Sequential Decoding: Computational Complexity and the Cutoff Rate

Lenny Grokop

May 13, 2005

Abstract

Sequential decoding algorithms decode convolutional codes by guessing their way through the expanding tree of possible transmitted sequences. In this way computational complexity is reduced. Generally it comes at the cost of having to communicate at rates distinctly below capacity. The computational cutoff rate R_{comp} delineates the border between those rates for which the average decoding time is bounded and those for which it is infinite. A series of papers establishes $R_{\text{comp}} = E_0(1)$, where $E_0(\rho)$ is the Gallager function.

1 Convolutional Codes

Convolutional codes use memory to spread the message digits over time in order to average out the channel noise. An example of a convolutional encoder is shown in figure (1). The encoder is defined by three parameters λ -the number of digits inputed at each time point, L -the constraint length and ν -the number of digits outputed at each time point. The rate of the code is defined as $R \triangleq \lambda/\nu \log(2)$ in nats. The constraint length plays the same role as the length of a block code. If the sequence of message digits is denoted

$$u_1^{(1)}, u_1^{(2)}, \ldots, u_1^{(\lambda)}, u_2^{(1)}, u_2^{(2)}, \ldots, u_2^{(\lambda)}, \ldots$$

then the sequence of channel digits outputed by the convolutional encoder

$$x_1^{(1)}, x_1^{(2)}, \ldots, x_1^{(\nu)}, x_2^{(1)}, x_2^{(2)}, \ldots, x_2^{(\nu)}, \ldots$$

is given by

$$x_n^{(i)} = \sum_{l=0}^{L-1} \sum_{j=1}^{\lambda} g_{j,i}(l) u_{n-l}^{(j)}, \quad 1 \leq i \leq \nu$$

where L is the constraint length of the encoder and $g_{j,i}(l)$ are the encoder taps.

2 Sequential decoding

Sequential decoding was proposed by Wozencraft [1] as a suboptimal method of decoding convolutional codes. The underlying idea is that the decoder retrieves the message sequence by guessing its way through the time-expanding tree of possible transmitted sequences. The complexity of such decoders is substantially better than MLSD enabling much larger constraint lengths to be used. For this reason, and for the purposes of later analysis, we will take L to be infinite in this report. In doing so we assume that the encoder and decoder have access to an infinite amount of common randomness with which they generate the sequence of taps $g_{j,i}(l)$.

2.1 Decoding illustration

Sequential decoding (with infinite constraint length) works as follows. Firstly, the encoder generates its taps randomly and independently as described above. Consider the convolutional encoder from figure (2) and its associated tree structure in figure (3). Suppose we encode the message sequence (1,0,0) as **111 010 100** but receive **001 010 100**. Upon reception of **001** the decoder selects the most likely transmitted symbol, a **0**, and heads into the tree in that direction. It stores the cost of this move as 1 -the hamming distance between the received 3 bits and the selected 3 bits. The next two received symbols lead it to decode two more zeros. These decisions, unavoidable as they are given the first choice of received symbol, cause the decoder's cost to rise at an unusually high rate. Sooner or later it realists this (depending on how quickly the decoder starts second guessing itself), and decides to turn back and try a different path. After it returns to the beginning and takes the upward path it has a clean run to the end and incurs a total Hamming cost of 1 for the decoded sequence. More importantly the decoder will correctly decode the sequence without examining all 8 paths.

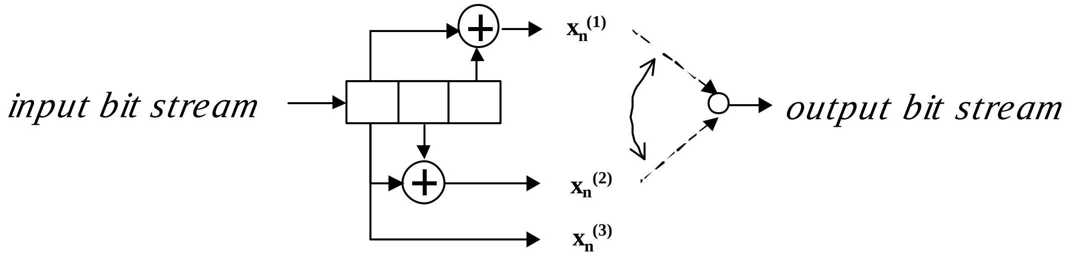

Figure 1: Convolutional encoder

The idea behind sequential decoding is that if the noise level is small relative to the rate of the code, the decoder can guess its way through the tree of possible received sequences -it need not examine every path in order to decode with decreasing probability of error. The decoder keeps a running metric designed to, on average, increase when it is travelling along the correct path, and decrease otherwise (in the above example the metric was simplified so that it increased quickly along false paths and slowly along true paths). At each tree depth it compares the metric to an adpating threshold and decides to either move forward, deeper into the tree, laterally to the nearest branch or backwards. At the same time it may raise or lower its threshold if it deems it to be too constraining.

2.2 The algorithm

The algorithm used nowadays is due to Fano [2]. We briefly describe it here. Our description is based on [3].

At each time point the decoder moves through the tree from node to node. It only moves in one of three directions, forwards (F), backwards (B) or laterally (L). It does not really need to keep track of the time, only its depth in the tree, which is denoted l. Forward moves increase l, backwards moves decrease l. Upon arriving at a new value of l the decoder computes the path metric Γ_l corresponding to the node it is currently on. Denoting the first νl digits of an encoded sequence by $x_1^{(1)}, \ldots, x_1^{(\nu)}, x_2^{(1)}, \ldots, x_l^{(\nu)}$ and the first νl digits of the received sequence by $y_1^{(1)}, \ldots, y_1^{(\nu)}, y_2^{(1)}, \ldots, y_l^{(\nu)}$ the path metric is given by

$$\Gamma_l = \sum_{n=1}^{l} \sum_{i=1}^{\nu} \left[\log \frac{P(y_n^{(i)} | x_n^{(i)})}{\omega(y_n^{(i)})} - B \right]. \tag{1}$$

In this expression B is an arbitrary bias term to be selected later and $\omega(j)$ is the nominal probability of the jth letter of the channel output alphabet

$$\omega(j) = \sum_{k=0}^{K-1} Q(k)P(j|k)$$

where $Q(k)$ is the relative frequency of the letter k in the mapping from binary digits to channel inputs. To save computation the path metric is computed recursively

$$\Gamma_{l+1} = \Gamma_l + \sum_{i=1}^{\nu} \left[\log \frac{P(y_{l+1}^{(i)}|x_{l+1}^{(i)})}{\omega(y_{l+1}^{(i)})} - B \right].$$

It turns out that the computational complexity of the decoder is relatively insensitive to B. For the BSC with crossover probability ϵ

$$\Gamma_{l+1} = \Gamma_l - d(\mathbf{x};\mathbf{y}) \ln \frac{1-\epsilon}{\epsilon} + \nu \ln[2(1-\epsilon)] - B$$

where $d(\mathbf{x};\mathbf{y})$ denotes the Hamming distance between the lth group of ν received bits and the hypothesized bits corresponding to the current node (refer to the example from the previous section). The path metric Γ_l is then compared to a threshold T and the decoder determines where to move to next, and how to update the threshold, based on the rule set listed in the table in figure 3.

Roughly speaking, the operational of the algorithm is such that with little noise rule 1 is the most common move. Severe noise causes the decoder to make regular use of the other rules. Under these conditions it will usually move forward a tad, increasing the threshold for a while, then retreat as it realizes it has been travelling down the wrong path. It will then oscillate using rules 3 and 4 so as to lower its threshold before heading down a new path.

Throughout the operation of the algorithm the decoder must keep track of the following variables.

1. The path metric Γ_{l-1} for the previous node at depth $l-1$.

2. The threshold T.

3. The previous move ('F', 'B' or 'L').

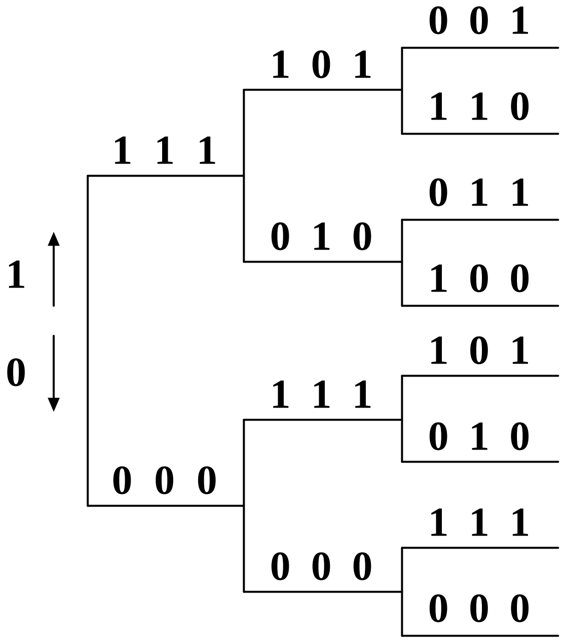

Figure 2: Decoding tree

	Node Conditions		Action	
	Previous Move	Metric Comparison	Final Threshold	Move
1	F or L	$\Gamma_{l-1} < T + \delta$, $\Gamma_l \geq T$	Raise*	F†
2	F or L	$\Gamma_{l-1} \geq T + \delta$, $\Gamma_l \geq T$	No change	F†
3	F or L	$\Gamma_l < T$	No change	L or B‡
4	B	$\Gamma_{l-1} < T$	Lower by Δ	F†
5	B	$\Gamma_{l-1} \geq T$	No change	L or B‡

Figure 3: Rule set for Fano decoding algorithm. *Add $j\delta$ to threshold, where j is chosen to satisfy $\Gamma_l - \delta < T + j\delta \leq \Gamma_l$. †Move forward to first node stemming from current node. ‡Move laterally to next node differing from current node only in the final branch. If current node is the last, move backward.

4. The estimate of the transmitted sequence. Note -this storage requirement is growing linearly in time.

5. Practical systems need a counter to help determine when to give up.

2.3 Error probability for two codewords

Before discussing the analysis of the above algorithm we present a result which lies at its heart. It concerns the probability that given a codebook consisting of only two codewords, a ML decoder will output one codeword when in actuality the other was transmitted. We follow [6] (pages 122-139).

Let $\mathbf{x_1}$ and $\mathbf{x_2}$ be two code words of length ν. Suppose message 1 was sent over a discrete memoryless channel and ML detection is used to decode. The probability of error $P_{e,1}$ can be written as the probability of receiving a particular sequence $\mathbf{y}$ conditioned on the transmitted message, summed over the set of received sequences that cause the decoder to choose message 2 –a set we denote by Y_1^c

$$P_{e,1} = \sum_{Y_1^c} P(\mathbf{y}|\mathbf{x_1}).$$

For any $0 < s < 1$ we can upper bound this expression by

$$P_{e,1} \leq \sum_{\mathbf{y}} P(\mathbf{y}|\mathbf{x_1})^{1-s} P(\mathbf{y}|\mathbf{x_2})^s,$$

where the sum is now over all $\mathbf{y}$. Now suppose $\mathbf{x_1}$ and $\mathbf{x_2}$ are chosen independently at random according to probability distribution $Q_\nu(\mathbf{x})$. The probability of confusing a particular codeword $\mathbf{x_1}$ with a particular codeword x_2, or vice versa is upper bounded by

$$P_e(\mathbf{x_1}, \mathbf{x_2}) \leq \sum_{\mathbf{y}} P(\mathbf{y}|\mathbf{x_1})^{1-s} P(\mathbf{y}|\mathbf{x_2})^s$$

for any $0 < s < 1$ so that the average error probability over the ensemble is given by

$$\begin{aligned} \overline{P_e} &= \sum_{\mathbf{x_1},\mathbf{x_2}} Q_\nu(\mathbf{x_1}) Q_\nu(\mathbf{x_2}) P_e(\mathbf{x_1}, \mathbf{x_2}) && (2) \\ &\leq \sum_{\mathbf{y}} \left[\sum_{\mathbf{x_1}} Q_\nu(\mathbf{x_1}) P(\mathbf{y}|\mathbf{x_1})^{1-s} \right] \left[\sum_{\mathbf{x_2}} Q_\nu(\mathbf{x_2}) P(\mathbf{y}|\mathbf{x_2})^s \right]. && (3) \end{aligned}$$

By symmetry, convexity and the fact that $\mathbf{x_1}$ and $\mathbf{x_2}$ are dummy variables, the minimum over s occurs at $s = 1/2$ and

$$\overline{P_e} \leq \sum_{\mathbf{y}} \left[\sum_{\mathbf{x}} Q_\nu(\mathbf{x}) \sqrt{P(\mathbf{y}|\mathbf{x})} \right]^2 .$$

For the discrete memoryless channel with input alphabet $(0, 1, ..., K-1)$ and output alphabet $(0, 1, ..., J-1)$, channel transition matrix $P(j|k)$ and codewords generated i.i.d according to $Q(k)$ this becomes

$$\overline{P_e} \leq \left(\sum_{j=0}^{J-1} \left(\sum_{k=0}^{K-1} Q(k) \sqrt{P(j|k)} \right)^2 \right)^\nu .$$

which is written as

$$\overline{P_e} \leq e^{-\nu E_0(1,\mathbf{Q})}$$

where

$$E_0(1, \mathbf{Q}) \triangleq -\ln \sum_{j=0}^{J-1} \left(\sum_{k=0}^{K-1} Q(k) \sqrt{P(j|k)} \right)^2 .$$

The bound can be tightened by maximizing $E_0(1, \mathbf{Q})$ over $\mathbf{Q}$. Define

$$E_0(1) \triangleq \max_{\mathbf{Q}} E_0(1, \mathbf{Q}).$$

Then we have

$$\overline{P_e} \leq \exp\{-\nu E_0(1)\}.$$

With this bound at hand we discuss the analysis of the sequential decoding algorithm.

2.4 Cutoff rate

A little thought reveals that if the noise level is too high relative to the rate of the code, the decoder will backtrack frequently and search through an increasing number of nodes in its quest to find the correct path. By the nature of its construction, the sequential decoder cannot avoid this effect.

It turns out though, for rates below a certain threshold R_{comp} known as the *computational cutoff rate* (or just the cutoff rate) the average amount of backtracking per bit decoded is bounded. Above the cutoff rate the decoder backtracks almost as often as it progresses forward, resulting in unbounded computational complexity. The parameter R_{comp} is a function of the channel and the noise level.

More specifically define the random variable W_n as the number of forward hypotheses required to decode the nth subblock. Then ([3] p279)

$$\overline{W}_n \leq 4\{1 - \exp[-\nu E_0(1) + \nu R]\}^{-2}.$$

The important thing to observe is that $\overline{W}_n$ is bounded for $R < R_{\text{comp}} = E_0(1)$. To get a feel for why this is the case let us examine the effect of the particular choice of metric in equation (1) on the decoding process. Suppose the decoder is on a particular node of depth n into the tree. Assume the depth $n-1$ node the decoder came from corresponds to the true path and denote this event by e^c_{n-1}. The decoder will proceed forward only if the likelihood that the corresponding source bits $\hat{u}^{(1)}_n, \ldots, \hat{u}^{(\lambda)}_n$ were transmitted given the received bits $y^{(1)}_n, \ldots, y^{(\nu)}_n$, is greater than the threshold T. The value of T should be thought of as roughly corresponding to the likelihood of the true sequence $u^{(1)}_n, \ldots, u^{(\lambda)}_n$ given the received bits. Thus the decoder will proceed down the current path provided

$$P\left(\hat{u}^{(1)}_n, \ldots, \hat{u}^{(\lambda)}_n \middle| \, y^{(1)}_n, \ldots, y^{(\nu)}_n\right) \geq P\left(u^{(1)}_n, \ldots, u^{(\lambda)}_n \middle| \, y^{(1)}_n, \ldots, y^{(\nu)}_n\right).$$

This probability is essentially the pairwise error probability between two codewords chosen independently at random and hence the probability of proceeding forward down a false branch of the tree is $\exp -\nu E_0(1)$. As there are $2^\lambda - 1 = \exp \nu R - 1 \approx \exp \nu R$ such false branches stemming from every node, applying the union bound, the probability of following a false path at depth n given the true path was followed at depth $n-1$ is

$$P(e_n | e^c_{n-1}) \leq \exp\{\nu(R - E_0(1))\}.$$

Thus

$$\begin{aligned} P(e_n) &= P(e_n|e^c_{n-1})P(e^c_{n-1}) + P(e_n|e_{n-1})P(e_{n-1}) \\ &\leq P(e_n|e^c_{n-1}) + P(e_{n-1}) \\ &\leq \exp\{\nu(R - E_0(1))\} + P(e_{n-1}) \\ \implies P(e_n) &\leq n \exp\{\nu(R - E_0(1))\}. \end{aligned}$$

Thus for $R < E_0(1)$, the probability of winding up on a false node at depth n goes to zero as $\nu \to \infty$, and we expect the average number of forward hypotheses required per decoded subblock to be small.

Arikan [4] was the first to prove the converse result in full generality: for $R > E_0(1)$

$$\lim_{N\to\infty} \overline{\left(\frac{1}{N}\sum_{n=0}^{N-1} W_n\right)} = \infty.$$

In doing so he completed the proof that the cutoff for sequential decoding $R_{\text{comp}} = E_0(1)$. For the BSC, the cutoff rate is given by the following expression

$$E_0(1) = 1 - 2\log_2[\sqrt{\epsilon} + \sqrt{1-\epsilon}]. \tag{4}$$

The capacity of the BSC is

$$C = 1 + \epsilon\log_2(\epsilon) + (1-\epsilon)\log_2(1-\epsilon). \tag{5}$$

These functions are plotted in figure (4) on the same graph for comparison.

One can ask a more general question about the statistics of the decoding time: below what rate is the ρth moment of W_n bounded? We denote this quantity $R_{\text{comp}}(\rho)$. The works of Falconer [5], Savage [6], Jelinek [7] and Hashimoto and Arimoto [8] established an achievability bound

$$R_{\text{comp}}(\rho) \geq E_0(\rho)/\rho, \quad \rho > 0.$$

Here $E_0(\rho)$ is defined as the maximum over all input distributions $\mathbf{Q}$ of the function

$$E_0(\rho, \mathbf{Q}) = -\log\sum_{j=0}^{J-1}\left[Q(k)P(j|k)^{1/(1+\rho)}\right]^{1+\rho}.$$

A converse result for channels where $E_0(\rho)$ is concave was given by Jacobs and Berlekamp [9]: for $R > E_0(\rho)/\rho$

$$\lim_{N\to\infty} \overline{\left(\frac{1}{N}\sum_{n=0}^{N-1} W_n\right)^{\rho}} = \infty.$$

The converse for arbitrary channels was not proved until much later (Arikan [10]). This established

$$R_{\text{comp}}(\rho) = E_0(\rho)/\rho.$$

2.5 Computational complexity

Although W_n is uniformly bounded for all n at rates below cutoff, the average number of computations per bit decoded is not. This is because the decoder must hypothesize increasingly long sequences and run them through its version of the convolutional encoder. For the same reason the computational complexity of the encoder is also unbounded with increasing n. Ultimately the average number of computations per bit decoded grows in time like $\mathcal{O}(n)$. This is a substantial improvement over the ML decoder which has computational complexity that grows in time like $\mathcal{O}(2^L) = O(2^n)$.

2.6 Guessing and its relationship to sequential decoding

It is interesting to examine a later paper by Arikan [11], that establishes the converse bound $R_{\text{comp}}(\rho) \leq E_0(\rho)/\rho$ using a simple but powerful lemma concerning the number of attempts one needs to guess the value of a random variable. Let (X, Y) be a pair of discrete random variables with X taking one of M possible values. Suppose the value of X is to be determined, given the value of Y, by asking questions of the form "Is X equal to x?" until the answer is "Yes". Let $G(x|y)$ denote the number of guesses in any such guessing scheme when $X = x, Y = y$. Then it can be shown that

$$E[G(X|Y)^\rho] \geq (1 + \log M)^{-\rho} \sum_y \left[\sum_x P_{X,Y}(x, y)^{\frac{1}{1+\rho}} \right]^{1+p} \tag{6}$$

for any $\rho \geq 0$. The trick in the proof of this result is to take the expression

$$E[G(X|Y)^\rho] = \sum_y P(y) \sum_x P(x) G(X = x|Y = y)^\rho \tag{7}$$

and bound it in such a way that the guessing r.v. term $G(X = x|Y = y)$ is isolated within the summation over x, i.e. $\sum_x f(G(X = x|Y = y))$. Because the enumeration of all possible values of $G(X = x|Y = y)$ is just the enumeration of all integers from 1 to M, we have

$$\sum_x f(G(X = x|Y = y)) = \sum_{i=1}^{M} f(i)$$

which does not depend on the specific guessing function G that is used. We can rewrite equation (6) as

$$E[G(X|Y)^{\rho}] \geq (1 + \log M)^{-\rho} \exp E_{\rho}(X|Y) \tag{8}$$

where

$$E_{\rho}(X|Y) \triangleq \sum_{y} \left[\sum_{x} P_{X,Y}(x,y)^{\frac{1}{1+\rho}} \right]^{1+p}.$$

Observe that the sequential decoding operation at a particular depth n involves guessing the correct node out of $M = \exp nR$ possibilities. Observe also that

$$E_{\rho}(X|Y) = \rho n R - E_0(\rho, P_X).$$

where P_X denotes the uniform distribution. It can be shown for any input distribution $\mathbf{Q}$ that $E_0(\rho, \mathbf{Q}) \leq N E_0(\rho)$ (see [13] theorem 5) and hence

$$E_{\rho}(X|Y) \geq \rho n R - N E_0(\rho).$$

So we have

$$E[G(X|Y)^{\rho}] \geq (1 + nR)^{-\rho} \exp[n(\rho R - E_0(\rho))] \tag{9}$$

and thus at rates $R > E_0(\rho)/\rho$, the ρth moment of computation performed at depth n of the tree must go to infinity exponentially as n is increased. Hence $R_{\text{comp}}(\rho) \leq E_0(\rho)/\rho$. This style of prove has also been used by Arikan to establish the cutoff rate region of sequential decoding for multiaccess channels [11] and to derive bounds to the complexity of sequential decoding in joint source-channel coding systems [12].

2.7 Simulations

Simulating the Fano algorithm provides a practical analysis that complements the theoretical one discussed in the preceding section. A convolutional code with infinite constraint, $\lambda = 1$ and $\nu = 15$ was used. The taps were generated uniformly at random from $\{0, 1\}$, and regenerated independently each simulation. The rate of the code was $R = 1/15 = 0.0667$. The input sequences consisted of 1000 binary digits selected uniformly at random from $\{0, 1\}$. The input sequences were regenerated independently each simulation.

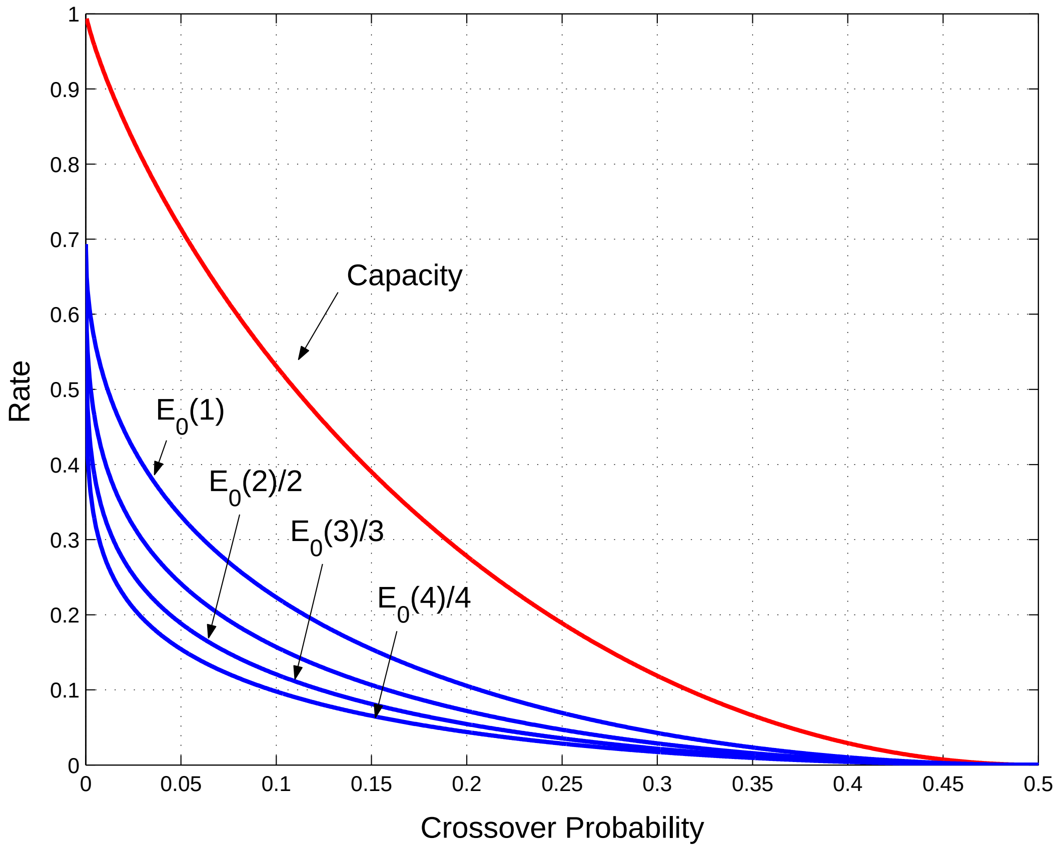

Figure 4: Cutoff rates versus capacity for the binary symmetric channel.

The channel was binary symmetric with crossover probability ϵ. A total of six simulations were run. In the first three $\epsilon = 0.2$ yielding $R < R_{\text{comp}} = 0.1054$. In the second three $\epsilon = 0.3$ yielding $R > R_{\text{comp}} = 0.0426$. The results are presented in figure 6 in the form of a graph plotting the depth the algorithm is at after a certain number of moves. Notice for $R < R_{\text{comp}}$ the algorithm finds the true path in roughly 4 times the number steps but for $R > R_{\text{comp}}$ it sooner or later gets stuck and begins to backtrack frequently.

3 Conclusion

The decoding of convolutional codes with large constraint lengths ($L > 20$) cannot be practically achieved using the Viterbi algorithm. Sequential algorithms such the one by Fano make this decoding possible as their average decoding time grows only linearly with the constraint length so long as the communication rate R is less than the cutoff rate of the channel $E_0(1)$, which is typically of the order of 1/2 the capacity. Furthermore, the ρth moment of the decoding time is bounded for $R < E_0(\rho)/\rho$. This indicates that in practical systems it may be wise to select a rate well below $E_0(1)$ in order to ensure the decoding delay is reasonably consistent.

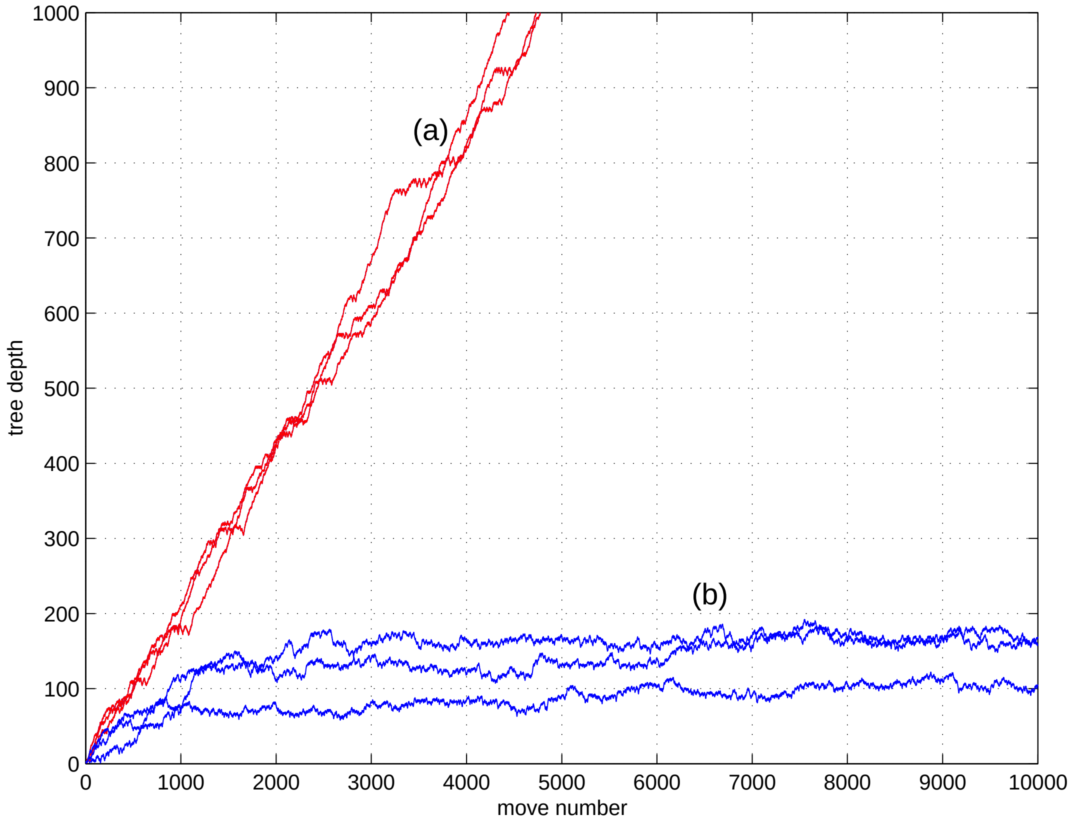

Figure 5: Tree depth versus the number of moves made by the decoder, for the Fano sequential decoding algorithm. In these simulations $\lambda = 1$ and $\nu = 15$. Thus $R = 1/15 = 0.0667$. The length of the input sequence is 1000. The channel is binary symmetric. For the three simulations in (a) the crossover probability is $\epsilon = 0.2$, for the three in (b) it is $\epsilon = 0.3$. Thus for the simulations in (a) $R_{\text{comp}} = 0.1054$ so that $R < R_{\text{comp}}$ but for the simulations in (b) $R_{\text{comp}} = 0.0426$ so that $R > R_{\text{comp}}$.

References

[1] J. M. Wozencraft, "Sequential Decoding for Reliable Communication," *1957 National IRE Convention Record*, **5**, Part 2, 11-25, 1957.

[2] R. M. Fano, "A Heuristic Discussion of Probabilistic Decoding," *IEEE Trans. Inform. Theory*, **IT-9**, 64-74, 1963.

[3] R. G. Gallager, *Information Theory and Reliable Communication*, Wiley, New York, 1968.

[4] E. Arikan, "An Upper Bound on the Cutoff Rate of Sequential Decoding," *IEEE Trans. Inform. Theory*, **IT-34**, 55-63, 1988.

[5] D. D. Falconer, "A hybrid coding scheme for discrete memoryless channels," *Bell Syst. Tech. J.*, vol. 48, pp. 691-728, Mar 1969.

[6] J. E. Savage, "Sequential decoding the computational problem," *Bell Syst. Tech. J.*, vol 45, pp. 149-175, 1966.

[7] F. Jelinek, "An upper bound on moments of sequential decoding effort," *IEEE Trans. Inform. Theory*, vol. IT-15, pp.140-149, Jan 1969.

[8] T. Hashimoto and S. Arimoto, "Computational moments for sequential decoding of convolutional codes," *IEEE Trans. Inform. Theory*, vol IT-25, pp 584-591, Sep. 1979.

[9] I. M. Jacobs and E. R. Berlekamp, "A Lower bound to the Distribution of Computation for Sequential Decoding," *IEEE Trans. Inform. Theory*, **IT-13**, 167-174, 1967.

[10] E. Arikan, "Lower bounds to moments of list size," in *Abstracts of Papers, IEEE Int. Symp. on Inform. Theory*, San Diego, 1990, pp 145-146.

[11] E. Arikan, "An inequality on guessing and its application to sequential decoding," *IEEE Trans. Inform. Theory*, vol. 42, No. 1, Jan 1996

[12] E. Arikan, "Joint Source-Channel Coding and Guessing with Application to Sequential Decoding," *IEEE Trans. Inform. Theory.*, Vol. 44, No. 5, Sep 1998.

[13] R. G. Gallager, "A Simple Derivation of the Coding Theorem and some Applications," *IEEE Trans. Inform. Theory*, **IT-11**, 3-18, 1965.