Sequentially finding the N-Best List in Hidden Markov Models

Dennis Nilsson
Aalborg University
Bajers Vej 7 E, 9220 Aalborg Øst
Denmark
nilsson@math.auc.dk

Jacob Goldberger
The Weizmann Institute of Science
Rehovot, 76100
Israel
jacob@wisdom.weizmann.ac.il

Abstract

We propose a novel method to obtain the N-best list of hypotheses in hidden Markov model (HMM). We show that the entire information needed to compute the N-best list from the HMM trellis graph is encapsulated in entities that can be computed in a single forward-backward iteration that usually yields the most likely state sequence. The hypotheses list can then be extracted in a sequential manner from these entities without the need to refer back to the original data of the HMM. Furthermore, our approach can yield significant savings of computational time when compared to traditional methods.

1 Introduction

In many tasks of large vocabulary speech recognition it is desirable to find from the HMM graph the N most likely state sequences given the observed acoustic data. The recognizer chooses the utterance hypotheses on the basis of acoustic information and a relatively simple language model. The existence of an N-best list enable us to combine additional knowledge sources such as complicated acoustic and language models into the recognition process [Ostendorf 1991]. Given the additional knowledge sources the list of sentence can be rescored and reordered. Even without additional knowledge sources the N-best paradigm can be used to improve the recognition rate [Stolcke *et al*. 1997]. In this paper we concentrate on the step of computing the N-best list. The most likely state sequence can be found using a single iteration of the Viterbi algorithm [Rabiner 1989]. A direct generalization of this algorithm can be used to obtain the N best state sequences. The only change is that for each time index t and for each state we have to keep the N best subsequences terminating at this state. However in this generalized Viterbi approach we have to decide in advance on the size of N and we can not change it in the middle of the process. Several modification of this algorithm have been proposed in the last decade. These algorithms are based either on a Viterbi search of a trellis or on A^* search [Schwartz and Chow 1991] [Schwartz *et al*. 1996].

We propose a novel method to obtain the N-best list in HMM. The obtained algorithm is inspired by the divide and conquer algorithm in [Nilsson (1998] for finding the M most likely configurations in probabilistic expert systems. The present algorithm also has the advantage of being an anytime algorithm since we need not in forehand specify the number of N best hypothesis that is wanted. Furthermore, our algorithm can yield significant savings in computational time compared to the traditional Viterbi algorithm.

2 Basic Structure

Consider a HMM with m hidden Markovian random variables $X_1, \ldots, X_m$ and m observed variables $Y_1, \ldots, Y_m$ such that the distribution of Y_t is determined by X_t. Denote $X = \{X_1, \ldots, X_m\}$ and $Y = \{Y_1, \ldots, Y_m\}$. Typical values that X and Y can take are denoted $x = (x_1, \ldots, x_m)$ and $y = (y_1, \ldots, y_m)$ respectively. The joint probability function is:

$$P(x,y) = P(x_1)\prod_{t=2}^{m} P(x_t|x_{t-1})\prod_{t=1}^{m} P(y_t|x_t) \quad (1)$$

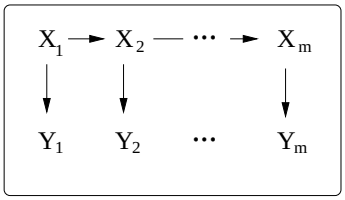

Figure 1: The HMM structure

This paper deals with the following decoding problem: Given observations $y_1 \ldots y_m$, find the N most likely state sequences of the unobserved state-variables. In other words we want to find the N values of x that maximize the conditional probability function $P(x,y)$ (viewed as a function of x). A single iteration of the forward-backward algorithm [Rabiner 1989] yields the following terms for each time index t:

$$\begin{aligned} f_t(s) &= \max_{\{x|x_t=s\}} P(x,y) \quad (2)\\ f_{t,t+1}(s,s') &= \max_{\{x|(x_t,x_{t+1})=(s,s')\}} P(x,y) \end{aligned}$$

We shall show that the entire information needed to compute the N-best list is encapsulated in the expressions defined in

(2). In other words, once $f_t(s)$ and $f_{t,t+1}(s,s')$ are given, there is no need to refer again to the trellis graph.

As a first example of the usefulness of f_t and $f_{t,t+1}$, we apply them to obtain the most likely state sequence. One can observe that the probability of the most likely state sequence is :

$$\max_x P(x,y) = \max_s f_t(s)$$

This equality remains the same for each index $t = 1,\ldots,m$. To find the most likely state sequence $\hat{x} = (\hat{x}_1,\ldots,\hat{x}_m)$, when this is uniquely determined, we note that

$$\hat{x}_t = \arg\max_s f_t(s),\ \ t = 1,\ldots,m$$

However to allow for the possibility that the maximizing state sequence is not unique, it is better to apply the following routine:

$$\begin{aligned} &(\hat{x}_1,\hat{x}_2) = \arg\max_{(s,s')} f_{1,2}(s,s') \\ &\hat{x}_t = \arg\max_s f_{t-1,t}(\hat{x}_{t-1},s),\ \ t > 2 \end{aligned} \tag{3}$$

3 Introducing the algorithm

Denote the entire ensemble of possible state sequences by $\mathcal{X}$ and let the Lth most likely state sequence be denoted

$$x^L = (x_1^L,\ldots,x_m^L).$$

Suppose at some stage in the algorithm that $x^1,\ldots,x^{L-1}$ have been identified. Then x^L is found by performing the following steps:

PARTITION PHASE**:** Here $\mathcal{X}\setminus\{x^1,\ldots,x^{L-1}\}$ is partitioned into subsets.

CANDIDATE PHASE**:** For each subset in the above partitioning we compute the probability of its most likely state sequence. These probabilities are referred to as the 'candidates'.

IDENTIFICATION PHASE**:** The state sequence associated with the highest candidate is identified.

In PARTITION PHASE, there is a large number of possible partitions that one may consider. We seek a partitioning with two properties. Firstly, it must be easy to compute the candidates. Secondly, the number of subsets in the partitioning must be small, since eventually we need to compare the different candidates.

In the successively subsections it is shown how the second and third most likely state sequences are found. In Section 4 the general case is considered.

3.1 The second most likely state sequence

Suppose the most likely state sequence x^1 has now been identified. To identify x^2 we carry out the following steps.

PARTITION PHASE

Here, $\mathcal{X}\setminus\{x^1\}$ is partitioned into subsets $A_1,\ldots,A_m$ defined by

$$\begin{aligned} &A_1 = \{x \mid x_1 \neq x_1^1\} \\ &A_i = \{x \mid x_1 = x_1^1,\ldots,x_{i-1} = x_{i-1}^1, x_i \neq x_i^1\},\ \ i \geq 2. \end{aligned} \tag{4}$$

CANDIDATE PHASE

We let $\hat{P}(A_i)$ stand for the probability of the most likely state sequence in the subset A_i, i.e.

$$\hat{P}(A_i) = \max_{x\in A_i} P(x,y),$$

and use a similar notation for other subsets.

The functions $f_{t,t+1}$ satisfy that

$$\begin{aligned} &\hat{P}(A_1) = \max_{\{(s,s')|s\neq x_1^1\}} f_{1,2}(s,s') \\ &\hat{P}(A_i) = \max_{s\neq x_i^1} f_{i-1,i}(x_{i-1}^1,s),\quad i > 2 \end{aligned} \tag{5}$$

So, the second most likely state sequence has probability

$$P(x^2,y) = \max_i \hat{P}(A_i).$$

IDENTIFICATION PHASE

If $\hat{P}(A_i) = \max\{\hat{P}(A_j) : j = 1,\ldots,m\}$, then $x^2 \in A_i$, and x^2 can be identified by carrying out the following steps:

- $x_j^2 = x_j^1$ for $j = 1,\ldots,i-1$.
- $x_i^2 = \arg\max_{s\neq x_i^1} f_{i-1,i}(x_{i-1}^2,s)$
- $x_k^2 = \arg\max_s f_{k-1,k}(x_{k-1}^2,s)$ for $k = i+1,\ldots,m$.

3.2 The third most likely state sequence

The identification of the third most likely state sequence is done by a similar procedure as the case with x^2.

PARTITION PHASE

Here, a partition of $\mathcal{X}\setminus\{x^1,x^2\}$ is constructed by refining the above partitioning of $\mathcal{X}\setminus\{x^1\}$. This is done in the following way.

Suppose $x^2 \in A_i$. Then we partitioning $A_i \setminus \{x^2\}$ into subsets $B_i,\ldots,B_m$ defined by

$$\begin{aligned} &B_i = \left\{x \mid x_1 = x_1^2,\ldots,x_{i-1} = x_{i-1}^2, x_i \notin \{x_i^1,x_i^2\}\right\} \\ &B_k = \left\{x \mid x_1 = x_1^2,\ldots,x_{k-1} = x_{k-1}^2, x_k \neq x_k^2\right\},\ k > i. \end{aligned} \tag{6}$$

Thus, the subsets in $\{B_j : j \geq i\}$) and $\{A_j : j \neq i\}$) constitute a partitioning of $\mathcal{X}\setminus\{x^1,x^2\}$.

CANDIDATE PHASE

As we shall prove in the next section (Theorem 1), the probability $\hat{P}(B_k)$ for $k \geq i$ is a simple function of $\hat{P}(A_i)$:

$$\begin{aligned} \hat{P}(B_i) &= \hat{P}(A_i)\frac{\max_{s\notin\{x_i^1,x_i^2\}} f_{i-1,i}(x_{i-1}^2,s)}{f_{i-1,i}(x_{i-1}^2,x_i^2)} \\ \hat{P}(B_k) &= \hat{P}(A_i)\frac{\max_{s\neq x_k^2} f_{k-1,k}(x_{k-1}^2,s)}{f_{k-1,k}(x_{k-1}^2,x_k^2)},\ \ k > i. \end{aligned} \tag{7}$$

Now, the third most likely state sequence has probability

$$P(x^3,y) = \max\left\{\max_{j:j\neq i}\hat{P}(A_j)\ ,\ \max_{j:j\geq i}\hat{P}(B_j)\right\}.$$

IDENTIFICATION PHASE

If the third most likely state sequence x^3 belongs to one of the subsets A_j ($j \neq i$), then it can be identified in a similar way as x^2 was identified. On the other hand, if $x^3 \in B_j$ for some $j \geq i$, then

$$(x_1^3, \ldots, x_{j-1}^3) = (x_1^2, \ldots, x_{j-1}^2),$$

and the sequence $x_j^3, \ldots, x_m^3$ is now successively identified as follows:

$$x_j^3 = \begin{cases} \arg\max_{s \notin \{x_j^1, x_j^2\}} f_{j-1,j}(x_{j-1}^3, s) & \text{if } j = i \\ \arg\max_{s \neq x_j^2} f_{j-1,j}(x_{j-1}^3, s) & \text{if } j > i \end{cases}$$

$$x_k^3 = \arg\max_s f_{k-1,k}(x_{k-1}^3, s), \quad \text{for } k = j+1, \ldots, m$$

Example To illustrate our algorithm, we consider a simple HMM with 10 hidden variables $X_1, \ldots, X_{10}$, and 10 observed variables $Y_1, \ldots, Y_{10}$. Each of the variables are assumed binary with possible states y and n. The initial conditional probability functions associated with the HMM are given in Table 1.

$P(x_1)$	n	y
	.6	.4

$P(x_t \mid x_{t-1})$		x_t	
		n	y
x_{t-1}	n	.6	.4
	y	.2	.8

$P(x_t \mid y_t)$		x_t	
		n	y
y_t	n	.9	.1
	y	.3	.7

Table 1: The intial probabilities in the HMM

Suppose we obtain 'evidence' of the form

$$y = (y_1, y_2, \ldots, y_{10}) = (n, n, y, y, y, y, n, n, y, y).$$

To compute the N-best list given evidence, we initially compute the $f_{t,t+1}$ functions shown in Table 2. By applying the routine in (3), we can now obtain the most likely state sequence:

$$x^1 = (n, n, y, y, y, y, y, y, y, y)$$

with probability $P(x^1, y) = .2591 \cdot 10^{-3}$.

For the computation of the second most likely state sequence we proceed as follows. First, we partition the space $\mathcal{X} \setminus \{x^1\}$ into subsets $A_1, \ldots A_{10}$ defined in (4):

$$\begin{aligned}
A_1 &= \{x \mid x = (y, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot) \\
A_2 &= \{x \mid x = (n, y, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot) \\
A_3 &= \{x \mid x = (n, n, n, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot) \\
A_4 &= \{x \mid x = (n, n, y, n, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot) \\
A_5 &= \{x \mid x = (n, n, y, y, n, \cdot, \cdot, \cdot, \cdot, \cdot) \\
A_6 &= \{x \mid x = (n, n, y, y, y, n, \cdot, \cdot, \cdot, \cdot) \\
A_7 &= \{x \mid x = (n, n, y, y, y, y, n, \cdot, \cdot, \cdot) \\
A_8 &= \{x \mid x = (n, n, y, y, y, y, y, n, \cdot, \cdot) \\
A_9 &= \{x \mid x = (n, n, y, y, y, y, y, y, n, \cdot) \\
A_{10} &= \{x \mid x = (n, n, y, y, y, y, y, y, y, n)
\end{aligned}$$

$f_{1,2}$		x_2	
		n	y
x_1	n	.2590	.1151
	y	.0192	.0512

$f_{2,3}$		x_3	
		n	y
x_2	n	.0278	.2590
	y	.0021	.1151

$f_{3,4}$		x_4	
		n	y
x_3	n	.0030	.0278
	y	.0046	.2590

$f_{4,5}$		x_5	
		n	y
x_4	n	.0005	.0046
	y	.0046	.2590

$f_{5,6}$		x_6	
		n	y
x_5	n	.0025	.0046
	y	.0234	.2590

$f_{6,7}$		x_7	
		n	y
x_6	n	.0234	.0046
	y	.2185	.2590

$f_{7,8}$		x_8	
		n	y
x_7	n	.2185	.0971
	y	.0971	.2590

$f_{8,9}$		x_9	
		n	y
x_8	n	.0234	.2185
	y	.0046	.2590

$f_{9,10}$		x_{10}	
		n	y
x_9	n	.0050	.0234
	y	.0093	.2590

Table 2: The $f_{t,t+1}$ functions (the numbers are multiplied by 10^3)

Next, we use (5) to compute the probability of the most likely state sequence in each subsets A_i:

$$\begin{aligned}
\hat{P}(A_1) &= \max\{f_{1,2}(y,n), f_{1,2}(y,y)\} &&= .0512 \\
\hat{P}(A_2) &= f_{1,2}(n,y) &&= .1151 \\
\hat{P}(A_3) &= f_{2,3}(n,n) &&= .0278 \\
\hat{P}(A_4) &= f_{3,4}(y,n) &&= .0046 \\
\hat{P}(A_5) &= f_{4,5}(y,n) &&= .0046 \\
\hat{P}(A_6) &= f_{5,6}(y,n) &&= .0234 \\
\hat{P}(A_7) &= f_{6,7}(y,n) &&= .2185 \\
\hat{P}(A_8) &= f_{7,8}(y,n) &&= .0971 \\
\hat{P}(A_9) &= f_{8,9}(y,n) &&= .0046 \\
\hat{P}(A_{10}) &= f_{9,10}(y,n) &&= .0093
\end{aligned}$$

Thus, the second most like state sequence x^2 belongs to subset A_7, and can be found by carrying out steps (see Section 3.1):

$$(x_1^2, x_2^2, x_3^2, x_4^2, x_5^2, x_6^2, x_7^2) = (n, n, y, y, y, y, n),$$

and

$$\begin{aligned}
x_8^2 &= \arg\max_s\{f_{7,8}(n,s)\} = n \\
x_9^2 &= \arg\max_s\{f_{8,9}(n,s)\} = y \\
x_{10}^2 &= \arg\max_s\{f_{9,10}(y,s)\} = y
\end{aligned}$$

This identifies the second most likely state sequence.

To compute the third most likely state sequence we proceed in a similar manner: First, we partition $A_7 \setminus \{x^2\}$ in subsets of the form in (6):

$$
\begin{aligned}
B_7 &= \{x \mid x = (n,n,y,y,y,y,\cdot,\cdot,\cdot,\cdot) \wedge x_7 \notin \{n,y\}\} \\
B_8 &= \{x \mid x = (n,n,y,y,y,y,n,y,\cdot,\cdot)\} \\
B_9 &= \{x \mid x = (n,n,y,y,y,y,n,n,n,\cdot)\} \\
B_{10} &= \{x \mid x = (n,n,y,y,y,y,n,n,y,n)\}
\end{aligned}
$$

Then, we compute the probability of the most likely state sequence within each of the subsets B_i. From (7) we immediately obtain ($\hat{P}(B_7) = 0$ since $B_7 = \emptyset$):

$$
\begin{array}{llll}
\hat{P}(B_8) & = \hat{P}(A_7)\frac{f_{7,8}(n,y)}{f_{7,8}(n,n)} & = .2185\frac{.0971}{.2185} & = .0971 \\
\hat{P}(B_9) & = \hat{P}(A_7)\frac{f_{8,9}(n,n)}{f_{8,9}(n,y)} & = .2185\frac{.0234}{.2185} & = .00234 \\
\hat{P}(B_{10}) & = \hat{P}(A_7)\frac{f_{9,10}(y,n)}{f_{9,10}(y,y)} & = .2185\frac{.0093}{.2591} & = .0078
\end{array}
$$

Thus, the third most like state sequence x^3 belongs to subset A_2, since

$$\max\{\max_{i\neq 7} \hat{P}(A_i), \hat{P}(B_8), \hat{P}(B_9), \hat{P}(B_{10})\} = \hat{P}(A_2),$$

and can be found by carrying out the steps described in Section 3.2. This is left to the reader.

Table 3 shows the three most likely state sequences.

	Configuration	Probability (multiplied by 10^3)
x^1	(n,n,y,y,y,y,y,y,y,y)	$P(x^1,y) = .2591$
x^2	(n,n,y,y,y,y,n,n,y,y)	$P(x^2,y) = .2185$
x^3	(n,y,y,y,y,y,n,n,y,y)	$P(x^3,y) = .1151$

Table 3: The 3-best list

4 The general algorithm

The general algorithm for computing the N-best list identifies the N most likely state sequences in a sequentialy manner.

Let $\mathcal{X}_i$ denote the possible states that the variable X_i can take.

Suppose at some stage in the algorithm that

- $x^1, \ldots, x^L$ have been identified;
- A partitioning, say $\mathcal{P}^L$, of $\mathcal{X} \setminus \{x^1, \ldots, x^{L-1}\}$ is given.
- For each element D in $\mathcal{P}^L$, $\hat{P}(D)$ is known.
- $x^L \in D$ for some element D in the partitioning $\mathcal{P}^L$, and D has the following form:

$$D = \{x \mid x_1 = x'_1, \ldots, x_{i-1} = x'_{i-1}, x_i \notin \mathcal{X}'_i\}, \quad (8)$$

where $x'_1, \ldots, x'_{i-1}$ are known states, and $\mathcal{X}'_i$ is a known subset of $\mathcal{X}_i$.

The tasks involved in proving the correcteness of our algorithm consist of:

Partition-phase: Construct a partitioning of $D \setminus \{x^L\}$;

Candidate-phase: Compute the probability of the most likely state-sequence in each element in the partitioning of $D \setminus \{x^L\}$;

Identification-phase:

- Compare the 'new candidates' computed above with the remaining candidates.
- Identify x^{L+1}.

These three tasks are described in subsequent subsections.

4.1 Partition phase

Suppose $x^L \in D$, where D has the form in (8). For the following analysis it is convenient to write D as

$$D = \{x \mid x_1 = x_1^L, \ldots, x_{i-1} = x_{i-1}^L, x_i \notin \mathcal{X}_i'\}. \quad (9)$$

For $k = i, \ldots, m$ we define the subsets D_k as

$$D_k = \{x \mid x_1 = x_1^L, \ldots, x_{k-1} = x_{k-1}^L, x_k \notin \bar{\mathcal{X}}_k\}, \quad (10)$$

where $\bar{\mathcal{X}}_k$ is a subset of $\mathcal{X}_k$ given by

$$\bar{\mathcal{X}}_k = \begin{cases} \mathcal{X}_k' \cup \{x_k^L\} & \text{if } k = i \\ \{x_k^L\} & \text{if } k > i \end{cases}$$

The reader may easily verify that the subsets $D_i, \ldots, D_m$ partition $D \setminus \{x^L\}$. Furthermore, it can be seen that each element D_k has a similar form as that of D.

4.2 Candidate phase

In this section we provide an efficient method to compute the probabilities $\hat{P}(D_k)$ ($k = i, \ldots, m$). The method presented here is similar to the the method in [Nilsson 1998], in that it locally computes the probabilities of all candidates directly from the functions $f_{t,t+1}$.

A keypoint in proving our main result is that we can write the joint probability function expressed in (1) as

$$P(x,y) = \frac{\prod_{t=1}^{m-1} f_{t,t+1}(x_t, x_{t+1})}{\prod_{t=2}^{m-1} f_t(x_t)}. \quad (11)$$

For a proof, the interested reader is referred to [Dawid 1992], where it is shown for more general graphical models, the junction trees.

In addition, we will use that the functions $f_{t,t+1}$ have the following property, termed *max-consistency*:

$$\max_{x_{t+1}} f_{t,t+1}(x_t, x_{t+1}) = f_t(x_t). \quad (12)$$

Now we have

Lemma 1 *Let E be a subset given by*

$$E = \{x : x_1 = x_1^*, \ldots, x_i = x_i^*\}.$$

Then

$$\hat{P}(E) = \max_{x \in E} P(x,y) = \frac{\prod_{t=1}^{i-1} f_{t,t+1}(x_t^*, x_{t+1}^*)}{\prod_{t=2}^{i-1} f_t(x_t^*)}.$$

Proof: The probability $\hat{P}(E)$ can be found by instantiating $x_1 = x_1^*, \ldots, x_i = x_i^*$ in (11), and then maximize over the remaining variables. Because of max-consistency (12) this reduces to

$$\max_{x \in E} P(x,y) = \frac{\prod_{t=1}^{i-1} f_{t,t+1}(x_t^*, x_{t+1}^*)}{\prod_{t=2}^{i-1} f_t(x_t^*)},$$

which completes the proof. •

Lemma 2 *Let D be given as in (9), and suppose*

$$x^L = \arg\max_{x \in D} P(x, y).$$

Then for all $k = i, \ldots, m$ we have

$$\hat{P}(D) = \max_{x \in D} P(x, y) = \frac{\prod_{t=1}^{k-1} f_{t,t+1}(x_t^L, x_{t+1}^L)}{\prod_{t=2}^{k-1} f_t(x_t^L)}.$$

Proof: Let $k \geq i$, and define D' as

$$D' = \{x : x_1 = x_1^L, \ldots, x_k^L\}.$$

By Lemma 1 it suffices to prove that

$$\max_{x \in D'} P(x, y) = \max_{x \in D} P(x, y).$$

Clearly,

$$\max_{x \in D'} P(x, y) \leq \max_{x \in D} P(x, y)$$

since $D' \subseteq D$. Furthermore, we have that

$$\max_{x \in D'} P(x, y) \geq \max_{x \in D} P(x, y)$$

because $x^L \in D'$ and $x^L = \arg\max_{x \in D} P(x, y)$. The result follows. •

Now we are ready to state the main theorem. It presents a straightforward procedure for finding the probability of each subset D_k (defined in (10)) from the probability $\hat{P}(D)$.

Theorem 1 *Let D be given as in (9), and suppose*

$$x^L = \arg\max_{x \in D} P(x, y).$$

Then for all D_k defined in (10) we have:

$$\hat{P}(D_k) = \hat{P}(D) \frac{\max_{s \notin \bar{\mathcal{X}}_k} f_{k,k+1}(x_k^L, s)}{f_{k,k+1}(x_k^L, x_{k+1}^L)}.$$

Proof: By Lemma 2 we have

$$\hat{P}(D) = \frac{\prod_{t=1}^{k-1} f_{t,t+1}(x_t^L, x_{t+1}^L)}{\prod_{t=2}^{k-1} f_t(x_t^L)}, \tag{13}$$

Furthermore, since D_k can be written as

$$D_k = \cup_{s: s \notin \bar{\mathcal{X}}_k} \{x : x_1 = x_1^L, \ldots, x_{k-1} = x_{k-1}^L, x_k = s\}.$$

we have from Lemma 1 that

$$\hat{P}(D_k) = \frac{\prod_{t=1}^{k-2} f_{t,t+1}(x_t^L, x_{t+1}^L)}{\prod_{t=2}^{k-1} f_t(x_t^L)} \max_{s \notin \bar{\mathcal{X}}_k} f_{k-1,k}(x_{k-1}^L, s) \tag{14}$$

The result now follows from (13) and (14). •

4.3 Identification phase

Suppose that we are given a subset D of the form

$$D = \{x \mid x_1 = x_1', \ldots, x_{i-1} = x_{i-1}', x_i \notin \mathcal{X}_i'\}.$$

and want to identify the most likely state sequence, say x^*, in D. First, one note that

$$x_j^* = x_j' \text{ for } j = 1, \ldots, i-1.$$

Now the sequence $x_i^*, \ldots, x_m^*$ is successively found by

$$x_i^* = \arg\max_{s \notin \mathcal{X}_i'} f_{i-1,i}(x_{i-1}^*, s)$$

$$x_k^* = \arg\max_s f_{k-1,k}(x_{k-1}^*, s), \;\; k = i+1, \ldots, m.$$

4.4 The algorithm

A pseudo code for our N-best list algorithm is given below.

Procedure N-best list:

Step 1 (Identify x^1):

1. *Compute* the functions $f_{t,t+1}$;
2. *Identify* the most likely state sequences x^1 as shown in Section 3.1;

Step 2 (Identify x^2):

•PARTITION PHASE Partition $\mathcal{X} \setminus \{x^1\}$ in subsets $A_1, \ldots, A_m$ as in (4);

•CANDIDATE PHASE *Compute* $\hat{P}(A_i)$ as in Section 3.1.

•IDENTIFICATION PHASE

1. Set the Candidate List $\mathrm{CL} = \{\hat{P}(A_i)\}$;
2. *pick* the highest candidate, say $\hat{P}(A_i)$, in CL;
3. *identify* $x^2 \in A_i$ as in Section 3.1; .

Step 3 (Identify $x^3, \ldots, x^N$)

For $L = 2, \ldots, N-1$ do

•PARTITION PHASE**:**

1. Suppose $x^L \in D$;
2. *Partition* $D \setminus \{x^L\}$ in subsets $D_k, \ldots, D_m$ as in (10).

•CANDIDATE PHASE**:** *Compute* $\hat{P}(D_k), \ldots, \hat{P}(D_m)$ as in Theorem 1.

•IDENTIFICATION PHASE**:**

1. *Augment* the Candidate List with the new candidates:
$$\mathrm{CL} := \mathrm{CL} \cup \{\hat{P}(D_k), \ldots, \hat{P}(D_m)\};$$
2. *pick* the highest candidate, say $\hat{P}(D^*)$, from CL;
3. *identify* $x^{L+1} = \arg\max_{x \in D^*} P(x, y)$ as in Section 4.3;
4. *retract* $\hat{P}(D^*)$ from the Candidate List: $\mathrm{CL} := \mathrm{CL} \setminus \{\hat{P}(D^*)\}$.

END

Accordingly, the probability of the Lth most likely state sequence is either one of the candidates computed in the Lth step of the algorithm or it is one of the candidates computed earlier and still on the Candidate List. The search over all the candidates can be effectively performed in the following way. We can sort the candidates on the Candidate List with their associated subsets. In PARTITION PHASE we split the subsets that contained x^{L-1} (and hence was at the top of the list) into several subsets. In CANDIDATE PHASE we merge the new candidates into the sorted list according to their probabilities. Now the state sequence x^L belongs to the subset at the top of the updated list.

Because the algorithm finds the N-best list sequentially, we need not in forehand specify the number of N best hypothesis that is wanted. In stead, we may chose to find the N most likely hypothesis whose total probability is at least α, where $0 < \alpha \leq 1$.

5 Complexity issues

We would claim that our method can be implemented with lower complexity than traditional methods for finding the N-best list.

To support this, we would briefly discuss the computational complexity of our method by computing the number of elementary operations needed to perform the various tasks in the algorithm. Let

$$\gamma = \max_i |\mathcal{X}_i| = \text{the maximum number of elements that a variable can have.}$$

The operations performed by our algorithm can be divided into three parts

1 Computing the functions $f_{t,t+1}$ and f_t defined in (2).

and for each $L = 1, \ldots, N$:

2 In CANDIDATE PHASE, to compute the candidates as in Theorem 1.

3 In IDENTIFICATION PHASE, to compare the candidates on the Candidate List, and identify x^L.

Finding the functions in part 1 can be done by less than

$$\gamma^2 m \text{ operations.}$$

In CANDIDATE PHASE, we generate for each step L at most m new candidates. The computation of a candidate takes place via Theorem 1, and can be done with γ computaions, i.e. computing all candidates in step $L = 1, \ldots, N$ demands at most

$$\gamma m N \text{ computations.}$$

In IDENTIFICATION PHASE, we compare, in step L at most mL elements (since at most m candidates are generated in each step). If we store the elements in a 2-3 search tree, then the comparisons needed for inserting m new elements and update the search tree is $\mathrm{O}(m \log(mL))$. Doing this for all $L = 1, \ldots, N$, the total number of comparisons needed is in the order of

$$\mathrm{O}\left(\sum_{L=2}^{N} m \log(mL)\right) = \mathrm{O}\left(mN \log(mN)\right).$$

Finally, in IDENTIFICATION PHASE, we identify x^L as described in Section 4.3, and the number of comparisons needed is no larger than $m\gamma$. Thus the total number of comparisons cannot exceed

$$\gamma m N.$$

Adding up these terms, we obtain that the whole process of finding $x^1, \ldots, x^N$ is in the order of

$$\mathrm{O}\left(\gamma^2 m\right) + \mathrm{O}\left(m\gamma N\right) + \mathrm{O}\left(mN \log(mN)\right).$$

To compare, a straightforward implementation of the Viterbi search as described in [Forney (1973] uses in the order of

$$\mathrm{O}\left(\gamma^2 N m\right).$$

We conclude that for large γ, our method can yield significant savings of computational time when comparing to traditional Viterbi search.

6 Conclusion

We have presented in this paper a novel method to compute the N most likely state sequences in HMMs. The algorithm has two advantages compared to traditional methods. Firstly it can yield significant gain in computational time. Secondly, it is an anytime algorithm, and thus is also effective in cases where we do not know in advance how many solutions are needed. The main concept is to perform a small preprocessing computation and then we can produce the sequences in an incremental manner. We have concentrated in this paper on applications of the algorithm to speech recognition problems. The proposed algorithm, however, can be applied to many other sources of information that are organized in a hidden Markov model e.g. analysis of DNA sequences and real time robot navigation.

Acknowledgment

This research was supported by DINA (Danish Informatics Network in the Agricultural Sciences), funded by the Danish Research Councils through their PIFT programme.

References

Dawid, A. P. (1992). Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, **2**, 25–36.

Forney, G. (1973). The viterbi algorithm. *Proc. IEEE*, **61**, 268–78.

Nilsson, D. (1998). An efficient algorithm for finding the m most probable configurations in probabilistic expert systems. *Statistics and Computing*, **8**, 159–73.

Ostendorf, M. e. a. (1991). Integration of diverse recognition methodologies through reevaluation of n-best sentence hypotheses. In *Proceedings, DARPA Speech and Natural language Processing Workshop*, pp. 83–7.

Rabiner, L. (1989). A tutorial on hidden markov models and selected application in speech recognition. *Proceedings of the IEEE*, **37**, (2), 257–86.

Schwartz, R. and Chow, Y. (1991). A comparison of several approximate algorithms for finding multiple (n-best) sentence hypotheses. In *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 701–4.

Schwartz, R., Nguyen, L., and Makhoul, J. (1996). Multiple-pass search strategies. In *Automatic Speech and Speaker recognition*, pp. 429–56. Kluwer Academic Publishers.

Stolcke, A., Konig, Y., and Weintraub, M. (1997). Explicit word error minimization in n-best list rescoring. *Proc. EUROSPEECH*, **1**, 163–6.