Choosing a Reliable Hypothesis

(Extended Abstract)

William Evans[*] **Sridhar Rajagopalan**[†] **Umesh Vazirani**[†]

Department of Computer Science,
University of California at Berkeley, CA 94720

Abstract

We study the problem of inferring an accurate model for a stochastic process from its output. We identify two desirable properties — resoluteness and reliability — of any identification algorithm. We prove that for any countable class of stochastic processes, there is an identification algorithm that has these properties. This result also formulates an optimization problem whose solution is sufficent to solve the identification problem. In this sense, our result provides an analogue to the Occam principle in a probabilistic setting.

1 Introduction

We consider the general problem of inferring an accurate model for a stochastic process from its output. For example, the stochastic process could be a dynamical system. The goal is to infer an accurate model for the system from a sequence of experimental observations made as the system evolves. Notice that we cannot assume any reliable reset on the underlying process. Therefore, the output sequence does not consist of independent observations but a single time series.

This situation is naturally expressed in an on-line setting as follows. We consider a countable class $\mathcal{M}$ of stochastic machines. We are given one of the machines $M^* \in \mathcal{M}$ in a black box. The machine M^* corresponds to the dynamical system in the example above. The output of M^* is a binary string $\mathbf{x} = x_1 x_2 \cdots$. This string corresponds to the time series of experimental observations. The goal is to infer the identity of M^* (or some $\hat{M} \in \mathcal{M}$ which is a close approximation to M^*) from $\mathbf{x}$.

The problem that we have sketched above is quite different from the corresponding problem of predicting $\mathbf{x}$ in an on-line fashion. The goal in the on-line prediction problem is to predict the next bit of $\mathbf{x}$ having seen $x_1 x_2 \cdots x_i$. For this problem, it is well known that for every class $\mathcal{M}$ of stochastic machines, there is a universal prediction mechanism $\mathcal{U}_{\mathcal{M}}$ usually called the Bayes' predictor for $\mathcal{M}$ that predicts almost optimally [5, 6, 10]. By contrast, we wish to find an $\hat{M} \in \mathcal{M}$ such that $\hat{M}$ is a good predictor for the sequence $\mathbf{x}$. An obvious reason to prefer such an hypothesis $\hat{M}$ over the Bayesian predictor $\mathcal{U}_{\mathcal{M}}$ is that $\mathcal{U}_{\mathcal{M}}$ is typically much more complex than any stochastic machine

*Supported in part by NSF grant CCR 92-01092.

†Supported by NSF PYI Award CCR 88-96202 and NSF grant IRI 91-20074.

$M \in \mathcal{M}$. Furthermore, returning to the example of inferring a dynamical system [3], our goal is not simply to predict the output of the system, but also to formulate an accurate model of the underlying mechanism. More generally, it can be argued that the fundamental goal of science is not just to predict, but also to posit a model that helps to understand the phenomenon being studied. Thus, any rigorous justification of the methodology of science must include an understanding of the identification problem. Such an interest in predictive models was also proposed in [4].

In the case that machines in $\mathcal{M}$ are all deterministic, there is a well known principle, Occam's Razor [1, 2], that states that any hypothesis that is consistent with the output, and whose description is "short" is a good approximation to the true machine - thus the criterion is to minimize $|M|$ among all consistent machines. In the stochastic setting, consistency is an extremely weak restriction. For example the machine that outputs the flips of a coin is consistent with every output.

A closely related setting in which the identification problem has been studied is the inference of finite state Markov sources with arbitrary (real) transition probabilities [9]. Rudich exhibited an algorithm that (in the limit) converges to the structure of the minimum state Markov chain that is equivalent to the target Markov chain. However, there do not appear to be explicit bounds on the number of mispredictions made by the working hypotheses of the algorithm (in a suitable on line model). On the other hand, [10] and [5] show how to predict as well as any Markov source in the limit using a very efficient algorithm based on Ziv-Lempel data compression [11]. However, they do not identify a good predictive hypothesis that is also a Markov source.

2 The Problem and Results

We are given a black box that contains an unknown randomized machine M^* chosen from some countable class $\mathcal{M}$. $\mathbf{x}$ is a string that is generated by the machine (M^*) in the black box. Our goal is to identify a machine $\hat{M} \in \mathcal{M}$ such that the output distribution of $\hat{M}$ is the same as the output distribution of M^*. Notice that if $\mathbf{x}$ began with a 0, then we get no information about the output distribution on strings starting with a 1. This is a serious problem if there are machines whose output distributions are identical conditioned on the first output bit being 0, and different conditioned on the first output bit being 1. One way of dealing with this problem is to place further restrictions on $\mathcal{M}$ and retain the goal of identifying a machine $\hat{M}$ equivalent to M^*. Instead, we will define a weaker criterion that an identification algorithm should satisfy — resoluteness and reliability. We will prove that this weaker goal can be achieved for any class of stochastic machines $\mathcal{M}$. Further, it is possible to show that the weaker goal implies the stronger one for any class $\mathcal{M}$ that satisfies a reasonable restriction.

An *identification algorithm* outputs a hypothesis machine $G(x_1 \cdots x_i) \in \mathcal{M}$ after observing the first i bits of output from the black box. We require that the expected number of times the identification algorithm changes its hypothesis is finite. Here the expectation is taken over the distribution on $\mathbf{x}$, the output of the black box. This distribution depends on the choice of M^*. Thus the expectation should be finite regardless of the choice of M^*. We shall call algorithms that have this property *resolute.* This formalizes our notion of identification. It implies that with probability 1, after a finite number of outputs the identification algorithm will never change its hypothesis. However, we cannot demand that the identification algorithm "know" when it has converged, since the distribution of two machines in $\mathcal{M}$ may agree on arbitrarily long strings.

We now formalize what it means for the algorithm to identify a *reliable* model of M^*: We imagine running a prediction tournament between the true machine M^* and the algorithm's current hypothesis $G(x_1 \cdots x_i)$ for each value of i from 1 to n. Let $\alpha(n)$ be the expected number of mispredictions of M^* on x, and $\beta(n)$ be the expected number of mispredictions when we use $G(x_1 \cdots x_i)$ to predict bit x_{i+1}. We say that the algorithm identifies good predictive hypotheses if $\beta(n) - \alpha(n) = o(n)$. Note that this condition is equivalent to saying that the prediction rate of the identification algorithm approaches the optimal rate; moreover, bounding this quantity (by say $n^{1-\epsilon}$) gives a bound on the convergence rate.

We prove that for any class $\mathcal{M}$, there is an identification algorithm in the above sense - the number of times the identification algorithm changes its hypothesis depends only upon the index of the target machine M^* in the enumeration of $\mathcal{M}$. Also, the expected number of mispredictions of the hypotheses for the first n bits is within (an additive) $O(n^{2/3})$ of the number of mispredictions of M^*.

The algorithm is based on a Bayesian approach, where we pick the machine with the highest posterior probability; however, the choice of the prior is of crucial importance. We must pick a prior so that small machines are favored sufficiently to guarantee convergence, but not so much that we lose predictive power. In addition, to achieve convergence, we modify the algorithm so that it only switches its hypothesis if the current machine is significantly less likely (a posteriori) than the most likely machine.

We first describe our identification algorithm, which we call the Stubborn algorithm. We then bound the expected number of times that this algorithm switches its guess. We bound the number of mispredictions of our identification algorithm via a series of reductions - the first reduction shows that the number of mispredictions of the Stubborn algorithm can be bounded in terms of the number of mispredictions of an algorithm that simply picks the machine with the highest a posteriori probability (we call this the Max algorithm). Next, we introduce the notion of an ϵ-Bayes algorithm. Recall that Bayes' algorithm just predicts the most likely outcome (one with probability at least 1/2) for the next bit, based on the weighted average, according to the prior, of all the machines in $\mathcal{M}$. The ϵ-Bayes algorithm is similar, except that instead of always using 1/2 as the threshold, an adversary gets to choose the threshold from $\{\frac{1+\epsilon}{2}, \frac{1}{2}, \frac{1-\epsilon}{2}\}$. We relate the error rate of the Max algorithm to that of ϵ-Bayes for a suitable choice of ϵ, and finally analyze the error rate of ϵ-Bayes to complete our analysis.

3 Model

Since we make no assumptions about the randomized algorithms in the class $\mathcal{M}$, it is natural to model a source $M \in \mathcal{M}$ by an infinite binary tree with probabilities on its edges. For every vertex v in the tree, the probabilities of the left and right edges descending from v sum to 1. A source produces a sequence by starting at the root and choosing a path down the tree according to the edge probabilities. Whenever the walk traverses a left or right edge, the source outputs 0 or 1 respectively.

Alternatively M may be viewed as a probability distribution on $[0, 1)$. This source produces a sequence by choosing $\mathbf{x} \in [0, 1)$ according to its distribution and revealing the binary expansion of $\mathbf{x}$ (following the decimal point) one bit at a time.

Let $M[x]$ be the probability that $M \in \mathcal{M}$ generates a string $\mathbf{x}$ which has x as a (finite) prefix. Let $|M|$ be the size of the index of M. Hence, if $\mathcal{M} = \{M_0, M_1 \cdots\}$ then $|M_i| = \lceil \log i \rceil$.

4 The Stubborn algorithm

We now introduce our identification algorithm - the Stubborn algorithm. The algorithm is defined by a function $G : \{0, 1\}^* \mapsto \mathcal{M}$, where $G(x)$ is the machine that is chosen by the algorithm after seeing x. Let,

$$\phi(M, x) \overset{\text{def}}{=} c|M| + \log\left(\frac{1}{M[x]}\right)$$

One way to view ϕ is as the length of an encoding of x (MDL principle [8]). We define G inductively.

- $G(\epsilon) = M_1$.
- Let $H(x)$ be the smallest such that $\phi(H(x), x) = \min\{\phi(M, x) : M \in \mathcal{M}\}$
- $G(xa) =$
$$\begin{cases} G(x) & \text{If } \phi(G(x), xa) < \phi(H(xa), xa) + 1 \\ H(xa) & \text{otherwise.} \end{cases}$$

Notice that the min operation in the computation of $H(x)$ can be reduced to a finite one assuming $c > 0$. Also, the quantity $\pi(M) = 2^{-c|M|}$ can be viewed as a Bayesian prior in concert with existing literature (see [7]). The choice of constant c is critical in obtaining our results.

The Stubborn algorithm guesses the source $G(x_1 \cdots x_i)$ after seeing the ith output bit. It then predicts bit x_{i+1} according to its guess. Predicting according to a source M means predicting 0 if $M[x0] > M[x1]$ and predicting 1 otherwise.

Theorem 1 *The Stubborn algorithm guesses a source at each step and predicts according to that source so that,* *A. The expected number of times the algorithm changes its guess is finite.*

B. The expected number of mispredictions in excess of the optimal predictor is $O(n^{2/3}2^{c|M^*|/3})$ where c is the aforementioned constant.

We should emphasize that our algorithm depends upon the ability to calculate the minimum over all $M \in \mathcal{M}$ of the quantity $c|M| + \log(1/M[x])$. Thus a class of machines $\mathcal{M}$ can be efficiently identified if this optimization problem can be efficiently solved. This is analogous to Occam's Razor, which identifies an optimization problem for the deterministic case - minimizing $|M|$ for an M consistent with the output - such that efficiently identifying an unknown $M^* \in \mathcal{M}$ is computationally equivalent to efficiently solving the corresponding optimization problem.

5 Resoluteness

In this section we will focus on the first part of Theorem 1. Consider a string $\mathbf{x} = x_1x_2x_3\cdots$. Here each $x_i \in \{0,1\}$. Let $C(\mathbf{x}) = \{x_1\cdots x_i : G(x_1\cdots x_i) \neq G(x_1\cdots x_{i-1})\}$. The set $C(\mathbf{x})$ contains those prefixes of $\mathbf{x}$ on which the Stubborn algorithm changes its hypothesis. The first part of Theorem 1 can be precisely stated as,

Theorem 1A. *If $c > 3$ then for any $M^* \in \mathcal{M}$,*

$$E_{\mathbf{x}\sim M^*}|C(\mathbf{x})| \in O\left(\frac{1}{\pi(M^*)}\right)$$

The proof of the theorem follows from two lemmas. Let the *active set* $\mathcal{A}(\mathbf{x}) \subseteq \mathcal{M}$ be defined as,

$$\mathcal{A}(\mathbf{x}) \overset{\text{def}}{=} \{M \in \mathcal{M} : M = G(x), x \in \mathsf{pref}(\mathbf{x})\}$$

where $\mathsf{pref}(\mathbf{x}) = \{x_1x_2\cdots x_i : \forall i\}$. The active set contains those sources that are the Stubborn algorithm's guess at some point on seeing $\mathbf{x}$. We obtain the following surprising lemma about the active set which in particular implies that the expected size of the active set is small.

Lemma 2

$$prob_{\mathbf{x}\sim M^*}(\exists M \in \mathcal{A}(\mathbf{x}) : |M| \geq t) \leq \frac{2^{t(1-c)}}{\pi(M^*)(1-2^{1-c})}$$

Proof: Let F_M be the set of x such that $\phi(M,x)$ minimizes $\phi(\cdot,x)$ and no prefix of x does. Note that $M \in \mathcal{A}(\mathbf{x})$ only if some prefix of $\mathbf{x}$ is in F_M. For every $x \in F_M$, $\phi(M,x) \leq \phi(M^*,x)$ or equivalently,

$$\pi(M)M[x] \geq \pi(M^*)M^*[x]$$

Summing the above inequality over all $x \in F_M$, we obtain

$$\pi(M)\sum_{x\in F_M} M[x] \geq \pi(M^*)\sum_{x\in F_M} M^*[x]$$

Since F_M is prefix free, it follows that $\sum_{x\in F_M} M[x] \leq 1$. Also, $\sum_{x\in F_M} M^*[x]$ is an upper bound on the probability that $\mathcal{A}(\mathbf{x})$ contains M. Therefore,

$$prob_{\mathbf{x}\sim M^*}(\mathcal{A}(\mathbf{x}) \text{ contains } M) \leq \frac{\pi(M)}{\pi(M^*)}$$

Finally, summing over all machines of size more than t, we get the lemma. ■

If t is the size of the largest machine in the active set then the size of the active set is less than 2^{t+1}. Using this fact and the previous lemma, it is simple to show that the expected size of the active set is finite (for $c > 2$).

We now show how to use the bound on the size of the active set to bound the number of times the Stubborn algorithm changes its hypothesis. Call xa an M, M' switch if $G(x) = M$ and $G(xa) = M'$ ($x \in \{0,1\}^*$ and $a \in \{0,1\}$). An M, M' switch is an *up* switch if $M < M'$, otherwise it is a *down* switch. Each M, M' down switch has *weight* $M - M'$ (difference of indices in enumeration). We make the following claim about up and down switches and their relationship to the active set.

$$\begin{aligned} &E(\text{No. of up switches}) - E(\text{wt. of down switches}) \\ &\quad\leq\ E(|\mathcal{A}(\mathbf{x})|) \end{aligned}$$

The previous lemma upper bounds the expected size of $\mathcal{A}(\mathbf{x})$. The following lemma upper bounds the expected weight of down switches. Together the lemmas provide a bound on the number of up switches. In addition, since the weight of a down switch is at least one, the following lemma also bounds the expected number of down switches, thus proving the theorem.

Lemma 3 *Let $M > M'$.*

$$E_{\mathbf{x}\sim M^*}(\textit{No. of } M, M' \textit{ switches in } \mathbf{x}) \le 4\frac{\pi(M')}{\pi(M^*)}$$

Proof: Let B_k be the set of strings x such that x is an M, M' switch and x has exactly $k-1$ proper prefixes that are M, M' switches. Let A_k be the set of strings x such that x is the longest prefix of some $y \in B_k$ that is a $\cdot, M$ switch. The following facts about A_k and B_k are easily established.

1. B_k and A_k are prefix free sets (i.e. if $x \in A_k$ (B_k), then no prefix y of x is in A_k (B_k)).
2. Every x in B_k has a prefix y in A_k.
3. Every y in A_k has a prefix z in B_{k-1}.

The heart of the lemma is an inductive proof that

$$\sum_{x\in B_k} M[x] \le 2^{-k+1}$$

For $k=1$, the result follows since B_1 is prefix free. For $k>1$, since strings in B_k are M, M' switches, and fact 2.

$$\begin{aligned}\sum_{x\in B_k} M[x] &\le \sum_{x\in B_k} \frac{\pi(M')}{2\pi(M)} M'[y] \\ &\le \sum_{y\in A_k} M'[y]\frac{\pi(M')}{2\pi(M)}\end{aligned}$$

But since strings in A_k are $\cdot, M$ switches, and fact 3,

$$\sum_{y\in A_k} M'[y]\frac{\pi(M')}{2\pi(M)} \le \sum_{y\in A_k}\frac{1}{2}M[y] \le \sum_{z\in B_{k-1}}\frac{1}{2}M[z]$$

And this completes the induction. Fact 3, can now be used to show that

$$\begin{aligned}\sum_{w\in A_{k+1}} M^*[w] &\le \sum_{w\in A_{k+1}} \frac{\pi(M)}{\pi(M^*)}M[w] \\ &\le \frac{\pi(M)}{\pi(M^*)}2^{-k+1}\end{aligned}$$

which tells us that the number of M, M' switches is at best a geometric random variable and implies the lemma. ■

The lemma bounds the number of M, M' down switches. Multiplying by $M - M'$ and summing over all pairs $M > M'$ bounds the expected weight of down switches (for $c > 3$, the sum converges). This bound and the bound on the expected size of $\mathcal{A}(\mathbf{x})$ imply that the expected number of switches (both up and down) is finite which proves theorem 1.A.

6 Reliability

To facilitate the following discussion, we consider the Stubborn algorithm from a Bayesian viewpoint. The quantity $2^{-\phi(M,x)}$ corresponds to the Bayesian probability of M given x using prior $2^{-c|M|}$ (up to some normalizing constant). The machine that minimizes $\phi(\cdot, x)$ also maximizes $2^{-\phi(\cdot,x)}$. Whence, minimizing $\phi(\cdot, x)$ corresponds to identifying the most likely machine in the Bayesian setting. Notice also that the Stubborn algorithm changes its mind only when the Bayesian probability of its current hypothesis is less than half the maximum.

Let $F(A, x)$ be the number of mispredictions made by algorithm A on sequence x. Therefore,

$$F(A,x) \overset{\text{def}}{=} |\{i : A(x_1, x_2 \cdots x_{i-1}) \ne x_i\}|$$

The following theorem says that predicting according to the hypotheses of the Stubborn algorithm is almost as good as predicting according to the true source M^* (optimal prediction).

Theorem 1B. *If $c > 3$ then for any $M^* \in \mathcal{M}$,*

$$E[F(\textit{Stubborn}, x) - F(M^*, x)] \in O(n^{2/3}2^{c|M^*|/3}) \quad (1)$$

where the expectation is taken over $x \in \{0,1\}^n$ produced by M^*.

One implication of the theorem is that the rate of misprediction approaches the optimal rate in the limit. Results of this kind have been shown in the case where the optimal predictor is based on a finite state Markov source ([10, 5]). However, for our model, in which the optimal predictor is based on a general random source, these results do not apply. In addition, we require that our algorithm predict according to a hypothesis from the set $\mathcal{M}$. It is worth noting that if prediction were the only goal, then we can show a bound of $\Theta(\sqrt{n \log \frac{1}{\pi(M^*)}})$ (following lemma 3). Thus, we pay a price in order to accomplish both inference and prediction.

6.1 The Max algorithm

The Max algorithm guesses the most probable source (in the Bayesian sense). The Max algorithm lacks the stubbornness that was essential to obtain the first half of the theorem. We will see now that as far as the competitive error difference is concerned, the two algorithms are similar. Partition $\{0,1\}^*$ into (disjoint) sets X_i where X_i is the set of strings on which the Stubborn algorithm picks M_i. Consider the world where only two machines $M = M_i$ and M^* exist. Let

$$\pi'(M) = \frac{2\pi(M)}{W} \qquad \pi'(M^*) = \frac{\pi(M^*)}{W}$$

be the prior probabilities used by the Max algorithm[1]. W is a normalizing factor and is $2\pi(M) + \pi(M^*)$. For $x \in X_i$, we notice that

$$\pi'(M)M[x] \geq \pi'(M^*)M^*[x]$$

Therefore, the Max algorithm will choose M on X_i. View the total error of the Stubborn algorithm as a sum of individual contributions of machines $M_i \in \mathcal{M}$. The error of the Max algorithm in the two machine world M_i, M^* exceeds the contribution of M_i to the total error of Stubborn. Therefore, if we bound the error of the Max algorithm in each two machine world, we are done since we just sum over all the worlds.

It remains to bound the competitive difference of Max using the prior π' in the two machine world M, M^*.

6.2 The ϵ-Bayes algorithm

The competitive difference of the Max algorithm is hard to bound directly. So we introduce yet another algorithm, the ϵ-Bayes algorithm. We begin by comparing the Max algorithm to the ϵ-Bayes algorithm and then analyzing the latter.

Instead of guessing one of M or M^*, the ϵ-Bayes algorithm takes the weighted average of the two and predicts according to that average. Let

$$B[x] \overset{\text{def}}{=} \pi''(M^*)M^*[x] + \pi''(M)M[x]$$

where π'' are the prior probabilities used by the ϵ-Bayes algorithm. (B is just the Bayesian "average" machine.) Let $\beta(x)$ be the Bayesian (conditional) probability that the next bit is a 0 after seeing the string x.

$$\beta(x) \overset{\text{def}}{=} \frac{B[x0]}{B[x]}$$

In the classical Bayesian world, one would predict 0 iff $\beta(x) > 1/2$. In the ϵ-Bayes world, this rigid threshold is relaxed somewhat. Philosophically, the ϵ-Bayes algorithm asserts that if $\beta(x)$ is large, then predict 0, if it is small, then predict 1 but if β is roughly $\frac{1}{2}$, then what we predict is not important. We define ϵ-Bayes's prediction after string x to be,

$$\epsilon\text{-Bayes}(x) = \begin{cases} 0 & \text{if } \beta(x) > T_x \\ 1 & \text{otherwise} \end{cases}$$

where T_x is chosen from $\{\frac{1+\epsilon}{2}, \frac{1}{2}, \frac{1-\epsilon}{2}\}$. Indeed, the careful choice of ϵ, T_x and π'' is essential to the final result. Let

$$\pi''(M) = \frac{w\pi'(M)}{W'} \qquad \pi''(M^*) = \frac{\pi'(M^*)}{W'}$$

be the prior probabilities used by ϵ-Bayes. Here, W' is a normalizing factor and is $w\pi'(M) + \pi'(M^*)$. This definition of π'' boosts the relative weight of M by a factor of w. Also, we have reduced the issue of choosing π'' to one of choosing w. This choice of π'' has an immediate consequence expressed in the claim below.

Claim[2] If $\pi'(M)M[x] > \pi'(M^*)M^*[x]$ then $|\beta(x) - M[0|x]| \leq 1/w$

Proof:

$$|\beta(x) - M[0|x]| = \frac{\pi''(M^*)M^*[x]\,|M^*[0|x] - M[0|x]|}{B[x]} \leq 1/w$$

The claim states that whenever Max picks M, the Bayesian belief that the next bit is 0 is close to the probability M places on 0 (using π''). If $\beta(x)$ and $M[0|x]$ are both less than 1/2 or both greater than 1/2 then we choose $T_x = 1/2$ and Max and ϵ-Bayes make the same prediction. Otherwise, the claim implies that we can choose $T_x = \{\frac{1+\epsilon}{2}, \frac{1-\epsilon}{2}\}$ (for $\epsilon = 2/w$) so that Max and ϵ-Bayes make the same prediction.

We have reduced the task of choosing ϵ, T_x and π'' to choosing a single parameter w. The choice

[1] This technique of boosting the relative weight of M is a very useful technique and will appear again later.

[2] Here $M[0|x]$ has the obvious meaning. It is $\frac{M[x0]}{M[x]}$.

of w will be made following the analysis of the ϵ-Bayes algorithm. We are left with the task of bounding the performance of the ϵ-Bayes algorithm.

Lemma 4 *For any pair of sources M^* and M, for any prior distribution π'' on M^* and M, for any choice of $T_x \in \{\frac{1+\epsilon}{2}, \frac{1}{2}, \frac{1-\epsilon}{2}\}$ for every $x \in \{0,1\}^*$,*

$$\begin{aligned} & E[F(\epsilon\text{-}Bayes, x) - F(M^*, x)] \\ & \quad = O\left(n\epsilon + \sqrt{n \lg \frac{1}{\pi''(M^*)}}\right) \end{aligned} \tag{2}$$

where the expectation is taken over $x \in \{0,1\}^n$ produced by M^*.

Once the lemma is shown, theorem 1B follows easily. We choose $w = \sqrt[3]{2n\pi(M^*)/\pi(M)}$, Composing the reductions and applying the lemma and then summing over all $M \in \mathcal{M}$ gives the theorem for $c > 3$.

Proof (lemma 4):(Sketch) Since we are only concerned with sequences of length n, we truncate each source at a finite depth n. The sources are completely defined by the probability they place on sequences $x \in \{0,1\}^n$ (the leaves). So each source is a probability distribution on 2^n sequences. Our plan is to delineate the structure of the worst case machines M and M^*.

We claim that the worst case is achievable even if we assume that $T_x = (1+\epsilon)/2$ for all $x \in \{0,1\}^{<n}$.

We also claim that there exists a worst case pair M, M^* which is ϵ-balanced with respect to $\pi''(M)$ and $\pi''(M^*)$. By ϵ-balanced we mean that for $x \in \{0,1\}^{<n}$,

$$\beta(x) = \frac{1+\epsilon}{2} \tag{3}$$

If every $T_x = (1+\epsilon)/2$, ϵ-Bayes always predicts 1 in an ϵ-balanced pair. To obtain the worst case bound, we assume that M^* always predicts 0.

If M and M^* are not ϵ-balanced, let x ($|x| < n$) be a longest sequence which violates (3). Let M_0 and M_1 (M_0^* and M_1^*) be the left and right subtrees below x in M (M^*). The choice of x implies that the pair M_0, M_0^* is ϵ-balanced with respect to priors $\pi''(M)M(x0)$ and $\pi''(M^*)M^*(x0)$. Similarly, the pair M_1, M_1^* is ϵ-balanced with respect to $\pi''(M)M(x1)$ and $\pi''(M^*)M^*(x1)$.

We can assume without loss of generality that $\beta(x) < (1+\epsilon)/2$. This is since we can exchange the subtrees otherwise. To achieve ϵ-balance at x, we shift probability from M_1, M_1^* to M_0, M_0^* until we reach the balance point. The actual process is to move a fraction f of M_1 to M_0 and a fraction f of M_1^* to M_0^*. where f is chosen to balance the pair M,M^* at x.

Since x is the longest sequence violating (3), the left and right subtrees below x remain ϵ-balanced after this process. Consequently, ϵ-Bayes predicts 0 in these subtrees and M^* predicts 1. Since we increase M^*'s probability on sequences starting $x0$, the competitive difference can only increase.

Given these properties, the lemma follows easily. The strategy is for M^* to put as much probability as possible (without violating ϵ-balance) on sequences with the most 0's. The ϵ-balanced property implies that

$$\begin{aligned} & \pi''(M^*)M^*[x] + \pi''(M)M[x] = \\ & \quad \left(\frac{1+\epsilon}{2}\right)^{\#\text{ 0's in }x} \left(\frac{1-\epsilon}{2}\right)^{\#\text{ 1's in }x} \end{aligned}$$

which bounds the probability $M^*[x]$. It follows from Chernoff bounds that the expected number of 1's in a sequence produced by M^* is at least a constant times

$$\frac{1+\epsilon}{2}n + \sqrt{n\frac{1+\epsilon}{2}\frac{1-\epsilon}{2}\lg\frac{1}{\pi''(M^*)}}$$

The lemma follows. ■

7 Open questions and other work.

There are a couple of interesting open questions that are not answered by this research. Consider, for example, the number of times the Stubborn algorithm changes its mind. It can be shown that this number is essentially tight. However, what is not clear is whether this number is optimal. Is there some algorithm that changes its mind only $O\left(\text{polylog}\frac{1}{\pi(M^*)}\right)$ times in expectation?

Another issue that remains open is to close the gap between the $\Omega(\sqrt{n})$ lower bound and the $O(n^{2/3})$ upper bound for the competitive difference. (The first result does not appear in this abstract).

8 Acknowledgements

Eddie Grove was involved in the preliminary stages of this research. Leonard Schulman suggested the term 'resoluteness'. He also provided us with some very valuable criticism about the write up. Thanks to Manuel Blum, Jim Crutchfield, and David Haussler for useful discussions.

References

[1] A. Blumer, A. Ehrenfeucht, D. Haussler and M. Warmuth, Occam's Razor, *Inform. Processing Letters* **24**, 377-380.

[2] R. Board and L. Pitt, On the necessity of Occam Algorithms, *Proceedings, 22nd ACM Symposium on Theory of Computing*, pp. 54-63.

[3] J. Crutchfield and K. Young, Inferring Statistical Complexity, *Phys. Rev. Let.*, vol. 63, p 105, 1989

[4] A. DeSantis, G. Markowsky, M. Wegman, Learning Probabilistic Prediction Functions, *Proc. 1988 Workshop on Computational Learning Theory*, pages 312–328, 1988.

[5] M. Feder, N. Merhav, and M. Gutman, Universal Prediction of Individual Sequences *IEEE Trans. Inform. Theory*, vol. 38(4), pages 1258–1269, July 1992.

[6] D. Haussler, M. Kearns, and R. Schapire, Bounds on the Sample Complexity of Bayesian Learning Using Information Theory and the VC Dimension, *Proc. 4th Workshop on Computational Learning Theory*, pages 61–74, 1991.

[7] M. Li and P. Vitányi, Inductive Reasoning and Kolmogorov Complexity *Journal of Computer and System Sciences*, vol. 44, pages 343–384, 1992.

[8] J. Rissanen, A universal prior for integers and estimation by minimum description length, *Ann. Statist.*, vol. 11, pages 416–431, 1982.

[9] S. Rudich, Inferring the structure of a Markov Chain from its output, *Proc. 26th IEEE Symposium on Foundations of Computer Science*, pages 321–326, 1985.

[10] J. Vitter and P. Krishnan, Optimal Prefetching via Data Compression, *Proc. 32nd IEEE Symposium on Foundations of Computer Science*, pages 121–130, 1991.

[11] J. Ziv and A. Lempel, Compression of Individual Sequences via Variable-Rate Coding *IEEE Transactions on Information Theory*, vol. 24, pages 530–536, September 1978.