Probabilistic methods of identifying genes in prokaryotic genomes: Connections to the HMM theory

Rajeev K. Azad and Mark Borodovsky

Date received (in revised form): 9th February 2004

Rajeev K. Azad
is currently a postdoctoral fellow at School of Biology, Georgia Institute of Technology. His research interests include unravelling the complexity of genomes using statistical measures for DNA segmentation and identifying protein-coding genes in genomic sequences by application of Markov models and hidden Markov models.

Mark Borodovsky
is a Regents' Professor in Schools of Biology and Biomedical Engineering and Director of Center for Bioinformatics and Computational Biology, Georgia Institute of Technology. His research interests include modelling of cell system structural components and predicting structure and function by statistical pattern recognition, particularly, gene identification, protein secondary structure and function prediction by using Markov models.

Keywords: *prokaryotic gene prediction, Markov model, hidden Markov model, forward–backward algorithm*

Mark Borodovosky,
School of Biology and School of Biomedical Engineering,
Georgia Institute of Technology,
Atlanta, GA 30332–0230, USA

Tel: +1 404 894 8432
Fax: +1 404 894 0519
E-mail: mark@amber.
biology.gatech.edu

Abstract

In this paper, we review developments in probabilistic methods of gene recognition in prokaryotic genomes with the emphasis on connections to the general theory of hidden Markov models (HMM). We show that the Bayesian method implemented in GeneMark, a frequently used gene-finding tool, can be augmented and reintroduced as a rigorous forward-backward (FB) algorithm for *local* posterior decoding described in the HMM theory. Another earlier developed method, prokaryotic GeneMark.hmm, uses a modification of the Viterbi algorithm for HMM with duration to identify the most likely *global* path through hidden functional states given the DNA sequence. GeneMark and GeneMark.hmm programs are worth using in concert for analysing prokaryotic DNA sequences that arguably do not follow any exact mathematical model. The new extension of GeneMark using the FB algorithm was implemented in the software program GeneMark.fba. Given the DNA sequence, this program determines an *a posteriori* probability for each nucleotide to belong to coding or non-coding region. Also, for any open reading frame (ORF), it assigns a score defined as a probabilistic measure of all paths through hidden states that traverse the ORF as a coding region. The prediction accuracy of GeneMark.fba determined in our tests was compared favourably to the accuracy of the initial (standard) GeneMark program. Comparison to the prokaryotic GeneMark.hmm has also demonstrated a certain, yet species-specific, degree of improvement in raw gene detection, ie detection of correct reading frame (and stop codon). The accuracy of exact gene prediction, which is concerned about precise prediction of gene start (which in a prokaryotic genome unambiguously defines the reading frame and stop codon, thus, the whole protein product), still remains more accurate in GeneMarkS, which uses more elaborate HMM to specifically address this task.

INTRODUCTION

A primary goal of genome annotation project is to locate all protein-coding genes. Currently, this task is too expensive to be solved by experimental means. Therefore, the major tools of gene annotation are computational gene-finding programs with algorithms using both intrinsic (*ab initio*, statistical) and extrinsic (sequence similarity) measures.

The first step in developing an *ab initio* gene-finding algorithm is to perform statistical analysis of DNA sequences of interest (protein-coding and non-coding) and to identify statistical determinants, such as in-frame frequencies of oligonucleotides, that can help recognise sequences of these two types. The second step is to build statistical models, such as Markov models for all sequence categories, particularly gene models. The third step is to integrate the models into a pattern recognition algorithm.

Development of computational

Gene prediction can be done by extrinsic and intrinsic approaches

methods of gene prediction has a history of over two decades, with pioneering works published in 1980s.[1–7] These early approaches used statistical patterns of nucleotide ordering in a DNA sequence alone, and, thus, could be classified into the category of intrinsic or *ab initio* methods.

Inhomogeneous Markov chain models provide a mathematical framework to account for in-frame oligonucleotide statistics of protein-coding regions

Among various statistical determinants of protein-coding potential examined by Fickett and Tung[8] in-frame hexamer frequencies were shown to possess the highest predictive power. A mathematical framework of non-uniform (inhomogeneous) Markov chain models[9,10] had been suggested earlier to rigorously incorporate frame-dependent oligonucleotide frequencies into a Bayesian pattern recognition algorithm. This advance in modelling led to the development of the GeneMark algorithm and program.[11] Over the years, inhomogeneous Markov models have been found to be very efficient for gene modelling and have been used in several popular algorithms and programs for prokaryotic as well as eukaryotic gene prediction.

The extrinsic methods use conservation of protein-coding sequences in evolution. Many of them use search for similarity of a protein product of predicted gene to other protein sequences in a database, an accumulation of substantial amount of sequence data in databases was required before considerable work was eventually and successfully done in this direction (eg, see Robison *et al.*[12] and Frishman *et al.*[13]).

INHOMOGENEOUS MARKOV MODELS AND THE GENEMARK ALGORITHM

By incorporating a gene shadow model in the Bayesian formalism, GeneMark identifies genes on both strands using a DNA sequence of the strand only

The GeneMark algorithm integrates an inhomogeneous (three-periodic) Markov model for protein-coding DNA sequence and a homogeneous Markov model for non-coding sequence in a Bayesian formalism to calculate the posterior probability of a sequence segment to belong to the following categories (or states): coding region, reverse complement of coding region (coding region shadow), non-coding region.[11]

For a DNA sequence segment, $S = \{s_1, s_2, \ldots, s_n\}$ (n is a multiple of 3), the algorithm determines the *a posteriori* probabilities $P(\mathrm{cod} \mid S)$, $P(\mathrm{shadow} \mid S)$ and $P(\mathrm{non} \mid S)$ for S to belong to the coding, shadow and non-coding regions respectively. A protein-coding sequence is said to be situated in phase i, $I = 1, 2, 3$ if the first nucleotide of the sequence is located in the position i of a codon. Given a three-periodic Markov model of order m, the probability of sequence S to be generated by this model in *phase 1*, $P(S \mid \mathrm{cod}_1)$, is defined by the equation:

$$\begin{aligned} P(S|\mathrm{cod}_1) = {} & P^1(s_1^m) * P^1(s_{m+1}|s_1^m) \\ & * P^2(s_{m+2}|s_2^{m+1}) * P^3(s_{m+3}|s_3^{m+2}) \\ & * \ldots * P^t(s_n|s_{n-m}^{n-1}) \end{aligned} \tag{1}$$

Here m is the model order, s_l^k designates an oligonucleotide starting in position l and ending in position k, $P^i(s_1^m)$ is the initial probability of oligonucleotide s_1^m situated in phase i and $P^i(s_k \mid s_{k-m}^{k-1})$ is the transition probability of base s_k to follow the oligonucleotide s_{k-m}^{k-1} situated in phase i, $t = 2, 1, 3$ if $m \pmod 3 = 1, 2, 0$, respectively. Note that values of parameters of the Markov model, the initial and transition probabilities are estimated using maximum likelihood approach. Equations for $P(S \mid \mathrm{cod}_2)$ and $P(S \mid \mathrm{cod}_3)$ can be obtained from (1) by cyclic permutations of the superscripts. In a similar way, $P(S \mid \mathrm{shadow}_i)$ can be computed. The equation for $P(S \mid \mathrm{non})$ is similar to (1) and even simpler as it is omitting the phase consideration.

Finally, the *a posteriori* probability of the event of observing a protein-coding sequence in phase i, given sequence S, $P(\mathrm{cod_i} \mid S)$, $(I = 1, 2, 3)$ is determined by the Bayesian equation:

$$P(\mathrm{cod}_i|S) = \frac{P(S|\mathrm{cod}_i) \ast P(\mathrm{cod}_i)}{\sum_{j=1}^{3} P(S|\mathrm{cod}_j) \ast P(\mathrm{cod}_j) + \sum_{j=1}^{3} P(S|\mathrm{shadow}_j) \ast P(\mathrm{shadow}_j) + P(S|\mathrm{non}) \ast P(\mathrm{non})} \tag{2}$$

where $P(\mathrm{cod}_i)$, $P(\mathrm{shadow}_i)$ and $P(\mathrm{non})$ are *a priori* probabilities of observing sequence of each category respectively upon a 'random pick' of a segment of length n. Equations that define the *a posteriori* probabilities of observing a coding region shadow sequence (in three phases) given sequence S or observing a non-coding sequence are similar to (2).

The GeneMark program uses a sliding window technique to calculate 'profiles' of *a posteriori* probabilities for a sequence of any length, up to a whole prokaryotic genome.[11] The window defines a sequence segment (of a default length, 96 nt) in a DNA sequence. These *a posteriori* probability profiles identifying local coding property of DNA sequence have been also useful in detecting frame shifts that occur in experimentally determined nucleotide sequences due to sequencing errors or as a natural consequence.

Sliding window approach of the GeneMark programme also helps in frameshift detection

The score of an ORF in GeneMark is defined as the average value of the *a posteriori* probabilities (for being coding) of the sequences in the windows that fall inside the open reading frame (ORF) and have the same reading frame (phase). If the score exceeds an established threshold, the ORF is predicted as a gene. GeneMark has been successfully used in a number of genome sequencing projects (see, for example, Fleischmann *et al.*,[14] Bult *et al.*,[15] Blattner *et al.*,[16] Kunst *et al.*,[17] Tomb *et al.*[18]).

Parameters of Markov models can be derived by machine learning approaches using an anonymous genomic sequence

Probabilistic parameters of Markov models, in an ideal case, should be determined from experimentally validated sets of protein-coding and non-coding DNA sequences. However, for a largely anonymous prokaryotic genome with no sufficient number of experimentally confirmed genes, parameter estimation is still feasible by the unsupervised training procedure suggested by Audic andClaverie.[19] In their approach the Markov models were built from randomly partitioned genomic sequences. Then, after clustering into three groups, the three sequence models were defined and integrated in the GeneMark-like algorithm that classified sequence segments into one of the three categories. The new predictions were used to refine the clusters and models and this process was iterated to convergence (ie the current predictions classified all sequences from a current cluster into the same cluster). Eventually, these three clusters accumulated the three types of sequences, coding, coding shadow and non-coding, and parameters of the three models corresponding to these three categories were determined. It was reported that convergence was typically reached in less than 50 iterations and resulted in up to 90 per cent gene detection accuracy. Another machine-learning method for deriving models from genomic sequence without knowing experimentally validated genes, GeneMark-Genesis,[20] used the observation that ORFs longer than 1000 nt are predominantly protein-coding. The goal of that algorithm was to build several types of gene models such as 'typical' and 'atypical' with the notion that 'atypical' genes are possibly horizontally transferred genes. GeneMark-Genesis employed an iterative k-means clusterisation procedure with relative entropy (Kullback–Leibler distance) used as a distance function to monitor the convergence of the clustering procedure in the space of model parameters. A learning procedure using 'long' ORFs to derive gene models was also used in the Glimmer program.[21,22]

INTERPOLATED MARKOV MODELS

Markov models of several orders were combined in the 'interpolated' model for gene prediction in the Glimmer algorithm.[21,22] The parameters of the interpolated model were defined by the following equation:

Interpolated Markov models are obtained by combining higher and lower order models with appropriate weight factors

$$P^{\mathrm{IMM}}(b|c_k) = \lambda(c_k) \ ^* \ P(b|c_k) + [1 - \lambda(c_k)] \ ^* \ P^{\mathrm{IMM}}(b|c_k - 1) \quad (3)$$

Here, $P^{\mathrm{IMM}}(b \,|\, c_k)$ is the interpolated transition probability of a nucleotide b given the preceding k-mer c_k, $P(b \,|\, c_k)$ is the maximum likelihood estimate of the value of transition probability of a regular Markov chain model, and $\lambda(c_k)$ is the interpolation parameter corresponding to c_k, $0 \leqslant \lambda(c_k) \leqslant 1$.

Generally, if the frequency of oligonucleotide c_k in the training sequence is sufficiently high, the value of $\lambda(c_k)$ is close to 1; for oligonucleotides with very low frequency, the value $\lambda(c_k)$ is close to 0, and the interpolated probability $P^{\mathrm{IMM}}(b \,|\, c_k)$ gains most of its value from $P^{\mathrm{IMM}}(b \,|\, c_{k-1})$. The estimation of interpolation parameters $\lambda(c_k)$ is a non-trivial issue and several alternative approaches have been proposed earlier.[21,23]

HIDDEN MARKOV MODELS

The hidden Markov model theory used initially in speech recognition was adapted later to develop gene prediction algorithms

The theory of hidden Markov models (HMM) was developed in the early 1970s.[24–26] Around that time the HMM theory started to be applied to speech recognition problems[27–30] as it was realised that a speech process can be characterised by a stochastic Markov process and statistical algorithms could be used in a speech recognition to identify the sequence of spoken phonemes or words. The subsequent two decades saw a significant progress in applications of HMMs to speech recognition. For substantial reviews of these studies we refer to publications by Rabiner and Juang,[31] Rabiner[32] and Jelinek.[33] Following the success in speech recognition applications, the HMM theory was soon introduced for developing algorithms of pattern recognition in biological sequences (see Durbin *et al.*[34] and Krogh[35] for consistent text and review).

Maximum likelihood parse of a DNA sequence into protein-coding and non-coding regions can be done by the Viterbi algorithm

The application of HMM in gene-finding started with the pioneering work on the ECOPARSE program by Krogh *et al.*[36] Subsequently, a number of algorithms employing HMM were developed for gene identification in genomes of prokaryotes[37–40] as well as eukaryotes.[41–43]

An HMM introduces a state sequence $A = \{A_1, \ldots, A_n\}$, where A_i denotes the hidden states that 'emit' the observed (given) DNA sequence $S = \{S_1, \ldots, S_n\}$. For example, the hidden states can be protein-coding, protein-coding shadow and non-coding in a hidden state model shown in Figure 1(a). As transitions between hidden states and emissions of nucleotides are governed by probabilistic rules, one could think about the state sequence $A*$ that is most likely associated with the observed sequence S. Mathematically, $A*$ could be found by maximisation of the conditional probability $P(A \,|\, S)$ with respect to A. This task is solved by a dynamic programming algorithm called the Viterbi algorithm.[32,34]

The ECOPARSE program[36] and later the GeneHacker program[38] were using the classic HMM model, with each hidden state emitting one nucleotide triplet (codon) from the coding state and one nucleotide from non-coding state. The Viterbi algorithm implemented in ECOPARSE and GeneHacker found the maximum likelihood parse of a DNA sequence given the sequence and the model. In a later development of another HMM-based algorithm, GeneMark.hmm,[37] the inhomogeneous Markov models used in GeneMark were explicitly utilised. However, that required a different type of HMM called HMM with duration. In Figure 1(b) we show the architecture of an HMM with duration, which implies that a DNA sequence consists of non-overlapping coding,

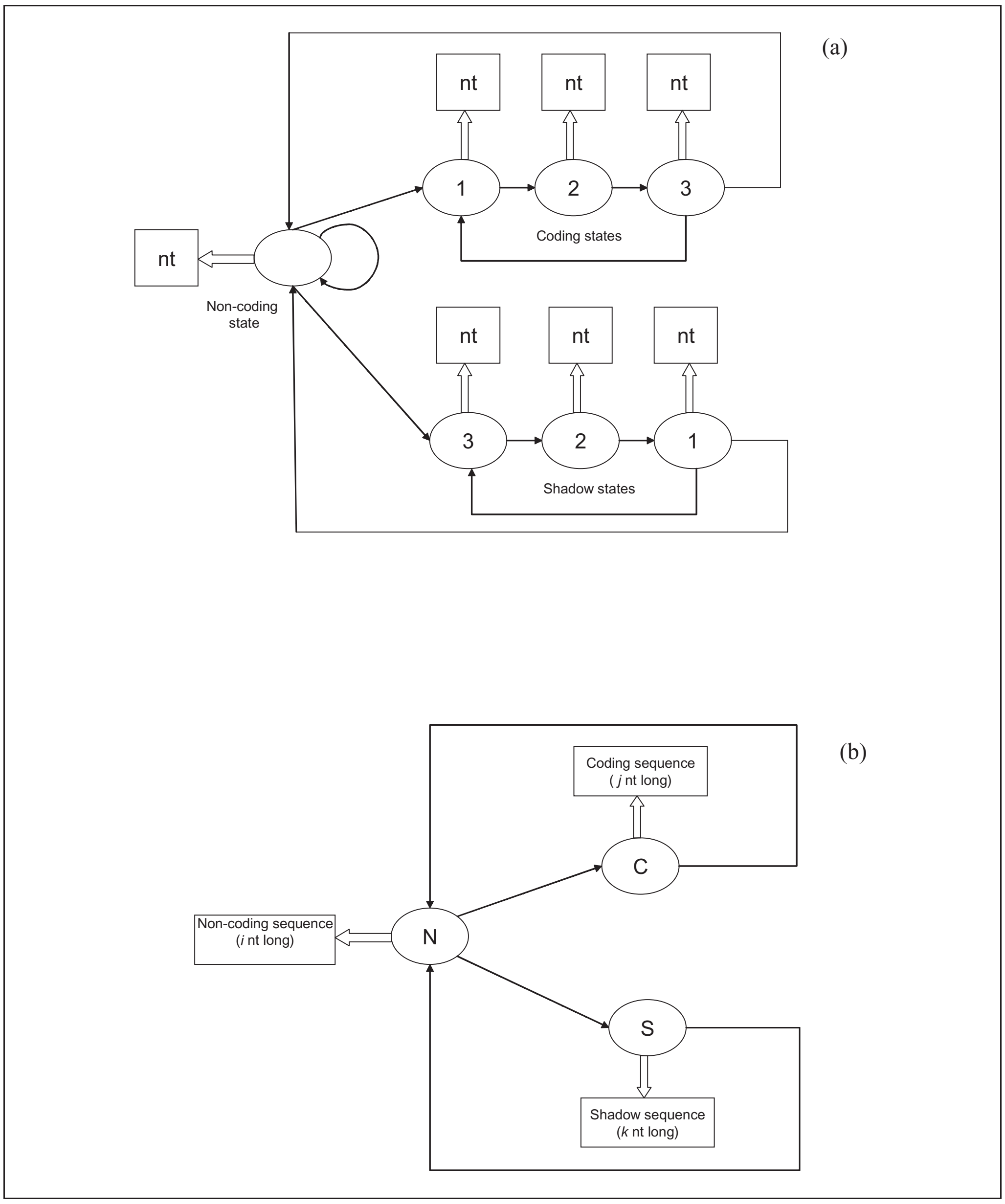

Figure 1: The architecture of (a) standard HMM and (b) generalised HMM. The standard HMM has three hidden states each for coding sub-model as well as shadow sub-model and one hidden state for the non-coding sub-model. Each state emits a nucleotide and then transition occurs to the next state. The generalised HMM has three hidden states corresponding to the three sub-models. Each state emits a string of nucleotides and then transition to another state occurs. The allowed transitions are shown by line arrows and the emissions by block arrows

shadow and non-coding regions. Each region corresponds to single coding, shadow or non-coding hidden state. Thus each state emits a string of nucleotides (coding or non-coding) and then transition occurs to a different hidden state (the allowed transitions are shown by line arrows and the emissions by block arrows). Advantages of HMM with duration include easiness of using any predefined length distribution of coding and non-coding fragments as well as opportunity to readily utilise inhomogeneous Markov models for different types of genes (typical and atypical). To improve prediction accuracy of the position of gene start, several algorithms including ECOPARSE, EasyGene, Glimmer, GeneMark.hmm and GeneHacker Plus used ribosomal binding site (RBS) models.

Use of ribosomal binding site models improves gene start prediction

An iterative procedure, GeneMarkS,[44] was developed to derive parameters for GeneMark.hmm by unsupervised learning. This procedure is using models with heuristically defined pseudo-counts in the initial step of the DNA sequence parsing into coding and non-coding regions.

With the fast growth of DNA and protein sequence databases, the sequence similarity is increasingly used to identify genes

PROKARYOTIC GENE PREDICTION USING SIMILARITY SEARCH

Several powerful gene-finding programs using extrinsic information have been developed in recent years. For a given genome the ORPHEUS program[13] selects the ORFs whose protein translations exhibit significant similarity to known proteins. DNA sequences of these ORFs are used as training data for deriving statistical models of protein coding sequence. In the ORPHEUS gene prediction algorithm the coding potential of a DNA sequence is measured by a quantity $\Omega = R(a_1 a_2 \ldots a_n) - \max\{R(b_1 b_2 \ldots b_n), R(c_1 c_2 \ldots c_n)\}$, where $R(a_1 a_2 \ldots a_n)$ is the normalised form (in standard deviation units) of the quantity $\sum_{i=1}^{n} \log f(a_i)$.

Here, $(a_1 \ldots a_n)$, $(b_1 \ldots b_n)$, $(c_1 \ldots c_n)$ denote the sequences of codons appearing in three possible frames and $f(a_i)$ is the frequency of the a_i codon.

The CRITICA algorithm[45] computes a coding score for each codon using database search and combines this score with a dicodon frequency score that depends on the log likelihood function $\ln[f_{\text{coding}}(a_i|a_{i-1})/f_{\text{non-coding}}(a_i|a_{i-1})]$. Here $f(a_i\,|a_{i\,-\,1})$ denotes the frequency of codon $a_{i,}$ given the preceding codon $a_{i\,-\,1}$. CRITICA uses an iterative procedure for learning dicodon usage statistics and thus belongs to the class of self-training algorithms as well.

Bio-Dictionary, a database of patterns describing a sequence space of proteins, was developed by Shibuya and Rigoutsos.[46] The patterns called 'seqlets' were derived by processing database of proteins using Teiresias algorithm.[47] When used for gene finding, the seqlets are matched against amino acid translation of an ORF. If the number of matching seqlets significantly exceeds a predetermined threshold, the ORF is predicted as a gene.

While the accuracy of detection of a prokaryotic gene location (with its 3′-end) is close to the theoretical limit the exact prediction of a gene start has greater room for improvement

The most recent prokaryotic gene finder, EasyGene,[40] uses a database similarity search information similar to one in ORPHEUS machine-learning procedure for deriving the model's parameters. EasyGene uses a non-looped HMM architecture to assign to each ORF a log-odds score. For instance, for a sequence S containing an ORF, the score $W = \log[P(S\,|\,M)/P(S\,|\,N)]$, where $P(S\,|\,M)$ and $P(S\,|\,N)$ are the posterior probabilities of S being generated by coding hidden Markov model M and null hidden Markov model N, respectively. EasyGene also implements a procedure to compute a statistical significance of the score of a predicted gene, a unique feature among gene prediction programs.

ACCURACY ASSESSMENT

The performance of a gene-finding algorithm is usually assessed by two accuracy parameters: sensitivity (Sn) and specificity (Sp). Sensitivity is defined as the percentage of real genes that were correctly identified by the algorithm in a test set. Specificity is defined as the percentage of predicted genes that match the real genes. The average value of Sn and Sp can also be used as a single accuracy parameter.

The accuracy assessments reported in recent papers on prokaryotic gene-finding indicate high level of sensitivity. Most of the algorithms have attained sensitivity above 90 per cent. On the tests done up to date, EasyGene, GeneHacker Plus, GeneMarkS and Glimmer performed comparably in detecting the genes in terms of sensitivity.[39,41,44] However, a high-quality prediction algorithm is supposed to minimise the number of false positive predictions, therefore specificity is equally important. In this respect, the predictions of EasyGene, GeneHacker Plus and GeneMarkS have been reported to have higher specificity than other programs.[39,40,44] For exact gene prediction (with correctly identified gene start), on test sets comprising 195 experimentally verified genes from the *Escherichia coli* genome,[48] EasyGene and GeneMarkS have shown a high performance with above 93 per cent of precise predictions.[40]

POSTERIOR DECODING IN GENEMARK AN APPROXIMATION OF THE FORWARD–BACKWARD ALGORITHM

In this and subsequent sections we will draw more parallels between the two major algorithms of the HMM theory, the FB algorithm and the Viterbi algorithm, and two gene-finding algorithms, GeneMark and GeneMark.hmm. These parallels are, however, not entirely complete unless we develop full extension of GeneMark to the FB algorithm. Indeed, even though fairly successful in detecting genes in prokaryotic genomes, the GeneMark algorithm has heuristic elements that deviate from a rigorous theory. For example, a whole sequence segment within the window is assigned to a single functional category, though the function in reality can vary inside the sequence segment (as a gene boundary may fall inside). Averaging *a posteriori* probabilities to obtain the score for an ORF is a heuristic solution as well. A rigorous assignment of an *a posteriori* probability of a functional state to each nucleotide in a DNA sequence could be provided by the FB algorithm.[32] Also the FB algorithm can be used to score ORFs in prokaryotic genomes. In what follows we reintroduce GeneMark by extending its procedure to full FB algorithm and show that this extension leads to improvement in gene prediction accuracy.

There is a parallel between the Bayesian methodology used in the GeneMark program and the forward-backward procedure of the HMM theory

While GeneMark determines the *a posterior* probability of the state of a short sequence segment, the forward-backward algorithm determines the *a posterior* probability of a state of a single nucleotide at a given position

Predicting the state of a nucleotide with the FB algorithm

Let us determine forward and backward variables $\alpha_t(i)$ and $\beta_t(i)$ as follows: $\alpha_t(i)$ is the probability of nucleotide S_t being associated with hidden state *i* given the observation of nucleotide sequence $S_1,\ldots,S_t$ and the model λ; $\beta_t(i)$ is the probability of nucleotide S_t being associated with hidden state *i* given the observation of nucleotide sequence $S_{t+1},\ldots,S_n$ (from $t+1$ to the end position) and the model λ. For a DNA sequence *S* and a model λ, the variables $\alpha_t(i)$ and $\beta_t(i)$ can be computed by the FB algorithm (for details, we refer to Rabiner[32] and Durbin *et al.*[34]).

$$\alpha_t(i) = \left(\sum_j \alpha_{t-1}(j)^* T_{j.i}\right)^* P_i(S_t) \tag{4}$$

$$\beta_t(i) = \sum_j T_{i,j}{}^* P_j(S_{t+1})^* \beta_{t+1}(j) \tag{5}$$

Here $T_{i,j}$ is the transition probability from hidden state *i* to state *j*. $P_i(S_t)$ is the probability of emitting the nucleotide S_t from state *i*.

The probability of a nucleotide S_t to be in state *i*, given model λ and the observation of sequence *S*, can be now defined by the equation:

$$\gamma_t(i) = \frac{\alpha_t(i)^* \beta_t(i)}{P(S)} \tag{6}$$

Here *P*(*S*), the probability of the sequence *S* given the model λ, can be computed using the forward step of the FB algorithm.[32]

The HMM architecture shown in Figure 1(a) was used to calculate the posterior probabilities $\gamma_t(i)$ of hidden states in a given position of DNA sequence. This HMM architecture contains three hidden states (corresponding to each phase) for the protein-coding sub-model as well as for protein-coding shadow sub-model, and a single hidden state for non-coding sub-model. Each state emits a nucleotide and then transition to next state occurs. The allowed transitions are indicated by the arrows in Figure 1(a). The overlap of coding sequences located either in the same strand (in different frames) or in the opposite strands is not allowed. Given the three Markov models derived for coding and coding shadow and non-coding sequences (the second order models were used in this implementation), $\gamma_t(i)$ was calculated for three phase-dependent states of protein-coding sub-model and three phase-dependent states of protein-coding shadow sub-model as well as for the non-coding state.

In Figure 2, we show the plot of posterior probabilities of six coding states (in six possible coding frames) for the *E. coli* genomic sequence containing genes *nhaR, insB1, insA1* and *rpsT*. For comparison, the plot of posterior probabilities produced for the same sequence by the GeneMark program is shown in Figure 3. As it can be seen in Figures 2 and 3, both algorithms identify the same coding regions, though larger fluctuations in posterior probability values are observed in the GeneMark output.

Determining scores for predicted genes using the FB algorithm

The forward-backward algorithm can be used to measure the coding potential of an ORF

The FB algorithm was applied previously for computing probabilistic scores for exons predicted in eukaryotic genomic sequences.[41] A similar approach can be used to score ORFs in prokaryotic genomes, except that this goal can be reached by using a standard HMM with architecture shown in Figure 1(a). To quantify the coding potential associated with a subsequence, $S_{t-d+1}, \ldots, S_t$, of length d, the standard HMM (Figure 1a) can be employed to get the quantity:

$$\begin{aligned} Q = \sum_{\substack{i=\mathrm{cod}_3,\mathrm{non}\\ j=\mathrm{non},\mathrm{cod}_1}} & \alpha_{t-d}(i) * T_{i,\mathrm{cod}_1} \\ & * P_{\mathrm{cod}_1}(S_{t-d+1}) * T_{\mathrm{cod}_1,\mathrm{cod}_2} \\ & * P_{\mathrm{cod}_2}(S_{t-d+2}) * T_{\mathrm{cod}_2,\mathrm{cod}_3} \\ & * P_{\mathrm{cod}_3}(S_{t-d+3}) * \ldots * T_{\mathrm{cid}_2,\mathrm{cod}_3} \\ & * P_{\mathrm{cod}_3}(S_t) * T_{\mathrm{cod}_3,j} * \beta_t^*(j) \qquad (7) \end{aligned}$$

where $\beta_t^*(j) = \beta_{t+1}(j) * P_j(S_{t+1})$. Here, Q defines the sum of probabilities of all possible parses of the sequence S having a coding subsequence from position $t-d+1$ to t. cod$_i$ denotes the ith state (corresponding to ith phase, $I = 1, 2, 3$) of a codon sub-model; non stands for the non-coding state. The forward variable α and backward variable β can be determined using equations (4) and (5).

To get a coding measure for an ORF,

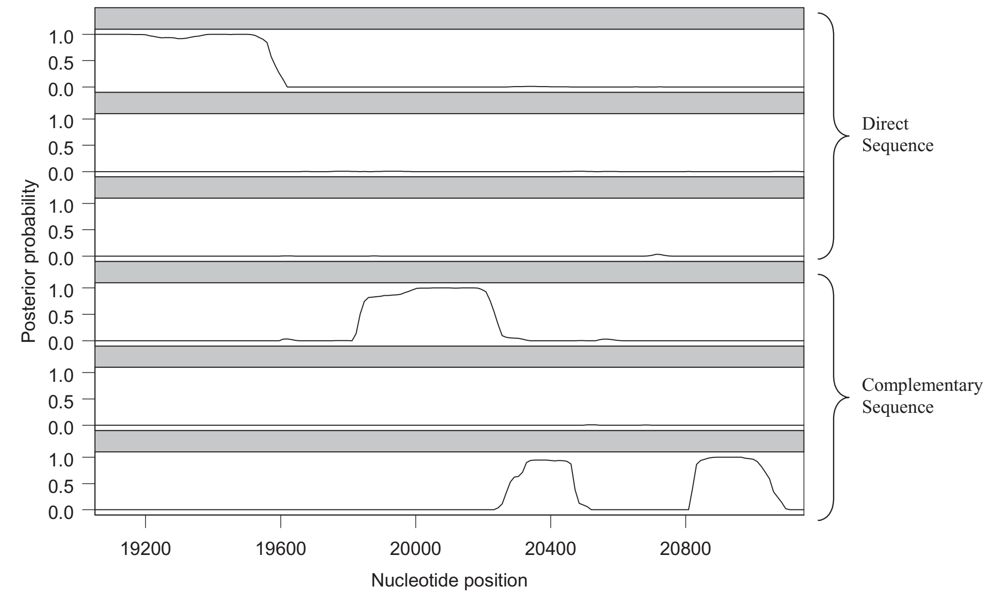

Figure 2: The posterior probability of coding hidden states as determined by the GeneMark.fba program implementing the FB algorithm. The probability values are shown for the six reading frames of the *E. coli* DNA sequence. Each 12th data point of the actual FB algorithm output was used for plotting. In the GenBank annotation there is one annotated gene *nhaR* (18715–19620) in the direct strand and three annotated genes *insB1, insA1, rpsT* (19811–20314, 20233–20508 and 20815–21078) in the complementary strand

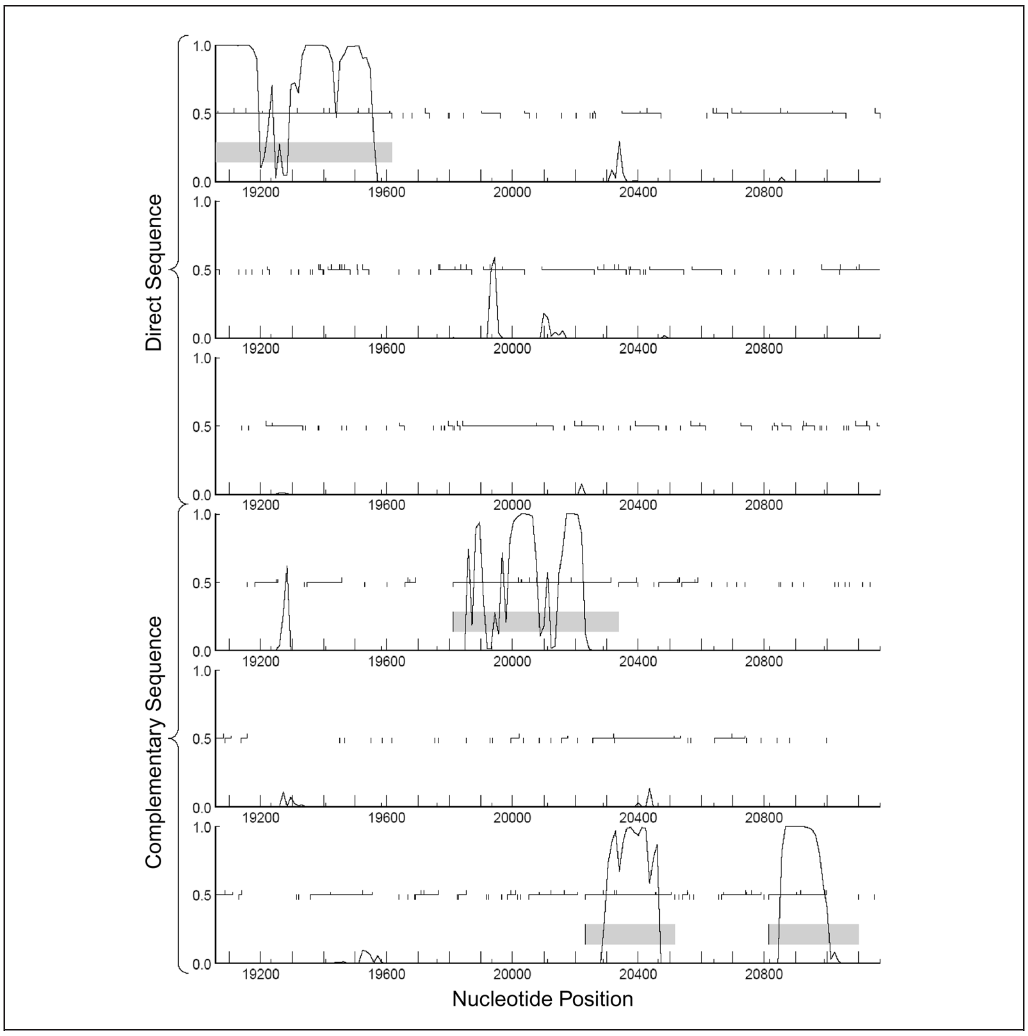

Figure 3: The posterior probabilities of a protein-coding property (protein-coding potential) as computed by GeneMark in six reading frames for the same *E. coli* DNA sequence as in Figure 2. A window of 96 nt moves at a step of 12 nt and the posterior probability is plotted at sequence position corresponding to the centre of the window

we compute Q for the subsequence $S_{t-d+1}, \ldots, S_t$ corresponding to the ORF (without stop codon) and also for all possible sequence segments of the ORF (with minimum length of 3), having the same phase as the ORF. Thus we take into account coding potential associated with the ORF as well as with its in-phase segments. We sum the Qs thus obtained to get the coding potential measure, Q_T, for the ORF in question. The ORFs are scored using one strand only approach;[11] the score Q for the ORFs in the complementary strand is computed by replacing coding state by shadow state in (7). This procedure gives good practical results, though it can be improved further. If Q_T is greater than a threshold, then the ORF is predicted as a gene. The threshold is set at $0.75 * P_0$. $P_0$is the probability of the test set sequence (including the ORF), given the model.

Accuracy tests require cross-validation approach

We tested the performance of Q_T scores computed by the GeneMark.fba algorithm for gene prediction in complete genomes of *Archaeoglobus fulgidus, Bacillus subtilis, E. coli, Haemophilus influenzae, Helicobacter pylori, Methanococcus jannaschii, Methanobacterium thermautotrophicum* and *Synechocystis*. The genomic sequences and annotations were downloaded from the NCBI database.[49]

For these tests, we used a six-fold cross-validation procedure: the models were built from the training set of sequences with genes annotated as given in GenBank and then the program was used to predict genes in the test set non-overlapping with the training set. Here we assumed that the GeneBank annotation is accurate. This assumption is not entirely valid, but it was sufficient for our purposes.

The genes were identified by the GeneMark.fba algorithm as follows. Since for a given stop codon, there can be more than one upstream in-frame start codon, the region where a gene can be residing is bounded by the stop codon and the furthest in-frame start codon (ie the longest ORF between two in frame stop codons). We measured the Q_T corresponding to this region and if the value Q_T exceeded the established threshold, the ORF identified by its stop codon was predicted as gene.

Table 1 shows the results of gene prediction accuracy assessment. The performance (in terms of average value of Sn and Sp) of GeneMark.fba was better than GeneMark for all the genomes shown in Table 1 except ones of *M. thermautotrophicum* and *M. jannaschii*. GeneMark.fba performed better than GeneMarkS for the genome of *E. coli*, *M. jannaschii* and *Synechocystis,* while GeneMarkS performed better for *A. fulgidus*, *B. subtilis* and *H. influenzae* genomes; both algorithms performed equally well for *H. pylori* and *M. thermautotrophicum* genomes. Note that GeneMarkS uses a different training procedure and has a more developed HMM architecture including besides models of protein-coding sequence and non-coding sequence additional two-component models of RBS to identify gene starts. One of our objectives was to draw a parallel between the Bayesian methodology used in GeneMark program and the HMM theory as implemented in GeneMark.fba program. Comparison of performance of GeneMark.fba with GeneMarkS is of interest since we compare two mainstream types of HMM algorithms for hidden state prediction: the forward–backward algorithm and the Viterbi algoriothms (though for HMM with a bit different architecture). The comparison shows that GeneMark.fba performs comparably with GeneMarkS

The forward-backward algorithm implemented in the GeneMark.fba program improves the gene prediction accuracy compared to the GeneMark program

Table 1: Comparison of the gene prediction accuracy of the GeneMark.fba program employing forward–backward algorithm with prediction accuracy of GeneMark and GeneMarkS programs

Organism	Prediction algorithm	Genes annotated	Genes predicted	Annotated genes detected	Sn (%)	Sp (%)	(Sn+Sp)/2 (%)
A. fulgidus	GeneMark	2406	2389	2260	93.93	**94.60**	94.26
	GeneMarkS		2462	2309	95.96	93.78	**94.87**
	GeneMark.fba		2545	2341	**97.29**	91.98	94.64
B. subtilis	GeneMark	4105	3759	3651	88.94	**97.12**	93.03
	GeneMarkS		4217	3968	**96.66**	94.09	**95.37**
	GeneMark.fba		4294	3939	95.95	93.91	94.93
E. coli	GeneMark	4255	3734	3655	85.89	**97.88**	91.89
	GeneMarkS		4069	3919	92.10	96.31	94.20
	GeneMark.fba		4153	3993	**93.84**	96.14	**94.99**
H. influenzae	GeneMark	1687	1712	1631	96.68	**95.26**	95.97
	GeneMarkS		1762	1657	**98.22**	94.04	**96.13**
	GeneMark.fba		1766	1656	98.16	93.77	95.96
H. pylori	GeneMark	1572	1514	1461	92.93	**96.49**	94.71
	GeneMarkS		1593	1510	**96.05**	94.78	**95.42**
	GeneMark.fba		1590	1509	95.99	94.90	**95.44**
M. jannaschii	GeneMark	1723	1750	1688	97.96	**96.45**	**97.21**
	GeneMarkS		1834	1709	**99.18**	93.18	96.18
	GeneMark.fba		1821	1709	**99.18**	93.84	96.51
M. thermautotrophicum	GeneMark	1872	1802	1762	94.12	**97.78**	**95.95**
	GeneMarkS		1825	1767	94.39	96.82	95.60
	GeneMark.fba		1909	1807	**96.52**	94.65	95.59
Synechocystis	GeneMark	3166	2942	2891	91.31	**98.26**	94.79
	GeneMarkS		3005	2922	92.29	97.23	94.76
	GeneMark.fba		3143	3048	**96.27**	96.97	**96.62**

(Table 1). Additionally, for the genome of *H. pylori* we have compared performance of the GeneMark.fba program with performances of the EasyGene and Glimmer 2.0 program. The Sn and Sp values obtained for EasyGene program were 94.52 and 97.25 per cent respectively. For the Glimmer 2.0 program, Sn and Sp values were 96.6 and 84.3 per cent respectively. The Sn and Sp values observed for the GeneMark.fba program (95.99 and 94.90 per cent) were comparable to EasyGene and Glimmer 2.0.

In Table 2, we show the results of the programs testing on three sets of annotated *B. subtilis* genes shorter than 300 nt with at least one, at least two and at least ten significant similarities to known proteins determined by BLAST analysis (Besemer *et al.*;[44] the data are available at the website.[50] GeneMarkS performed the best on all three sets, consistent with its best performance shown on the set of annotated *B. subtilis* genes (see Table 1).

The GeneMark.fba program performs comparably with other existing HMM based prediction programs

On a set of experimentally verified set of 850 genes of *E. coli* genome (see Rudd,[51] the data are available at the website[52]), GeneMarkS and GeneMark.fba performed equally well, identifying 842 genes. The GeneMark program identified 825 genes. The computational time of GeneMark.fba program is, however, greater than ones of GeneMark and GeneMarkS programs. For example, for the *E. coli* genome (4.6 Mbp), GeneMark.fba computations take about one hour.

Table 2: Comparison of gene prediction accuracy of GeneMark, GeneMarkS and GeneMark.fba programs tested on three sets of *B. subtilis* genes shorter than 300 nt with at least one (Set 1), at least two (Set 2), at least ten (Set 3) significant sequence similarities to known proteins as determined by BLAST analysis (Besemer *et al.*[44]). The number of genes identified by each program in each of the three test sets is shown

Test set	GeneMark 4th order	GeneMarkS 2nd order	GeneMark.fba 2nd order
Set 1 (123 genes)	84	113	99
Set 2 (72 genes)	54	68	61
Set 3 (51 genes)	39	48	42

A comparison of the Bayesian method implemented in GeneMark with the FB algorithm implemented in GeneMark.fba shows that GeneMark provides a computationally efficient procedure to generate *a posteriori* probability values that are approximations to *a posteriori* probabilities determined by a rigorous HMM method implemented in GeneMark.fba. For this reason, we believe the GeneMark method was characterised as 'HMM-like' one by Durbin *et al.*[34] (page 75) although GeneMark did not explicitly used an HMM approach. The ORF scores determined by the FB algorithm as implemented in GeneMark.fba improved the gene prediction accuracy in several genomes as compared to GeneMark. The extra computational time required for GeneMark.fba program is currently within reasonable limits and should further decrease as the speed of computers increases (not mentioning room for the code optimisation).

CONCLUSION

Statistical pattern recognition methods have achieved a high level of accuracy in prokaryotic gene finding. A number of algorithms using Markov models or hidden Markov models for gene identification have emerged and have been improved over the years. The latest algorithms have been reporting above 90 per cent sensitivity and specificity values in gene detection accuracy. As the perfect method that would correctly identify all protein-coding genes in a genome without false positives has not been invented yet, there is a point to try to deal with this challenge. However, this problem may not be precisely defined unless we have experimental evidence for all the genes in the genome. Absence of this information creates an uncertainty in the exact level of the performance of the best prokaryotic gene finders. With this caveat we believe that there is still a need to introduce new methods or improve existing methods to raise both the sensitivity and specificity bars firmly above 95 per cent level. The accuracy of prediction of gene starts in prokaryotic genomes is yet to be improved as well, though such programs as GeneMarkS and EasyGene have already raised the level of gene start prediction accuracy to 90 per cent level. As the amount of DNA sequences in the databases grows further, in concert use of extrinsic and intrinsic methods and further innovations should bring us in a rather short term to solving the prokaryotic gene-finding problem in a practical sense, though reaching perfect 100 per cent limit (with no false positives) would require indefinite time.

Acknowledgments
We thank Alexandre Lomsadze and John Besemer for useful discussions. RA and MB were supported in part by grants to MB from the US National Institutes of Health and the US Department of Energy.

References

1. Fickett, J. W. (1982), 'Recognition of protein coding regions in DNA sequences', *Nucleic Acids Res.*, Vol. 10, pp. 5303–5318.
2. Staden, R. (1984), 'Measurements of the effects that coding for a protein has on a DNA sequence and their use for finding genes', *Nucleic Acids Res.*, Vol. 12, pp. 551–567.
3. Gribskov, M., Devereux, J. and Burgess, R. R. (1984), 'The codon preference plot: Graphic analysis of protein coding sequences and prediction of gene expression', *Nucleic Acids Res.*, Vol. 12, pp. 539–549.
4. Almagor, H. (1985), 'Nucleotide distribution and the recognition of coding regions in DNA sequences: An information theory approach', *J. Theor. Biol.*, Vol. 117, pp. 127–136.
5. Claverie, J. M. and Bougueleret, L. (1986), 'Heuristic informational analysis of sequences', *Nucleic Acids Res.*, Vol. 14, pp. 179–196.
6. Borodovsky, M., Sprizhitsky, Yu, A., Golovanov, E. I. and Alexandrov, A. A. (1986c), 'Statistical patterns in the primary structures of functional regions of the genome in *Escherichia coli*: III. Computer recognition of coding regions', *Mol. Biol.*, Vol. 20, pp. 1144–1150.
7. Fichant, G. and Gautier, C. (1987), 'Statistical method for predicting protein coding regions in nucleic acid sequences', *Comput. Appl. Biosci.*, Vol. 3, pp. 287–295.
8. Fickett, J. W. and Tung, C. S. (1992), 'Assessment of protein coding measures', *Nucleic Acids Res.*, Vol. 20, pp. 6441–6450.
9. Borodovsky, M., Sprizhitsky, Yu, A. *et al.* (1986), 'Statistical patterns in the primary structures of functional regions of the genome in *Escherichia coli*: II. Nonuniform Markov models', *Mol. Biol.*, Vol. 20, pp. 833–840.
10. Tavare, S. and Song, B. (1989), 'Codon preference and primary sequence structure in protein-coding regions', *Bull. Math. Biol.,* Vol. 51, pp. 95–115.
11. Borodovsky, M. and McIninch, J. (1993), 'GeneMark: Parallel gene recognition for both DNA strands', *Computers Chem.*, Vol. 17, pp. 123–133.
12. Robison, K., Gilbert, W. and Church, G. M. (1994), 'Large scale bacterial gene discovery by similarity search', *Nature Genet.*, Vol. 7, pp. 205–214.
13. Frishman, D., Mironov, A., Mewes, H.-W. and Gelfand, M. (1998), 'Combining diverse evidence for gene recognition in completely sequenced bacterial genomes', *Nucleic Acids Res.*, Vol. 26, pp. 2941–2947.
14. Fleischmann, R. D., Adams, M. D. and White, O. (1995), 'Whole-genome random sequencing and assembly of *Haemophilus influenzae* Rd', *Science*, Vol. 269, pp. 496–512.
15. Bult, C. J., White O., Olsen G. J. *et al.* (1996), 'Complete genome sequence of the methanogenic archaeon, *Methanococcus jannaschii*', *Science*, Vol. 273, pp. 1058–1073.
16. Blattner, F. R., Plunkett, G. and Bloch, C. A. (1997), 'The complete genome sequence of *Escherichia coli* K-12', *Science*, Vol. 277, pp. 1453–1474.
17. Kunst, F., Ogasawara, N., Moszer, I. *et al.* (1997), 'The complete genome sequence of the gram-positive bacterium *Bacillus subtilis*', *Nature*, Vol. 390, pp. 249–256.
18. Tomb, J.-F., White, O. and Kerlavage, A. R. (1997), 'The complete genome sequence of the gastric pathogen *Helicobacter pylori*', *Nature*, Vol. 388, pp. 539–547.
19. Audic, S. and Claverie, J. M. (1998), 'Self-identification of protein-coding regions in microbial genomes', *Proc. Natl Acad. Sci. USA*, Vol. 95, pp. 10026–10031.
20. Hayes, W. S. and Borodovsky, M. (1998), 'How to interpret an anonymous bacterial genome: Machine learning approach to gene identification', *Genome Res.*, Vol. 8, pp. 1154–1171.
21. Salzberg, S. L., Delcher, A. L., Kasif, S. and White, O. (1998), 'Microbial gene identification using interpolated Markov models', *Nucleic Acids Res.*, Vol. 26, pp. 544–548.
22. Delcher, A. L., Harmon, D., Kasif, S. *et al.* (1999), 'Improved microbial gene identification with GLIMMER', *Nucleic Acids Res.*, Vol. 27, pp. 4636–4641.

23. Potamianos, G. and Jelinek, F. (1998), 'A study of N-gram and decision tree letter language modeling methods', *Speech Comm.*, Vol. 24, pp. 171–192.

24. Baum, L. E. and Petrie, T. (1966), 'Statistical inference for probabilistic functions of finite state Markov chains', *Ann. Math. Stat.*, Vol. 37, pp. 1554–1563.

25. Baum, L. E., Petrie, T., Soules, G. and Weiss, N. (1970), 'A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains', *Ann. Math. Stat.*, Vol. 41, pp. 164–171.

26. Baum, L. E. (1972), 'An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes', *Inequalities*, Vol. 3, pp. 1–8.

27. Baker, J. K. (1975), 'The dragon system – An overview', *IEEE Trans. Acoust. Speech Signal Processing*, Vol. ASSP-23, pp. 24–29.

28. Bahl, L. R. and Jelinek, F. (1975), 'Decoding for channels with insertions, deletions, and substitutions with applications to speech recognition', *IEEE Trans Informat. Theory*, Vol. IT-21, pp. 404–411.

29. Jelinek, F. (1976), 'Continuous speech recognition by statistical methods', *Proc. IEEE*, Vol. 64, pp. 532–536.

30. Bahl, L. R., Jelinek, F. and Mercer, R. L. (1983), 'A maximum likelihood approach to continuous speech recognition', *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 5, pp. 179–190.

31. Rabiner, L. R. and Juang, B. H. (1986), 'An introduction to hidden Markov models', *IEEE ASSP Mag.*, Vol. 3, pp. 4–16.

32. Rabiner, L. (1989), 'A tutorial on hidden Markov models and selected applications in speech recognition', *Proc. IEEE*, Vol. 77, pp. 257–286.

33. Jelinek, F. (1997), 'Statistical Methods for Speech Recognition', MIT Press, Cambridge, MA.

34. Durbin, R., Eddy, S.R., Krogh, A. and Mitchison G. (1998), 'Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids', Cambridge University Press, Cambridge.

35. Krogh, A. (1998), 'An introduction to hidden Markov models for biological sequences', in Salzberg, S., Searls, D. and Kasif, S., Eds, 'Computational Methods in Molecular Biology', Elsevier, Amsterdam.

36. Krogh, A., Mian, I. S. and Haussler, D. (1994), 'A hidden Markov model that finds genes in *E. coli* DNA', *Nucleic Acids Res.*, Vol. 22, pp. 4768–4778.

37. Lukashin, A. V. and Borodovsky, M. (1998), 'GeneMark.hmm: New solutions for gene finding', *Nucleic Acids Res.*, Vol. 26, pp. 1107–1115.

38. Yada, T. and Hirosawa, M. (1996), 'Detection of short protein coding regions within the cyanobacterium genome: Application of the hidden Markov model', *DNA Res.*, Vol. 3, pp. 355–361.

39. Yada, T., Totoki, Y., Takagi, T. and Nakai, K. (2001), 'A novel bacterial gene-finding system with improved accuracy in locating start codons', *DNA Res*, Vol. 8, pp. 97–106.

40. Larsen, T. S. and Krogh, A. (2003), 'EasyGene – a prokaryotic gene finder that ranks ORFs by statistical significance', *BMC Bioinformatics*, Vol. 4:21.

41. Burge, C. and Karlin, S. (1997), 'Prediction of complete gene structures in human genomic DNA', *J. Mol. Biol.*, Vol. 268, pp.78–94.

42. Krogh, A. (1997), 'Two methods for improving performance of an HMM and their application for gene finding', *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, Vol. 5, pp. 179–186.

43. Krogh, A. (2000), 'Using database matches with HMMGene for automated gene detection in *Drosophila*', *Genome Res.*, Vol. 10, pp. 523–528.

44. Besemer, J., Lomsadze, A. and Borodovsky, M. (2001), 'GeneMarkS: A self-training method for prediction of gene starts in microbial genomes. Implications for finding sequence motifs in regulatory regions', *Nucleic Acids Res.*, Vol. 29, pp. 2607–2618.

45. Badger, J. H. and Olsen, G. J. (1999), 'CRITICA: Coding region identification tool invoking comparative analysis', *Mol. Biol. Evol.*, Vol. 16, pp. 512–524.

46. Shibuya, T. and Rigoutsos, I. (2002), 'Dictionary-driven prokaryotic gene finding', *Nucleic Acids Res.,* Vol. 30, pp. 2710–2725.

47. Rigoutsos, I. and Floratos, A. (1998), 'Combinatorial pattern discovery in biological sequences: The TEIRESIAS algorithm', *Bioinformatics*, Vol. 14, pp. 55–67.

48. Link, A. J., Robison, K. and Church, G. M. (1997), 'Comparing the predicted and observed properties of proteins encoded in the genome of *Escherichia coli* K-12', *Electrophoresis*, Vol. 18, pp. 1259–1313.

49. URL: http://www.ncbi.nlm.nih.gov/genomes/MICROBES/Complete.html

50. URL: http://opal.biology.gatech.edu/GeneMark/GeneMarkS/index.html

51. Rudd, K. E. (2000), 'EcoGene: A genome sequence database for *Escherichia coli* K-12', *Nucleic Acids Res.*, Vol. 28, pp. 60–64.

52. URL: http://bmb.med.miami.edu/EcoGene/EcoWeb