Sampling Algorithms: Lower Bounds and Applications

[Extended Abstract] *

Ziv Bar-Yossef†
Computer Science Division
U. C. Berkeley
Berkeley, CA 94720
zivi@cs.berkeley.edu

Ravi Kumar
IBM Almaden
650 Harry Road
San Jose, CA 95120
ravi@almaden.ibm.com

D. Sivakumar
IBM Almaden
650 Harry Road
San Jose, CA 95120
siva@almaden.ibm.com

ABSTRACT

We develop a framework to study probabilistic sampling algorithms that approximate general functions of the form $f : A^n \to B$, where A and B are arbitrary sets. Our goal is to obtain lower bounds on the query complexity of functions, namely the number of input variables x_i that any sampling algorithm needs to query to approximate $f(x_1, \ldots, x_n)$.

We define two quantitative properties of functions — the *block sensitivity* and the *minimum Hellinger distance* — that give us techniques to prove lower bounds on the query complexity. These techniques are quite general, easy to use, yet powerful enough to yield tight results. Our applications include the mean and higher statistical moments, the median and other selection functions, and the frequency moments, where we obtain lower bounds that are close to the corresponding upper bounds.

We also point out some connections between sampling and streaming algorithms and lossy compression schemes.

1. INTRODUCTION

The need for computing with massive data sets has sparked off much interest in novel computational paradigms. These include, but are not restricted to, algorithms that probe only small (random) portions of the data; algorithms that work by making a few passes over the data [17]; algorithms that operate on a stream of data with limited space and stringent constraints on time per data item [17, 1, 13]. Algorithms of this nature are typically intended to compute sufficiently good approximate solutions to the problem at hand. Several algorithmic and complexity questions arise in the context of these computational paradigms.

*A full version of this paper is available at http://www.cs.berkeley.edu/∼zivi

†Part of this work was done while the author was visiting IBM Almaden Research Center. Supported by NSF Grant CCR-9820897.

STOC'01, July 6-8, 2001, Hersonissos, Crete, Greece.

In this paper, we focus on one such paradigm: algorithms that work by randomly sampling a few entries of a large input. We develop the first systematic complexity theory of probabilistic sampling algorithms that approximate general functions of the form $f : A^n \to B$, where A and B are arbitrary sets. A highlight of our work is the level of generality we are able to achieve toward our goal: we consider functions where the domain and range are not necessarily metric spaces, which makes the issue of "approximate" computation rather subtle; even when the domain and range have metric properties, there are many standard notions of approximation, e.g., relative vs. additive, that need to be encompassed. Also, we include algorithms that may perform non-oblivious (adaptive) sampling of input entries.

Main contributions.

From the standpoint of modeling, we present a single unified view of what it means to approximate a general function on a possibly non-metric domain and range. As our computational model, we show how the standard decision tree model may be suitably adapted. Our main results are two techniques for obtaining lower bounds on the query complexity for approximating broad classes of functions. A primary advantage of these techniques is that the lower bounds are stated in terms of certain quantitative properties of functions, which make them quite general and very easy to use. We also present (general and specific) upper bounds that establish the tightness of our lower bounds. Finally, we point out connections between sampling algorithms, streaming algorithms, and lossy data compression.

Lower bounds.

We present two techniques for obtaining lower bounds on the query complexity of randomized sampling algorithms that approximately compute functions. Our first lower bound technique is based on an adaptation of the notion of *block sensitivity* [23]. The advantage of this technique is that it applies to *any* function, even though the lower bounds may be somewhat weak. Moreover, it applies to the *expected* query complexity, not just the worst case query complexity. Our second lower bound technique is based on the notion of *Hellinger distance* [21] between probability distributions, and yields much stronger bounds for symmetric functions. These results are described in Sections 3 and 4.

Using the method based on the Hellinger distance, we obtain (in many cases, optimal) lower bounds on the worst-case query complexity for several problems. These include approximate computations of the median, minimum, maximum, and other selection functions, the mean and higher statistical moments, and frequency moments F_k for $k \geq 2$. For the case of the mean, our lower bound matches the lower bound of Canetti, Even, and Goldreich [7], and (modulo the machinery) is substantially simpler. This lower bound has another powerful consequence: it implies the main technical result of Radhakrishnan and Ta-Shma [24], which they use to obtain lower bounds on extractor, disperser, and superconcentrator parameters.

Our lower and upper bounds for the frequency moments have some interesting implications. (Recall that given $X \in [m]^n$, for $k \geq 0$, the k-th frequency moment $F_k(X)$ is defined as $\sum_{i=1}^{m}(f_i(X))^k$, where $f_i(X)$ is the number of times i appears in the sequence X.) While any oblivious sampling algorithm that uses s samples can be simulated by a streaming algorithm that uses space (roughly) s, we show that the converse is not true for F_2: the streaming algorithm of Alon, Matias, and Szegedy [1] uses $O(\log m)$ space for approximating F_2, while we prove an $\Omega(\sqrt{m})$ lower bound for sampling algorithms. (This style of "separation" of the two models is also demonstrated in [14].) We then show that for $k \geq 2$, the k-th frequency moment can be approximated using $O(m^{1-\frac{1}{k}})$ samples, which immediately implies the space upper bound of [1] for all $k > 2$. Finally, we provide a simple proof for the sampling lower bound of Charikar et al. [9] for F_0. These results are described in Section 5.

We also investigate the question of how tight our lower bound methodologies are. We obtain a general theorem, which shows that at least in some special cases (symmetric functions on bounded domains), the lower and upper bounds are polynomially related. In addition, we point out some limitations of our techniques.

These are described in Section 6.

Connection to Lossy compression.

The question of how to compress data so that certain interesting functions of the data may be computed directly from the compressed data is of fundamental importance, both in theory and practice. The question becomes much more interesting (and harder) if we allow the compression to be *lossy*, that is, if we trade some qualitative degradation of the data for large compression factors. We present a formal model to study this question, and show that if a function has an efficient sampling algorithm or a streaming algorithm [17, 1, 13], then it admits lossy compression (in a technically precise sense).

These results are described in Section 7.

In Section 8 we discuss related prior work and in particular the relationship of our model to the areas of Boolean decision tree complexity [6], PAC and statistical learning Theory [27, 20, 29], statistical decision theory [4], statistical estimation theory [28], and statistical sequential analysis [26].

2. PRELIMINARIES

In this section we introduce a notion of approximation for functions $f : A^n \to B$, where A and B are arbitrary sets. We then generalize Boolean decision trees to decision trees that approximately compute such functions. These decision trees will be our model for sampling algorithms.

2.1 Approximation Notions

An approximation for a function $f : A^n \to B$ is a function that maps inputs of f to subsets of its range B. This function assigns to every input x a set of values in B that are considered a good approximation for $f(x)$. It has two natural requirements, as specified in the following definition:

Definition 1 (Approximation). *An* approximation *for a function $f : A^n \to B$ is a family of functions $\{C_{f,\epsilon} : A^n \to 2^B\}_{\epsilon \geq 0}$, parameterized by an error parameter ϵ, which satisfies the following two conditions:*
(1) $\forall x \in A^n$, $C_{f,0}(x) = \{f(x)\}$
(2) $\forall x \in A^n\ \forall \epsilon > \epsilon' \geq 0$, $C_{f,\epsilon}(x) \supseteq C_{f,\epsilon'}(x)$

One might wonder why we define $C_{f,\epsilon}$ as a function of the inputs rather than as a function of the output values, i.e., why wouldn't $C_{f,\epsilon}$ determine for each value in the range what other values in the range are considered a good approximation for it? The rank and property testing approximations described below show that in some cases the approximation is indeed a function of the inputs: there can be two different inputs with the same output value but with different approximation sets.

Throughout this paper, for simplicity of notation, we implicitly assume that every function f is associated with a single approximation $C_{f,\epsilon}$ (the "natural" approximation for f). A few examples of popular approximation notions interpreted in light of the above definition:

(1) *Additive approximation:* (B, d_B) is a metric space. $C_{f,\epsilon}(x) = \{y \in B \mid d_B(y, f(x)) \leq \epsilon\}$.

(2) *Relative approximation:* $B = \mathbf{R}$. $C_{f,\epsilon}(x) = \{y \in B \mid (1-\epsilon)f(x) \leq y \leq (1+\epsilon)f(x)\}$.

(3) *Ratio approximation:* $B = \mathbf{R}$. $C_{f,\epsilon}(x) = \{y \in B \mid \epsilon f(x) \leq y \leq (1/\epsilon)f(x)\}$.

(4) *Rank approximation:* An approximation notion for selection functions, like the median. The input for Select_q, the selection function of order q ($0 \leq q \leq 1$), is a set of n real numbers $a_1, \ldots, a_n$. If we order these numbers from smallest to largest: $a_{i_0} \leq \cdots \leq a_{i_{n-1}}$, then $\mathrm{Select}_q(a_1, \ldots, a_n) = a_{i_{\lfloor q(n-1) \rfloor}}$. For example, the median is $\mathrm{Select}_{\frac{1}{2}}$, minimum is Select_0, and maximum is Select_1. The approximation is $C_{\mathrm{Select}_q,\epsilon}(a_1, \ldots, a_n) = \{a_{i_j} \mid j \in [\lfloor q'(n-1) \rfloor, \lfloor q''(n-1) \rfloor], q' = \max(q-\epsilon, 0), q'' = \min(q+\epsilon, 1)\}$.

(5) *Property testing approximation:* An approximation notion for functions with a Boolean range. We assume some distance measure D on the input domain. The approximation is: $C_{f,\epsilon}(x) = \{1\}$, if $D(x,y) \leq \epsilon/2$ for some input y with $f(y) = 1$; $C_{f,\epsilon}(x) = \{0\}$, if $D(x,y) \geq \epsilon$ for all inputs y with $f(y) = 1$; and $C_{f,\epsilon}(x) = \{0, 1\}$ otherwise.

A special kind of approximation is one that satisfies the "sunflower property," defined below. It is easy to check that approximations (2)–(5) above satisfy the sunflower property. Approximation (1) satisfies it, if $B = \mathbf{R}$.

Definition 2 (Sunflower property). *An approximation $C_{f,\epsilon}$ is said to satisfy the* sunflower property*, if for every $D \subseteq A^n$, the following condition holds:*
$(\forall x, x' \in D)[C_{f,\epsilon}(x) \cap C_{f,\epsilon}(x') \neq \emptyset] \Rightarrow \bigcap_{x \in D} C_{f,\epsilon}(x) \neq \emptyset$.

A notion that will play an important role in our discussion is the following:

Definition 3 (Disjoint inputs). *Two inputs $x, y \in A^n$ are said to be ϵ-disjoint with respect to approximation $C_{f,\epsilon}$, if $C_{f,\epsilon}(x) \cap C_{f,\epsilon}(y) = \emptyset$.*

2.2 Decision Trees for General Functions

A *randomized decision tree* for a function $f : A^n \to B$ is a rooted labeled tree, not necessarily binary. Similar to Boolean decision trees, internal nodes denote either queries of input locations or random coin tosses, and leaves denote outputs. Given an input x, we use the queries and the random coin tosses to determine a path from the root to one of the leaves. This leaf specifies the output on x.

Formally, for every node v we denote by $\deg(v)$ the number of children it has and by $S(v)$ the number of query nodes on the path from the root to v. If v is a query node, it is labeled by an input variable $i \in [n]$ and a function $N_v : A^{S(v)} \to [\deg(v)]$. If the path reaches v, we apply N_v on the values queried so far to determine which child of v should be the next node on the path. If v is a random coin node, it is labeled by the special character \$, instructing the path to pick one of its children uniformly at random. If v is a leaf, it is labeled by a function $O_v : A^{S(v)} \to B$, which specifies the output of the tree, based on the values queried along the path. Note that any input x may be associated with several possible paths leading from the root to a leaf, depending on the random choices made in the random coin nodes. These random choices induce a distribution over the paths corresponding to x.

We discuss two notions of complexity for decision trees: the *expected query complexity* of a tree T on input x, denoted $S^e(T, x)$, is the expected number of query nodes on paths corresponding to x. The *worst-case query complexity* of T on x, denoted $S^w(T, x)$, is the maximum number of query nodes on paths corresponding to x. Here, the expectation and the maximum are over the distribution of paths. The expected and worst-case query complexity of T, $S^e(T)$ and $S^w(T)$, are the maximum of $S^e(T, x)$ and $S^w(T, x)$, respectively, over all inputs $x \in A^n$.

Notice that this model can simulate, with the same efficiency, various models of decision trees, including Boolean, comparison, and algebraic decision trees.

Yao's Theorem [30] gives an equivalent characterization of a randomized decision tree as a distribution μ over deterministic decision trees. The expected query complexity of the tree on input x is the expected length (over μ) of the paths corresponding to x in these trees (and similarly for the worst-case query complexity).

Let $\epsilon \geq 0$ be an error parameter, $0 \leq \delta \leq 1$ a confidence parameter, and $f : A^n \to B$ a function with approximation $C_{f,\epsilon}$. A decision tree is said to (ϵ, δ)-*approximate* f, if for every input $x \in A^n$ the probability of paths corresponding to x that output a value $y \in C_{f,\epsilon}(x)$ is at least $1 - \delta$. The (ϵ, δ) *expected query complexity* of f is: $S^e_{\epsilon,\delta}(f) \stackrel{\text{def}}{=} \min\{S^e(T) \mid T \ (\epsilon, \delta)\text{-approximates } f\}$. We similarly define the worst-case query complexity of functions.

3. A GENERAL LOWER BOUND VIA BLOCK SENSITIVITY

In this section we generalize the notion of block sensitivity (defined by Nisan [23] for Boolean functions) and obtain a lower bound on the expected query complexity in terms of block sensitivity.

A Boolean function $f : \{0, 1\}^n \to \{0, 1\}$ is said to be sensitive to a subset (a "block") $I \subseteq [n]$ on input x, if f flips its value when flipping all the bits in I. The block sensitivity of f on x, $bs(f, x)$, is the maximum number t of pairwise disjoint subsets $I_1, \ldots, I_t \subseteq [n]$, to which f is sensitive on x. The block sensitivity of f, $bs(f)$, is the maximum, over all inputs x, of $bs(f, x)$.

For general domains and ranges, the basic intuition we carry over from the Boolean case is that a function is sensitive to a block of variables if a change to the corresponding input elements results in a significant change to the function value. In the following definition we denote by $x^{(I \leftarrow Q)}$ the input obtained from x by changing the elements in $I \subseteq [n]$ to the values in $Q \in A^{|I|}$:

Definition 4 (Block sensitivity). *f is* ϵ-sensitive to a subset of variables $I \subseteq [n]$ *on input x, if there exists $Q \in A^{|I|}$ such that x and $x^{(I \leftarrow Q)}$ are ϵ-disjoint. $bs_\epsilon(f, x)$, the* ϵ-block sensitivity of f on x, *is the maximum number t of pairwise disjoint subsets $I_1, \ldots, I_t \subseteq [n]$, such that f is ϵ-sensitive to each of them on x. $bs_\epsilon(f)$, the* ϵ-block sensitivity of f, *is* $\max_{x \in A^n} bs_\epsilon(f, x)$.

Nisan [23] proved that for Boolean functions f, $S^e_{0,\delta}(f) \geq (1 - 2\delta)bs(f)$. We generalize the proof of this theorem to our case.

Theorem 5. *For every $\epsilon \geq 0$, $0 \leq \delta \leq 1/2$, and $f : A^n \to B$, $S^e_{\epsilon,\delta}(f) \geq (1 - 2\delta)bs_\epsilon(f)$.*

Proof. Let x be the input that achieves the block sensitivity of f. Let $I_1, \ldots, I_t \subseteq [n]$ be the disjoint variable blocks to which f is ϵ-sensitive on x (note that $t = bs_\epsilon(f)$). Let $Q_1, \ldots, Q_t$ be the assignments to these blocks, for which x and $x^{(I_k \leftarrow Q_k)}$ are ϵ-disjoint.

Consider any decision tree T that (ϵ, δ)-approximates f. We will use the view of T as a distribution μ over deterministic decision trees $T_1, \ldots, T_m$. Consider one such tree T_j. There is exactly one path in T_j from the root to a leaf corresponding to x. Assume this path does not query any of the variables in I_k for some $k \in [t]$. This means that the same path will correspond to $x^{(I_k \leftarrow Q_k)}$. Moreover, since all the values queried in this path are the same at x and at $x^{(I_k \leftarrow Q_k)}$, the leaf will output the same value y for both. However, since $C_{f,\epsilon}(x) \cap C_{f,\epsilon}(x^{(I_k \leftarrow Q_k)}) = \emptyset$ either $y \notin C_{f,\epsilon}(x)$ or $y \notin C_{f,\epsilon}(x^{(I_k \leftarrow Q_k)})$. So T_j belongs to the "bad" set of trees (i.e., trees that output a non-approximate value) of either x or $x^{(I_k \leftarrow Q_k)}$. The probability of the bad set of each input can be at most δ, implying that the probability of trees T_j, in which the path corresponding to x does not query any variable in I_k can be at most 2δ. This is true for all $k \in [t]$, hence the expected length of paths corresponding to x is at least $(1 - 2\delta)t$. $\square$

Nisan [23] proved that for Boolean functions, block sensitivity and (deterministic/randomized) decision tree complexity are polynomially related. A particular corollary of this is that for Boolean functions, deterministic and randomized decision tree complexities are polynomially related. It is natural to ask if block sensitivity also gives a (polynomial) characterization of randomized decision tree complexity for approximating general functions, but it turns out not to be the case. While the ϵ-approximate median problem has a sampling algorithm that makes $O(\epsilon^{-2})$ probes (independent of n), it is easy to show that any deterministic algorithm needs at least $\Omega(n(1 - \epsilon))$ probes.

4. A LOWER BOUND FOR SYMMETRIC FUNCTIONS

The query complexity lower bound via block sensitivity is very general but is also weak. Its dependence on both the error ϵ and the confidence δ frequently turn out to be sub-optimal. In this section we obtain a stronger lower bound that works only for *symmetric functions* — functions that are invariant under permutations of the input elements:

Definition 6 (Symmetric functions). *A function $f : A^n \to B$ is* ϵ-symmetric *if for every input $x \in A^n$ and every permutation $\pi \in S_n$, $C_{f,\epsilon}(x) = C_{f,\epsilon}(\pi(x))$ (where $\pi(x)_i = x_{\pi(i)}$).*

The basic idea for the lower bound is to use a probabilistic view of inputs as distributions. An input $x \in A^n$ induces a distribution P_x on A, defined by picking an entry of x uniformly at random. The crucial point is that a decision tree that approximates the function f gives a procedure that *distinguishes* every two input distributions P_x and P_y for which $C_{f,\epsilon}(x)$ and $C_{f,\epsilon}(y)$ are disjoint. Clearly, the closer P_x and P_y are, the harder it is to distinguish between them. Thus, by giving a lower bound on the number of samples required to distinguish any two such input distributions, we obtain a lower bound on the query complexity of the function.

We state the lower bound in terms of the following measure. Here, $h(P,Q)$ denotes the Hellinger distance between distributions P and Q (see Section 4.2).

Definition 7 (Minimum Hellinger distance). *For a function $f : A^n \to B$, the* ϵ-minimum Hellinger distance of f *is $h_\epsilon(f) \stackrel{\text{def}}{=} \min\{h(P_x, P_y) \mid x, y \in A^n$ are ϵ-disjoint$\}$.*

Our main theorem gives a lower bound on the worst-case query complexity of symmetric functions in terms of their minimum Hellinger distance:

Theorem 8 (Main theorem). *For all $\epsilon \geq 0$, $0 \leq \delta < 1/4$, and every ϵ-symmetric function $f : A^n \to B$ with $S^w_{\epsilon,\delta}(f) \leq n/4$ and $h_\epsilon(f) \leq 1/2$,*

$$S^w_{\epsilon,\delta}(f) \geq \frac{1}{4h^2_\epsilon(f)} \ln \frac{1}{4\delta + O(1/n)}$$

Our strategy for the proof of Theorem 8 is the following: First we show (Lemma 9) that, essentially without any loss in efficiency, any symmetric function can be computed by decision trees that only probe the input at uniformly chosen positions. The consequence of this lemma is that if a decision tree makes k queries to an input x, the distribution it "sees" is the distribution P^k_x, the k-fold product of P_x. Next we derive a lower bound (Lemma 12) on the number of samples needed to distinguish two distributions (with a certain advantage) in terms of their Hellinger distance. Lemma 13 shows that a decision tree that (ϵ, δ)-approximates f using k (worst-case) queries can be used to distinguish the distributions P^k_x and P^k_y for any ϵ-disjoint x, y, with advantage 2δ. Lemma 14 handles some technical complications.

For simplicity of notation, we will assume throughout this section that the domain A is finite. Everything we prove in the sequel can be generalized to arbitrary domains by replacing summation by integration.

4.1 Uniform Decision Trees

A *uniform decision tree* is identical to a standard decision tree with the difference that every query node v is labeled not with an input variable but rather with the special character $\pounds$. This character instructs a path reaching v from the root to query an input variable chosen uniformly from all variables that have not been queried so far. The distribution of paths corresponding to an input x is now determined by both the random coin tosses of the tree and the random samples drawn at its query nodes.

It is convenient to think of a uniform decision tree as picking a random sample of input elements *without replacement*, and deciding the output according to the values of these elements. Note that these samples may be somewhat adaptive; for instance, the size of the sample can depend on the values of the sampled elements. We show next that for symmetric functions uniform decision trees can do as well as general decision trees:

Lemma 9. *Let $f : A^n \to B$ be an ϵ-symmetric function with $S^e_{\epsilon,\delta}(f) = k$. Then, there exists a uniform decision tree of expected query complexity k that (ϵ, δ)-approximates f.*

Proof. Let T be an optimal decision tree that (ϵ, δ)-approximates f. Without loss of generality, we may assume that T never queries the same index more than once. We will show a construction of a uniform decision tree T' that simulates T without loss of efficiency.

Given an input $x \in A^n$, T' picks a random permutation $\pi \in S_n$ and simulates T on the input $y = \pi(x)$. Whenever T queries y_i, T' returns $x_{\pi(i)}$. With probability at least $1 - \delta$, T' outputs an ϵ-approximation of $f(y)$. Due to symmetry, such an output is also an ϵ-approximation of $f(x)$.

The main point is that however adaptively T queries y, the distribution of the queries to x is uniform without replacement. Note that T' makes exactly the same number of queries as T, hence its complexity is k. □

A similar result holds for the worst case query complexity.

4.2 Sample Complexity for Distinguishing Distributions

The main aim of this section is to obtain a lower bound on the number of samples required in order to distinguish between two distributions. We will use two measures of distance between distributions: the variation distance and the Hellinger distance [21].

Definition 10 (Distance measures). *Let P and Q be two distributions on the same probability space Ω. The* variation distance d *and the* Hellinger distance h *between P and Q are defined as follows:*

$$\begin{aligned} d(P,Q) &\stackrel{\text{def}}{=} \frac{1}{2}\sum_{\omega\in\Omega} |P(\omega) - Q(\omega)| \\ &= \max_{D\subseteq\Omega} |P(D) - Q(D)| \\ h(P,Q) &\stackrel{\text{def}}{=} (1 - \sum_{\omega\in\Omega} \sqrt{P(\omega)Q(\omega)})^{\frac{1}{2}} \\ &= (\frac{1}{2}\sum_{\omega\in\Omega} (\sqrt{P(\omega)} - \sqrt{Q(\omega)})^2)^{\frac{1}{2}} \end{aligned}$$

Variation distance is a measure of distinguishability between distributions. In order to get a lower bound on the number of samples required to distinguish two distributions P and Q with some advantage α, we need to find a lower bound on the minimal integer k for which $d(P^k,Q^k) \geq \alpha$. Here, P^k and Q^k are distributions on Ω^k obtained by picking k independent random samples from P and Q respectively. However, variation distance does not behave well under product distributions, and in particular there is no simple formula that describes $d(P^k,Q^k)$ in terms of $d(P,Q)$. We therefore use the Hellinger distance, which has a nice multiplicativity property and bounds the variation distance from both sides: (the proof is easy and appears, e.g., in [21])

Proposition 11. *(1)* $1-h^2(P^k,Q^k) = (1-h^2(P,Q))^k$*; (2)* $h^2(P,Q) \leq d(P,Q) \leq h(P,Q)\sqrt{2-h^2(P,Q)}$.

The following Lemma provides the desired lower bound in terms of the distributions' Hellinger distance. It was pointed out to us by David Zuckerman [personal communication, September 2000].

Lemma 12. *Let P and Q be two distributions on Ω with $h^2(P,Q) \leq 1/2$, let $0<\alpha<1$, and let k be an integer such that $d(P^k,Q^k) \geq \alpha$. Then:* $k \geq \frac{1}{4h^2(P,Q)} \ln \frac{1}{1-\alpha^2}$

Proof. By Proposition 11 (part (2)), we have

$$h(P^k,Q^k)\sqrt{2-h^2(P^k,Q^k)} \geq d(P^k,Q^k) \geq \alpha,$$

which implies that $h^4(P^k,Q^k) - 2h^2(P^k,Q^k) + \alpha^2 \leq 0$. The solution to this quadratic inequality gives $h^2(P^k,Q^k) \geq 1-\sqrt{1-\alpha^2}$.

By Proposition 11 (part (1)), we have $1-(1-h^2(P,Q))^k \geq 1-\sqrt{1-\alpha^2}$. Using the inequality $1-x \geq e^{-2x}$ for $0 \leq x \leq 1/2$, we have:

$$\begin{aligned} k &\geq \ln{(1-\alpha^2)^{-1/2}} / \ln{(1-h^2(P,Q))^{-1}} \\ &\geq \frac{1}{4h^2(P,Q)} \ln \frac{1}{1-\alpha^2}. \end{aligned}$$

□

4.3 The Lower Bound

The main lemma of this section proves that any uniform decision tree of worst-case query complexity k that approximates a function f can be used to distinguish, using k samples, the distributions P_x and P_y of any two ϵ-disjoint inputs $x,y \in A^n$. Since the tree draws uniform samples *without replacement*, the distributions it distinguishes between are not P_x^k and P_y^k (the independent product distributions), but rather $\tilde{P}_x^k$ and $\tilde{P}_y^k$, the distributions on A^k obtained by picking k elements of x and y uniformly without replacement.

Lemma 13. *Let T be a uniform decision tree of worst-case query complexity k that (ϵ,δ)-approximates a function $f: A^n \rightarrow B$. Then for any two ϵ-disjoint inputs $x,y \in A^n$, $d(\tilde{P}_x^k,\tilde{P}_y^k) \geq 1-2\delta$.*

Proof. Without loss of generality, we assume that T is *regular*, that is, it always makes exactly k queries. As in [7], the basic idea underlying the proof will be to show that the distribution of values read off the input by the random queries along the tree's paths is very different when x is the input from when y is the input.

Observe that if we fix the internal random bits of T and the outputs of the queries it makes, then the output of the tree is fully determined. We can describe T as function that maps a pair (r,a), where $r \in \{0,1\}^m$ is a string of random bits and $a \in A^k$, to a value $b \in B$. The pair (r,a) determines a unique path in T from the root to a leaf v, and the value output by this leaf (which is obtained by applying O_v to a) is the value b. Note that the distribution of the string a is $\tilde{P}_x^k$ if x is the input, and $\tilde{P}_y^k$ if y is the input. The distribution of r is uniform.

Let $q_z \stackrel{\text{def}}{=} \Pr_{r,a}(T(r,a) \notin C_{f,\epsilon}(z))$ for an input z. By definition, $q_z \leq \delta$ for every input z. We write q_z as $q_z = \sum_{a\in A^k} \tilde{P}_z^k(a) Q_z(a)$, where $Q_z(a) \stackrel{\text{def}}{=} \Pr_r(T(r,a) \notin C_{f,\epsilon}(z))$.

Note that since $C_{f,\epsilon}(x) \cap C_{f,\epsilon}(y) = \emptyset$, we have $Q_y(a) \geq \Pr_r(T(r,a) \in C_{f,\epsilon}(x)) = 1-Q_x(a)$.

We can rewrite the sum q_x+q_y as follows:

$$\begin{aligned} q_x+q_y &= \sum_{a\in A^k} \tilde{P}_x^k(a)Q_x(a) + \tilde{P}_y^k(a)Q_y(a) \\ &\geq \sum_{a\in A^k} \tilde{P}_x^k(a)Q_x(a) + \tilde{P}_y^k(a)(1-Q_x(a)) \\ &\geq \sum_{a\in A^k} \min(\tilde{P}_x^k(a),\tilde{P}_y^k(a)) \\ &= \frac{1}{2}\sum_{a\in A^k} \tilde{P}_x^k(a)+\tilde{P}_y^k(a) - |\tilde{P}_x^k(a)-\tilde{P}_y^k(a)| \\ &= 1-d(\tilde{P}_x^k,\tilde{P}_y^k) \end{aligned}$$

Since $q_x+q_y \leq 2\delta$, then $d(\tilde{P}_x^k,\tilde{P}_y^k) \geq 1-2\delta$. □

Lemma 13 yields distinguishability of $\tilde{P}_x^k$ and $\tilde{P}_y^k$. In order to use Lemma 12, we need to show that this also implies distinguishability between P_x^k and P_y^k. The proof of the following lemma is technical and can be found in the full version of the paper.

Lemma 14. *For all $k \leq n/4$ and for large enough n, if $d(\tilde{P}_x^k,\tilde{P}_y^k) \geq \alpha > 1/2$, then*

$$d(P_x^{2k},P_y^{2k}) \geq (1-O(\frac{1}{n}))\, d(\tilde{P}_x^k,\tilde{P}_y^k).$$

We use this lemma to complete the proof of Theorem 8:

Proof of Theorem 8. Let T be the optimal decision tree that (ϵ,δ)-approximates f. Denote by k the complexity of T ($k = S_{\epsilon,\delta}^w(f)$). By Lemma 9, we can assume without loss of generality that T is uniform.

Let x and y be the ϵ-disjoint inputs that achieve the function's minimum Hellinger distance: $h(P_x,P_y) = h_\epsilon(f)$. Lemma 13 implies $d(\tilde{P}_x^k,\tilde{P}_y^k) \geq 1-2\delta$. Since $k \leq n/4$, and $1-2\delta > 1/2$, we can apply Lemma 14, which implies $d(P_x^{2k},P_y^{2k}) \geq (1-O(1/n))(1-2\delta) \geq 1-2\delta-O(1/n)$. Application of the distinguishability lower bound (Lemma 12) completes the proof. □

5. APPLICATIONS

5.1 Statistical Moments

The k-th statistical moment, $\mu_k : [0,1]^n \rightarrow \mathbf{R}$, is defined as $\mu_k(x_1,\ldots,x_n) = \frac{1}{n}\sum_{i=1}^n x_i^k$. The first moment, for example, is simply the mean. An additive approximation of μ_k can be easily performed by taking a random sample of size $O(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$, and computing the sample's k-th moment. Chernoff-Heoffding bound [10, 18] implies that this yields an ϵ-additive approximation with probability $1-\delta$. Canetti et al. [7] prove that the mean (and, implicitly, all the other moments too) require $\Omega(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$ samples. Using Theorem 8 we obtain a much simpler proof for this result:

THEOREM 15. *For every* $\Omega(1/\sqrt{n}) \le \epsilon \le 1/\sqrt{8}, \Omega(1/n) \le \delta < 1/4$, $S^w_{\epsilon,\delta}(\mu_k) \ge \Omega(\frac{1}{\epsilon^2}\ln\frac{1}{\delta})$

PROOF. Consider the two inputs (or equivalently, distributions) x and y. x is defined to be 0 with probability $1/2-\epsilon$ and 1 otherwise; y is defined to be 0 with probability $1/2+\epsilon$ and 1 otherwise. Since $\mu_k(x) = 1/2+\epsilon$ and $\mu_k(y) = 1/2-\epsilon$, x and y are ϵ-disjoint. We bound the Hellinger distance of x and y as follows: $h^2(P_x,P_y) = 1 - 2\sqrt{(1/2+\epsilon)(1/2-\epsilon)} = 1-\sqrt{1-4\epsilon^2} \le 1-(1-4\epsilon^2) = 4\epsilon^2$. The lower bound follows from Theorem 8. □

5.2 Frequency Moments

Given a sequence $x = x_1,\ldots,x_n$ where $x_i \in [m]$, the k-th frequency moment of x is defined to be $F_k(x) \stackrel{\text{def}}{=} \sum_{i=1}^m q_i^k$, where $q_i = |\{x_j \mid x_j = i\}|$, the number of occurrences of i in the sequence. We prove upper and lower bounds for a relative approximation of F_k that are almost matching.

THEOREM 16. *Let* $r = \min\{m,n\}$. *Then, for any* $k \ge 2$, *and for all* $\Omega(1/n^{1/2k}) \le \epsilon < 1$, $\Omega(1/n) \le \delta < 1/4$,

(1) $\Omega(r^{1-1/k}\frac{1}{\epsilon^{1/k}}\log\frac{1}{\delta}) \le S^w_{\epsilon,\delta}(F_k)$

(2) $S^w_{\epsilon,\delta}(F_k) \le O(kr^{1-1/k}\frac{1}{\epsilon^{2/(k-1)}}\log\frac{1}{\delta})$

PROOF. For simplicity, we assume $m \ge n$. For the lower bound, we consider two inputs. The first input $x = 1,\ldots,n$. The second input is constructed from x as follows. Divide the input of length n into $n^{1-1/k}/(3\epsilon)^{1/k}$ blocks of size $(3\epsilon n)^{1/k}$. y is x except in the first block where it is all 1's. Clearly, $F_k(x) = n$ while $F_k(y) = 3\epsilon n + (n-(3\epsilon n)^{1/k}) \approx n(1+3\epsilon)$. Since $C_{F_k,\epsilon}(x) \cap C_{F_k,\epsilon}(y) = \emptyset$ and $h^2(P_x,P_y) \le O(\epsilon^{1/k}/n^{1-1/k})$, the lower bound follows from Theorem 8.

Now, we show that this lower bound is almost tight. Note that for any input x, $P_x(i) = q_i/n$. Thus, computing $F_k(x)$ is equivalent to estimating the k-wise collision probability of P_x, since $F_k(x) = \sum_{i=1}^m q_i^k = n^k\sum_{i=1}^n P_x(i)^k$. So, the sampling algorithm picks ℓ samples $x_1,\ldots,x_\ell$ using P_x and counts the k-wise collisions C_k, our basic estimator. It is clear that $E(C_k) = \binom{\ell}{k}\|P_x\|_k^k$. The following lemma, whose proof generalizes that of [16] for F_2, helps us to bound the variance of this estimator.

LEMMA 17. $Var(C_k) \le O(E(C_k)^{1+1/k})$.

Now, noting that $\|P_x\|_k^k \ge 1/r^{k-1}$, by Chebyshev inequality, $\Pr(|C_k - E(C_k)| \ge \epsilon E(C_k)) \le O(1/(\epsilon^2 E(C_k)^{1-1/k})) \le O(k^{k-1}r^{(k-1)^2/k}/(\epsilon^2\ell^{k-1}))$. For this to be a constant, say, 3/4, we get $\ell \ge \Omega(kr^{1-1/k}/\epsilon^{2/(k-1)})$. Now, this experiment can be repeated $O(\log 1/\delta)$ times and the median of these various trials is correct with probability at least $1-\delta$. □

We next give a simple new proof for the sampling lower bound of Charikar et al. [9] for a ratio approximation of F_0:

THEOREM 18. *For all* $\Omega(1/\sqrt{n}) \le \epsilon \le 1/2$, *and* $\Omega(1/n) \le \delta < 1/4$, $S^w_{\epsilon,\delta}(F_0) \ge \Omega(\epsilon^2 n\log\frac{1}{\delta})$.

PROOF. Let x be the input that is all 1's and let y be the input that has $n-\lfloor(1/\epsilon^2)\rfloor$ 1's, followed by $2,3,\ldots,\lfloor(1/\epsilon^2)\rfloor+1$. Note that $F_0(z)$ counts the number of distinct elements in z, and thus $F_0(x) = 1, F_0(y) \ge 1/\epsilon^2$, implying x and y are ϵ-disjoint. Now, $h^2(P_x,P_y) = 1-\sqrt{(n-\lfloor(1/\epsilon^2)\rfloor)/n} \le 1/(n\epsilon^2)$. The lower bound follows from Theorem 8. □

5.3 Selection Functions

Chernoff bound implies that the median of a random sample of size $O(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$ yields an ϵ-approximate median. The same result holds for Select_q for any constant $0 < q < 1$. However, for $q = 0$ (minimum) and for $q = 1$ (maximum), a sample of size $O(\frac{1}{\epsilon}\log\frac{1}{\delta})$ is sufficient. We next show that these bounds are tight:

THEOREM 19. *For every* $\Omega(1/\sqrt{n}) \le \epsilon \le 1/2, \Omega(1/n) \le \delta < 1/4$,

(1) For $\epsilon \le q \le 1-\epsilon$, $S^w_{\epsilon,\delta}(Select_q) \ge \Omega(\frac{q(1-q)}{\epsilon^2}\log\frac{1}{\delta})$

(2) For $q < \epsilon, q > 1-\epsilon$, $S^w_{\epsilon,\delta}(Select_q) \ge \Omega(\frac{1}{\epsilon}\log\frac{1}{\delta})$

PROOF. Assume, initially, that $\epsilon \le q \le 1-\epsilon$. We pick $x,y \in \{0,1\}^n$ as follows: x has $(q+\epsilon)n$ 0's and the rest are 1's, and y has $(q-\epsilon)n$ 0's and the rest are 1's. Clearly, $C_{\text{Select}_{q,\epsilon}}(x) = \{0\}$ and $C_{\text{Select}_{q,\epsilon}}(y) = \{1\}$, implying x and y are ϵ-disjoint. Since $h^2(P_x,P_y) \le \epsilon^2/(q(1-q))$, the bound follows from Theorem 8.

When $q < \epsilon$, we pick x as before and $y = 1^n$, and when $q > 1-\epsilon$, we pick y as before and $x = 0^n$. The fact that $h^2(x,y) \le 2\epsilon$ implies the lower bound. □

5.4 Others

There are several problems like the longest increasing subsequence, Spearman's distance metrics between permutations [12], and L_p distances between vectors, for which we can easily show an $\Omega(n)$ lower bound. This is mainly because there is no $\omega(1)$ lower bound on the values of these functions. In fact, the lower bounds can be stated in terms of a lower bound on their range size.

6. TIGHTNESS OF THE BOUNDS

In this section, we discuss the tightness of our bounds. First, we show that for symmetric functions the lower bound achieved in Section 4 is always at least as strong as the lower bound via block sensitivity. The tightness is stated in terms of the function's *minimum variation distance* (denoted $d_\epsilon(f)$), which is the analogue of $h_\epsilon(f)$ for the variation distance.

THEOREM 20. *For all* $\epsilon \ge 0$ *and every* ϵ*-symmetric function* $f : A^n \to B$, $\frac{1}{h^2_\epsilon(f)} \ge \frac{1}{d_\epsilon(f)} \ge bs_\epsilon(f)$.

PROOF. The first inequality follows from Proposition 11. For the second inequality let x be the input that achieves the block sensitivity of f, and let $I \subseteq [n]$ be the largest block to which f is ϵ-sensitive on x. Note that $bs_\epsilon(f) = bs_\epsilon(f,x) \le n/|I|$.

Let $Q \in A^{|I|}$ be the assignment to I such that x and $y = x^{(I\leftarrow Q)}$ are ϵ-disjoint. Since x and y differ in $|I|$ positions, then $d(P_x,P_y) \le |I|/n$. Therefore, by definition, $d_\epsilon(f) \le |I|/n$. The theorem follows. □

We now show that the lower bound achieved in Section 4 is reasonably tight for functions over small finite domains (e.g., functions over a Boolean domain). The proof is due to Luca Trevisan [personal communication, March 2001].

THEOREM 21. *If $f : A^n \to B$ is an ϵ-symmetric function with approximation $C_{f,\epsilon}$ that satisfies the sunflower property, then $S^w_{\epsilon,\delta}(f) \le O(|A| \frac{1}{d^2_\epsilon(f)} \ln \frac{1}{\delta})$.*

PROOF. Given input x, the algorithm for approximating f will try to construct a distribution Q_x which approximates P_x in variation distance to within an additive factor of $\beta = d_\epsilon(f)/18$.

The algorithm constructs Q_x by picking k independent samples $x_1, \ldots, x_k$ uniformly at random from x, and computing for all $a \in A$: $Q_x(a) = (1/k)|\{x_i \mid x_i = a\}|$. We choose $k = |A|/\beta^2$.

Define $e_a = |Q_x(a) - P_x(a)|$, the error made on a; we bound $E(e_a)$ as follows: $E(e_a)^2 \le E(e_a^2) = E(|Q_x(a) - P_x(a)|^2) = \mathrm{Var}(Q_x(a))$. Define for all $i \in [k]$, $Z_i = 1$, if $x_i = a$ and $Z_i = 0$, otherwise. Then, $Q_x(a) = \frac{1}{k}\sum_{i\in[k]} Z_i$. We obtain $E(e_a)^2 \le \mathrm{Var}(\frac{1}{k}\sum_{i\in[k]} Z_i) = \frac{1}{k^2}\sum_{i\in[k]}\mathrm{Var}(Z_i) = \frac{1}{k^2}\cdot k \cdot P_x(a)(1-P_x(a)) \le P_x(a)/k$. Now, if $P_x(a) \le 1/|A|$, then $E(e_a) \le \sqrt{(1/|A|)(\beta^2/|A|)} = \beta/|A|$. If $P_x(a) > 1/|A|$, then $E(e_a) \le \sqrt{P_x(a)(\beta^2/|A|)} \le \sqrt{P_x(a)^2\beta^2} = \beta P_x(a)$. Thus, for all $a \in A$, $E(e_a) \le \max\{\beta/|A|, \beta P_x(a)\}$.

Note that $d(Q_x, P_x) = \frac{1}{2}\sum_{a\in A} e_a$. Thus,

$$\begin{aligned} E(d(Q_x,P_x)) &= \frac{1}{2}\sum_{a\in A} E(e_a) \\ &\le \frac{1}{2}\left(\sum_{a:P_x(a)\le\frac{1}{|A|}} \frac{\beta}{|A|} + \sum_{a:P_x(a)>\frac{1}{|A|}} \beta P_x(a)\right) \\ &\le \frac{1}{2}(|A|\cdot(\beta/|A|) + \beta\cdot 1) = \beta. \end{aligned}$$

Therefore, by Markov's inequality, $\Pr(d(Q_x,P_x) > 3\beta) < 1/3$.

The algorithm will generate ℓ independent approximations $Q_x^1, \ldots, Q_x^\ell$ for P_x, where $\ell = O(\ln(1/\delta))$. We call an approximation Q_x^i "good", if it is at distance of at most 3β from P_x. Chernoff bound implies that with probability at least $1-\delta$, more than half of the approximations are good. Note that by the triangle inequality every two good approximations are at distance of at most 6β from each other. The algorithm thus picks an approximation Q_x^j that is of distance at most 6β from more than half of the other approximations, if one exists. If indeed more than half of the approximations are good, then (1) Q_x^j exists (each good approximation can serve as Q_x^j) and (2) any Q_x^j we pick has at least one good approximation at distance of at most 6β; therefore, Q_x^j is at distance of at most 9β from P_x. We conclude that with probability $1-\delta$ our algorithms finds an approximation Q_x that satisfies $d(Q_x,P_x) \le 9\beta = d_\epsilon(f)/2$.

The algorithm now looks for the set D of all inputs y for which $d(Q_x,P_y) < d_\epsilon(f)/2$. What we proved above shows that with probability $1-\delta$, $x \in D$. Moreover, by the triangle inequality for every two inputs $y, y' \in D$, $d(P_y, P_{y'}) < d_\epsilon(f)$. It follows from the definition of $d_\epsilon(f)$ that $C_{f,\epsilon}(y) \cap C_{f,\epsilon}(y') \ne \emptyset$. Using now the sunflower property of $C_{f,\epsilon}$, we obtain that there exists a value $b \in \bigcap_{y\in D} C_{f,\epsilon}(y) \subseteq C_{f,\epsilon}(x)$. The algorithm outputs b. □

The analysis of the technique applied in this theorem is tight in terms of its dependence on $|A|$, since there is a lower bound of $\Omega(|A|)$ on the number of samples required to approximate an unknown distribution in L_1 distance [3]. Therefore, in order to obtain a better upper bound one should apply a different technique.

For small domains this upper bound can be tight up to a constant factor. The tightness depends on the difference between $d_\epsilon(f)$ and $h_\epsilon(f)$. In the Boolean median, for example, $d_\epsilon(\text{median}) = h_\epsilon(\text{median}) = \epsilon$, which means that the bound is tight. In any case, since $d_\epsilon(f) \ge h^2_\epsilon(f)$, the difference between the upper and lower bounds is at most quadratic for small domains.

Finally, we show that block sensitivity is not a good lower bound in some cases.

PROPOSITION 22. *There is a function $f : A^n \to B$ such that $S^w_{\epsilon,\delta}(f) = \Omega(\sqrt{n})$, while $bs_\epsilon(f) = O(1/\epsilon)$.*

PROOF. Let f be the Boolean function on bounded-degree d graphs with n vertices such that $f(x) = 1$ iff x is a bipartite graph. x is represented by the graph's edge list (a list of dn integers in $[n]$). We consider the property testing approximation. If x is bipartite, to make it ϵ-far from being bipartite, we need to make at least ϵdn modifications to its edge list, implying $\Omega(\epsilon dn)$-sized blocks, which implies only $O(1/\epsilon)$ block sensitivity. On the other hand, Goldreich and Ron [15] prove that $S^w_{\epsilon,\delta}(f) = \Omega(\sqrt{n})$. □

7. LOSSY COMPRESSION

In the context of computing with massive data sets, the following question is fairly natural and extremely important: what computations can be performed efficiently when data is available in compressed form? Specifically, it is of much interest to find algorithms that are able to compute interesting functions without completely decompressing the compressed data. Applications include pattern matching on compressed text files [2], operations such as scene change detection and abrupt lighting change detection on video sequences [8], and nearest neighbor computations.

This question becomes more interesting if, to gain larger compression factors, we are willing to accept some loss in the precision of the computation. The idea of *lossy compression* has been around quite awhile, and is used in multimedia applications, where images and video sequences are routinely stored in compressed form. The guideline in these areas is that a lossy compression method is good if a typical human observer cannot tell the difference between an image and its version stored using the compression technique.

To our knowledge, there is no formal treatment of the notion of approximate computation from data stored using a lossy compression technique. The closest is the notion of computing from sketches, described in [13, 5], and our definition below may be thought of as an extension of theirs.

DEFINITION 23 (LOSSY COMPRESSION SCHEME). *For $\epsilon, \delta > 0$, a function $f : A^n \times F \to B$ is said to* admit a (c,ϵ,δ)-lossy compression scheme *if there exists a (probabilistic) compression function $\phi : A^n \to E$ and a (probabilistic) approximating function $\widetilde{f} : E \times F \to B$ such that:*

(1) $\log|E|/(n\log|A|) \le c$;

(2) for all inputs $x \in A^n, y \in F$, $\widetilde{f}(\phi(x), y) \in C_{f,\epsilon}(x,y)$ with probability at least $1-\delta$.

It is natural to wonder why f is taken to be a bivariate function in the definition above. If not, the trivial compression function $\phi = f$ and the trivial approximating function $\widetilde{f} = \text{IDENTITY}$ would be a perfectly good compression scheme. That is, if we wish to be able to compute $f(x)$ from the compression $\phi(x)$ of x, simply precompute $f(x)$ and let $\phi(x) = f(x)$. The point, in all real applications, is to be able to compress x without the knowledge of which y will be the argument for the computation of $f(x, y)$ (possibly in the future). For example, with image compression, the role of y is played by (reasonable) human observers, and the goal is to compress an image so that for any observer, the image and its compressed version look similar. In nearest neighbor computation, the role of y is played by a future "query" input for which we need to find the nearest database point from a set of database points stored in compressed form. The situation is similar for pattern matching.

We now point out that under certain conditions, a sampling algorithm for f implies a lossy compression scheme for f. The proof of the following proposition is straightforward.

Proposition 24. *If $f : A^n \times F \to B$ has an (ϵ, δ) sampling algorithm that, to compute $f(x, y)$, makes oblivious, uniformly random queries to entries of x, then f admits a (c, ϵ, δ)-lossy compression scheme, where c depends on the maximum space needed to store the internal state of the sampling algorithm.*

7.1 Streaming Algorithms

Consider a function $f : A^n \to B$. In the basic streaming model, a stream is an ordered sequence of pairs (i, x_i) where $i \in [n], x_i \in A$ that is presented on a unidirectional read-once tape. Each $i \in [n]$ occurs exactly once in the stream. The requirement of a streaming algorithm is that it should output the correct (possibly approximate) value of f irrespective of the ordering of the input sequence.

Definition 25. *An (s, t)-*streaming algorithm *for a function $f : A^n \to B$ is a streaming algorithm for f that, for every $\epsilon, \delta \geq 0$ uses work space $s = s(n, |A|, \epsilon, \delta)$, time $t = t(n, |A|, \epsilon, \delta)$ per data item, and for any input x with probability at least $1 - \delta$, outputs a $y \in C_{f,\epsilon}(x)$.*

Our next observation is that every sampling algorithm (a randomized decision tree) yields a streaming algorithm in a natural way.

Proposition 26. *A non-adaptive sampling algorithm for f that works for all $\epsilon, \delta \geq 0$ implies a streaming algorithm for f that uses space at most $S^w_{\epsilon,\delta}(f) \cdot O(\log |A| + \log n)$.*

Proof. Construct a streaming algorithm in the following manner. Without loss of generality, let the non-adaptive sampling algorithm toss coins to pick $S^w_{\epsilon,\delta}(f)$ indices to sample. Let this index set be I. Now, on an input stream consisting of tuples (i, x_i), if $i \in I$, the streaming algorithm stores x_i. After it has read all the samples, it runs the rest of the sampling algorithm. Clearly, the space used by the streaming algorithm is $S^w_{\epsilon,\delta}(f) \cdot O(\log |A| + \log n)$. $\square$

Finally, we note that a streaming algorithm for a function yields a lossy compression scheme for the function. In particular, if the function is a metric, this yields a nearest neighbor scheme as well.

Proposition 27. *A streaming algorithm for $f : A^n \times F \to B$ that uses space s implies, for every $\epsilon, \delta \geq 0$ a (c, ϵ, δ)-lossy compression scheme for f, where $c = O(s/(n \log |A|))$.*

Proof. Given a streaming algorithm T for f that runs in space s, the compression function $\phi(x)$ is given by the contents of the work tape of T (of size $\leq s$) when T is run on x, pretending that it arrives as a stream. The approximating function $\widetilde{f}$, given $\phi(x), y$, resumes the streaming algorithm T initializing the work tape to $\phi(x)$, and supplying y in a stream (in arbitrary order), and outputs the output of T. $\square$

A partial converse to the above was proved by making additional assumptions about the encoding function of the lossy compression [13]. We omit all the parameters in the following statement.

Proposition 28. *[13] If f has a lossy compression scheme whose compression function has a streaming algorithm, then f has a streaming algorithm.*

7.2 Nearest Neighbor Schemes

We point out one interesting application of the connection to lossy compression. We define a metric nearest neighbor scheme and show that a lossy compression scheme for the metric yields an efficient nearest neighbor scheme.

Definition 29 (Nearest neighbor scheme). *For a metric space A^d with metric f, an (s, t)-*nearest neighbor scheme *is an algorithm that, for any $\epsilon, \delta \geq 0$, preprocesses N points $X = \{x_1, \ldots, x_N\} \subseteq A^d$ into a storage of size $s = s(N, d, \epsilon, \delta)$ such that for any query $y \in A^d$, with probability at least $1 - \delta$, it outputs an $x \in X$ in time $t = t(N, d, \epsilon, \delta)$ such that for all $x' \in X$, $f(x, y) \leq (1 + \epsilon) f(x', y)$.*

Proposition 30. *If $f : A^d \times A^d \to \mathbf{R}$ is a metric and has a (c, ϵ, δ)-lossy compression scheme with compression function ϕ and approximating function $\widetilde{f}$, then there is an $(Ncd \log |A|, Nt(\epsilon, \delta/N))$-nearest neighbor scheme for A^d with f as the metric, where $t(\epsilon, \delta)$ denotes the time for one application of $\widetilde{f}$ to succeed with error ϵ and confidence $1 - \delta$.*

Proof. We can take each $x_i \in X$ and compress under C to get a signature z_i of size $cd \log |A|$ and store it. Upon a query $y \in A^d$, for each stored signature z_i, we use the approximating function $\widetilde{f}$ to find the x_i such that $\widetilde{f}(z_i, y)$ is the minimum. The proposition follows from the properties of lossy compression. $\square$

The parameters of the approximate nearest neighbor algorithm from Proposition 30 are quite poor in terms of N, since the algorithm simply applies $\widetilde{f}$ to each database item. If $\widetilde{f}$ is also a distance function in a smaller dimensional space, then one may recursively apply any of the known approximate nearest neighbor computation techniques to improve the speed. Unfortunately, $\widetilde{f}$ often involves the median operator, which is not a distance function. In an earlier version of this paper, we had claimed that this problem could be circumvented, but this claim appears to be erroneous. Independently, Indyk [19] has shown how to handle some of the complications that arise due to the median; he applies the streaming algorithms of [1, 13] for distance functions to obtain embeddings of metric spaces into metric spaces.

8. DISCUSSION AND RELATED WORK

This paper concentrates on a worst-case analysis of the (expected and worst-case) query complexity. Specifically, we defined the query complexity of a decision tree as its query complexity on the "worst" input. However, sometimes the query complexity varies significantly for different inputs and a worst-case analysis gives too pessimistic a view. For example, the query complexity of a relative approximation of the mean is $\Omega(n)$, since in such an approximation the inputs 0^n and $0^{n-1}1$ are ϵ-disjoint, while having Hellinger distance of $O(1/\sqrt{n})$. However, the mean of an input $x \in \{0,1\}^n$ with a constant number of 1's can be approximated with only $O(\epsilon^{-2} \log \delta^{-1})$ queries. This motivates the definition of the query complexity of a function $f : A^n \to B$ on a *specific* input $x \in A^n$, as $S^e_{\epsilon,\delta}(f,x) = \min\{S^e_{\epsilon,\delta}(T,x) \mid T \ (\epsilon,\delta)\text{-approximates } f\}$ (and similarly for $S^w_{\epsilon,\delta}(f,x)$). Our two lower bounds (Theorem 5 and Theorem 8) can be stated in terms of the query complexity of f on x: $S^e_{\epsilon,\delta}(f,x) \geq (1-2\delta)bs_\epsilon(f,x)$ and $S^w_{\epsilon,\delta}(f,x) \geq \Omega(\frac{1}{h^2_\epsilon(f,x)} \log \frac{1}{\delta})$, where $h^2_\epsilon(f,x) = \min\{h^2(P_x,P_y) \mid y \text{ is } \epsilon\text{-disjoint from } x\}$. Using this formulation, one can prove, for example, that a relative approximation of the mean on input $x \in \{0,1\}^n$ requires $\Omega(\frac{n}{\epsilon^2|x|} \log \frac{1}{\delta})$ queries, where $|x|$ denotes the Hamming weight of x.

Our decision tree model is a generalization of Boolean decision trees, which are suitable for exact computation of Boolean functions. Boolean query complexity was extensively studied, and is known to be related to various Boolean function properties, such as sensitivity, certificate complexity, and degree of representing and approximating polynomials (see [6] for a survey). In this paper we explored only the generalization of block sensitivity and its connection to query complexity of general functions. Similar generalizations are possible also for the other properties; however, as mentioned in Section 3, not all the connections between these properties and the query complexity seem to carry over to the general case.

We are aware of only a few lower bounds in the theoretical computer science literature on the query complexity of non-Boolean function approximations; all of these are tailored to specific functions. Canetti et al. [7] show a lower bound for additive approximation of the mean; Dagum et al. [11] (and also, implicitly, Schulman and Vazirani [25]) give lower bounds for relative approximation of the mean on any input x. Charikar et al. [9] prove a lower bound for ratio approximation of the frequency moment of order 0. Nayak and Wu [22] give a lower bound on the *quantum* query complexity of the median and some other statistics.

Statistical decision theory [4] studies the process of making decisions under incomplete information. The decision procedure can gain information about the environment by sampling from a sample space that depends on the true state of the environment. It then makes a decision so as to maximize its expected utility with respect to this true state. Our function approximation problem can be formulated in this setting, by associating the input with the unknown environment; the decision we wish to make is the value of the function on the unknown input; our utility is 1 if we succeed to output a good approximation and 0 otherwise. Classical decision theory is mainly concerned with finding methods for maximizing the expected utility for a given fixed sample space. In particular, there is no notion of cost associated with the sample (e.g., the sample size). Thus, this theory does not provide explicit bounds on the number of samples needed to guarantee a prescribed utility.

A classical result in estimation theory, a variant of decision theory, is the Cramér-Rao inequality (see, e.g., [28], Chapter 2.4) which provides a lower bound on the variance of any unbiased estimator for an unknown parameter (that is, an estimator whose expectation is the true value of the parameter) in terms of the *Fisher information* of the sample distribution. It is not clear how, if at all, to translate such a bound to a lower bound on the number of samples needed to make the estimate have a prescribed error with a prescribed confidence. In any case, since the bound handles only unbiased estimators, it is not appropriate for our general setting.

Statistical learning theory [27, 20, 29] is concerned with learning an unknown function from a class of target functions. It uses a small sample of inputs labeled with the true value of the unknown function on them to determine a hypothesis function that approximates the unknown function on any input. To compare this model to ours, one can view the inputs of our function f as functions: an input $x \in A^n$ is a mapping from $[n]$ to A that maps i to x_i. The main difference between our model and the learning model is thus, that in learning one is interested in approximating the function itself, whereas in our model one is interested in approximating some parameter of the function (e.g., its mean). In particular, the VC dimension based lower bounds on the number of samples needed to learn a function do not seem to carry over to our general case.

Statistical sequential analysis [26] is a variant of decision theory, in which the sample space is composed of separate observations that are performed sequentially. At each step the decision procedure can either continue sampling or stop and output a decision. The number of observations made is incorporated into the utility function. Thus, a primary goal of the procedure is to minimize the expected number of samples made. A fundamental result in sequential analysis (see, e.g., [26], Chapter 1) is the optimality of the sequential probability ratio test, which gives a lower bound on the expected number of samples needed to solve the following decision problem: a distribution P is known to be either P_0 or P_1; the procedure can draw independent samples from P, and has to decide with confidence $1-\delta$ whether $P = P_0$ or $P = P_1$. The lower bound is given in terms of δ and the KL (Kullback-Leibler) distance between P_0 and P_1.

Acknowledgments

We thank Ron Fagin, Michael Jordan, Christos Papadimitriou, Moni Naor, Andrew Ng, Luca Trevisan, and David Zuckerman for useful discussions.

9. REFERENCES

[1] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1):137–147, 1999.

[2] A. Amir, G. Benson, and M. Farach. Let sleeping files lie: Pattern matching in Z-compressed files. *Journal of Computer and System Sciences*, 52(2):299–307, 1996.

[3] T. Batu, L. Fortnow, R. Rubinfeld, W. D. Smith, and P. White. Testing that distributions are close. In *Proceedings of the 41st IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 259–269, 2000.

[4] J. O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, 1985.

[5] A. Broder, M. Charikar, A. Frieze, and M. Mitzenmacher. Min-wise independent permutations. In *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 327–336, 1998.

[6] H. Buhrman and R. de Wolf. Complexity measures and decision tree complexity: A survey, 1999. Available at `http://www.cwi.nl/~rdewolf`.

[7] R. Canetti, G. Even, and O. Goldreich. Lower bounds for sampling algorithms for estimating the average. *Information Processing Letters*, 53:17–25, 1995.

[8] S.-F. Chang. Compressed-domain techniques for image/video indexing and manipulation. In *Invited article in IEEE International Conference on Image Processing, Special Session on Digital Image/Video Libraries and Video-on-demand*, 1995.

[9] M. Charikar, S. Chaudhuri, R. Motwani, and V. Narasayya. Towards estimation error guarantees for distinct values. In *Proceedings of the 19th Annual ACM Symposium on Principles of Database Systems (PODS)*, pages 268–279, 2000.

[10] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *American Mathematical Society*, 23:493–507, 1952.

[11] P. Dagum, R. Karp, M. Luby, and S. Ross. An optimal algorithm for monte carlo estimation. In *Proceedings of the 36th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 142–149, 1995.

[12] P. Diaconis. *Group Representation in Probability and Statistics*. IMS Lecture Series 11, Institute of Mathematical Statistics, 1999.

[13] J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan. An approximate L^1-difference algorithm for massive data streams. In *Proceedings of the 40th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 501–511, 1999.

[14] J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan. Testing and spot-checking of data streams. In *Proceedings of the 11th IEEE Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 165–174, 2000.

[15] O. Goldreich and D. Ron. Property testing in bounded degree graphs. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 406–415, 1997.

[16] O. Goldreich and D. Ron. On testing expansion in bounded-degree graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 2000. TR00-020.

[17] M. Henzinger, P. Raghavan, and S. Rajagopalan. Computing on data streams. In *DIMACS series in Discrete Mathematics and Theoretical Computer Science*, volume 50, pages 107–118, 1999.

[18] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

[19] P. Indyk. Stable distributions, pseudorandom generators, embeddings and data stream computation. In *Proceedings of the 41st IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 189–197, 2000.

[20] M. J. Kearns and U. V. Vazirani. *An Introduction to Computational Learning Theory*. The MIT Press, 1994.

[21] L. Le Cam and G. Lo Yang. *Asymptotics in Statistics - Some Basic Concepts*, pages 24–30. Springer-Verlag, 1990.

[22] A. Nayak and F. Wu. The quantum query complexity of approximating the median and related statistics. In *Proceedings of the 31st Annual ACM Symposium on the Theory of Computing (STOC)*, pages 384–393, 1999.

[23] N. Nisan. CREW PRAMs and Decision Trees. *SIAM Journal on Computing*, 20(6):999–1007, 1991.

[24] J. Radhakrishnan and A. Ta-Shma. Tight bounds for depth-two superconcentrators. In *Proceedings of the 38th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 585–594, 1997.

[25] L. Schulman and V. V. Vazirani. Majorizing estimators and the approximation of #P-complete problems. In *Proceedings of the 31st Annual ACM Symposium on the Theory of Computing (STOC)*, pages 288–294, 1999.

[26] D. Siegmund. *Sequential Analysis - Tests and Confidence Intervals*. Springer-Verlag, 1985.

[27] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[28] H. L. Van Trees. *Detection, Estimation, and Modulation Theory*. Jon Wiley & Sons, Inc., 1968.

[29] V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, Inc., 1998.

[30] A.-C. Yao. Probabilistic computations: toward a unified measure of complexity. In *Proceedings of the 18th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 222–227, 1977.