[2] C. Møller, *The Theory of Relativity*. London, England: Oxford Univ. Press, 1952, pp. 274–276, 302–309.
[3] E. J. Post, *Formal Structure of Electromagnetics*. Amsterdam, The Netherlands: North-Holland, 1962, chs. 3, 6, 7.
[4] A. Sommerfeld, *Elektrodynamik*. Wiesbaden, Germany: Dieterich'sche Verlag, 1948, pp. 281–289.
[5] T. Schlomka, "Das Ohmsche Gesetz bei bewegten Körpern," *Ann. Phys.*, vol. 6, no. 8, pp. 246–252, 1951.
[6] H. Epheser and T. Schlomka, "Flächengrössen und elektrodynamische Grenzbedingungen bei bewegten Körpern," *Ann. Phys.*, vol. 6, no. 8, pp. 211–220, 1951.
[7] J. L. Anderson and J. W. Ryon, "Electromagnetic radiation in accelerated systems," *Phys. Rev.*, vol. 181, pp. 1765–1775, 1969.
[8] C. V. Heer, "Resonant frequencies of an electromagnetic cavity in an accelerated system of reference," *Phys. Rev.*, vol. 134, pp. A799–A804, 1964.
[9] T. C. Mo, "Theory of electrodynamics in media in noninertial frames and applications," *J. Math. Phys.*, vol. 11, pp. 2589–2610, 1970.
[10] R. M. Fano, L. J. Chu, and R. B. Adler, *Electromagnetic Fields, Energy and Forces*. New York: Wiley, 1960, pp. 407–416.
[11] L. Robin, *Fonctions Sphériques de Legendre et Fonctions Sphéroidales*, vol. 3. Paris, France: Gauthier-Villars, 1959, pp. 94–100.
[12] W. R. Symthe, *Static and Dynamic Electricity*. New York: McGraw-Hill, 1950, p. 208.
[13] M. Faraday, *Experimental Researches in Electricity*, vol. 3. New York: Dover, 1965, pp. 362–368.
[14] E. G. Cullwick, *Electromagnetism and Relativity*. London, England: Longmans, pp. 36–141, 167.

# The Viterbi Algorithm

G. DAVID FORNEY, JR.

*Invited Paper*

***Abstract*—The Viterbi algorithm (VA) is a recursive optimal solution to the problem of estimating the state sequence of a discrete-time finite-state Markov process observed in memoryless noise. Many problems in areas such as digital communications can be cast in this form. This paper gives a tutorial exposition of the algorithm and of how it is implemented and analyzed. Applications to date are reviewed. Increasing use of the algorithm in a widening variety of areas is foreseen.**

## I. Introduction

THE VITERBI algorithm (VA) was proposed in 1967 [1] as a method of decoding convolutional codes. Since that time, it has been recognized as an attractive solution to a variety of digital estimation problems, somewhat as the Kalman filter has been adapted to a variety of analog estimation problems. Like the Kalman filter, the VA tracks the state of a stochastic process with a recursive method that is optimum in a certain sense, and that lends itself readily to implementation and analysis. However, the underlying process is assumed to be finite-state Markov rather than Gaussian, which leads to marked differences in structure.

This paper is intended to be principally a tutorial introduction to the VA, its structure, and its analysis. It also purports to review more or less exhaustively all work inspired by or related to the algorithm up to the time of writing (summer 1972). Our belief is that the algorithm will find application in an increasing diversity of areas. Our hope is that we can accelerate this process for the readers of this paper.

*This invited paper is one of a series planned on topics of general interest—The Editor.*
Manuscript received September 20, 1972; revised November 27, 1972.
The author is with Codex Corporation, Newton, Mass. 02195.

## II. Statement of the Problem

In its most general form, the VA may be viewed as a solution to the problem of maximum *a posteriori* probability (MAP) estimation of the state sequence of a finite-state discrete-time Markov process observed in memoryless noise. In this section we set up the problem in this generality, and then illustrate by example the different sorts of problems that can be made to fit such a model. The general approach also has the virtue of tutorial simplicity.

The underlying Markov process is characterized as follows. Time is discrete. The state $x_k$ at time $k$ is one of a finite number $M$ of states $m$, $1 \leq m \leq M$; i.e., the state space $X$ is simply $\{1, 2, \cdots, M\}$. Initially we shall assume that the process runs only from time 0 to time $K$ and that the initial and final states $x_0$ and $x_K$ are known; the state sequence is then represented by a finite vector $\boldsymbol{x} = (x_0, \cdots, x_K)$. We see later that extension to infinite sequences is trivial.

The process is Markov, in the sense that the probability $P(x_{k+1} | x_0, x_1, \cdots, x_k)$ of being in state $x_{k+1}$ at time $k+1$, given all states up to time $k$, depends only on the state $x_k$ at time $k$:

$$P(x_{k+1} | x_0, x_1, \cdots, x_k) = P(x_{k+1} | x_k).$$

The transition probabilities $P(x_{k+1} | x_k)$ may be time varying, but we do not explicitly indicate this in the notation.

It is convenient to define the *transition* $\xi_k$ at time $k$ as the pair of states $(x_{k+1}, x_k)$:

$$\xi_k \triangleq (x_{k+1}, x_k).$$

We let $\Xi$ be the (possibly time-varying) set of transitions

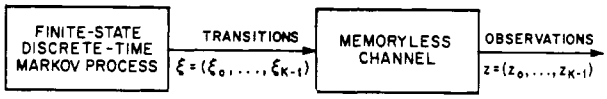


Fig. 1. Most general model.

$\xi_k = (x_{k+1}, x_k)$ for which $P(x_{k+1} | x_k) \neq 0$, and $|\Xi|$ their number. Clearly $|\Xi| \leq M^2$. There is evidently a one-to-one correspondence between state sequences $\boldsymbol{x}$ and transition sequences $\xi = (\xi_0, \cdots, \xi_{K-1})$. (We write $\boldsymbol{x} \underset{\leftrightarrow}{1-1} \xi$.)

The process is assumed to be observed in memoryless noise; that is, there is a sequence $z$ of observations $z_k$ in which $z_k$ depends probabilistically only on the transition $\xi_k$ at time $k$:[1]

$$P(z | x) = P(z | \xi) = \prod_{k=0}^{K-1} P(z_k | \xi_k).$$

In the parlance of information theory, $z$ can be described as the output of some memoryless channel whose input sequence is $\xi$ (see Fig. 1). Again, though we shall not indicate it explicitly, the channel may be time varying in the sense that $P(z_k | \xi_k)$ may be a function of $k$. This formulation subsumes the following as special cases.

1) The case in which $z_k$ depends only on the state $x_k$:

$$P(z | x) = \prod_k P(z_k | x_k).$$

2) The case in which $z_k$ depends probabilistically on an output $y_k$ of the process at time $k$, where $y_k$ is in turn a deterministic function of the transition $\xi_k$ or the state $x_k$.

*Example:* The following model frequently arises in digital communications. There is an input sequence $u = (u_0, u_1, \cdots)$, where each $u_k$ is generated independently according to some probability distribution $P(u_k)$ and can take on one of a finite number of values, say $m$. There is a noise-free signal sequence $y$, not observable, in which each $y_k$ is some deterministic function of the present and the $\nu$ previous inputs:

$$y_k = f(u_k, \cdots, u_{k-\nu}).$$

The observed sequence $z$ is the output of a memoryless channel whose input is $y$. We call such a process a *shift-register process*, since (as illustrated in Fig. 2) it can be modeled by a shift register of length $\nu$ with inputs $u_k$. (Alternately, it is a $\nu$th-order $m$-ary Markov process.) To complete the correspondence to our general model we define:

1) the state

$$x_k \triangleq (u_{k-1}, \cdots, u_{k-\nu})$$

2) the transition

$$\xi_k \triangleq (u_k, \cdots, u_{k-\nu}).$$

The number of states is thus $|X| = m^\nu$, and of transitions, $|\Xi| = m^{\nu+1}$. If the input sequence "starts" at time 0 and "stops" at time $K-\nu$, i.e.,

$$u = (\cdots, 0, u_0, u_1, \cdots, u_{K-\nu}, 0, 0, \cdots)$$

then the shift-register process effectively starts at time 0 and ends at time $K$ with $x_0 = x_K = (0, 0, \cdots, 0)$.

[1] The notation is appropriate when observations are discrete-valued; for continuous-valued $z_k$, simply substitute a density $p(z_k | \xi_k)$ for the distribution $P(z_k | \xi_k)$.

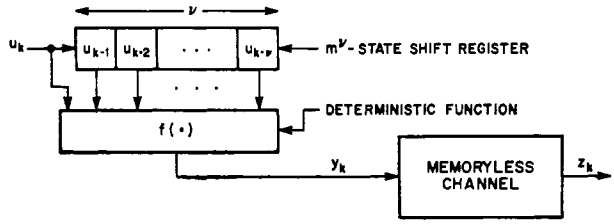


Fig. 2. Shift-register model.

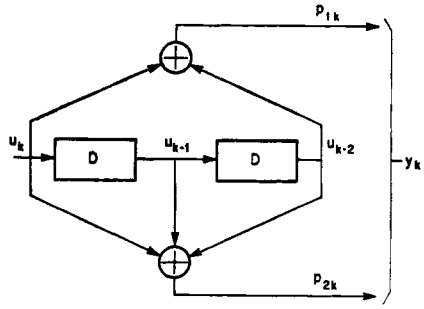


Fig. 3. A convolutional encoder.

Finally, we state the problem to which the VA is a solution. Given a sequence $z$ of observations of a discrete-time finite-state Markov process in memoryless noise, find the state sequence $\boldsymbol{x}$ for which the *a posteriori* probability $P(x | z)$ is maximum. Alternately, find the transition sequence $\xi$ for which $P(\xi | z)$ is maximum (since $\boldsymbol{x} \underset{\leftrightarrow}{1-1} \xi$). In the shift-register model this is also the same as finding the most probable input sequence $u$, since $u \underset{\leftrightarrow}{1-1} \boldsymbol{x}$; or also the most probable signal sequence $y$, if $y \underset{\leftrightarrow}{1-1} \boldsymbol{x}$. It is well known that this MAP rule minimizes the error probability in detecting the whole sequence (the block-, message-, or word-error probability), and thus is optimum in this sense. We shall see that in many applications it is effectively optimum in any sense.

*Application Examples:* We now give examples showing that the problem statement above applies to a number of diverse fields, including convolutional coding, intersymbol interference, continuous-phase frequency-shift keying (FSK), and text recognition. The adequately motivated reader may skip immediately to the next section.

### A. Convolutional Codes

A rate-$1/n$ binary convolutional encoder is a shift-register circuit exactly like that of Fig. 2, where the inputs $u_k$ are information bits and the outputs $y_k$ are blocks of $n$ bits, $y_k = (p_{1k}, \cdots, p_{nk})$, each of which is a parity check on (modulo-2 sum of) some subset of the $\nu+1$ information bits $(u_k, u_{k-1}, \cdots, u_{k-\nu})$. When the encoded sequence (codeword) $y$ is sent through a memoryless channel, we have precisely the model of Fig. 2. Fig. 3 shows a particular rate-$\frac{1}{2}$ code with $\nu = 2$. (This code is the only one ever used for illustration in the VA coding literature, but the reader must not infer that it is the only one the VA can handle.)

More general convolutional encoders exist: the rate may be $k/n$, the inputs may be nonbinary, and the encoder may even contain feedback. In every case, however, the code may be taken to be generated by a shift-register process [2].

We might also note that other types of transmission codes (e.g., dc-free codes, run-length-limited codes, and others) can be modeled as outputs of a finite-state machine and hence fall into our general setup [49].

### B. Intersymbol Interference

In digital transmission through analog channels, we frequently encounter the following situation. The input sequence $u$, discrete-time and discrete-valued as in the shift-register model, is used to modulate some continuous waveform which is transmitted through a channel and then sampled. Ideally, samples $z_k$ would equal the corresponding $u_k$, or some simple function thereof; in fact, however, the samples $z_k$ are perturbed both by noise and by neighboring inputs $u_{k'}$. The latter effect is called intersymbol interference. Sometimes intersymbol interference is introduced deliberately for purposes of spectral shaping, in so-called partial-response systems.

In such cases the output samples can often be modeled as

$$z_k = y_k + n_k$$

where $y_k$ is a deterministic function of a finite number of inputs, say, $y_k = f(u_k, \cdots, u_{k-\nu})$, and $n_k$ is a white Gaussian noise sequence. This is precisely Fig. 2.

To be still more specific, in pulse-amplitude modulation (PAM) the signal sequence $y$ may be taken as the convolution of the input sequence $u$ with some discrete-time channel impulse-response sequence $(h_0, h_1, \cdots)$:

$$y_k = \sum_i h_i u_{k-i}.$$

If $h_i = 0$ for $i > \nu$ (finite impulse response), then we obtain our shift-register model. An illustration of such a model in which intersymbol interference spans three time units ($\nu = 2$) appears in Fig. 4.

It was shown in [29] that even problems where time is actually continuous—i.e., the received signal $r(t)$ has the form

$$r(t) = \sum_{k=0}^{K} u_k h(t - kT) + n(t)$$

for some impulse response $h(t)$, signaling interval $T$, and realization $n(t)$ of a white Gaussian noise process—can be reduced without loss of optimality to the aforementioned discrete-time form (via a "whitened matched filter").

### C. Continuous-Phase FSK

This example is cited not for its practical importance, but because, first, it leads to a simple model we shall later use in an example, and, second, it shows how the VA may lead to fresh insight even in the most traditional situations.

In FSK, a digital input sequence $u$ selects one of $m$ frequencies (if $u_k$ is $m$-ary) in each signaling interval of length $T$; that is, the transmitted signal $\eta(t)$ is

$$\eta(t) = \cos\left[\omega(u_k)t + \theta_k\right], \qquad kT \le t < (k+1)T$$

where $\omega(u_k)$ is the frequency selected by $u_k$, and $\theta_k$ is some phase angle. It is desirable for reasons both of spectral shaping and of modulator simplicity that the phase be continuous at the transition interval; that is, that

$$\omega(u_{k-1})kT + \theta_{k-1} \equiv \omega(u_k)kT + \theta_k \text{ modulo } 2\pi.$$

This is called continuous-phase FSK.

The continuity of the phase introduces memory into the modulation process; i.e., it makes the signal actually transmitted in the $k$th interval dependent on previous signals. To take the simplest possible case ("deviation ratio" $= \frac{1}{2}$), let the input sequence $u$ be binary and let $\omega(0)$ and $\omega(1)$ be chosen so that $\omega(0)$ goes through an integer number of cycles in $T$ seconds and $\omega(1)$ through an odd half-integer number; i.e., $\omega(0)T \equiv 0$ and $\omega(1)T = \pi$ modulo $2\pi$. Then if $\theta_0 = 0$, $\theta_1 = 0$ or $\pi$, according to whether $u_0$ equals zero or one, and similarly $\theta_k = 0$ or $\pi$, according to whether an even or odd number of ones has been transmitted.

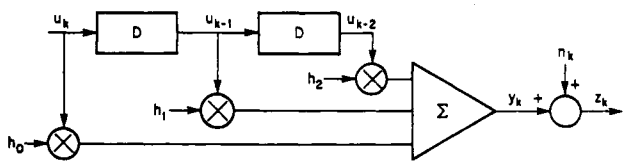


Fig. 4. Model of PAM system subject to intersymbol interference and white Gaussian noise.

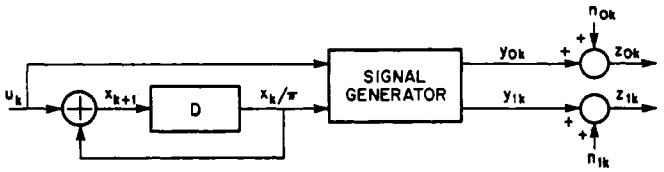


Fig. 5. Model for binary continuous-phase FSK with deviation ratio $\frac{1}{2}$ and coherent detection in white Gaussian noise.

Here we have a two-state process, with $X = \{0, \pi\}$. The transmitted signal $y_k$ is a function of both the current input $u_k$ and the state $x_k$:

$$y_k = \cos\left[\omega(u_k)t + x_k\right] = \cos x_k \cos \omega(u_k)t, \qquad kT \le t < (k+1)T.$$

Since transitions $\xi_k = (x_{k+1}, x_k)$ are one-to-one functions of the current state $x_k$ and input $u_k$, we may alternately regard $y_k$ as being determined by $\xi_k$.

If we take $\eta_0(t) \triangleq \cos \omega(0)t$ and $\eta_1(t) \triangleq \cos \omega(1)t$ as bases of the signal space, we may write

$$y_k = y_{0k}\eta_0(t) + y_{1k}\eta_1(t)$$

where the coordinates $(y_{0k}, y_{1k})$ are given by

$$(y_{0k}, y_{1k}) = \begin{cases} (1, 0), & \text{if } u_k = 0, x_k = 0 \\ (-1, 0), & \text{if } u_k = 0, x_k = \pi \\ (0, 1), & \text{if } u_k = 1, x_k = 0 \\ (0, -1), & \text{if } u_k = 1, x_k = \pi. \end{cases}$$

Finally, if the received signal $\xi(t)$ is $\eta(t)$ plus white Gaussian noise $\nu(t)$, then by correlating the received signal against both $\eta_0(t)$ and $\eta_1(t)$ in each signal interval (coherent detection), we may arrive without loss of information at a discrete-time output signal

$$z_k = (z_{0k}, z_{1k}) = (y_{0k}, y_{1k}) + (n_{0k}, n_{1k})$$

where $n_0$ and $n_1$ are independent equal-variance white Gaussian noise sequences. This model appears in Fig. 5, where the signal generator generates $(y_{0k}, y_{1k})$ according to the aforementioned rules.

### D. Text Recognition

We include this example to show that the VA is not limited to digital communication. In optical-character-recognition (OCR) readers, individual characters are scanned, salient

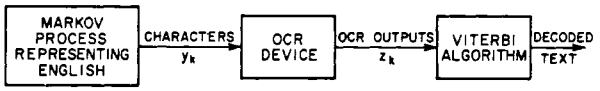


Fig. 6. Use of VA to improve character recognition by exploiting context.

features isolated, and some decision made as to what letter or other character lies below the reader. When the characters actually occur as part of natural-language text, it has long been recognized that contextual information can be used to assist the reader in resolving ambiguities.

One way of modeling contextual constraints is to treat a natural language like English as though it were a discrete-time Markov process. For instance, we can suppose that the probability of occurrence of each letter depends on the $\nu$ previous letters, and estimate these probabilities from the frequencies of $(\nu+1)$-letter combinations $[(\nu+1)$-grams$]$. While such models do not fully describe the generation of natural language (for examples of digram and trigram English, see Shannon [45], [46]), they account for a large part of the statistical dependencies in the language and are easily handled.

With such a model, English letters are viewed as the outputs of an $m^{\nu}$-state Markov process, where $m$ is the number of distinguishable characters, such as 27 (the 26 letters and a space). If it is further assumed that the OCR output $z_k$ is dependent only on the corresponding input character $y_k$, then the OCR reader is a memoryless channel to whose output sequence we may apply the VA to exploit contextual constraints; see Fig. 6. Here the "OCR output" may be anything from the raw sensor data, possibly a grid of zeros and ones, to the actual decision which would be made by the reader in the absence of contextual clues. Generally, the more raw the data, the more useful the VA will be.

### *E. Other*

Recognizing certain similarities between magnetic recording media and intersymbol interference channels, Kobayashi [50] has proposed applying the VA to digital magnetic recording systems. Timor [51] has applied the algorithm to a sequential ranging system. Use of the algorithm in source coding has been proposed [52]. Finally, Preparata and Ray [53] have suggested using the algorithm to search "semantic maps" in syntactic pattern recognition. These exhaust the applications known to the author.

## III. The Algorithm

We now show that the MAP sequence estimation problem previously stated is formally identical to the problem of finding the shortest route through a certain graph. The VA then arises as a natural recursive solution.

We are accustomed to associating with a discrete-time finite-state Markov process a state diagram of the type shown in Fig. 7(a), for a four-state shift-register process like that of Fig. 3 or Fig. 4 (in this case, a de Bruijn diagram [54]). Here nodes represent states, branches represent transitions, and over the course of time the process traces some path from state to state through the state diagram.

In Fig. 7(b) we introduce a more redundant description of the same process, which we have called a *trellis* [3]. Here each node corresponds to a distinct state at a given time, and each branch represents a transition to some new state at the next

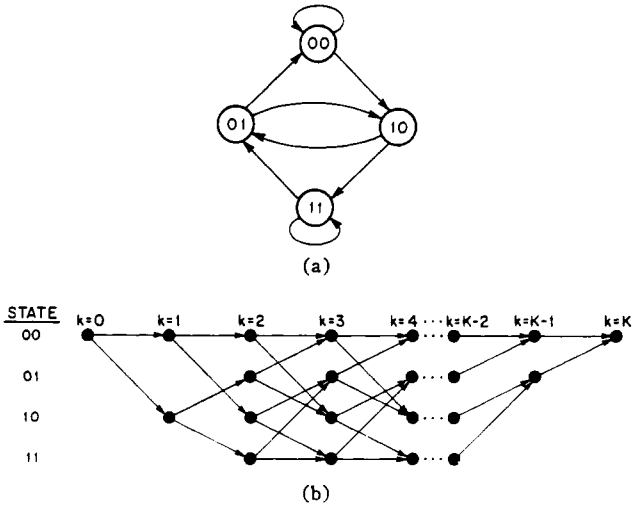


Fig. 7. (a) State diagram of a four-state shift-register process. (b) Trellis for a four-state shift-register process.

instant of time. The trellis begins and ends at the known states $x_0$ and $x_K$. Its most important property is that to every possible state sequence $\boldsymbol{x}$ there corresponds a unique path through the trellis, and vice versa.

Now we show how, given a sequence of observations $\boldsymbol{z}$, every path may be assigned a "length" proportional to $-\ln P(\boldsymbol{x}, \boldsymbol{z})$, where $\boldsymbol{x}$ is the state sequence associated with that path. This will allow us to solve the problem of finding the state sequence for which $P(\boldsymbol{x} \mid \boldsymbol{z})$ is maximum, or equivalently for which $P(\boldsymbol{x}, \boldsymbol{z}) = P(\boldsymbol{x} \mid \boldsymbol{z})P(\boldsymbol{z})$ is maximum, by finding the path whose length $-\ln P(\boldsymbol{x}, \boldsymbol{z})$ is minimum, since $\ln P(\boldsymbol{x}, \boldsymbol{z})$ is a monotonic function of $P(\boldsymbol{x}, \boldsymbol{z})$ and there is a one-to-one correspondence between paths and sequences. We simply observe that due to the Markov and memoryless properties, $P(\boldsymbol{x}, \boldsymbol{z})$ factors as follows:

$$\begin{aligned} P(\boldsymbol{x}, \boldsymbol{z}) &= P(\boldsymbol{x})P(\boldsymbol{z} \mid \boldsymbol{x}) \\ &= \prod_{k=0}^{K-1} P(x_{k+1} \mid x_k) \prod_{k=0}^{K-1} P(z_k \mid x_{k+1}, x_k). \end{aligned}$$

Hence if we assign each branch (transition) the "length"

$$\lambda(\xi_k) \triangleq -\ln P(x_{k+1} \mid x_k) - \ln P(z_k \mid \xi_k)$$

then the total length of the path corresponding to some $\boldsymbol{x}$ is

$$-\ln P(\boldsymbol{x}, \boldsymbol{z}) = \sum_{k=0}^{K-1} \lambda(\xi_k)$$

as claimed.

Finding the shortest route through a graph is an old problem in operations research. The most succinct solution was given by Minty in a quarter-page correspondence in 1957 [55], which we quote almost in its entirety:

> The shortest-route problem . . . can be solved very simply . . . as follows: Build a string model of the travel network, where knots represent cities and string lengths represent distances (or costs). Seize the knot "Los Angeles" in your left hand and the knot "Boston" in your right and pull them apart. If the model becomes entangled, have an assistant untie and re-tie knots until the entanglement is resolved. Eventually one or more paths will stretch tight—they then are alternative shortest routes. . . . It is well to label the knots since after one or two uses of the model their identities are easily confused.

Unfortunately, the Minty algorithm is not well adapted to modern methods of machine computation, nor are assistants as pliable as formerly. It therefore becomes necessary to move on to the VA, which is also well known in operations research [56]. It requires one additional observation.

We denote by $x_0^k$ a segment $(x_0, x_1, \cdots, x_k)$ consisting of the states to time $k$ of the state sequence $x=(x_0, x_1, \cdots, x_k)$. In the trellis $x_0^k$ corresponds to a path segment starting at the node $x_0$ and terminating at $x_k$. For any particular time-$k$ node $x_k$, there will in general be several such path segments, each with some length

$$\lambda(x_0^k) = \sum_{i=0}^{k-1} \lambda(\xi_i).$$

The shortest such path segment is called the *survivor* corresponding to the node $x_k$, and is denoted $\hat{x}(x_k)$. For any time $k>0$, there are $M$ survivors in all, one for each $x_k$. The observation is this: the shortest complete path $\hat{x}$ must begin with one of these survivors. (If it did not, but went through state $x_k$ at time $k$, then we could replace its initial segment by $\hat{x}(x_k)$ to get a shorter path—contradiction.)

Thus at any time $k$ we need remember only the $M$ survivors $\hat{x}(x_k)$ and their lengths $\Gamma(x_k) \triangleq \lambda[\hat{x}(x_k)]$. To get to time $k+1$, we need only extend all time-$k$ survivors by one time unit, compute the lengths of the extended path segments, and for each node $x_{k+1}$ select the shortest extended path segment terminating in $x_{k+1}$ as the corresponding time-$(k+1)$ survivor. Recursion proceeds indefinitely without the number of survivors ever exceeding $M$.

Many readers will recognize this algorithm as a simple version of forward dynamic programming [57], [58]. By this or any other name, the algorithm is elementary once the problem has been cast in the shortest route form.

We illustrate the algorithm for a simple four-state trellis covering 5 time units in Fig. 8. Fig. 8(a) shows the complete trellis, with each branch labeled with a length. (In a real application, the lengths would be functions of the received data.) Fig. 8(b) shows the 5 recursive steps by which the algorithm determines the shortest path from the initial to the final node. At each stage only the 4 (or fewer) survivors are shown, along with their lengths.

A formal statement of the algorithm follows:

*Viterbi Algorithm*

*Storage:*

| | |
|---|---|
| $k$ | (time index); |
| $\hat{x}(x_k)$, $1 \le x_k \le M$ | (survivor terminating in $x_k$); |
| $\Gamma(x_k)$, $1 \le x_k \le M$ | (survivor length). |

*Initialization:*

$$k = 0;$$
$$\hat{x}(x_0) = x_0; \quad \hat{x}(m) \text{ arbitrary}, \quad m \neq x_0;$$
$$\Gamma(x_0) = 0; \quad \Gamma(m) = \infty, \quad m \neq x_0.$$

*Recursion:* Compute

$$\Gamma(x_{k+1}, x_k) \triangleq \Gamma(x_k) + \lambda[\xi_k = (x_{k+1}, x_k)]$$

for all $\xi_k = (x_{k+1}, x_k)$.

Find

$$\Gamma(x_{k+1}) = \min_{x_k} \Gamma(x_{k+1}, x_k)$$

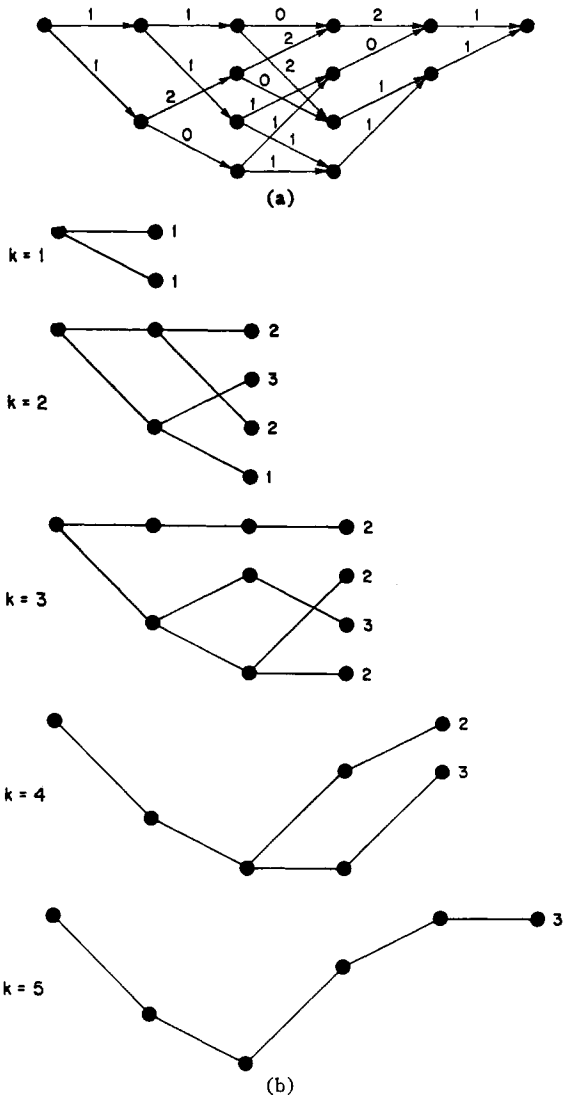


Fig. 8. (a) Trellis labeled with branch lengths; $M=4$, $K=5$. (b) Recursive determination of the shortest path via the VA.

for each $x_{k+1}$; store $\Gamma(x_{k+1})$ and the corresponding survivor $\hat{x}(x_{k+1})$.

Set $k$ to $k+1$ and repeat until $k=K$.

With finite state sequences $x$ the algorithm terminates at time $K$ with the shortest complete path stored as the survivor $\hat{x}(x_K)$.

Certain trivial modifications are necessary in practice. When state sequences are very long or infinite, it is necessary to truncate survivors to some manageable length $\delta$. In other words, the algorithm must come to a definite decision on nodes up to time $k-\delta$ at time $k$. Note that in Fig. 8(b) all time-4 survivors go through the same nodes up to time 2. In general, if the truncation depth $\delta$ is chosen large enough, there is a high probability that all time-$k$ survivors will go through the same nodes up to time $k-\delta$, so that the initial segment of the maximum-likelihood path is known up to time $k-\delta$ and can be put out as the algorithm's firm decision; in this case, truncation costs nothing. In the rare cases when survivors disagree, any reasonable strategy for determining the algorithm's time-$(k-\delta)$ decision will work [20], [21]: choose an arbitrary time-$(k-\delta)$ node, or the node associated with the shortest survivor, or a node chosen by majority vote,

etc. If $\delta$ is large enough, the effect on performance is negligible.

Also, if $k$ becomes large, it is necessary to renormalize the lengths $\Gamma(m)$ from time to time by subtracting a constant from all of them.

Finally, the algorithm may be required to get started without knowledge of the initial state $x_0$. In this case it may be initialized with any reasonable assignment of initial node lengths, such as $\Gamma(m)=0$, all $m$, or else $\Gamma(m)=-\ln \pi_m$ if the states are known to have *a priori* probabilities $\pi_m$. Usually, after an initial transient, there is a high probability that all survivors will merge with the correct path. Thus the algorithm synchronizes itself without any special procedures.

The complexity of the algorithm is easily estimated. First, memory: the algorithm requires $M$ storage locations, one for each state, where each location must be capable of storing a "length" $\Gamma(m)$ and a truncated survivor listing $\hat{x}(m)$ of $\delta$ symbols. Second, computation: in each unit of time the algorithm must make $|\Xi|$ additions, one for each transition, and $M$ comparisons among the $|\Xi|$ results. Thus the amount of storage is proportional to the number of states, and the amount of computation to the number of transitions. With a shift-register process, $M=m^{\nu}$ and $|\Xi|=m^{\nu+1}$, so that the complexity increases exponentially with the length $\nu$ of the shift register.

In the previous paragraph, we have ignored the complexity involved in generating the incremental lengths $\lambda(\xi_k)$. In a shift-register process, it is typically true that $P(x_{k+1}|x_k)$ is either $1/m$ or 0, depending on whether $x_{k+1}$ is an allowable successor to $x_k$ or not; then all allowable transitions have the same value of $-\ln P(x_{k+1}|x_k)$ and this component of $\lambda(\xi_k)$ may be ignored. Note that in more general cases $P(x_{k+1}|x_k)$ is known in advance; hence this component can be precomputed and "wired in." The component $-\ln P(z_k|\xi_k)$ is the only component that depends on the data; again, it is typical that many $\xi_k$ lead to the same output $y_k$, and hence the value $-\ln P(z_k|y_k)$ need be computed or looked up for all these $\xi_k$ only once, given $z_k$. (When the noise is Gaussian, $-\ln P(z_k|y_k)$ is proportional simply to $(z_k-y_k)^2$.) Finally, all of this can be done outside the central recursion ("pipelined"). Hence the complexity of this computation tends not to be significant.

Furthermore, note that once the $\lambda(\xi_k)$ are computed, the observation $z_k$ need not be stored further, an attractive feature in real-time applications.

A closer look at the trellis of a shift-register process reveals additional detail that can be exploited in implementation. For a binary shift register, the transitions in any unit of time can be segregated into $2^{\nu-1}$ disjoint groups of four, each originating in a common pair of states and terminating in another common pair. A typical such cell is illustrated in Fig. 9, with the time-$k$ states labeled $x'0$ and $x'1$ and the time-$(k+1)$ states labelled $0x'$ and $1x'$, where $x'$ stands for a sequence of $\nu-1$ bits that is constant within any one cell. For example, each time unit in the trellis of Fig. 8 is made up of two such cells. We note that only quantities within the same cell interact in any one recursion. Fig. 10 shows a basic logic unit that implements the computation within any one cell. A high-speed parallel implementation of the algorithm can be built around $2^{\nu-1}$ such identical logic units; a low-speed implementation, around one such unit, time-shared; or a software version, around the corresponding subroutine.

Many readers will have noticed that the trellis of Fig. 7(b) reminds them of the computational flow diagram of the fast

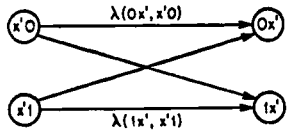


Fig. 9. Typical cell of a shift-register process trellis.

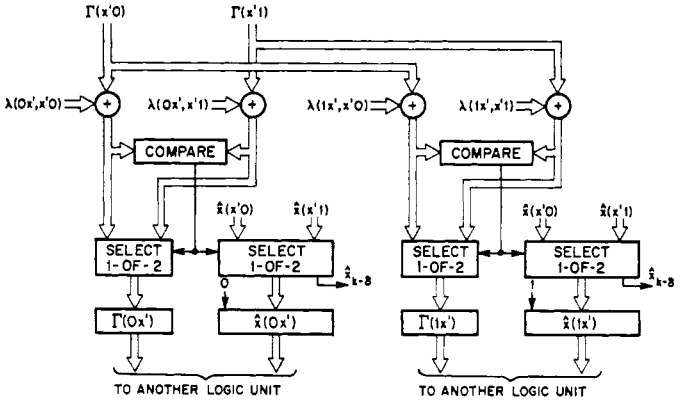


Fig. 10. Basic VA logic unit for binary shift-register process.

Fourier transform (FFT). In fact, it is identical, except for length, and indeed the FFT is also ordinarily organized cell-wise. While the add-and-compare computations of the VA are unlike those involved in the FFT, some of the memory-organization tricks developed for the FFT may be expected to be equally useful here.

Because of its highly parallel structure and need for only add, compare, and select operations, the VA is well suited to high-speed applications. A convolutional decoder for a $\nu=6$ code ($M=64$, $|\Xi|=128$) that is built out of 356 transistor–transistor logic circuits and that can operate at up to 2 Mbits/s perhaps represents the current state of the art [22]. Even a software decoder for a similar code can be run at a rate the order of 1000 bits/s on a minicomputer. Such moderate complexity qualifies the VA for inclusion in many signal-processing systems.

For further details of implementation, see [22], [23], and the references therein.

## IV. Analysis of Performance

Just as important as the straightforwardness of implementation of the VA is the straightforwardness with which its performance can be analyzed. In many cases, tight upper and lower bounds for error probability can be derived. Even when the VA is not actually implemented, calculation of its performance shows how far the performance of less complex schemes is from ideal, and often suggests simple suboptimum schemes that attain nearly optimal performance.

The key concept in performance analysis is that of an error event. Let $\boldsymbol{x}$ be the actual state sequence, and $\hat{\boldsymbol{x}}$ the state sequence actually chosen by the VA. Over a long time $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ will typically diverge and remerge a number of times, as illustrated in Fig. 11. Each distinct separation is called an error event. Error events may in general be of unbounded length if $\boldsymbol{x}$ is infinite, but the probability of an infinite error event will usually be zero.

The importance of error events is that they are probabilistically independent of one another; in the language of probability theory they are *recurrent*. Furthermore, they allow us to calculate error probability per unit time, which is neces-

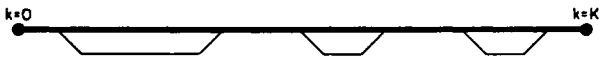


Fig. 11. Typical correct path $x$ (heavy line) and estimated path $\hat{x}$ (lighter line) in the trellis, showing three error events.

sary since usually the probability of *any* error in MAP estimation of a block of length $K$ goes to 1 as $K$ goes to infinity. Instead we calculate the probability of an error event starting at some given time, given that the starting state is correct, i.e., that an error event is not already in progress at that time.

Given the correct path $x$, the set $\mathcal{E}_k$ of all possible error events starting at some time $k$ is a treelike trellis which starts at $x_k$ and each of whose branches ends on the correct path, as illustrated in Fig. 12, for the trellis of Fig. 7(b). In coding theory this is called the incorrect subset (at time $k$).

The probability of any particular error event is easily calculated; it is simply the probability that the observations will be such that over the time span during which $\hat{x}$ is different from $x$, $\hat{x}$ is more likely than $x$. If the error event has length $\tau$, this is simply a two-hypothesis decision problem between two sequences of length $\tau$, and typically has a standard solution.

The probability $P(\mathcal{E}_k)$ that *any* error event in $\mathcal{E}_k$ occurs can then be upper-bounded, usually tightly, by a union bound, i.e., by the sum of the probabilities of all error events in $\mathcal{E}_k$. While this sum may well be infinite, it is typically dominated by one or a few large leading terms representing particularly likely error events, whose sum then forms a good approximation to $P(\mathcal{E}_k)$. True upper bounds can often be obtained by flow-graph techniques [4], [5], [29].

On the other hand, a lower bound to error-event probability, again frequently tight, can be obtained by a genie argument. Take the particular error event that has the greatest probability of all those in $\mathcal{E}_k$. Suppose that a friendly genie tells you that the true state sequence is one of two possibilities: the actual correct path, or the incorrect path corresponding to that error event. Even with this side information, you will still make an error if the incorrect path is more likely given $z$, so your probability of error is still no better than the probability of this particular error event. In the absence of the genie, your error probability must be worse still, since one of the strategies you have, given the genie's information, is to ignore it. In summary, the probability of any particular error event is a lower bound to $P(\mathcal{E}_k)$.

An important side observation is that this lower bound applies to any decision scheme, not just the VA. Simple extensions give lower bounds to related quantities like bit probability of error. If the VA gives performance approaching these bounds, then it may be claimed that it is effectively optimum with respect to these related quantities as well [30].

In conclusion, the probability of any error event starting at time $k$ may be upper- and lower-bounded as follows:

$$\max P(\text{error event}) \leq P(\mathcal{E}_k) \leq \max P(\text{error event}) + \text{other terms.}$$

With luck these bounds will be close.

*Example*

For concreteness, and because the result is instructive, we carry through the calculation for continuous-phase FSK of the particularly simple type defined earlier. The two-state trellis for this process is shown in Fig. 13, and the first part of

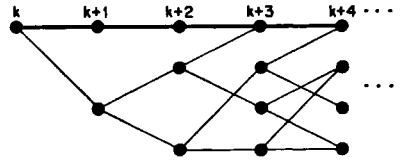


Fig. 12. Typical correct path $x$ (heavy line) and time-$k$ incorrect subset for trellis of Fig. 7(b).

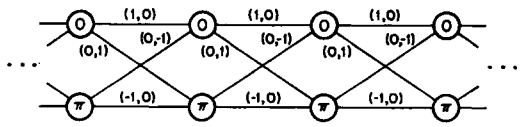


Fig. 13. Trellis for continuous-phase FSK.

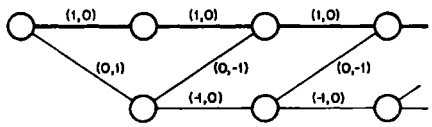


Fig. 14. Typical incorrect subset. Heavy line: correct path. Lighter lines: incorrect subset.

a typical incorrect subset in Fig. 14. The shortest possible error event is of length 2 and consists of a decision that the signal was $\{\cos \omega(1)t, -\cos \omega(1)t\}$ rather than $\{\cos \omega(0)t, \cos \omega(0)t\}$, or in our coordinate notation that $\{(0, 1), (0, -1)\}$ is chosen over $\{(1, 0), (1, 0)\}$. This is a two-hypothesis decision problem in a four-dimensional signal space [59]. In Gaussian noise of variance $\sigma^2$ per dimension, only the Euclidean distance $d$ between the two signals matters; in this case $d = \sqrt{2}$, and therefore the probability of this particular error event is $Q(d/2\sigma) = Q(1/\sigma\sqrt{2})$, where $Q(x)$ is the Gaussian error probability function defined by

$$Q(x) \triangleq \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy.$$

By examination of Fig. 14 we see that the error events of lengths 3, 4, $\cdots$ lie at distances $\sqrt{6}$, $\sqrt{10}$, $\cdots$ from the correct path; hence we arrive at upper and lower bounds on $P(\mathcal{E}_k)$ of

$$Q(\sqrt{2}/2\sigma) \leq P(\mathcal{E}_k) \leq Q(\sqrt{2}/2\sigma) + Q(\sqrt{6}/2\sigma) + Q(\sqrt{10}/2\sigma) + \cdots.$$

In view of the rapid decrease of $Q(x)$ with $x$, this implies that $P(\mathcal{E}_k)$ is accurately estimated as $Q(\sqrt{2}/2\sigma)$ for any reasonable noise variance $\sigma^2$. It is easily verified from Fig. 13 that this result is independent of the correct path $x$ or the time $k$.

This result is interesting in that the best one can do with coherent detection in a single symbol interval is $Q(1/2\sigma)$ for orthogonal signals. Thus exploiting the memory doubles the effective signal energy, or improves the signal-to-noise ratio by 3 dB. It may therefore be claimed that continuous-phase FSK is inherently 3 dB better than noncontinuous for deviation ratio $\frac{1}{2}$, or as good as antipodal phase-shift keying. (While we have proved this only for a deviation ratio of $\frac{1}{2}$, it holds for nearly any deviation ratio.) Even though the VA is quite simple for a two-state trellis, the fact that only one type of error event has any significant probability permits still simpler sub-

optimum schemes to achieve effectively the same performance [43], [44].[2]

## V. Applications

We conclude with a review of the results, both theoretical and practical, obtained with the VA to date.

### A. Convolutional Codes

It was for convolutional codes that the algorithm was first developed, and naturally it has had its greatest impact here.

The principal theoretical result is contained in Viterbi's original paper [1]; see also [5]–[7]. It shows that for a suitably defined ensemble of random trellis codes and MAP decoding, the error probability can be made to decrease exponentially with the constraint length $\nu$ at all code rates $R$ less than channel capacity. Furthermore, the rate of decrease is considerably faster than for block codes with comparable decoding complexity (although the same as that for block codes with the same decoding delay). In our view, the most telling comparison between block and convolutional codes is that an effectively optimum block code of any specified length and rate can be created by suitably terminating a convolutional (trellis) code, but with a lower rate for the block code of course [7]. In fact, if convolutional codes were any better, they could be terminated to yield better block codes than are theoretically possible—an observation which shows that the bounds on convolutional code performance must be tight.

For fixed binary convolutional codes of nonasymptotic length on symmetric memoryless channels, the principal result [6] is that $P(\mathcal{E}_k)$ is approximately given by

$$P(\mathcal{E}_k) \approx N_d 2^{-dD}$$

where $d$ is the free distance, i.e., the minimum Hamming distance of any path in the incorrect subset $\mathcal{E}_k$ from the correct path; $N_d$ is the number of such paths; and $D$ is the Bhattacharyya distance

$$D = \log_2 \sum_z P(z \mid 0)^{1/2} P(z \mid 1)^{1/2}$$

where the sum is over all outputs $z$ in the channel output space $Z$.

On Gaussian channels

$$P(\mathcal{E}_k) \approx N_d \exp(-dRE_b/N_0)$$

where $E_b/N_0$ is the signal-to-noise ratio per information bit. The tightness of this bound is confirmed by simulations [8], [9], [22]. For a $\nu=6$, $d=10$, $R=\frac{1}{2}$ code, for example, error probabilities of $10^{-3}$, $10^{-5}$, and $10^{-7}$ are achieved at $E_b/N_0 = 3.0$, 4.3, and 5.5 dB, respectively, which is within a few tenths of a dceibel of this bound [22].

Channels available for space communications are frequently accurately modeled as white Gaussian channels. The VA is attractive for such channels because it gives performance superior to all other coding schemes save sequential decoding, and does this at high speeds, with modest complexity, and with considerable robustness against varying channel parameters. A number of prototype systems have been implemented and tested [21]–[26], some quite original, and it seems likely that Viterbi decoders will become common in space communication systems.

Finally, Viterbi decoders have been used as elements in very-high-performance concatenated coding schemes [10]–[12], [27] and in decoding of convolutional codes in the presence of intersymbol interference [35], [60].

### B. Intersymbol Interference

Application of the VA to intersymbol interference problems is more recent, and the main achievements have been theoretical. The principal result, for PAM in white Gaussian noise, is that $P(\mathcal{E}_k)$ can be tightly bounded as follows:

$$K_L Q(d_{\min}/2\sigma) \leq P(\mathcal{E}_k) \leq K_U Q(d_{\min}/2\sigma)$$

where $K_L$ and $K_U$ are small constants, $Q(x)$ is the Gaussian error probability function defined earlier, $\sigma^2$ is the noise variance, and $d_{\min}$ is the minimum Euclidean distance between any two distinct signals [29]. This result implies that on most channels intersymbol interference need not lead to any significant degradation in performance, which comes as rather a surprise.

For example, with the most common partial response systems, the VA recovers the 3-dB loss sustained by conventional detectors relative to full-response systems [29], [32]. Simple suboptimum processors [29], [33] can do nearly as well.

Several workers [34], [35], [42] have proposed adaptive versions of the algorithm for unknown and time-varying channels. Ungerboeck [41], [42] and Mackechnie [35] have shown that only a matched filter rather than a whitened matched filter is needed in PAM.

It seems most likely that the greatest effect of the VA on digital modulation systems will be to reveal those instances in which conventional detection techniques fall significantly short of optimum, and to suggest effective suboptimum methods of closing the gap. PAM channels that cannot be linearly equalized without excessive noise enhancement due to nulls or near nulls in the transmission band are the likeliest candidates for nonlinear techniques of this kind.

### C. Text Recognition

An experiment was run in which digram statistics were used to correct garbled text produced by a simulated noisy character recognizer [47]. Results were similar to those of Raviv [48] (using digram statistics), although the algorithm was simpler and only had hard decisions rather than confidence levels to work with. It appears that the algorithm may be a useful adjunct to sophisticated character-recognition systems for resolving ambiguities when confidence levels for different characters are available.

## VI. Conclusion

The VA has already had a significant impact on our understanding of certain problems, notably in the theories of convolutional codes and of intersymbol interference. It is beginning to have a substantial practical impact as well in the engineering of space-communication links. The amount of work it has

[2] De Buda [44] actually proves that an optimum decision on the phase $x_k$ at time $k$ can be made by examining the received waveform only at times $k-1$ and $k$; i.e., $(z_{0,k-1}, z_{1,k-1})$, $(z_{0k}, z_{1k})$. The proof is that the log likelihood ratio

$$-\ln \frac{P(z_{0,k-1}, z_{1,k-1}, z_{0k}, z_{1k} \mid x_{k-1}, x_k = 0, x_{k+1})}{P(z_{0,k-1}, z_{1,k-1}, z_{0k}, z_{1k} \mid x_{k-1}, x_k = \pi, x_{k+1})}$$

is proportional to $-z_{0,k-1}+z_{1,k-1}-z_{0k}-z_{1k}$ for any values of the pair of states $(x_{k-1}, x_{k+1})$. For this phase decision (which differs slightly from our sequence decision) the error probability is exactly $Q(\sqrt{2}/2\sigma)$.

inspired in the intersymbol interference area suggests that here too practical applications are not far off. The generality of the model to which it applies and the straightforwardness with which it can be analyzed and implemented lead one to believe that in both theory and practice it will find increasing application in the years ahead.

## Appendix
## Related Algorithms

In this Appendix we mention some processing structures that are closely related to the VA, chiefly sequential decoding and minimum-bit-error-probability algorithms. We also mention extensions of the algorithm to generate reliability information, erasures, and lists.

When the trellis becomes large, it is natural to abandon the exhaustive search of the VA in favor of a sequential trial-and-error search that selectively examines only those paths likely to be the shortest. In the coding literature, such algorithms are collectively known as sequential decoding [13]–[16]. The simplest to explain is the "stack" algorithm [15], [16], in which a list is maintained of the shortest partial paths found to date, the path on the top of the list is extended, and its successors reordered in the list until some path is found that reaches the terminal node, or else decreases without limit. (That some path will eventually do so is ensured in coding applications by the subtraction of a bias term such that the length of the correct path tends to decrease while that of all incorrect paths tends to increase.) Searches that start from either end of a finite trellis [17] are also useful.

In coding applications, sequential decoding has many of the same properties as Viterbi decoding, including the same error probability. It allows the decoding of longer and therefore more powerful codes, at the cost of a variable amount of computation necessitating buffer storage for the incoming data $z$. It is probably less useful outside of coding, since it depends on the decoder's ability to recognize when the best path has been found without examining other paths, and therefore requires either a finite trellis or a very large distance between the correct path and possible error events.

In the intersymbol interference literature, many of the early attempts to find optimum nonlinear algorithms used bit-error probability as the optimality criterion. The Markov property of the process leads to algorithms that are manageable but less attractive than the Viterbi [36]–[42].

The general principle of several of these algorithms is as follows.[3] First, we calculate the joint probability $P(x_k, z)$ for every state $x_k$ in the trellis, or alternately $P(\xi_k, z)$ for every transition $\xi_k$. This is done by observing that

$$\begin{aligned} P(x_k, z) &= P(x_k, z_0^{k-1})P(z_k^K \mid x_k, z_0^{k-1}) \\ &= P(x_k, z_0^{k-1})P(z_k^K \mid x_k) \end{aligned}$$

since, given $x_k$, the outputs $z_k^K$ from time $k$ to $K$ are independent of the outputs $z_0^{k-1}$ from time 0 to $k-1$. Similarly

$$\begin{aligned} P(\xi_k, z) &= P(x_k, x_{k+1}, z) \\ &= P(x_k, z_0^{k-1})P(x_{k+1}, z_k \mid x_k, z_0^{k-1}) \\ &\quad \cdot P(z_{k+1}^K \mid x_{k+1}, x_k, z_0^k) \\ &= P(x_k, z_0^{k-1})P(x_{k+1}, z_k \mid x_k)P(z_{k+1}^K \mid x_{k+1}). \end{aligned}$$

[3] The author is indebted to Bahl *et al.* [18] for a particularly lucid exposition of this type of algorithm.

Now we note the recursive formula

$$\begin{aligned} P(x_k, z_0^{k-1}) &= \sum_{x_{k-1}} P(x_k, x_{k-1}, z_0^{k-1}) \\ &= \sum_{x_{k-1}} P(x_{k-1}, z_0^{k-2})P(x_k, z_{k-1} \mid x_{k-1}) \end{aligned}$$

which allows us to calculate the $M$ quantities $P(x_k, z_0^{k-1})$ from the $M$ quantities $P(x_{k-1}, z_0^{k-2})$ with $|\Xi|$ multiplications and additions using the exponentiated lengths

$$e^{-\lambda(\xi_{k-1})} = P(x_k \mid x_{k-1})P(z_{k-1} \mid \xi_{k-1}).$$

Similarly we have the backward recursion

$$\begin{aligned} P(z_k^K \mid x_k) &= \sum_{x_{k+1}} P(z_k^K, x_{k+1} \mid x_k) \\ &= \sum_{x_{k+1}} P(z_k, x_{k+1} \mid x_k)P(z_{k+1}^K \mid x_{k+1}) \end{aligned}$$

which has a similar complexity. Completion of these forward and backward recursions for all nodes allows $P(x_k, z)$ and/or $P(\xi_k, z)$ to be calculated for all nodes.

Now, to be specific, let us consider a shift-register process and let $S(u_k)$ be the set of all states $x_{k+1}$ whose first component is $u_k$. Then

$$P(u_k, z) = \sum_{x_{k+1} \in S(u_k)} P(x_{k+1}, z).$$

Since $P(u_k, z) = P(u_k \mid z)P(z)$, MAP estimation of $u_k$ reduces to finding the maximum of this quantity. Similarly, if we wish to find the MAP estimate of an output $y_k$, say, then let $S(y_k)$ be the set of all $\xi_k$ that lead to $y_k$ and compute

$$P(y_k, z) = \sum_{\xi_k \in S(y_k)} P(\xi_k, z).$$

A similar procedure can be used to estimate any quantity which is a deterministic function of states or transitions.

Besides requiring multiplications, this algorithm is less attractive than the VA in requiring a backward as well as a forward recursion and consequently storage of all data. The following amended algorithm [39], [48] eliminates the latter ugly feature at the cost of suboptimal performance and additional computation. Let us restrict ourselves to a shift-register process with input sequence $u$ and agree to use only observations up to time $k+\delta$ in estimating $u_k$, say, where $\delta \geq \nu - 1$. We then have

$$P(u_k, z_0^{k+\delta}) = \sum_{u_{k+1}} \cdots \sum_{u_{k+\delta}} P(u_k^{k+\delta}, z_0^{k+\delta}).$$

The $m^{\delta+1}$ quantities in the sum can be determined recursively by

$$\begin{aligned} P(u_k^{k+\delta}, z_0^{k+\delta}) &= \sum_{u_{k-1}} P(u_{k-1}^{k+\delta}, z_0^{k+\delta}) \\ &= \sum_{u_{k-1}} P(u_{k-1}^{k+\delta-1}, z_0^{k+\delta-1}) \\ &\quad \cdot P(u_{k+\delta}, z_{k+\delta} \mid u_{k+\delta-1}, \cdots, u_{k+\delta-\nu}). \end{aligned}$$

While the recursion is now forward only, we now must store $m^{\delta+1}$ quantities rather than $m^\nu$. If $\delta$ is large, this is most unattractive; if $\delta$ is close to $\nu-1$, the estimate may well be decidedly suboptimum.

These variations thus seem considerably less attractive than the VA. Nonetheless, something like this may need to be hybridized with the VA in certain situations such as track-

ing a finite-state source over a finite-state channel, where only the state sequence of the source is of interest.

Finally, we can consider augmented outputs from the VA. A good general indication of how well the algorithm is doing is the depth at which all paths are merged; this can be used to establish whether or not a communications channel is on the air, in synchronism, etc. [11], [28]. A more selective indicator of how reliable particular segments are is the difference in lengths between the best and the next-best paths at the point of merging; this reliability indicator can be quantized into an erasure output. Lastly, the algorithm can be altered to store the $L$ best paths, rather than the single best path, as the survivors in each recursion, thus eventually generating a list of the $L$ most likely path sequences.

## References


### *Convolutional Codes*

[1] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 260–269, Apr. 1967.

[2] G. D. Forney, Jr., "Convolutional codes I: Algebraic structure," *IEEE Trans. Inform. Theory*, vol. IT-16, pp. 720–738, Nov. 1970.

[3] ——, "Review of random tree codes," NASA Ames Res. Cen., Moffett Field, Calif., Contract NAS2-3637, NASA CR 73176, Final Rep., Dec. 1967, appendix A.

[4] G. C. Clark, R. C. Davis, J. C. Herndon, and D. D. McRae, "Interim report on convolution coding research," Advanced System Operation, Radiation Inc., Melbourne, Fla., Memo Rep. 38, Sept. 1969.

[5] A. J. Viterbi and J. P. Odenwalder, "Further results on optimal decoding of convolutional codes," *IEEE Trans. Inform. Theory* (Corresp.), vol. IT-15, pp. 732–734, Nov. 1969.

[6] A. J. Viterbi, "Convolutional codes and their performance in communication systems," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 751–772, Oct. 1971.

[7] G. D. Forney, Jr., "Convolutional codes II: Maximum likelihood decoding," Stanford Electronics Labs., Stanford, Calif., Tech. Rep. 7004-1, June 1972.

[8] J. A. Heller, "Short constraint length convolutional codes," in *Space Program Summary 37-54*, vol. III. Jet Propulsion Lab., Calif. Inst. Technol., pp. 171–177, Oct.–Nov. 1968.

[9] ——, "Improved performance of short constraint length convolutional codes," in *Space Program Summary 37-56*, vol. III. Jet Propulsion Lab., Calif. Inst. Technol., pp. 83–84, Feb.–Mar. 1969.

[10] J. P. Odenwalder, "Optimal decoding of convolutional codes," Ph.D. dissertation, Dep. Syst. Sci., Sch. Eng. Appl. Sci., Univ. of California, Los Angeles, 1970.

[11] J. L. Ramsey, "Cascaded tree codes," MIT Res. Lab. Electron., Cambridge, Mass., Tech. Rep. 478, Sept. 1970.

[12] G. W. Zeoli, "Coupled decoding of block-convolutional concatenated codes," Ph.D. dissertation, Dep. Elec. Eng., Univ. of California, Los Angeles, 1971.

[13] J. M. Wozencraft, "Sequential decoding for reliable communication," in *1957 IRE Nat. Conv. Rec.*, vol. 5, pt. 2, pp. 11–25.

[14] R. M. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inform. Theory*, vol. IT-9, pp. 64–74, Apr. 1963.

[15] K. S. Zigangirov, "Some sequential decoding procedures," *Probl. Pered. Inform.*, vol. 2, pp. 13–25, 1966.

[16] F. Jelinek, "Fast sequential decoding algorithm using a stack," *IBM J. Res. Develop.*, vol. 13, pp. 675–685, Nov. 1969.

[17] L. R. Bahl, C. D. Cullum, W. D. Frazer, and F. Jelinek, "An efficient algorithm for computing free distance," *IEEE Trans. Inform. Theory* (Corresp.), vol. IT-18, pp. 437–439, May 1972.

[18] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate" (Abstract), in *1972 Int. Symp. Information Theory* (Pacific Grove, Calif., Jan. 1972), p. 90.

[19] P. L. McAdam, L. R. Welch, and C. L. Weber, "M.A.P. bit decoding of convolutional codes," in *1972 Int. Symp. Information Theory* (Pacific Grove, Calif., Jan. 1972), p. 91.

### *Space Applications*

[20] Linkabit Corp., "Coding systems study for high data rate telemetry links, "Final Rep. on NASA Ames Res. Cen., Moffett Field, Calif., Contract NAS2-6024, Rep. CR-114278, 1970.

[21] G. C. Clark, "Implementation of maximum likelihood decoders for convolutional codes," in *Proc. Int. Telemetering Conf.* (Washington, D. C., 1971).

[22] J. A. Heller and I. M. Jacobs, "Viterbi decoding for satellite and space communication," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 835–847, Oct. 1971.

[23] G. C. Clark, Jr., and R. C. Davis, "Two recent applications of error-correction coding to communications system design," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 856–863, Oct. 1971.

[24] I. M. Jacobs and R. J. Sims, "Configuring a TDMA satellite communication system with coding, in *Proc. 5th Hawaii Int. Conf. Systems Science.* Honolulu, Hawaii: Western Periodicals, 1972, pp. 443–446.

[25] A. R. Cohen, J. A. Heller, and A. J. Viterbi, "A new coding technique for asynchronous multiple access communication," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 849–855, Oct. 1971.

[26] D. Quagliato, "Error correcting codes applied to satellite channels," in *1972 IEEE Int. Conf. Communications* (Philadelphia, Pa.), pp. 15/13–18.

[27] Linkabit Corp., "Hybrid coding system study," Final Rep. on Contract NAS2-6722, NASA Ames Res. Cen., Moffett Field, Calif., NASA Rep. CR114486, Sept. 1972.

[28] G. C. Clark, Jr., and R. C. Davis, "Reliability-of-decoding indicators for maximum likelihood decoders," in *Proc. 5th Hawaii Int. Conf. Systems Science.* Honolulu, Hawaii: Western Periodicals, 1927, pp. 447–450.

### *Intersymbol Interference*

[29] G. D. Forney, Jr., "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 363–378, May 1972.

[30] ——, "Lower bounds on error probability in the presence of large intersymbol interference," *IEEE Trans. Commun. Technol.* (Corresp.), vol. COM-20, pp. 76–77, Feb. 1972.

[31] J. K. Omura, "On optimum receivers for channels with intersymbol interference," abstract presented at the IEEE Int. Symp. Information Theory, Noordwijk, The Netherlands, June 1970.

[32] H. Kobayashi, "Correlative level coding and maximum-likelihood decoding," *IEEE Trans. Inform. Theory*, vol. IT-17, pp. 586–594, Sept. 1971.

[33] M. J. Ferguson, "Optimal reception for binary partial response channels," *Bell Syst. Tech. J.*, vol. 51, pp. 493–505, Feb. 1972.

[34] F. R. Magee, Jr., and J. G. Proakis, "Adaptive maximum-likelihood sequence estimation for digital signaling in the presence of intersymbol interference," *IEEE Trans. Inform. Theory* (Corresp.), vol. IT-19, pp. 120–124, Jan. 1973.

[35] L. K. Mackechnie, "Maximum likelihood receivers for channels having memory," Ph.D. dissertation, Dep. Elec. Eng., Univ. of Notre Dame, Notre Dame, Ind., Jan. 1973.

[36] R. W. Chang and J. C. Hancock, "On receiver structures for channels having memory," *IEEE Trans. Inform. Theory*, vol. IT-12, pp. 463–468, Oct. 1966.

[37] K. Abend, T. J. Harley, Jr., B. D. Fritchman, and C. Gumacos, "On optimum receivers for channels having memory," *IEEE Trans. Inform. Theory* (Corresp.), vol. IT-14, pp. 819–820, Nov. 1968.

[38] R. R. Bowen, "Bayesian decision procedures for interfering digital signals," *IEEE Trans. Inform. Theory* (Corresp.), vol. IT-15, pp. 506–507, July 1969.

[39] K. Abend and B. D. Fritchman, "Statistical detection for communication channels with intersymbol interference," *Proc. IEEE*, vol. 58, pp. 779–785, May 1970.

[40] C. G. Hilborn, Jr., "Applications of unsupervised learning to problems of digital communication," in *Proc. 9th IEEE Symp. Adaptive Processes, Decision, and Control* (Dec. 7–9, 1970).
C. G. Hilborn, Jr., and D. G. Lainiotis, "Optimal unsupervised learning multicategory dependent hypotheses pattern recognition," *IEEE Trans. Inform. Theory*, vol. IT-14, pp. 468–470, May 1968.

[41] G. Ungerboeck, "Nonlinear equalization of binary signals in Gaussian noise," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 1128–1137, Dec. 1971.

[42] ——, "Adaptive maximum-likelihood receiver for carrier-modulated data transmission systems," in preparation.

### *Continuous-Phase FSK*

[43] M. G. Pelchat, R. C. Davis, and M. B. Luntz, "Coherent demodulation of continuous-phase binary FSK signals," in *Proc. Int. Telemetry Conf.* (Washington, D. C., 1971).

[44] R. de Buda, "Coherent demodulation of frequency-shift keying with low deviation ratio," *IEEE Trans. Commun. Technol.*, vol. COM-20, pp. 429–435, June 1972.

### *Text Recognition*

[45] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication.* Urbana, Ill.: Univ. of Ill. Press, 1949.

[46] C. E. Shannon, "Prediction and entropy of printed English," *Bell Syst. Tech. J.*, vol. 30, pp. 50–64, Jan. 1951.

[47] D. L. Neuhoff, "The Viterbi algorithm as an aid in text recognition," Stanford Electronic Labs., Stanford, Calif., unpublished.

[48] J. Raviv, "Decision making in Markov chains applied to the problem of pattern recognition," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 536–551, Oct. 1967.

*Miscellaneous*

[49] H. Kobayashi, "A survey of coding schemes for transmission or recording of digital data," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 1087–1100, Dec. 1971.

[50] ——, "Application of probabilistic decoding to digital magnetic recording systems," *IBM J. Res. Develop.*, vol. 15, pp. 64–74, Jan. 1971.

[51] U. Timor, "Sequential ranging with the Viterbi algorithm," Jet Propulsion Lab., Pasadena, Calif., JPL Tech. Rep. 32-1526, vol. II, pp. 75–79, Jan. 1971.

[52] J. K. Omura, "On the Viterbi algorithm for source coding" (Abstract), in *1972 IEEE Int. Symp. Information Theory* (Pacific Grove, Calif., Jan. 1972), p. 21.

[53] F. P. Preparata and S. R. Ray, "An approach to artificial nonsymbolic cognition," *Inform. Sci.*, vol. 4, pp. 65–86, Jan. 1972.

[54] S. W. Golomb, *Shift Register Sequences.* San Francisco, Calif.: Holden-Day, 1967, pp. 13–17.

[55] G. J. Minty, "A comment on the shortest-route problem," *Oper. Res.*, vol. 5, p. 724, Oct. 1957.

[56] M. Pollack and W. Wiebenson, "Solutions of the shortest-route problem—A review," *Oper. Res.*, vol. 8, pp. 224–230, Mar. 1960.

[57] R. Busacker and T. Saaty, *Finite Graphs and Networks: An Introduction with Applications.* New York: McGraw-Hill, 1965.

[58] J. K. Omura, "On the Viterbi decoding algorithm," *IEEE Trans. Inform. Theory*, vol. IT-15, pp. 177–179, Jan. 1969.

[59] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering.* New York: Wiley, 1965, ch. 4.

[60] J. K. Omura, "Optimal receiver design for convolutional codes and channels with memory via control theoretical concepts," *Inform. Sci.*, vol. 3, pp. 243–266, July 1971.