state may not be a single symbol. Instead, the output may be a string of finite length. For a particular current state, the length of the output string as well as the output string itself might be randomly chosen according to some probability distribution. The probability distribution need not be the same for all states. For example, one state might use a weight matrix model for generating the output string, while another might use a HMM. Formally a GHMM is described by a set of four parameters:

- A finite set Q of states.
- Initial state probability distribution π.
- Transition probabilities $T_{i,j}$ for $i, j \in Q$.
- Length distributions f of the states (f_q is the length distribution for state q).
- Probabilistic models for each of the states, according to which output strings are generated upon visiting a state.

GenScan Model

The probabilistic model for gene structure as suggested by Berge and Karlin [3], is based on a GHMM (see Figure 7.12).

The states of the GHMM correspond to the different functional units on a gene, like promoter regions, exon, intron etc. The transition between the states ensure that the order in which the model visits various states is biologically consistent.

The states for an intron and an internal exon are subdivided according to *phase* offset to the codon frames. For $0 \leq i \leq 2$, the state I_i (respectively, E_i) corresponds to introns (exons) starting i positions after a codon starts. Note that the only transition from I_i to any internal exon state is to E_i. Also note that the model is divided into two symmetric halves. The upper half of the figure (states with a "+" superscript) models a gene on the forward strand and the lower half models a gene on the backward strand of the genomic sequence. If the parameters (like π, $a_{i,j}$, etc.) are suitably determined, then the model can be used for gene structure prediction in the following manner.

prediction

Definition 7.1 A *parse* Φ of sequence S of length L is an ordered sequence of states $(q_1, \ldots, q_t)$ with an associated duration d_i to each state ($L = \sum_{i=1}^{t} d_i$). Parse is actually a possible annotation to a base squence, matching each subsequence with appropriete functional unit of a gene.

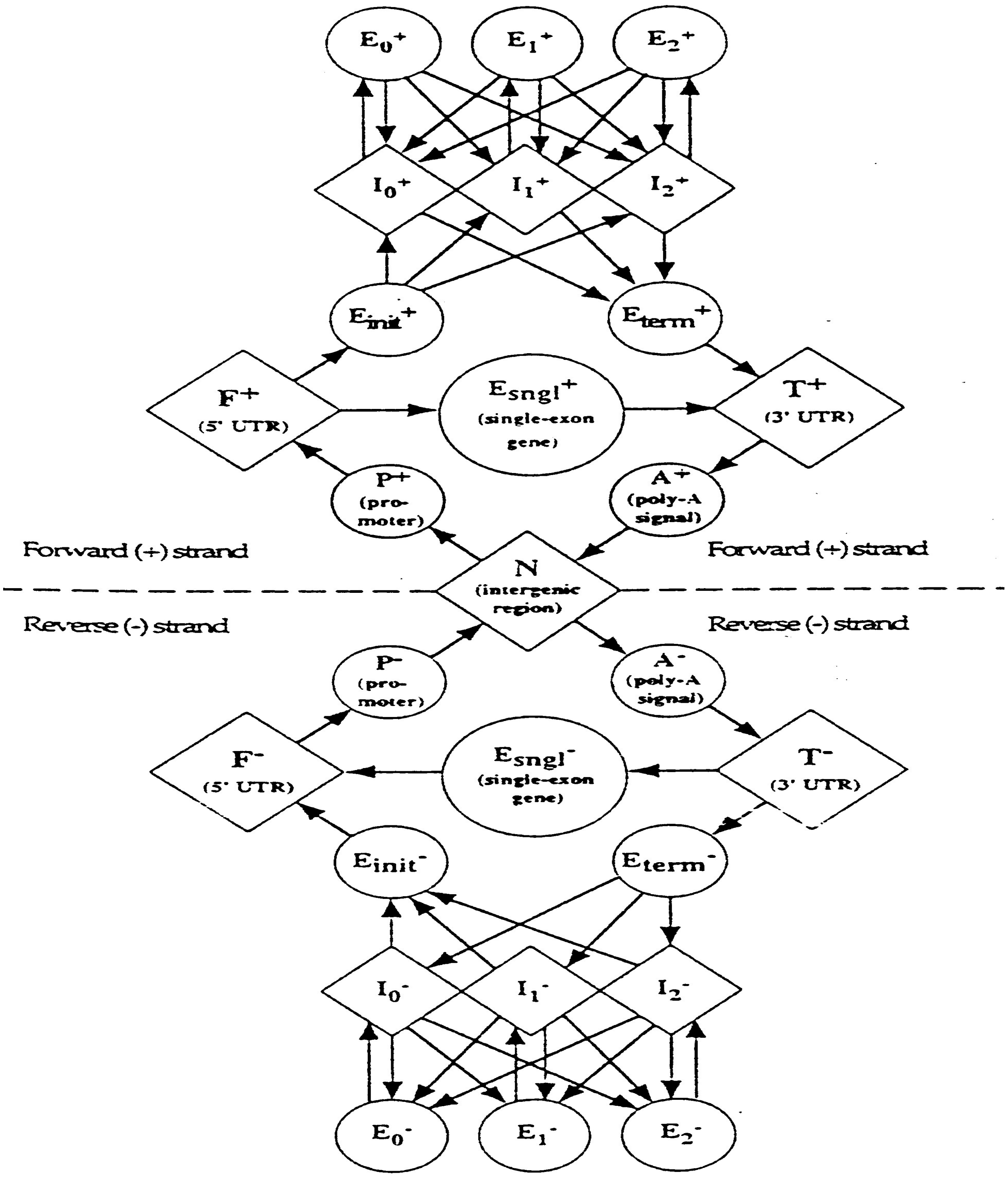


Figure 7.12: GenScan Model [3].

We are given a parse Φ and a sequence S. Let S_i be the segment of S produced by q_i, and let $P(S_i|d_i)$ be the probability of generating S_i by the sequence generation model of state q_i with length d_i. The probability the model went through the states to create S according to Φ is:

$$P(\Phi, S) = \pi_{q_1} f_{q_1}(d_1) P(S_1|d_1) \prod_{k=2}^{t} T_{q_{k-1}q_k} f_{q_k}(d_k) P(S_k|d_k)$$

Suppose we are given a DNA sequence S and a specific parse Φ, both of length L. The conditional probability of the parse Φ given that the sequence generated is S, can be computed as:

$$P(\Phi|S) = \frac{P(\Phi, S)}{P(S)} = \frac{P(\Phi, S)}{\sum_{\Phi_i \text{ is a parse of length L}} P(\Phi_i, S)}$$

The most probable parse, Φ_{opt}, can be computed by Viterbi like algorithm. $P(S)$ can be computed by a forward-like algorithm.

7.3.7 GENSCAN

GENSCAN, a computer program for gene identification, uses this model. In this section we shall consider how GENSCAN determines various parameters of the model to get meaningful results. The program uses a training set of completely sequenced genes from GenBank.

C+G Content

The C+G content (*isochore*) of the genomic sequence has a strong effect on gene density (see Figure 7.13), gene length etc. For example: gene density in C+G rich regions is 5 times higher than moderate C+G regions and 10 times higher than rich A+T regions. Thus, for training GENSCAN the training set is divided into four categories depending on the C+G content of the sequence. The categories are:

1. ($<$ 43% C+G)
2. (43 -51% C+G)
3. (51 - 57% C+G)
4. ($>$ 57% C+G)

For each of these categories, separate initial state probabilities, transition probabilities and state length distributions are computed.

Table 3. Gene density and structure as a function of C + G composition: derivation of initial and transition probabilities

Group	I	II	III	IV
C + G% range	<43	43-51	51-57	>57
Number of genes	65	115	99	101
Est. proportion single-exon genes	0.16	0.19	0.23	0.16
Codelen: single-exon genes (bp)	1130	1251	1304	1137
Codelen: multi-exon genes (bp)	902	908	1118	1165
Introns per multi-exon gene	5.1	4.9	5.5	5.6
Mean intron length (bp)	2069	1086	801	518
Est. mean transcript length (bp)	10866	6504	5781	4833
Isochore	L1 + L2	H1 + H2	H3	H3
DNA amount in genome (Mb)	2074	1054	102	68
Estimated gene number	22100	24700	9100	9100
Est. mean intergenic length	83000	36000	5400	2600
Initial probabilities:				
Intergenic (N)	0.892	0.867	0.540	0.418
Intron (I_0^+, I_1^+, I_2^+, I_0^-, I_1^-, I_2^-)	0.095	0.103	0.338	0.388
5′ Untranslated region (F^+, F^-)	0.008	0.018	0.077	0.122
3′ Untranslated region (T^+, T^-)	0.005	0.011	0.045	0.072

The top portion of the Table shows data from the learning set of 380 genes, partitioned into four groups according to the C + G% content of the GenBank sequence; the middle portion shows estimates of gene density from Duret *et al.* (1995) for isochore compartments corresponding to the four groups above; the bottom portion shows the initial probabilities used by GENSCAN for sequences of each C + G% compositional group, which are estimated using data from the top and middle portions of the Table. All of the values in the top portion are observed values, except the proportion of single-exon genes. Since single-exon genes are typically much shorter than multi-exon genes at the genomic level (due to the absence of introns) and hence easier to sequence completely, they are probably substantially over-represented in the learning set relative to their true genomic frequency; accordingly, the proportion of single-exon genes in each group was estimated (somewhat arbitrarily) to be one half of the observed fraction. Codelen refers to the total number of coding base-pairs per gene. Data for subsets III and IV are estimated from the Duret *et al.* (1995) data for isochore H3 assuming that one-half of the genes and 60% of the amount of DNA sequence in isochore H3 falls into the 51 to 57% C + G range. Mean transcript lengths were estimated assuming an average of 769 bp of 5′UTR and 457 bp of 3′UTR per gene (these values derived from comparison of the "prim_transcript" and "CDS" features of the GenBank annotation in the genes of the learning set). To simplify the model, the initial probabilities of the exon, polyadenylation signal and promoter states are set to zero. All other initial probabilities are estimated from the data shown above, assuming that all features are equally likely to occur on either DNA strand. The initial probability for all intron states was partitioned among the three intron phases according to the observed fraction of each phase in the learning set. Transition probabilities were estimated analogously.

Figure 7.13: Gene density and structure as a function of C+G composition [3].

Initial state probabilities

The initial state probabilities should be proportional to the frequencies with which various functional units occur in the actual genomic data. For example, if the estimated proportion of the non-coding intergenic region is 80%, then initial probability for the state N (see Figure 7.12) must be around 0.8.

Transition probabilities

Transition probabilities are also known to vary quite a bit with the C+G content (although not as much as the initial probabilities). Thus, transition probabilities are also separately computed for each of the categories. Of course, while estimating these probabilities, it is ensured that the transitions are biologically permissible. For example, some transitions are obligatory (like $P^+ \rightarrow F^+$). Such transitions are assigned probability one.

State length distributions

Different functional units on a gene have vastly different lengths. For example, an average internal exon is about 150bp long, while introns of the order of 1Kbp length are not uncommon. Thus, in our probabilistic model of gene structure, different states need to have different length distributions.

Intron lengths are known to vary dramatically with the C+G content category. For example, the mean intron length for category I ($<$ 43% C+G) of the training set is 2069bp as opposed to only 518bp for category IV ($>$ 57 % C+G) (see Figure 7.13). Thus, the program uses separate distributions for intron states in each category.

The learning set shows quite different length distributions for initial exons, internal exons and terminal exons. Consequently, different distributions are used for them. It is important to note here that the length of an internal exon has to be consistent with the phase of its adjacent introns. For example, if the preceding state is I_2 and the succeeding state is I_1, then the generated internal exon length (for state E_2 in this case) must be $3n + 2$ for some n. n is therefore generated randomly according to the length distribution and then a string of length $3n + 2$ is generated according to the string generating model for that state.

For the 5' UTR and 3' UTR states, geometric distributions with mean values of 769bp and 457bp are used.

Signal models

GENSCAN uses different signal models to model different functional units. One of the models is weight matrix model (WMM) in which every position has its own specific independent distribution. It is used for modeling polyadenylation signals, translation initiation signal, translation termination signal and promoters. Another model is the the weighted array model (WAM). The WAM model is a generalization of the WMM model that allows dependencies between adjacent positions. The WAM model is used for the recognition of the splice sites ([7] p. 61). Correct recognition of these sites greatly enhances ability to predict correct exon boundaries. This modeling of splice sites gave GENESCAN a substantial improvement in performance.

7.3.8 Performance of GENSCAN

Performance of GENSCAN has been compared to that of other computer programs written for gene finding. The Burset/Guigo set of 570 vertebrate multi-exon gene sequences [4] was used as the test set. The results (as reported in [4]) have been quite encouraging. Both at the nucleotide level as well as the exon-level, GENSCAN's accuracy has been significantly better than other programs.

Important features of GENSCAN include:

- Identification of complete intron/exon structures of a gene in genomic DNA.
- Ability to predict multiple genes and to deal with partial as well as complete genes.
- Ability to predict consistent sets of genes occurring on either or both strands of the DNA.
- Ability to predict both optimal annotation and sub-optimal exons.

Although the results are good they are still not good enough for massive gene finding. GENSCAN has 80% chance for detecting an exon. If a gene has more than one exon the probability of correctly detecting all of them declines rapidly. On the positive side a more permissive figure of merit, the prediction per bp, is over 90%.

7.3.9 Spliced Alignment

Given a genomic sequence and a set of candidate exons, the spliced alignment algorithm [5] explores all possible exon assemblies and finds a chain of exons which best fits a related target protein. The set of candidate exons is constructed by considering all blocks between

Accuracy of GENSCAN for different signal and exon types.

(a) Prediction of individual splice sites and translational signals.

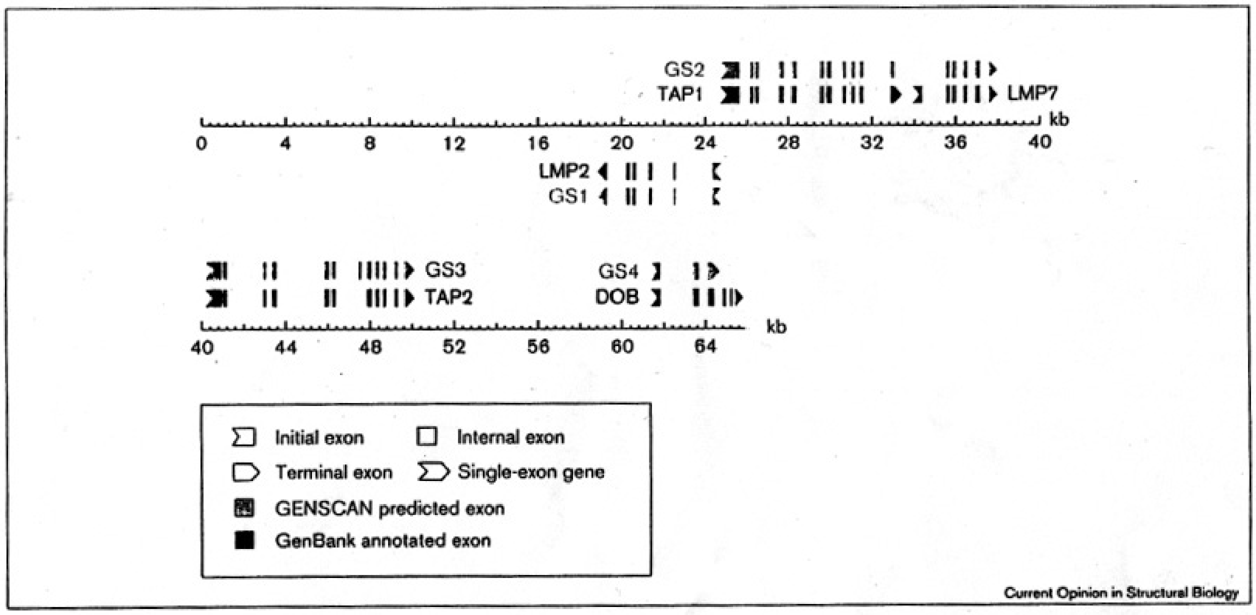
Type of signal	Type of exon	Annotated exons		Predicted exons	
		Number	% Correctly predicted	Number	% Correctly predicted
Initiation	Initial only	570	66	450	84
Termination	Terminal only	570	78	487	91
5′ splice site	Initial only	570	88	450	89
5′ splice site	Internal only	1510	93	1682	89
5′ splice site	Initial and internal	2080	91	2132	89
3′ splice site	Terminal only	570	81	487	92
3′ splice site	Internal only	1510	92	1682	83
3′ splice site	Internal and terminal	2080	89	2169	85

(b) Accuracy for initial, internal and terminal exons.

Exon type	Annotated exons				Predicted exons			
	Number	% Exactly	% Partially	% Missed	Number	% Exactly	% Partially	% Wrong
Initial	570	65	25	9	457	81	9	10
Internal	1510	90	5	4	1707	80	11	8
Terminal	570	76	8	15	509	84	6	8
All types	2650	81	10	8	2678	81	10	9

Accuracy statistics are shown for forward-strand exons predicted by the GENSCAN program [17••], as tested on the Burset and Guigo dataset of 570 vertebrate gene sequences [3••]. (a) Accuracy is shown for four types of signals: initiation codons, termination codons and 5′ and 3′ splice sites. For each signal type, the number of (true) sites according to the GenBank CDS (coding sequence) annotation and the percentage of sites predicted correctly by GENSCAN are shown in columns 3 and 4, respectively. Columns 5 and 6 show the number of sites predicted by GENSCAN and the percentage of predicted sites that were correct, respectively. For 5′ and 3′ splice sites, accuracy data are also shown separately for initial versus internal exons, and internal versus terminal exons, respectively. (b) Accuracy data at the exon level. The percentages of annotated exons that were predicted exactly (both endpoints correct), predicted partially (one endpoint correct) or missed (not overlapped by a predicted exon) are listed in columns 3, 4 and 5, respectively. Columns 7, 8 and 9 show the percentages of predicted exons that were exactly correct (both endpoints correct), partially correct (one endpoint correct), or wrong (not overlapping an annotated exon), respectively. In addition, five single-exon genes were predicted by GENSCAN in this set (not given a separate row, but included in the totals).

Figure 7.14: Accuracy of GENSCAN for different signal and exon types [3].



A schematic representation of predicted and annotated genes in a 66 kb portion of the human MHC II region (GenBank accession number X66401) is shown. Annotated genes are labeled according to the names given in the GenBank annotation; predicted genes are labeled GS1 through GS4 as they occur along the sequence. Genes coding on the forward and reverse DNA strands are shown above and below the sequence line, respectively. Predicted exons are shown in grey, annotated exons in black; the shape of the exon indicates its type, as shown in the key. Exon sizes and locations are drawn approximately to scale.

Figure 7.15: GENSCAN results [3].

candidate acceptor and donor sites (i.e., between AG dinucleotide at the intron-exon boundary and GU dinucleotide at exon-intron boundary) and further filtration of this set. To avoid losing true exons, the filtration procedure is designed to be very gentle, and the resulting set of blocks may contain a large number of false exons. Instead of trying to identify the correct exons by further pursuit of statistical methods, The algorithm considers all possible chains of candidate exons and finds a chain with the maximum global similarity to the target protein.

Definition of the spliced alignment problem

We start with the formal definitions and statement of the spliced alignment problem.

- Let $G = g_1 g_2 \dots g_n$ be a string of letters and let $B = g_i \dots g_j$ and $B' = g_{i'} \dots g_{j'}$ be substrings of G. We write $B < B'$ if $j < i'$, i.e. if B ends before B' begins.
- A sequence $\Gamma = \{B_1, \dots, B_k\}$ of substrings of G is a *chain* if $B_1 < B_2 < B_3 < \dots < B_k$.
- Let $B_1 * B_2$ denote the concatenation of B_1 and B_2. We denote the concatenation of all the strings from Γ by Γ^*.
- Given two strings G and T, let $S(G, T)$ be the score of the optimal global alignment between them.

Problem 7.1 Spliced alignment
Instance: A new genomic sequence $G = g_1 \dots g_n$. A target sequence (related protein) $T = t_1 \dots t_m$. A set $\mathcal{B} = B_1, \dots, B_L$ of blocks (substring of G).
Question: Find a chain Γ of blocks from $\mathcal{B}$ such that $S(\Gamma^*, T)$ is maximum.

Network formulation

The spliced alignment problem can be formulated in network terms. The set of blocks $\mathcal{B} = \{B_i\}_{i=1}^{L}$ is represented by a set of nodes $\{v_i\}_{i=1}^{L}$. A node v_i is connected to a node v_j if $B_i < B_j$. The requested solution is the best alignment between the reference sequence (T) and a path in the network.

Spliced alignment algorithm

To solve the problem, we use dynamic programming.

We start with a few notations:

- A *j-prefix* of block $B_k = g_i \dots g_j \dots g_l$ is $B_k(j) = g_i \dots g_j$.
- If B_k contains the position i, the *i-prefix* of $\Gamma^* = B_1 * ...B_k$ is $\Gamma^*(i) = B_1 * ...B_k(i)$.

- For block $B_k = g_i \dots g_j$ $\text{First}(B_k) = i$ and $\text{Last}(B_k) = j$.
- A Chain $\Gamma^* = B_1 * \dots * B_k$ *ends* at $\text{Last}(B_k)$, and it ends *before* position i, if $\text{Last}(B_k) < i$.
- $\mathcal{B}[i]$ is a set of blocks from $\mathcal{B}$ ending before i, i.e. if $B_k \in \mathcal{B}[i]$ than $\text{Last}(B_k) < i$.
- $S(i,j,k) = \max_{\Gamma \mid B_k \in \Gamma} S(\Gamma^*(i), T(j))$.

The recursive relation of $S(i,j,k)$ is :

$$S(i,j,k) = \max \begin{cases} S(i,j-1,k) + (-,t_j) & \text{if } i \geq \text{First}(B_k) \\ S(i-1,j-1,k) + (g_i,t_j) & \text{if } i > \text{First}(B_k) \\ S(i-1,j,k) + (g_i,-) & \text{if } i > \text{First}(B_k) \\ \max\limits_{B_l \in \mathcal{B}[i]} S(\text{Last}(B_l), j-1, l) + (g_i,t_j) & \text{if } i = \text{First}(B_k) \\ \max\limits_{B_l \in \mathcal{B}[i]} S(\text{Last}(B_l), j, l) + (g_i,-) & \text{if } i = \text{First}(B_k) \end{cases}$$

The Final score $\max\limits_{B_k \in \mathcal{B}} S(\text{Last}(B_k), m, k)$ can therefore be computed in time complexity of $O(nmL^2)$ and in space complexity of $O(nmL)$.

Improved spliced alignment algorithm

Define $P(i,j) = \max\limits_{B_l \in \mathcal{B}[i]} S(\text{Last}(B_l), j, l)$. The recursive relation for $S(i,j,k)$ can be presented as:

$$S(i,j,k) = \max \begin{cases} S(i,j-1,k) + (-,t_j) & \text{if } i \geq \text{First}(B_k) \\ S(i-1,j-1,k) + (g_i,t_j) & \text{if } i > \text{First}(B_k) \\ S(i-1,j,k) + (g_i,-) & \text{if } i > \text{First}(B_k) \\ P(i,j-1) + (g_i,t_j) & \text{if } i = \text{First}(B_k) \\ P(i,j) + (g_i,-) & \text{if } i = \text{First}(B_k) \end{cases}$$

The value of $P(i,j)$ can be achieved by an alignment of Γ ending <u>before</u> the $(i-1)$'th position ($=P(i-1,j)$) or exactly <u>at</u> the $(i-1)$'th position ($=\max\limits_k S(i,j,k)$). Therefore, the recursive relation for $P(i,j)$ is:

$$P(i,j) = \max\{P(i-1,j), \max_k S(i,j,k)\}$$

The space complexity remains $O(nmL)$, however the time complexity is now reduced to $O(nmL)$ (instead of $O(nmL^2)$).

Results

The number of exon assemblies is huge; However, the spliced alignment algorithm is fast enough to process large genomic fragments (up to 180,000 nucleotides) containing multi-exon genes (more than 30 exons). After the highest-scoring exon assembly is found, the hope is that it represents the correct exon-intron structure. This is almost guaranteed if a protein sufficiently similar to the one encoded in the analyzed fragment is available (99% correlation between predicted and actual genes with mammalian targets). This method is good only if you have the right protein to compare it with. It can identify up to 53% of all genes.

Bibliography

[1] Genetics computer group (CGC) user's guide, codonpreference(+). WWW:`http://dapsas.weizmann.ac.il/gcg_html/codonpreference.html`.

[2] The U.S. department of energy and the human genome project, introducing the human genom. WWW: `http://www.ornl.gov/hgmis/publicat/tko/03_introducing.html`.

[3] C.B. Burge and S. Karlin. Finding the genes in genomic DNA. *J. Mol. Bio*, 268:78–94, 1997.

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

[5] M.S. Gelfand, A.A. Mironov, and P.A. Pevzner. Gene recognition via spliced sequence alignmnet. In *Proc. Natl Acad Sci USA*, volume 93, pages 9061–9066, 1996.

[6] J. Witkowski J.D. Watson, M. Gilman and M. Zoller. *Recombinant DNA*. W.H. Freeman, New-York, 1992.

[7] Sumeet Sobti. Gene prediction, II - DRAFT. WWW: `http://www.cs.washington.edu/education/courses/590bi/98w/lect09.ps`, Feb 1998.