[8] ——, "Information bounds and quick detection of parameter changes in stochastic systems," *IEEE Trans. Inf. Theory*, vol. 44, no. 7, pp. 2917–2929, Nov. 1998.
[9] ——, "Sequential multiple hypothesis testing and efficient fault detection-isloation in stochastic systems," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 595–608, Mar. 2000.
[10] E. L. Lehmann, *Testing Statistical Hypothesis*. New York: Wiley, 1959.
[11] G. Lorden, "Procedures for reacting to a change in distribution," *Ann. Math. Statist.*, vol. 42, pp. 1897–1908, 1971.
[12] Y. Mei, "Asymptotically optimal methods for sequential change-point detection," Ph.D. dissertation, Calif. Inst. Technol., Pasadena, CA, 2003.
[13] G. V. Moustakides, "Optimal stopping times for detecting changes in distributions," *Ann. Statist.*, vol. 14, pp. 1379–1387, 1986.
[14] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, pp. 100–115, 1954.
[15] M. Pollak, "Optimal detection of a change in distribution," *Ann. Statist.*, vol. 13, pp. 206–227, 1985.
[16] D. Siegmund, "The variance of one-sided stopping rules," *Ann. Math. Statist.*, vol. 40, pp. 1074–1077, 1969.
[17] ——, *Sequential Analysis: Tests and Confidence Intervals*. New York: Springer-Verlag, 1985.
[18] D. Siegmund and E. S. Venkatraman, "Using the generalized likelihood ratio statistics for sequential detection of a change-point," *Ann. Statist.*, vol. 23, pp. 255–271, 1995.
[19] A. N. Shiryayev, "On optimum methods in quickest detection problems," *Theory Probab. Appl.*, vol. 8, pp. 22–46, 1963.
[20] ——, *Optimal Stopping Rules*. New York: Springer-Verlag, 1978.
[21] A. G. Tartakovsky and V. V. Veeravalli, "An efficient sequential procedure for detecting changes in multichannel and distributed systems," in *Proc. 5th Int. Conf. Information Fusion*, vol. 2, Annapolis, MD, Jul. 2002, pp. 1–8.
[22] J. N. Tsitsiklis, "Extremal properties of likelihood ratio quantizers," *IEEE Trans. Commun.*, vol. 41, no. 4, pp. 550–558, Apr. 1993.
[23] V. V. Veeravalli, "Sequential decision fusion: Theory and applications," *J. Franklin Inst.*, vol. 336, pp. 301–322, Feb. 1999.
[24] ——, "Decentralized quickest change detection," *IEEE Trans. Inf. Theory*, vol. 47, no. 4, pp. 1657–1665, May 2001.
[25] V. V. Veeravalli, T. Basar, and H. V. Poor, "Decentralized sequential detection with a fusion center performing the sequential test," *IEEE Trans. Inf. Theory*, vol. 39, no. 2, pp. 433–442, Mar. 1993.
[26] D. Williams, *Probabilty with Martingles*. Cambridge, U.K.: Cambridge Univ. press, 1991.

Efficient Computation of the Hidden Markov Model Entropy for a Given Observation Sequence

Diego Hernando, Valentino Crespi, *Member, IEEE*, and George Cybenko, *Fellow, IEEE*

***Abstract*—Hidden Markov models (HMMs) are currently employed in a wide variety of applications, including speech recognition, target tracking, and protein sequence analysis. The Viterbi algorithm is perhaps the best known method for tracking the hidden states of a process from a sequence of observations. An important problem when tracking a process with an HMM is estimating the uncertainty present in the solution. In this correspondence, an algorithm for computing at runtime the entropy of the possible hidden state sequences that may have produced a certain sequence of observations is introduced. The brute-force computation of this quantity requires a number of calculations exponential in the length of the observation sequence. This algorithm, however, is based on a trellis structure resembling that of the Viterbi algorithm, and permits the efficient computation of the entropy with a complexity linear in the number of observations.**

***Index Terms*—Entropy, hidden Markov model (HMM), performance measurement, process query system, Viterbi algorithm.**

I. Introduction

Hidden Markov models (HMMs) are often used to find the most likely hidden state sequence that produces a given sequence of observations. This can be done with the well-known Viterbi algorithm. Possible performance measures in this scenario include the probability of error on a single state and the probability of error on the whole sequence. An alternative measure is the entropy of the possible *solutions* (state sequences) that explain a certain observation sequence.

The entropy of a random variable provides a measure of its uncertainty. The entropy of the state sequence that explains an observation sequence, given a model, can be viewed as the minimum number of bits that, on average, will be needed to encode the state sequence (given the model and the observations) [1]. The higher this entropy, the higher the uncertainty involved in tracking the hidden process with the current model.

In this correspondence, we introduce an efficient algorithm for computing at runtime the entropy of the hidden state sequence that explains a given observation sequence.

The remainder of this document is organized as follows: Section II gives a brief introduction to HMMs and specifies the notation used in this document. Section III describes the algorithm for efficiently computing the entropy at runtime, along with a numerical example and a brief analysis of the algorithm's performance, in terms of the number of operations required. Finally, Section IV contains the conclusions and a discussion of the usefulness of our algorithm.

Manuscript received February 18, 2004; revised March 27, 2005. This work was supported in part by ARDA under Grant F30602-03-C-0248, DARPA Projects F30602-00-2-0585 and F30602-98-2-0107, and the National Institute of Justice, Department of Justice Award number 2000-DT-CX-K001.

D. Hernando is with the Electrical and Computer Engineering Department, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA (e-mail: diego.hernando@ieee.org).

V. Crespi is with the Department of Computer Science at the California State University, Los Angeles, CA 90032-8150 USA (e-mail: vcrespi@calstatela.edu).

G. Cybenko is with the Thayer School of Engineering, Dartmouth College, Hanover, NH 03755 USA (e-mail: george.cybenko@dartmouth.edu).

Communicated by X. Wang, Associate Editor for Detection and Estimation.
Digital Object Identifier 10.1109/TIT.2005.850223

II. Preliminaries

An HMM is, in the words of Ephraim [2], a discrete-time finite-state homogeneous Markov chain observed through a discrete-time memoryless invariant channel. The channel is characterized by a finite set of transition densities indexed by the states of the Markov chain.

Formally, an HMM consists of the following elements.

- A Markov chain $\{S_i\}$, represented by an $N \times N$ stochastic matrix A, which describes the transition probabilities $a_{ij} = P(S_t = j | S_{t-1} = i)$ between the N states of the model, together with a probability distribution π, where $\pi_i = P(S_1 = i)$.
- A set of probability distributions, one for each hidden state, $b_i(o_j) = P(O_t = o_j | S_t = i)$, that model the emission of such observations. If there are M possible distinct observations, we accommodate the probability distributions b_i in the rows of an $N \times M$ matrix B. Note that M is not necessarily bounded.

Throughout this correspondence we will adopt the following notation.

- Subscripts will be used to identify a particular component in a sequence.
- Superscripts will be used to denote sequences of variables or symbols. For example, by S^t we will mean the sequence of t random variables $(S_1, S_2, \ldots, S_t)$.
- Capital letters will be used to denote random variables while lower case letters will denote specific symbols of a probability source.

III. Runtime Computation of the Entropy

A. Introduction

When we apply the Viterbi algorithm to find the most likely state sequence for a certain observation sequence, given the HMM parameters, we need some measure of the quality of the results that we obtain.

One possible measure of performance could be the probability of error on a single state [3]. However, since the Viterbi algorithm produces the most likely state sequence for the observations (even if each single state in the sequence is not the most likely at that particular time), it is reasonable to search for measures that involve the sequence as a whole rather than individual states.

After running the Viterbi algorithm, we obtain a "solution" sequence S^T, along with its likelihood P^*. We could compute the probability of this solution using Bayes rule, and use this probability as a measure of performance (the higher the probability of our solution, the better)

$$P(S^T = \hat{s}^T | O^T = o^T) = \frac{P(O^T = o^T | S^T = \hat{s}^T)\, P(S^T = \hat{s}^T)}{P(O^T = o^T)}. \quad (1)$$

The denominator can be efficiently computed using the solution to "Problem 1" in [4], while the numerator is equal to P^*.

However, $P(S^T = \hat{s}^T | O^T = o^T)$ alone does not provide a very good measure of the performance of our model with the current observation sequence. For example, it may be to our advantage to compute not only the most likely state sequence, but rather the k most likely sequences (as in the list Viterbi algorithm [5]).

But, how large should k be? We could easily keep track of the number of tracks with probability greater than zero, which may be useful for a very sparse transition matrix A. But even the probability of the k most likely sequences, plus the number of possible sequences, do not provide a very clear picture of how the algorithm is performing.

Alternatively, we propose to use the entropy of the distribution of all the possible solutions s^T, i.e.,

$$H(S^T | O^T = o^T) = -\sum_{s^T} \Big[P(S^T = s^T | O^T = o^T) \cdot \log(P(S^T = s^T | O^T = o^T)) \Big]. \quad (2)$$

The entropy of a random variable provides a measure of its uncertainty. In our case, the entropy of the state sequences that can possibly generate o^T gives a measure of how well our HMM parameters are suited to track a certain observation sequence.

For example if, according to our model, there is just one state sequence that could have produced a certain observation sequence, then the entropy associated with tracking these observations is 0, as there is no uncertainty in the *solution*. On the other hand, if all N^T possible state sequences are equally likely to produce the observations, then the entropy is maximized and equals $T \log N$.

A direct evaluation of (2) would be infeasible as there are N^T terms to add. However, in the next section, we provide an efficient algorithm based on a generalization of the Viterbi algorithm on a proper trellis structure.

B. Efficient Computation of $H(S^t | O^t = o^t)$

A Hidden Markov Model can be defined by the following parameters [4]:

N : number of distinct states;
M : number of distinct observations;
A : transition matrix (size $N \times N$);
B : observation emission matrix (size $N \times M$);
π : initial probability vector (length N).

This algorithm relies on basic properties of the entropy [1]. Let us consider the random variables X and Y

$$H(X, Y) = H(X) + H(Y|X) \quad (3)$$

$$H(Y|X) = \sum_{x \in \mathcal{X}} P(x) H(Y | X = x). \quad (4)$$

Our algorithm uses the following intermediate variables.

- $H_t(j) = H(S^{t-1} | S_t = j, O^t = o^t)$: entropy of all paths (state sequences) that lead to state j at time t, given the observations up to time t. For example, if there is just one possible path that leads to state j at time t, then $H_t(j) = 0$.
- $c_t(j) = P(S_t = j | O^t = o^t)$: probability of being in state j at time t, given the observations up to time t.

The entropy $H_t(j)$ can be computed recursively using the values from the previous step, $H_{t-1}(i), 1 \le i \le N$. Let us assume that we are on state j at time t. Then, we can divide the path into two segments: the first contains the sequence of states up to time $t-2$ (we will call this random variable Y), and the second contains just the state at time $t-1$ (we will call this X). Then, $H(X, Y) = H(X) + H(Y|X)$. $H(X)$ is the entropy associated with $P(S_{t-1} = i | S_t = j)$, which we can compute easily using $c_t(k), 1 \le k \le N$ and $c_{t-1}(l), 1 \le l \le N$. $H(Y|X)$ can be computed from $H_{t-1}(k),\ 1 \le k \le N$, using

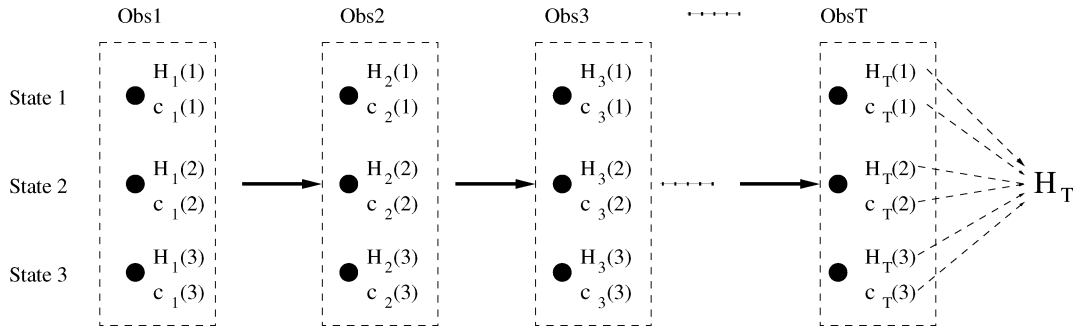

Fig. 1. Structure for the recursive computation of the entropy.

(4). Fig. 1 shows the structure of the intermediate variables used in the present algorithm.

Let us analyze the recursion in detail. First, note that $c_t(j)$ can be computed recursively [6]

$$c_t(j) = \frac{\sum_{i=1}^{N} c_{t-1}(i) a_{ij} b_j(o_t)}{\sum_{k=1}^{N} \sum_{i=1}^{N} c_{t-1}(i) a_{ik} b_k(o_t)}. \tag{5}$$

Next, for the recursion, we need the auxiliary probabilities $P(S_{t-1} = i | S_t = j, O^t = o^t)$. These can be computed as follows:

$$\begin{aligned}
&P(S_{t-1} = i | S_t = j, O^t = o^t) = \\
&\quad = P(S_{t-1} = i | S_t = j, O_t = o_t, O^{t-1} = o^{t-1}) \\
&\quad = \frac{P(S_t = j, O_t = o_t | S_{t-1} = i, O^{t-1} = o^{t-1})}{P(S_t = j, O_t = o_t | O^{t-1} = o^{t-1})} \\
&\qquad \cdot P(S_{t-1} = i | O^{t-1} = o^{t-1}) \\
&\quad = \frac{P(O_t = o_t | S_t = j) P(S_t = j | S_{t-1} = i)}{P(O_t = o_t | S_t = j) P(S_t = j | O^{t-1} = o^{t-1})} \\
&\qquad \cdot P(S_{t-1} = i | O^{t-1} = o^{t-1}) \\
&\quad = \frac{P(S_t = j | S_{t-1} = i) P(S_{t-1} = i | O^{t-1} = o^{t-1})}{\sum_{k=1}^{N} P(S_t = j | S_{t-1} = k) P(S_{t-1} = k | O^{t-1} = o^{t-1})} \\
&\quad = \frac{a_{ij} c_{t-1}(i)}{\sum_{k=1}^{N} a_{kj} c_{t-1}(k)}.
\end{aligned} \tag{6}$$

The recursion on the intermediate entropies can be derived as follows:

$$\begin{aligned}
H_t(j) &= H(S^{t-1} | S_t = j, O^t = o^t) \\
&= H(S^{t-2}, S_{t-1} | S_t = j, O^t = o^t) \\
&= H(S_{t-1} | S_t = j, O^t = o^t) \\
&\quad + H(S^{t-2} | S_{t-1}, S_t = j, O^t = o^t)
\end{aligned} \tag{7}$$

where

$$\begin{aligned}
&H(S_{t-1} | S_t = j, O^t = o^t) = \\
&\quad - \sum_{i=1}^{N} \big[P(S_{t-1} = i | S_t = j, O^t = o^t) \\
&\qquad \cdot \log(P(S_{t-1} = i | S_t = j, O^t = o^t)) \big]
\end{aligned}$$

and

$$\begin{aligned}
&H(S^{t-2} | S_{t-1}, S_t = j, O^t = o^t) = \\
&\quad = \sum_{i=1}^{N} \big[P(S_{t-1} = i | S_t = j, O^t = o^t) \\
&\qquad \cdot H(S^{t-2} | S_{t-1} = i, S_t = j, O^t = o^t) \big] \\
&\quad = \sum_{i=1}^{N} \big[P(S_{t-1} = i | S_t = j, O^t = o^t) H_{t-1}(i) \big] \ .
\end{aligned}$$

Lemma 1: The entropy of the state sequence up to time $t-2$, given the state at time $t-1$ and the observations up to $t-1$, is conditionally independent on the state and observation at time t

$$H(S^{t-2} | S_{t-1} = i, S_t = j, O^t = o^t) = H_{t-1}(i)$$

Proof:

$$\begin{aligned}
&H(S^{t-2} | S_{t-1} = i, S_t = j, O^t = o^t) = \\
&= H(S^{t-2} | S_{t-1} = i, O^{t-1} = o^{t-1}, S_t = j, O_t = o_t) \\
&= H(S^{t-2} | S_{t-1} = i, O^{t-1} = o^{t-1}) \\
&= H_{t-1}(i)
\end{aligned}$$

where the second step comes from the properties of HMMs: states S_{t-k}, $k \geq 2$, and S_t are statistically independent given S_{t-1}. The same applies to states S_{t-k}, $k \geq 2$, and observation O_t, given S_{t-1}. According to the basic properties of the entropy, $H(X|Y = y) = H(X)$ if X and Y are independent [1]. □

To finalize the algorithm, we need to compute $H(S^T | O^T = o^T)$, which can be expanded using the basic properties of the entropy

$$\begin{aligned}
H(S^T | O^T = o^T) &= H(S^{T-1} | S_T, O^T = o^T) + H(S_T | O^T = o^T) \\
&= \sum_{i=1}^{N} H_T(i) c_T(i) - \sum_{i=1}^{N} c_T(i) \log(c_T(i)).
\end{aligned} \tag{8}$$

This is the algorithm:

1. *Initialization.* For $1 \leq j \leq N$

$$H_1(j) = 0 \tag{9}$$

$$c_1(j) = \frac{\pi(j) b_j(o_1)}{\sum_{i=1}^{N} \pi(i) b_i(o_1)}. \tag{10}$$

2. *Recursion.* For $1 \leq j \leq N;\ 2 \leq t \leq T$

$$c_t(j) = \frac{\sum_{i=1}^{N} c_{t-1}(i) a_{ij} b_j(o_t)}{\sum_{k=1}^{N} \sum_{i=1}^{N} c_{t-1}(i) a_{ik} b_k(o_t)} \tag{11}$$

$$P(S_{t-1} = i | S_t = j, O^t = o^t) = \frac{a_{ij} c_{t-1}(i)}{\sum_{k=1}^{N} a_{kj} c_{t-1}(k)} \tag{12}$$

$$\begin{aligned}
H_t(j) &= \sum_{i=1}^{N} H_{t-1}(i) \, P(S_{t-1} = i | S_t = j, O^t = o^t) \\
&\quad - \sum_{i=1}^{N} \big[P(S_{t-1} = i | S_t = j, O^t = o^t) \\
&\qquad \cdot \log(P(S_{t-1} = i | S_t = j, O^t = o^t)) \big].
\end{aligned} \tag{13}$$

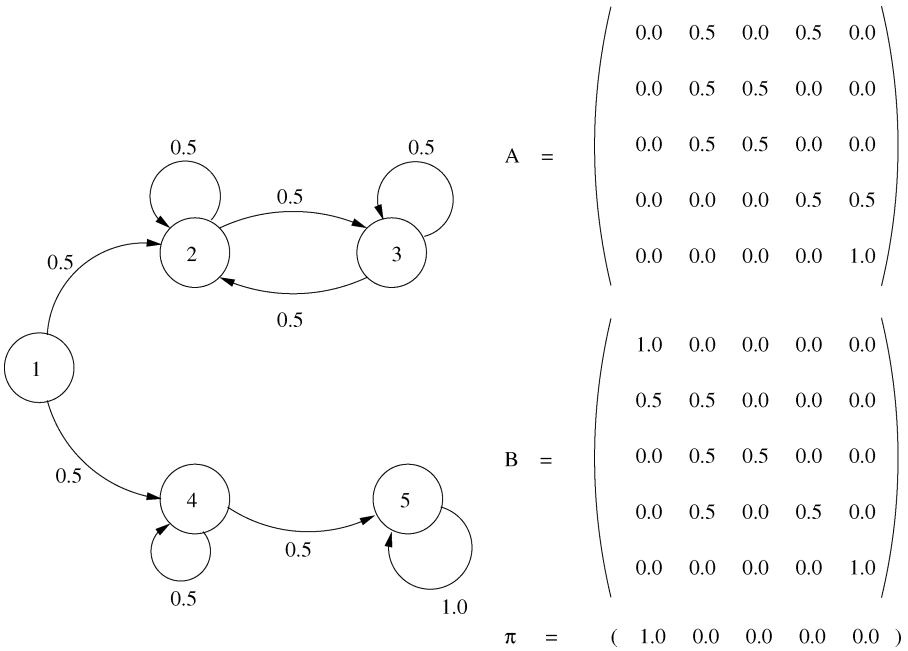

Fig. 2. Example set of parameters $\lambda$for an HMM with five hidden states and five different observation symbols ($N = M = 5$). The underlying Markov chain is described using a graph, which is equivalent to matrix A. The observation emission probabilities are contained in matrix B. The initial state probabilities are contained in vector π.

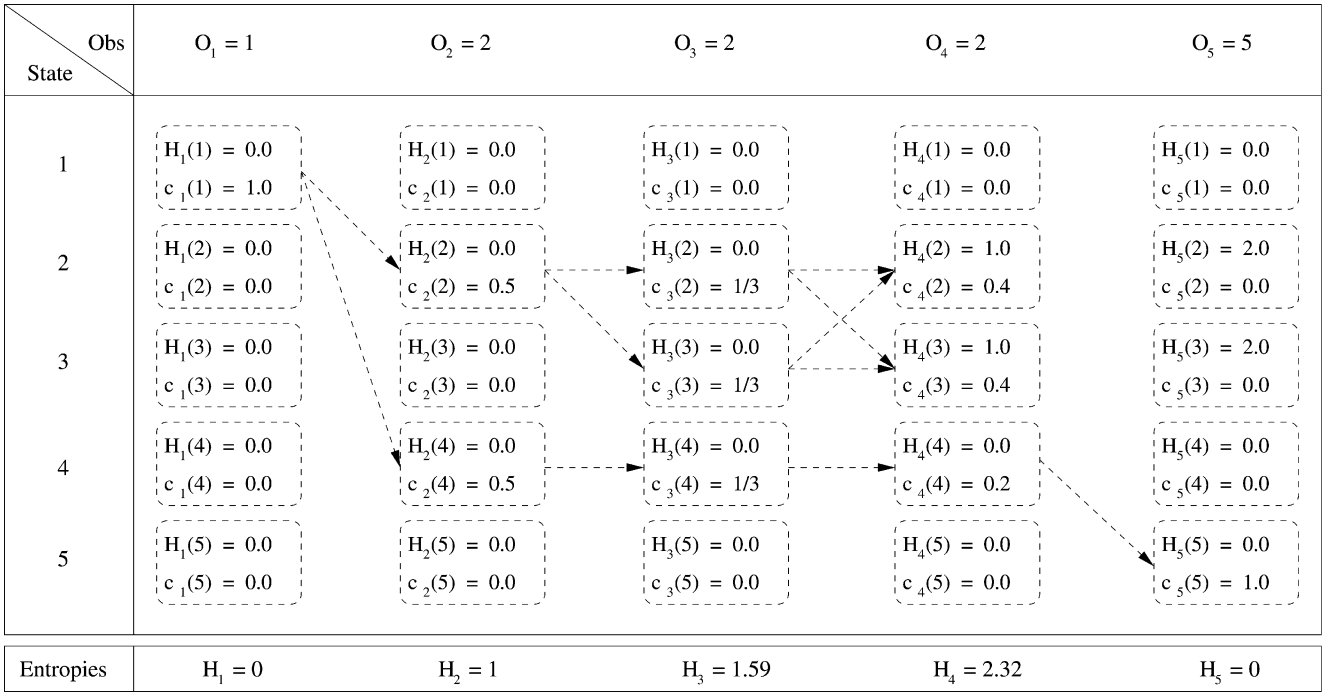

Fig. 3. Evolution of the trellis structure with the sequence of observations $o^5 = (1, 2, 2, 2, 5)$. The total entropy after each time step is displayed at the bottom of the figure. The arrows indicate the possible state transitions from time $t - 1$to t, given the model and the observations up to t.

3. *Termination*

$$H(S^T \mid O^T = o^T) = \sum_{i=1}^{N} H_T(i)\, c_T(i) - \sum_{i=1}^{N} c_T(i) \log(c_T(i)). \quad (14)$$

Fig. 2 shows an example set of parameters of an HMM with $N = M = 5$. Fig. 3 shows the evolution of the trellis structure for this HMM with a particular sequence of observations. The total entropy after each time step is displayed at the bottom of Fig. 3. For example, after receiving the second observation, there are two possible state sequences that could have produced $o^2 = (1, 2)$: $s^2 = (1, 2)$ and $s^2 = (1, 4)$, each with probability 0.5. Therefore, the entropy $H_2 = 1$. Also, when the fifth observation is received, all possible state sequences collapse onto $s^5 = (1, 4, 4, 4, 5)$, since there is no other state sequence that could have produced o^5. Therefore, this state sequence has probability 1 when $t = 5$, and the entropy $H_5 = 0$.

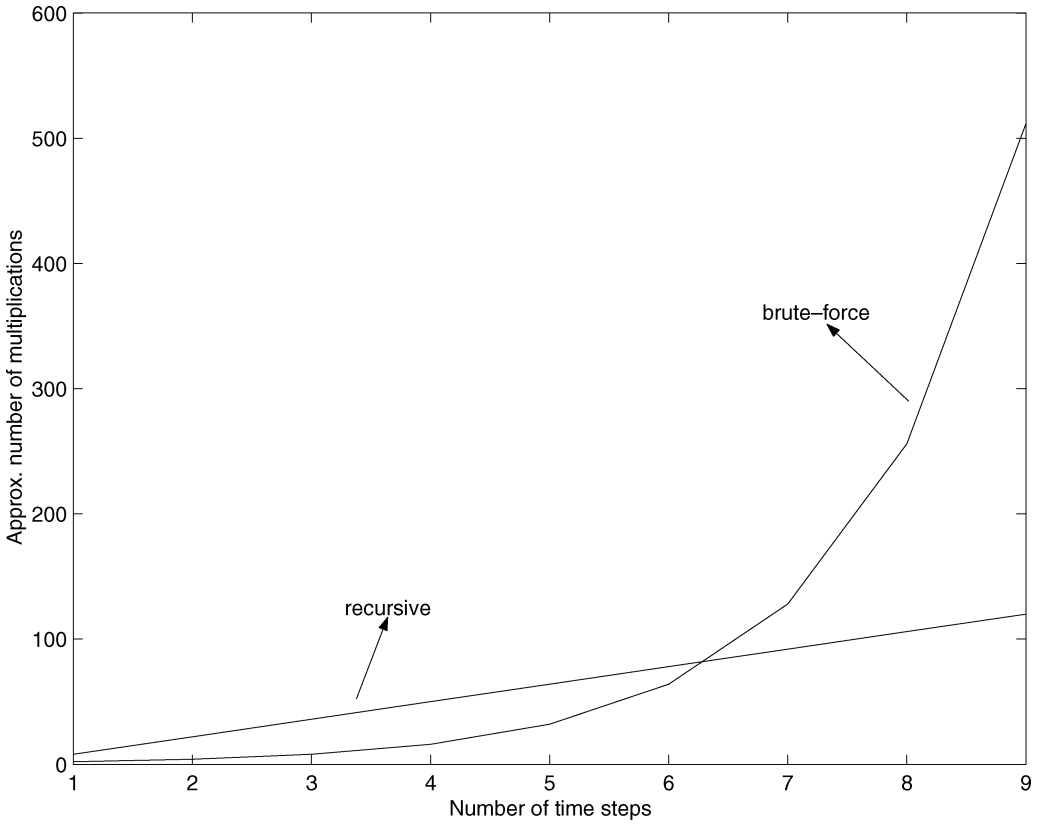

Fig. 4. Approximate number of multiplications required by each algorithm, depending on the number of time steps.

C. Performance

Fig. 4 shows a comparison of the performance of brute-force and recursive algorithms, for an example HMM where $N = M = 2$.

As in the case of the Viterbi algorithm [4], [7], this recursive entropy algorithm requires on the order of N^2T calculations, as opposed to the N^T calculations needed for the naïve approach.

IV. Conclusion

In this correspondence we have introduced an algorithm for computing the entropy of the possible state sequences that may produce a certain observation sequence, given an HMM. Our method is based on a trellis structure, similar to that of the Viterbi algorithm.

The intended use for this algorithm is to be running simultaneously with the Viterbi algorithm in a tracking system, so that after each observation is processed the user can have access to the (n) most likely state sequence(s) for the observations, as well as to the entropy of the distribution of possible state sequences. This measure allows the user to keep track of the performance of the current model.

Although we have defined this algorithm only for the case of discrete HMMs, extending it to the case with continuous observation emission probability densities (such as Gaussian mixtures) is straightforward.

This has been implemented in the Process Query System (PQS) framework [8], a generic architecture for process tracking and sensor information fusion. The kernel of such a system consists of a group of algorithms that can be used to track processes of very diverse nature, but which can be expressed in terms of abstract models such as HMMs. For example, our current system performs ground vehicle tracking and worm detection in computer networks. Generic performance measures such as the one presented in this correspondence are useful for providing measures of tracking performance that are independent of the particular domain in which the HMMs are employed.

References

[1] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York, NY: Wiley, 1991, Wiley Series in Telecommunications.

[2] Y. Ephraim and N. Merhav, "Hidden Markov processes," *IEEE Trans. Inf. Theory*, vol. 48, no. 6, pp. 1518–1569, Jun. 2002.

[3] J. Proakis and M. Salehi, *Communications Systems Engineering*. Upper Saddle River, NJ: Prentice-Hall, 2002.

[4] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[5] N. Seshadri and C. Sundberg, "List Viterbi decoding algorithms with applications," *IEEE Trans. Commun.*, vol. 42, no. 2/3/4, pp. 313–323, Feb./Mar./Apr. 1994.

[6] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. IT-20, no. 2, pp. 284–287, Mar. 1974.

[7] G. D. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.

[8] V. Berk *et al.*, "Process query systems for surveillance and awareness," in *Proc. 7th World Conf. Systemics, Cybernetics and Informatics (SCI2003)*, Orlando, FL, Jul. 2003.