Log-odd: A new method for improving hidden Markov model decoding for gene finding

AHMED M. KHEDR* AND MOHAMED HAMZA IBRAHIM**

*Computer Science Department, Faculty of Sciences, University of Sharjah, Sharjah, UAE, email: akhedr@Sharjah.ac.ae
*Mathematics Dept., Faculty of Science, Zagazig University, Zagazig, Egypt, email: amkhedr@zu.edu.eg
**Computer and Software Engineering department, Polytechnique Montreal, University of Montreal, Canada, email: mohamed.ibrahim@polymtl.ca

ABSTRACT

Hidden Markov models (HMMs) are applied to many problems of computational Molecular Biology. In a predictive task, the HMM is endowed with a decoding algorithm in order to assign the most probable path of states, and in turn the class labeling, to an unknown sequence. In this paper, we have introduced a new decoding algorithm called (Log odd-Viterbi (LV)) for gene finding, which combines the log odd of posterior probability and Viterbi algorithms, to avoid the drawbacks of using only Viterbi, or Posterior algorithms, and also to avoid under flow problem. LV is a two step process: In the first step, the log odd of posterior probability is computed at each state using posterior decoding algorithm and then the best allowed path through the model is evaluated by Viterbi algorithm. Our simulation results show that our proposed LV has better performance than other existing algorithms in the computational biological problems such as predicting coding regions in prokaryotic DNA sequences.

Keywords: Hidden Markov Model, viterbi algorithm, posterior algorithm, log-odd posterior, DNA sequences.

INTRODUCTION

A large number of ideas from the fields of pattern recognition and machine learning are being applied to solve problems relating to biological data. Biological data (DNA and protein) sequences are relatively large sequences defined over some alphabet. These biological sequences can be clustered into different groups based on the similarity of their biological functionality. HMMs are powerful machine learning tools that are applicable to the molecular biology problems. The computing problems in this context relate to processing of large
sequences to infer various complex interrelationships among the sequences, their functional manifestations, clustering or determination of their functional family. When HMMs are implemented for prediction task, a decoding algorithm is needed. Decoding phase is the process of finding the most suitable path of states that represents a given sequence.

In prediction tasks, the HMM is endowed with a decoding algorithm in order to assign the most probable state path, and in turn the class labelling to an unknown sequence. There are many argued decoding algorithms such as Viterbi, 1-best, and Posterior decoding algorithms. The decoding algorithm takes the observation sequence as input and returns the best path of states that represents it in the model. The Viterbi and the Posterior decoding algorithms are the most common algorithms used in decoding. The Viterbi decoding is very efficient in determining the best allowed path when one path dominates the rest, however it fails to return a path if concurring paths have similar probabilities. On the other hand, the posterior decoding in contrary is more effective when several concurring paths have similar probabilities, however it does not guarantee to preserve the automaton grammar, or in other words it may get a path that might be not allowed. The third alternative is 1-best, which was shown to perform equal or better than Viterbi (Krogh, 1997).

In this paper, we introduce a new decoding LV algorithm that combines the log odd of posterior decoding and Viterbi decoding algorithms. LV can be described as follows: at each state, the log odd of posterior probability is computed and then the best allowed path through the model is evaluated by Viterbi algorithm. The proposed algorithm uses posterior probability to solve the several paths of similar probabilities problem that occurs in Viterbi, and uses Viterbi to overcome the unallowed path problem that occurs in posterior decoding. Also, we use a log odd probability to make posterior probability more accurate and to overcome the problem of underflow that appears when dealing with very large sequences like DNA sequences. Posterior algorithm has accurate computations based on posterior probabilities; however it is not efficient in decoding. On the contrary, Viterbi algorithm has inaccurate computations, but it is efficient in decoding based on dynamic programming. So that taking into account Posterior-Viterbi ordering guides, LV exploits the advantages of Posterior and Viterbi algorithms and guarantees the improvement. Our simulation results show that the performance of LV is better than other decoding algorithms in the computational biological problems such as predicting coding regions in DNA sequences, and some simple random sequences.

Related research

Machine learning approaches have been shown to be very useful in the field of
computational Molecular Biology (Baldi & Brunak, 2001; Holems & Durbin, 1998; Khedr, 2011; El-Sayed & Khedr, 2007; Suhartati, 2011). Among these approaches HMMs have been proven to be especially successful method (Baldi & Brunak, 2001; Durbin et al., 1998; Anggi et al., 2008). HMMs were developed for alignments (Baldi et al., 1994; Krogh et al., 1994; Brown & Turszkowski, 2010), pattern detection (Bateman et al., 2002; Mamitsuka, 1998), predictions as in the case of the topology of all-alpha and all-beta membrane proteins (Bagos et al., 2004; Bigelow et al., 2004; Krogh et al., 2001; Martelli et al., 2002; Tusnády & Simon, 1998), and gene finding (Dosztányi et al., 2003).

The most famous decoding algorithms are Viterbi, Posterior, and 1-best. Viterbi finds the most probable allowed path through HMM model. Viterbi decoding is particularly effective when there is a single best path among other much less probable ones, however it is not effective when several best paths have similar probability, in this case, the posterior and the 1-best algorithms are more convenient (Krogh, 1997). The posterior decoding assigns the state path on the basis of the posterior probability, the selected path might be not allowed. For this reason, in order to recast the automaton constraints, a post-processing algorithm should be applied to the posterior decoding (Martelli et al., 2002). In this paper, we proposed a new algorithm to overcome the problems between Viterbi and Posterior decoding algorithms. Here, we summarize the main decoding algorithms that are used in our proposed algorithm:

- **Viterbi Decoding:** Viterbi decoding finds the path \( \pi \) through the model, which has the maximal probability with respect to all the others (Baldi et al., 1994; Durbin et al., 1998). This means that we look for the path which satisfies

\[
\pi^* = \arg \max_{\pi} P(\pi|O, M),
\]

where \( O = O_1, \ldots, O_L \) is the observed sequence of length \( L \) and \( M \) is the trained HMM model. Since the \( P(O|M) \) is independent of a particular path \( \pi \), Eq. (1) is equivalent to:

\[
\pi^* = \arg \max_{\pi} P(\pi, O|M),
\]

where \( P(\pi, O|M) \) can be easily computed as:

\[
P(\pi, O|M) = \left( \prod_{i=1}^{L} a_{\pi(i-1), \pi(i)} e_{\pi(i)}(O_i) \right) a_{\pi(L), \text{End}}
\]
Defining $v_k(i)$ as the probability of the most likely path ending in state $k$ at position $i$, and as the trace-back $P_i(k)$ pointer, $\pi^v$ can be obtained by running the following dynamic programming based algorithm called Viterbi decoding.

- **Initializations:**
  
  \[ v_{\text{Begin}}(0) = 1, \quad v_k(0) = 0 \quad \text{for} \quad k \neq \text{Begin} \]

- **Recursion:**
  
  \[ v_k(i) = \max_{(s)} (v_s(i - 1)a_{s,k})e_k(O_i). \]
  
  \[ P_i(k) = \arg \max_{(s)} (v_s(i - 1)a_{s,k}). \]

- **Termination:**
  
  \[ P(O, \pi^v|M) = \max_{(s)} [v_s(L)a_{s,\text{End}}]. \]
  
  \[ \pi^v_L = \arg \max_{(s)} [v_s(L) = a_{s,\text{End}}]. \]

- **Traceback:**
  
  \[ \pi^v_{i-1} = P_i(\pi^v_i). \]

- **Label assignment:**
  
  \[ \mu_i = \text{label}(\pi^v_i), \quad \text{for} \quad i = 1, 2, \ldots, L. \]

Viterbi generates all possible paths in HMM for query sequence $O$, then computes the probability of each path and chooses the path with large probability which is the most probable path. The problem occurs if there are two or more paths with the same probability. Our proposed algorithm uses posterior probability to solve the several paths of similar probabilities problem that occurs in Viterbi.

- **Posterior Decoding:** The posterior decoding finds the path that maximizes the product of the posterior probability of the states (Baldi et al., 2001; Durbin et al., 1998). Using the usual notation for forward $f_k(i)$ and backward $b_k(i)$, we have:

  \[ P(\pi_i = k | O, M) = \frac{f_k(i) \cdot b_k(i)}{P(O|M)} \quad (4) \]

The path $\pi^p$ that maximizes the posterior probability is then computed as:

\[ \pi^p_i = \arg \max_s P(\pi_i = s | O, M), \quad \text{for} \quad i = 1, 2, \ldots, L. \quad (5) \]

The corresponding label assignment is:

\[ \mu_i = \text{label}(\pi^p_i), \quad \text{for} \quad i = 1, 2, \ldots, L. \quad (6) \]
If we have more than one state sharing the same label, labeling can be improved by summing over the states that share the same label (posterior sum). This way we can have a path through the model which maximizes the posterior probability of being in a state with label \( \eta \) when emitting the observed sequence element, or more formally,

\[
\mu_i = \arg\max_{\eta} \sum_{\text{Label}(s) = \eta} P(\pi_i = s|O, M). \tag{7}
\]

The posterior-decoding drawback is that the state path sequences \( \pi^p \) or \( \mu \) may not be allowed paths. However, this decoding can perform better than Viterbi when there more than one high probable path (Baldi & Brunak, 2001; Durbin et al., 1998). In this case, the post-processing algorithm that recasts the original topological constraints is recommended (Fariselli et al., 2003). In sequel, if not differently indicate with the term posterior (posterior sum). Our proposed algorithm uses Viterbi to overcome the un-allowed path problem that occurs in posterior decoding.

- **1-best Decoding**: The 1-best labelling algorithm described here is the Krogh’s previously described variant of the N-best decoding (Krogh, 1997; Brown & Golod, 2010). Since there is no exact algorithm for finding the most probable labeling, 1-best is an approximate algorithm which usually achieves good results for solving the decoding problem (Krogh, 1997). Different from Viterbi, the 1-best algorithm ends when the most probable labeling is computed, so that no trace-back is needed.

For sake of clarity, we define \( H_i \) as the set of all labeling hypotheses surviving as 1-best for each state \( s \) up to sequence position \( i \). In the worst case, the number of distinct labeling-hypotheses is equal to the number of states. The term \( h_i^s \) is the current partial labeling hypothesis associated to the state \( s \) from the beginning to the \( i^{th} \) sequence position. In general, several states may share the same labeling hypothesis. Finally, we use \( \oplus \) as the string concatenation operator, so that ’AAAA’ \( \oplus \) ’B’ = ’AAAAB’. 1-best algorithm can then be described as follows:

- **Initialization:**

\[
\begin{align*}
\nu_{\text{Begin}}(0) & = 1, \quad \nu_k(0) = 0 \quad \text{for} \quad k \neq \text{Begin} \\
\nu_k(1) & = a_{\text{Begin}, k}.e_k(O_i), \quad H_1 = \{\text{label}(k) : \ a_{\text{Begin}, k} \neq 0\} \\
H_i & = \phi, \quad \text{for} \quad i = 2, 3, 4, \ldots, L.
\end{align*}
\]
• **Recursion:**

\[ v_k(i + 1) = \max_{h \in H_i} \left[ \sum_s v_s(i) \gamma(h_i^s, h) \cdot a_{s,k} \right] e_k(O_{i+1}) \]

\[ h_{i+1}^k = \arg \max_{h \in H_i} \left[ \sum_s v_s(i) \gamma(h_i^s, h) \cdot a_{s,k} \right] \oplus \text{label}(k) \]

\[ H_{i+1} \leftarrow H_{i+1} \cup \{ h_{i+1}^k \} \]

• **Termination:**

\[ \mu = \arg \max_{h \in H_L} \sum_s v_s(L) \gamma(h_L^s, h) \cdot a_{s,\text{End}}. \]

The function \( \gamma(a, b) = 1 \) when \( a = b \), 0 otherwise, and with 1-best decoding, we do not need to keep back trace matrix since it is computed during the forward steps.

**LOG-ODD VITERBI DECODING (LV)**

**Log-odd Ratio**

A better system is needed for describing and calculating probabilities on an order of magnitude basis, so that very rare or nearly certain events can be compared and comprehended in a more intuitive way. A new scale was proposed (Fleiss, 1973; Martin & Altmann, 1999), which can help the scientific and technical community to describe and compare these probabilities. It is hoped that this scale will provide a uniform standard with which the probabilities of both are very rare or nearly certain events can be communicated, intelligently compared.

The new scale can be referred to as the Log-odds scale "log-odd". It is logarithmic, and similar in spirit to the Richter scale. To convert a probability to Log-odds, first convert it into odds, and then take the logarithm. In algebraic terms, the scale value \( r \) for a probability of \( p \) is given by: \( r = \log-\text{odd}(p) = \log[p/(1-p)] \).

**Algorithm Description**

LV decoding is based on the combination of the log-odd posterior probability and Viterbi. Log-odd posterior is computed by applying log-odd with posterior decoding algorithm to get the log-odd posterior probability of each state at each position, and then Viterbi algorithm is applied to find the best allowed log-odd posterior path through the model.

In LV algorithm, the basic idea is to compute the path \( \pi^{LV} \) as:
\[
\pi^{LV} = \arg \max_{x \in F} \prod_{i=1}^{L} \log - odd(P(\pi_i|O, M)),
\]

(8)

Where \( x \) is the decoded path, \( F \) is set of allowed decoded paths through the model, and log-odd (\( P(\pi_i|O,M) \)) is the log-odd posterior probability of the state assigned by the path \( \pi \) at position \( i \) (as we take the log-odd in Eq. 8).

Defining the function \( \omega_{s,k} \) to be 1 if the transition from \( s \) to \( k \) is an allowed transition of the model \( M \), and 0 otherwise, \( v_k(i) \) is the probability of the most probable allowed log-odd path ending at state \( k \) having observed the partial \( O_1, \ldots, O_i \), and \( P_i \) is the trace-back pointer, we can compute the best path \( \pi^{LV} \) using the Viterbi algorithm as follows:

- **Initializations:**

\[
v_{\text{Begin}}(0) = 1, \quad v_k(0) = 0 \quad \text{for} \quad k \neq \text{Begin}
\]

(9)

- **Recursion:**

\[
v_k(i) = \max_{s} (v_s(i-1).w_{s,k}) \times \log - odd(P(\pi_i = k|O, M)) \quad \text{for} \quad k \neq \text{Begin}
\]

(10)

\[
P_i(k) = \arg \max_{s} (v_s(i-1).w_{s,k}).
\]

(11)

- **Termination:**

\[
P(\pi^{LV}|O, M) = \max_{s} (v_s(L).w_{s,\text{End}}),
\]

(12)

\[
\pi^{LV} = \arg \max_{s} (v_s(L).w_{s,\text{End}}).
\]

(13)

- **Traceback:**

\[
\pi^{LV}_{i-1} = P_i(\pi^{LV}_i), \quad \text{for} \quad i = 1, 2, 3, \ldots, L.
\]

(14)

- **Label assignment:**

\[
\mu_i = \text{label}(\pi^{LV}_i), \quad \text{for} \quad i = 1, 2, 3, \ldots, L,
\]

(15)

where \( \omega_{s,k} \) is defined as:

\[
w_{s,k} = \begin{cases} 
1 & \text{If the transition from } s \text{ to } k \text{ is allowed.} \\
0 & \text{Otherwise.}
\end{cases}
\]

(16)
From conditional probability rule we have:

\[ P(\pi_i = k, O) = P(\pi_i | O, M) \times P(O | M) \]

In terms of forward-backward probabilities, the posterior probability will be:

\[ P(\pi_i = k, O) = f_k(i) \times b_k(i) \]

Thus we have:

\[ f_k(i) \times b_k(i) = P(\pi_i | O, M) \times P(O | M) \]

\[ P(\pi_i | O, M) = \frac{f_k(i) \times b_k(i)}{P(O | M)} \]

\[ \log \text{- odd} \left( P(\pi_i | O, M) \right) = \log \text{- odd} \left( \frac{f_k(i) \times b_k(i)}{P(O | M)} \right) \]

\[ \log \text{- odd} \left( P(\pi_i | O, M) \right) = \log \text{- odd} \left( f_k(i) \times b_k(i) \right) - \log \text{- odd} \left( P(O | M) \right) \]

Then, finally we will have:

\[ \log \text{- odd} \left( P(\pi_i | O, M) \right) = \log \left( \frac{f_k(i) \times b_k(i)}{1 - (f_k(i) \times b_k(i))} \right) - \log \left( \frac{P(O | M)}{P(O | \text{Null Model})} \right), \]

where \( \log\text{-odd} \left( P \right) \) is the \( \log\text{ odd} \) ratio of the posterior probability \( P \), which can be calculated as \( \log\text{-odd} \left( P \right) = \log \left( \frac{P}{1-P} \right) \), \( P \) is the probability, and \( 1 - P \) is its null probability. For simplicity, the null probability model here can be chosen as uniform distribution.

One of the advantages of \( \log\text{-odd} \) probability applied on posterior is that it makes use of logarithm computation, which overcomes the occurrence of underflow problem, especially when we are dealing with extensive multiplication operations on probabilities.

**RESULTS AND DISCUSSIONS**

**Data sets**

The used data to score our proposed, Viterbi, and Posterior decoding algorithms are as follows:
1 - 60 sequences of 400 outputs are randomly generated from TF model which is represented in Fig.1. Initially, the emissions and transitions probabilities in the model set to uniform distributions. Then, the generated sequences are used to train the model using the standard Baum-Welch algorithm. After training, we tested the ability of decoding algorithms Viterbi, I-best, posterior, and LV to recover the original labeling from the observed sequences.

2 - A set of 40 DNA sequences, which are taken from http://www.ncbi.nlm.nih.gov/sites/entrez, for DUTPasecoli. These DNA sequences have length range from 800 bp to 1250 bp. Aligning these DNA sequences show that their conserved region represents less than 25% of their total length, in other words, these DNA sequences are less than 25% identical. Among these 40 DNA sequences, 30 were used randomly to train the HMM, the other 10 DNA sequences are used as a test set. Here, our focus is to test the ability of Viterbi, Posterior, I-best, and LV decoding algorithms to obtain accurate labeling decoded path which can be used for predicting gene in these DNA sequences.

**Measures of accuracy**

Here, we have used some measures to test the accuracy and performance of predictions for TF model and HMM of DNA as the following:

**TF model:** To test the prediction accuracy of underlying decoding algorithms within TF model, we used common measure accuracy ratio, abbreviated as MA1 measure, that counts the number of correctly predicted outcomes from Fair notated as M (i.e. the outcomes with label F on the sum of number of correctly predicted outcomes M from Fair plus number of non correctly outcomes predicted from F notated as A:

$$MA1 = \frac{M}{M + A}$$

(18)

**HMM model of DNA:** we use nucleotide level measure, where each nucleotide of testing sequence is classified as predicted positive (PP) if it is in predicted coding region, or predicted negative (PN) otherwise, and also as actual positive (AP) or actual negative (AN) according to annotation of the DNA sequence. These assignments are then compared to calculate the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). Accuracy is then measured by using the following formulas:
1 - Sensitivity

\[ SEN = \frac{TC}{TC + FN} \]  \hspace{1cm} (19)

2 - Specificity

\[ SPE = \frac{TC}{TC + FC} \]  \hspace{1cm} (20)

Sensitivity and specificity are statistical measures of the performance of a binary classification and prediction test. Sensitivity used to measures the proportion of actual positives which are correctly identified (here used to measure the ability of decoding algorithms within HMMs to sense the nucleotides in DNA sequences, and Specificity measures the proportion of negatives which are correctly identified (here we use it to judge the ability of decoding algorithms and HMMs to specify the correct labeling on the DNA sequences). We choose these two measures since they are closely related to the gene prediction, which is a kind of prediction problems.

\[ \text{Figure 1: Occasionally dishonest casino (TFModel)} \]

The emission probability of the fair state \( F \) is 1/6 for each possible outcome, for the \( T \) state is 1/2 for '2', and 1/10 for the other faces.

**TESTING RESULTS**

**Simple TF model testing**

We start using one of the simplest HMM model that can be thought of \( \text{TF} \), which is the occasionally dishonest casino. \( \text{TF} \) can generate any observed sequence of numbers ranging from 1 to 6 called outcomes. In \( \text{TF} \) model the labels are \( \text{Tor F} \), the outcome takes label \( T \) if it is generated from state \( T \), and labeled \( F \) if it is generated from state \( F \). We produced 60 sequences with 400 dice outcomes and we trained the model with them. Then, we tested the model with
the four decoding algorithms (Viterbi, Posterior, I-best, and LV) to reconstruct the correct labeling. Indeed, in such kind of benchmark problems, normally HMMs find the predicting labels in a very small amount of time, less than one second; thereby we run the experiment to test the ability of our proposed decoding algorithm and the other standard decoding algorithms within HMM to accurately find the predictions. In other words; taking into account the average time, all decoding algorithms including our proposed one approximately have the same time. The results that indicate the differentiation accuracy between the decoding algorithms are summarized in the following table:

![Table 1. The Performance of the four Decoding Algorithms on TF model](image)

<table>
<thead>
<tr>
<th>Meas.</th>
<th>Alg.</th>
<th>Viterbi</th>
<th>Posterior</th>
<th>I-Best</th>
<th>LV</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA1</td>
<td></td>
<td>0.51</td>
<td>0.50</td>
<td>0.57</td>
<td>0.80</td>
</tr>
</tbody>
</table>

From Table 1, it is clear to say that the performance of the LV is better than those of the other three decoding algorithms.

**DNA sequences testing**

Here, we have tested our proposed decoding algorithm LV on real biological data. We have used a previously developed HMM, devised for the prediction of gene finding (Krogh, 1997). Taking the advantage of emitting more than one symbol at each state, as described (Krogh, 1994), the labels in this case are "C" for coding regions, and "N" for non coding regions.

Table 2 shows the obtained predictions sensitivity, and specificity measurements for the four different decoding algorithms when applying with the 40 DNA sequences with their labels (previously mentioned in Data sets part 2). 30 out of 40 DNA sequences used as across validation test, and the latter 10 DNA sequences for blind-test.

From Table 2, it is obvious that the Viterbi decoding and the I-best are unreliable, since the blind test is less than 20 %. On the other hand, posterior decoding is more efficient and can correctly assigns more than 50 % and 29% of the DNA sequences, in cross-validation and blind test, respectively. LV has the best performance since its decoding achieves 80 % and 63 % in cross-validation and blind test, respectively.
Table 2. The Performance of the four Decoding Algorithms on DNA sequences

<table>
<thead>
<tr>
<th>Alg.</th>
<th>Viterbi</th>
<th>Posterior</th>
<th>1-Best</th>
<th>LV</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cross V.</td>
<td>Blind Test</td>
<td>Cross V.</td>
<td>Blind Test</td>
</tr>
<tr>
<td>SEN.</td>
<td>0.39</td>
<td>0.05</td>
<td>0.50</td>
<td>0.29</td>
</tr>
<tr>
<td>SPE.</td>
<td>0.51</td>
<td>0.10</td>
<td>0.53</td>
<td>0.30</td>
</tr>
</tbody>
</table>

From the testing results, we can say LV performs better than Viterbi, because it overcomes the problem of several paths that have same probabilities by including posterior probability. LV performs better than posterior decoding because it overcomes the problem of un-allowed paths that occur in posterior decoding by including Viterbi. Finally, LV uses log-odd to compute the probability in more accurate manner, and at the same time overcomes the underflow problem.

Complexity

**Time complexity:** the time complexity for Viterbi, 1-best, and posterior is $O(N^2.L)$. Although LV algorithm combines log-oddposterior probability and Viterbi algorithm, but its time-complexity order still the same as other decoding algorithms which is $O(N^2.L)$, where $L$ and $N$ are the DNA sequence length and the number of states in HMM, respectively.

$$\text{Time}(\text{Viterbi}) \leq \text{Time}(1-\text{best}) \leq \text{Time}(\text{Posterior}) \leq \text{Time}(\text{LV})$$

**Space complexity:** the space-complexity of Viterbi and Posterior algorithms is $O(N.L)$, while 1-best requires less memory space (Krogh, 1997). LV has space complexity near to the three other decoding algorithms, which is also $s$.

$$\text{Space}(1-\text{best}) \leq \text{Space}(\text{Viterbi}) \leq \text{Space}(\text{Posterior}) \leq \text{Space}(\text{LV})$$

CONCLUSION

In this paper, we have presented a new decoding algorithm, called log-oddViterbi (LV). LV satisfies any HMM grammar structures, and it is applicable to all the possible HMM models with an arbitrary number of labels. When one-state-path dominates, we may expect that LV does not perform better than the other existing decoding algorithms. However, when several concurring paths are present, LV performs better than the other decoding algorithms. We have
applied our proposed algorithm to predict gene as a motivation to improve the decoding algorithms within HMM. However, our proposed decoding algorithm is general and could be applicable with HMM in several fields such as telecommunications and speech recognition.

ACKNOWLEDGEMENT

The authors would like to thank University of Sharja and Zagazig University for supporting this work.

REFERENCES


Submitted : 4/1/2011
Revised : 10/12/2011
Accepted : 15/1/2012
طريقة جديدة لتحسين نماذج ماركوف المخفية لاكتشاف وتصنيف الجينات

أحمد محمد خضر و **محمد حمرة إبراهيم

قسم علوم الحاسب - كلية العلوم - جامعة الشارقة - الإمارات
قسم الحاسب و هندسة البرمجيات - جامعة مونتريال - كندا

خلاصة

تستخدم نماذج ماركوف المخفية في حل كثيراً من المشاكل الحسابية في مجال البيولوجيا الحزينة. فمثلاً تستخدم نماذج ماركوف للكشف وتصنيف السلاسل المجهولة (جينات - بروتينات) إلى عائلتها الصحيحة.

في هذا البحث تم تقديم خوارزمية جديدة تسمى Log odd Viterbi (LV) لتسريع عملية استخراج وتصنيف سلاسل الجينات المجهولة إلى عائلتها الصحيحة. تتجنب الخوارزمية المقترحة العيوب الناتجة عن استخدام خوارزمية Posterior لقيم Log-odd باختصار خوارزمية Viterbi. وكذلك تمنع حدوث مشكلة Underflow لاختلاف أفضل مسار داخل نموذج ماركوف.

تظهر نتائج المحاكاة أن الخوارزمية المقترحة تعطي نتائج أفضل من استخدام فقط خوارزمية Viterbi أو Posterior لتصنيف سلاسل الجينات المجهولة إلى عائلتها الصحيحة.