The complexity of distributions[*]

Emanuele Viola[†]

November 5, 2010

Abstract

Complexity theory typically studies the complexity of computing a function $h(x) : \{0,1\}^m \to \{0,1\}^n$ of a given input x. A few works have suggested to study the complexity of generating – or sampling – the distribution $h(x)$ for uniform x, given random bits. We further advocate this study, with a new emphasis on lower bounds for restricted computational models. Our main results are:

1. Any function $f : \{0,1\}^\ell \to \{0,1\}^n$ such that (i) each output bit f_i depends on $o(\log n)$ input bits, and (ii) $\ell \le \log_2 \binom{n}{\alpha n} + n^{0.99}$, has output distribution $f(U)$ at statistical distance $\ge 1 - 1/n^{0.49}$ from the uniform distribution over n-bit strings of hamming weight αn.

   We also prove lower bounds for generating $(X, b(X))$ for boolean b, and in the case in which each bit f_i is a small-depth decision tree.

   These lower bounds seem to be the first of their kind; the proofs use anti-concentration results for the sum of random variables.

2. Lower bounds for generating distributions imply succinct data structures lower bounds. As a corollary of (1), we obtain the first lower bound for the membership problem of representing a set $S \subseteq [n]$ of size αn, in the case where $1/\alpha$ is a power of 2: If queries "$i \in S$?" are answered by non-adaptively probing $o(\log n)$ bits, then the representation uses $\ge \log_2 \binom{n}{\alpha n} + \Omega(\log n)$ bits.

3. Upper bounds complementing the bounds in (1) for various settings of parameters.

4. Uniform randomized AC^0 circuits of poly(n) size and depth $d = O(1)$ with error ϵ can be simulated by uniform randomized AC^0 circuits of poly(n) size and depth $d+1$ with error $\epsilon + o(1)$ using $\le (\log n)^{O(\log \log n)}$ random bits.

   Previous derandomizations [Ajtai and Wigderson '85; Nisan '91] increase the depth by a constant factor, or else have poor seed length.

[*]An extended abstract of this work appeared in the *Proceedings of the 51th IEEE Symposium on Foundations of Computer Science (FOCS)*, see [Vio10].

[†]Supported by NSF grant CCF-0845003. Email: `viola@ccs.neu.edu`

1 Introduction

Complexity theory typically studies the complexity of computing a function $h(x) : \{0,1\}^m \to \{0,1\}^n$ of a given input x. A few works, such as the one by Goldreich, Goldwasser, and Nussboim [GGN10, §2.5], suggest to study instead the complexity of generating – or sampling – the output distribution $h(x)$ for random x, given random bits. This work further advocates this study, with a new emphasis on lower bounds for restricted models such as small bounded-depth circuits with unbounded fan-in (AC^0) or bounded fan-in (NC^0).

An interesting example of a function h for which computing $h(x)$ is harder than generating its output distribution is $h(x) := (x, \mathrm{parity}(x))$, where $\mathrm{parity}(x) := \sum_i x_i \mod 2$. Whereas small AC^0 circuits cannot compute parity (cf. [Hås87]), Babai [Bab87] and Boppana and Lagarias [BL87] show a function f whose output distribution equals that of $(x, \sum_i x_i \mod 2)$ for random $x \in \{0,1\}^n$, and each output bit f_i depends on just 2 input bits (so $f \in \mathrm{NC}^0$):

$$f(x_1, x_2, \ldots, x_n) := (x_1, x_1 + x_2, x_2 + x_3, \ldots, x_{n-1} + x_n,\ x_n). \tag{1}$$

This construction is useful for proving average-case lower bounds, see [Bab87] and [Bei93, Corollary 22].

Later, Impagliazzo and Naor [IN96] extend the construction (1) to show that small AC^0 circuits can even generate $(x, b(x))$ for more complicated functions, such as inner product $b(x) = x_1 \cdot x_2 + x_3 \cdot x_4 + \cdots + x_{n-1} \cdot x_n$. They use this to construct cryptographic pseudorandom generators computable by poly-size AC^0 circuits based on the hardness of the subset-sum problem, and similar techniques are useful in constructing depth-efficient generators based on other assumptions [AIK06, Vio05].

We mention that cryptography provides several candidate functions h for which computing $h(x)$ is harder than generating its output distribution (e.g., take h^{-1} to be a one-way permutation). However, in this work we focus on unconditional results.

The work by Mossel, Shpilka, and Trevisan [MST06] provides another example of the power of NC^0 circuits in generating distributions: NC^0 circuits can generate small-bias distributions with non-trivial stretch.

The surprising nature of the above constructions, and their usefulness (for example for average-case lower bounds and pseudorandom generators) raises the challenge of understanding the complexity of generating distributions, and in particular proving lower bounds:

Challenge 1.1. *Exhibit an explicit map $b : \{0,1\}^n \to \{0,1\}$ such that the distribution $(X, b(X)) \in \{0,1\}^{n+1}$ cannot be generated by $\mathrm{poly}(n)$-size AC^0 circuits given random bits.*

Current lower-bounding techniques appear unable to tackle questions such as Challenge 1.1 (which, to our knowledge, is open even for DNFs). As we have seen, standard "hard functions" b such as parity and inner product have the property that $(X, b(X))$ can be generated exactly by small AC^0 circuits. Along the way, in this work we point out that the same holds for any symmetric b (e.g., majority, mod 3) (up to an exponentially small error). In fact, weaker models often suffice.

This suggests that our understanding of even these simple models is incomplete, and that pursuing the above direction may yield new proof techniques.

1.1 Our results

In this work we prove several "first-of-their-kind" lower bounds for generating distributions. We also complement these with upper bounds, and establish connections to other areas such as succinct data structures, derandomization, and switching networks.

Lower bounds. We aim to bound from below the statistical (a.k.a. variation) distance Δ between a distribution D on n bits and the output distribution of a "simple" function $f : \{0,1\}^\ell \to \{0,1\}^n$ over random input $U \in \{0,1\}^\ell$:

$$\Delta(f(U), D) := \max_{T \subseteq \{0,1\}^n} \left| \Pr_U[f(U) \in T] - \Pr_D[D \in T] \right| = \frac{1}{2} \sum_a |\Pr[f(U) = a] - \Pr[D = a]|.$$

In addition to being a natural measure, small statistical distance (as opposed to equality) is sufficient in typical scenarios (e.g., pseudorandomness). Moreover, this work shows that statistical distance lower bounds imply lower bounds for succinct data structure problems, and uses this implication to derive a new lower bound for a central data structure problem (Corollary 1.7).

The next convenient definition generalizes NC^0 (which corresponds to $d = O(1)$).

Definition 1.2. *A function $f : \{0,1\}^\ell \to \{0,1\}^n$ is* d-local *if each output bit f_i depends on $\leq d$ input bits.*

Our first lower bound is for generating the uniform distribution $D_{=\alpha}$ over n-bit strings with αn ones (i.e., hamming weight αn). This distribution arises frequently. For example, we will see that it is related to generating $(X, b(X))$ for symmetric b, and to the membership problem in data structures.

Theorem 1.3 (Lower bound for generating "$= \alpha$" locally)**.** *For any $\alpha \in (0,1)$ and any $\delta < 1$ there is $\epsilon > 0$ such that for all sufficiently large n for which αn is an integer:*
Let $f : \{0,1\}^\ell \to \{0,1\}^n$ be an $(\epsilon \log n)$-local function where $\ell \leq \log_2 \binom{n}{\alpha n} + n^\delta$.
Let $D_{=\alpha}$ be the uniform distribution over n-bit strings with αn ones.
Then $\Delta(f(U), D_{=\alpha}) \geq 1 - O(1/n^{\delta/2})$.

For $\alpha = 1/2$, Theorem 1.3 matches the 1-local identity function $f : \{0,1\}^n \to \{0,1\}^n$, $f(u) := u$, achieving $\Delta(U, D_{=1/2}) \leq 1 - O(1/\sqrt{n})$ (a standard bound, see Fact 2.2). For $\alpha < 1/2$, upper bounds are a bit more involved. There are $\operatorname{poly}\log(n)$-local functions again achieving statistical distance $\leq 1 - O(1/\sqrt{n})$. We refine this to also obtain input length $\ell = \log_2 \binom{n}{\alpha n} + n/\operatorname{poly}\log n$ (Theorem 5.1).

For generating $(X, b(X))$ for boolean b obviously no lower bound larger than $1/2$ holds. We establish $1/2 - o(1)$ for the function which checks if the hamming weight of X modulo p is between 0 and $(p-1)/2$. We call it "majority modulo p," majmod for short.

Theorem 1.4 (Lower bound for generating $(X, \text{majmod } X)$ locally)**.** *For any $\delta < 1$ there is $\epsilon > 0$ such that for all sufficiently large n: Let $p \in [0.25 \log n, 0.5 \log n]$ be a prime number, and let* majmod : $\{0,1\}^n \to \{0,1\}$ *be defined as*

$$\text{majmod}(x) = 1 \Leftrightarrow \sum_{i \le n} x_i \mod p \in \{0, 1, \ldots, (p-1)/2\}.$$

Let $f : \{0,1\}^\ell \to \{0,1\}^{n+1}$ be an $(\epsilon \log n)$-local function where $\ell \le n + n^\delta$.
Then $\Delta(f(U), (X, \text{majmod } X)) \ge 1/2 - O(1/\log n)$.

Theorem 1.4 is tight up to the $O(.)$: it can be verified that $\Pr_X[\text{majmod}(X) = 0] = 1/2 - \Theta(1/\log n)$, hence $\Delta((X, 1), (X, \text{majmod } X)) \le 1/2 - O(1/\log n)$. Moreover, we show a poly $\log(n)$-local function with statistical distance $\le 1/n$ (Theorem 5.4).

Theorems 1.3 and 1.4 may hold even when the input length ℓ is unbounded, but it is not clear to us how to prove such statistical bounds in those cases. However we can prove weaker statistical bounds when the input length ℓ is unbounded, and these hold even against the stronger model where each bit of the function f is a *decision tree.* We call such a function *forest*, to distinguish it from a function computable by a single decision tree.

Definition 1.5. *A function $f : \{0,1\}^\ell \to \{0,1\}^n$ is a d-forest if each bit f_i is a decision tree of depth d.*

A d-forest function is also 2^d local, so the previous theorems yield bounds for $d = (\log(\epsilon \log n))$-forests. We prove bounds for $d = \epsilon \log n$ with a different argument.

Theorem 1.6 (Lower bound for generating "$= 1/2$" or $(X, \text{majority } X)$ by forest)**.** *Let $f : \{0,1\}^* \to \{0,1\}^n$ be a d-forest function. Then:*

(1) $\Delta(f(U), D_{=1/2}) \ge 2^{-O(d)} - O(1/n)$, where $D_{=1/2}$ is the uniform distribution over n-bit strings with $n/2$ ones.

(2) $\Delta(f(U), (X, \text{majority } X)) \ge 2^{-O(d)} - O(1/n)$.

A similar bound to (1) also holds for generating $D_{=\alpha}$; we pick $\alpha = 1/2$ for simplicity.

Theorem 1.6 complements the existence of d-forest functions achieving statistical distance $O(1/n)$ where $d = O(\log n)$ for (1) and $d = O(\log^2 n)$ for (2). (In fact, $d = O(\log n)$ may hold for both, see §6.) We obtain such functions by establishing a simple connection with results on *switching networks*, especially by Czumaj et al. [CKKL99]: we prove they imply forest upper bounds. These upper bounds are not explicit; explicit upper bounds are known for $d = \text{poly} \log n$, see §6.

For AC^0 circuits, there are constructions that are both explicit and achieve exponentially small error. In particular, building on results by Matias and Vishkin [MV91] and Hagerup [Hag91], we exhibit AC^0 circuits of size poly(n) and depth $O(1)$ whose output distribution has statistical distance $1/2^n$ from the distribution $(X, \sum_i X_i) \in \{0,1\}^n \times \{0, 1, \ldots, n\}$ for uniform $X \in \{0,1\}^n$.

The above lower bounds are obtained via new proof techniques also using anti-concentration results for the sum of random variables. We provide an overview of the proof of Theorem 1.3 in §2.

Motivation: Succinct data structures lower bounds. Succinct data structures aim to compress data using a number of bits close to the information-theoretic minimum while at the same time supporting interesting queries. For a number of problems, tight bounds are known, cf. [Păt08, Vio, PV10, DPT10]. But there remains a large gap for the notable *membership* problem which asks to store a subset x of $[n]$ of size ℓ (think of x as an n-bit string of weight ℓ) in $\lceil \log_2 \binom{n}{\ell} \rceil + r$ bits, where r is as small as possible, while being able to answer the query "is i in x" by reading few bits of the data structure [BMRS02, Pag01a, Pag01b, Păt08, Vio]. In particular, previous to this work there was no lower bound in the case when $\ell := \alpha n$ for $1/\alpha = 2^a$ a fixed power of two. Note that the lower bound in [Vio] does not apply to that case; intuitively, that is because the techniques there extend to the problem of succinctly storing arrays over the alphabet $[1/\alpha]$, but when $1/\alpha = 2^a$ no lower bound holds there: using a bits per symbol yields redundancy $r = 0$.

Using different techniques, as a corollary to our lower bound for generating the "$= \alpha$" distribution (Theorem 1.3) we obtain the first lower bound for the membership problem in the case where the set-size is a power-of-two fraction of the universe.

Corollary 1.7 (Lower bound for membership)**.** *For any $\alpha \in (0,1)$ there is $\epsilon > 0$ such that for all large enough n for which αn is an integer:*

Suppose one can store subsets x of $[n]$ of size αn in $m := \lceil \log_2 \binom{n}{\alpha n} \rceil + r$ bits, while answering "is i in x" by non-adaptively reading $\leq \epsilon \log n$ bits of the data structure. Then $r \geq 0.49 \log n$.

Again, Corollary 1.7 is tight for $\alpha = 1/2$ up to the constant 0.49, since $\log_2 \binom{n}{n/2} = n - \Theta(\log n)$, and using $m = n$ bits the problem is trivial. For $\alpha < 1/2$ it is not clear what lower bound on r one should expect, as surprising upper bounds hold for related problems [BMRS02, Pag01b, DPT10]. In particular, the recent work by Dodis, Pătraşcu, and Thorup [DPT10] yields $r = 1$ for storing arrays (non-adaptively reading $O(\log n)$ bits). It remains to be seen whether their techniques apply to the membership problem too.

We obtain Corollary 1.7 from Theorem 1.3 by establishing the simple and general fact that lower bounds for generating distributions somewhat close to a distribution D imply succinct data structure lower bounds for storing support(D). The following claim formalizes this for the membership problem, where $D = D_{=\alpha}$ is the uniform distribution over n-bit strings with αn ones.

Claim 1.8. *Suppose one can store subsets x of $[n]$ of size αn in $m := \lceil \log_2 \binom{n}{\alpha n} \rceil + r$ bits, while answering "is i in x" by non-adaptively reading q bits of the data structure. Then there is a q-local function $f : \{0,1\}^m \to \{0,1\}^n$ such that $\Delta(f(U), D_{=\alpha}) \leq 1 - 2^{-r-1}$.*

Proof. The i-th output bit of f is the algorithm answering "is i in x." Feed f random bits. With probability $\binom{n}{\alpha n} / 2^{\lceil \log_2 \binom{n}{\alpha n} \rceil + r} \geq 1/2^{r+1}$ the input is uniform over encodings of subsets of $[n]$ of size αn, in which case the statistical distance is 0. If we distinguish in every other case, the distance is at most $1 - 1/2^{r+1}$. □

Similar considerations extend to adaptive bit-probes and cell probes, corresponding to forest functions (in the latter case, over the alphabet $[n]$ instead of $\{0,1\}$). While one could prove lower bounds for data structures without using this approach, Claim 1.8 and Corollary 1.7 appear to suggest an uncharted direction. Finally, we note that none of the upper bounds mentioned earlier is an obstacle to using Claim 1.8, since those upper bounds use input length that is larger than the information-theoretic minimum by a quantity polynomial in the statistical distance gap, while for Claim 1.8 a logarithmic dependence suffices. Whether the lower bounds for generating $D_{=\alpha}$ can be improved in this case is an interesting open problem.

Pseudorandom generators. The ability to generate a distribution efficiently has obvious applications in pseudorandomness which we now elaborate upon. The ultimate goal of derandomization of algorithms is to remove, or reduce, the amount of randomness used by a randomized algorithm while incurring the *least possible* overhead in other resources, such as time. Typically, this is achieved by substituting the needed random bits with the output of a pseudorandom generator. That of pseudorandom generators is a general paradigm with many incarnations (cf. [Gol08, Chapter 8]). To lead to our result we would like to consider the distinction between two types of generators that can be used to derandomize relatively robust classes of algorithms. The first type is that of cryptographic generators [BM84, Yao82] (a.k.a. Blum-Micali-Yao). These use less resources than the algorithm to be derandomized, and in fact computing these generators can even be done in the restricted circuit class NC^0 [AIK06]. However, unconditional instantiations of these generators are rare, and in particular we are unaware of any unconditional cryptographic generator with large stretch, a key feature for derandomization. By contrast, Nisan-Wigderson generators [NW94] use more resources than the algorithm to be derandomized, and this looser notion of efficiency allows for more unconditional results [Nis91, NW94, LVW93, Vio07]. Moreover, all of these results yield generators with large, superpolynomial stretch.

In particular, Nisan [Nis91] shows a generator that fools small AC^0 circuits of depth d with exponential stretch, or seed length $\log^{O(d)} n$. As mentioned above, this generator uses more resources than the circuits to be derandomized. Specifically, it computes the parity function on $\geq \log^d n$ bits, which requires AC^0 circuits that have either depth $\geq d$ or superpolynomial size. Thus, if one insists on polynomial-size circuits, the derandomized circuit, consisting of the circuit computing the generator and the original circuit, has depth at least twice that of the original circuit. This constant factor blow-up in depth appears necessary for Nisan-Wigderson constructions.

In this work we exhibit, for any d, a generator computable by depth-2 circuits that fools circuits of depth d, using a number of random bits close to Nisan's (an improvement in the tools we use would let us match the number of random bits in Nisan's result). As a corollary, we obtain a derandomization which only blows up the depth by 1.

Theorem 1.9 (Depth-efficient generator against AC^0)**.** *The following holds for every d. There is a generator $G : \{0,1\}^\ell \to \{0,1\}^n$ such that:*
(i) each output bit of G can be written explicitly as both a DNF and a CNF of size $n^{O(1)}$,
(ii) any circuit of depth d and size n has advantage at most $o(1)$ in distinguishing the uniform distribution from $G(U)$ for random $U \in \{0,1\}^\ell$, and

(iii) $\ell \leq (\log n)^{O(\log \log n)}$.

Corollary 1.10 (Depth-efficient derandomization of AC^0)**.** *The following holds for every d. Let $f : \{0,1\}^* \to \{0,1\}^*$ be computable by uniform randomized AC^0 circuits of $\mathrm{poly}(n)$-size and depth d with error ϵ. Then f is computable by uniform randomized AC^0 circuits of $\mathrm{poly}(n)$-size and depth $d+1$ with error $\epsilon + o(1)$ using $\leq (\log n)^{O(\log \log n)}$ random bits.*

Corollary 1.10 follows easily from Theorem 1.9 by collapsing two adjacent layers of gates.

Some evidence that a generator as efficient as that in Theorem 1.9 may exist comes from Example (1), which implies a generator mapping $n-1$ bits to n bits that can be shown to look random to AC^0 circuits, and yet each output bit just depends on 2 inputs bits. However, the seed length of this generator is very poor, and it is not clear how to improve on it. Intuitively, one would like to be able to generate the output distribution of Nisan's generator [Nis91] more efficiently than shown in [Nis91]. We were not able to do so, and we raise this as another challenge. (Some recent progress on this question appears in [LV10].)

For Theorem 1.9, we notice that the recent line of work by Bazzi, Razborov, and Braverman [Bra09] shows that any distribution that is $(k := \log^c n)$-wise independent looks random to small AC^0 circuits of depth d, for a certain constant $c = c(d) \geq d$.

We show how such distributions can be generated by DNFs. Although the constructions of k-wise independent distributions in [CG89, ABI86, GV04] all require iterated sums of k bits, which for $k := \log^c n$ is unfeasible in our setting, we follow an approach of Mossel, Shpilka, and Trevisan [MST06] and give an alternative construction using unique-neighbor expanders. Specifically, we use the recent unique-neighbor expanders by Guruswami, Umans, and Vadhan [GUV09].

In the remainder of this section we discuss related work.

The work by Dubrov and Ishai [DI06]. The interesting work by Dubrov and Ishai [DI06] also addresses the problem of generating distributions, with a focus on the randomness complexity. In particular, [DI06] shows how non-boolean pseudorandom generators can be used to reduce the randomness necessary to generate a distribution. For the class AC^0, they show how any circuit generating a distribution can be combined with their non-boolean generator so that the randomness complexity is reduced to roughly the square of the entropy of the distribution. This square loss comes from the use of the Nisan-Wigderson generator [NW94]. Can this intriguing result in [DI06] be used to reduce the randomness used in some of our constructions, or even drop the bound on the input length in Theorem 1.3? For example, this work points out the fact that small AC^0 circuits can generate $(X, \text{majority } X)$ with exponentially small error. However, the circuits use $\Omega(|X| \log |X|)$ randomness. Can the randomness be reduced to, say, $O(|X|)$, or even $|X|$? Unfortunately, the aforementioned quadratic loss prevents us from using [DI06] to obtain any improvement, as all of our constructions use randomness which is less than quadratic in the output length. Still, we propose to explore further the approach in [DI06], and especially the construction of non-boolean pseudorandom generators for the local and the forest models, which may give new trade-offs in combination with our work.

More related work and discussion. A result (we already mentioned briefly) by Applebaum, Ishai, Kushilevitz [AIK06] shows, under standard assumptions, that there are pseudorandom distributions computable by NC^0 circuits. Their result is obtained via a generic transformation that turns a distribution D into another "padded" distribution D' that is computable in NC^0 and at the same time maintains interesting properties, such as pseudorandomness (but not stretch). The techniques in [AIK06] do not seem to apply to distributions such as $(x, \sum_i x_i)$ (Theorem 7.1), and they destroy stretch, which in particular prevents them from obtaining Corollary 1.10 (regardless of the stretch of the original generator, the techniques in [AIK06] always produce a generator with sublinear stretch).

Under an assumption on the hardness of decoding random linear codes, the same authors show in [AIK08] how to construct generators computable in NC^0 that have linear stretch. Their construction requires generating in NC^0 a uniform "noise vector" $e \in \{0,1\}^n$. They consider two types of noise vectors. The first type is when e has hamming weight exactly pn (think $p = 1/4$), i.e. e comes from the distribution $D_{=pn}$. The results in this paper show that it is impossible to generate such an e in NC^0, regardless of the input length, except with constant statistical distance, see Remark 4.2 related to Theorem 1.6. The second type of noise vector is when e is obtained by setting each bit to 1 independently with probability p. This distribution can be trivially generated in NC^0 when $p = 2^{-t}$, using tn bits of randomness, which is much larger than the entropy of the distribution. This loss in randomness is problematic for pseudorandom generator constructions, but the authors of [AIK08] make up for it by applying an extractor. (They use an extractor computable in NC^0 that is implied by [MST06]). Whether such a noise vector can be generated in NC^0 using randomness close to optimal is an interesting open question.

It is perhaps worthwhile to pause to make a philosophical remark. While the above mentioned works [AIK06, AIK08] show that, under various assumptions, one can locally generate distributions on n bits with small entropy that look random to any polynomial-time test, by contrast our results show that one cannot locally generate a distribution that is close to being uniform over n-bit strings with $n/2$ ones, which superficially seems a less demanding goal.

Recently, Lovett and the author [LV10] prove that small AC^0 circuits cannot generate the uniform distribution over any good error-correcting codes. This result does not solve Challenge 1.1 – it does not apply to distributions like $(X, b(X))$ – although it does answer a question asked in a preliminary version of this work.

Organization In §2 we provide the intuition and the proof of our lower bound for generating the "$= \alpha$" distribution locally (Theorem 1.3). The lower bound for generating $(X, \text{majmod } X)$ locally (Theorem 1.4) is in §3, and the lower bounds in the decision tree model (Theorem 1.6) are in §4. Upper bounds are in §5, §6, and §7, respectively for the local, decision-tree, and AC^0 models. In §8 we prove Theorem 1.9, our depth-efficient generator against AC^0 circuits. In §9 we conclude and summarize a few open problems.

2 Intuition and proof of lower bound for generating "$= \alpha$" locally

In this section we prove our lower bound for generating the "$= \alpha$" distribution, restated next.

Theorem 1.3 (Lower bound for generating "$= \alpha$" locally)**.** (Restated.) *For any $\alpha \in (0,1)$ and any $\delta < 1$ there is $\epsilon > 0$ such that for all sufficiently large n for which αn is an integer:*
Let $f : \{0,1\}^\ell \to \{0,1\}^n$ be an $(\epsilon \log n)$-local function where $\ell \le \log_2 \binom{n}{\alpha n} + n^\delta$.
Let $D_{=\alpha}$ be the uniform distribution over n-bit strings with αn ones.
Then $\Delta(f(U), D_{=\alpha}) \ge 1 - O(1/n^{\delta/2})$.

2.1 Intuition for the proof of Theorem 1.3.

We now explain the ideas behind the proof of Theorem 1.3. For simplicity, we consider the case $\ell = n$ and $\alpha = 1/2$, that is, we want to prove that any $(\epsilon \log n)$-local function $f : \{0,1\}^n \to \{0,1\}^n$ has output distribution $f(U)$ for uniform $U \in \{0,1\}^n$ at statistical distance $\ge 1 - 1/n^{\Omega(1)}$ from the distribution $D_{=1/2}$ uniform over n-bit strings with $n/2$ ones. For simplicity, we denote the latter by $D = D_{=1/2}$.

We start with two warm-up scenarios:

Low-entropy scenario. Suppose that f is the constant function $f(u) := 0^{n/2}1^{n/2}$. In this case, the simple test

$$T_F := \text{support}(f) = \{0^{n/2}1^{n/2}\}$$

gives $\Pr_U[f(U) \in T_F] = 1$ and $\Pr[D \in T_F] = 1/\binom{n}{n/2} \ll 1/n$, proving the theorem.

We call this the "low-entropy" scenario because $f(U)$ has low entropy.

Anti-concentration scenario. Suppose that $f(u) := u$. In this case we consider the test

$$T_S := \overline{\text{support}(D)} = \{z : \sum_i z_i \ne n/2\}.$$

Note $\Pr[D \in T_S] = 0$ by definition, while $\Pr_U[f(U) \in T_S] = \Pr[\sum_i U_i \ne n/2] = \binom{n}{n/2}/2^n \ge 1 - O(1/\sqrt{n})$ by a standard bound (Fact 2.2). (Taking T_S to be the complement of the support of D, rather than the support itself, is useful when pasting tests together.)

We call this the "anti-concentration" scenario because the bound $\Pr[\sum_i U_i \ne n/2] \ge 1 - O(1/\sqrt{n})$ is an instance of the general anti-concentration phenomenon that the sum of independent, non-constant, uniform random variables is unlikely to equal any fixed value. Specifically, the bound is a special case ($S_i = U_i \in \{0,1\}$) of the following anti-concentration inequality by Littlewood and Offord (later we use the general case).

Fact 2.1 (Littlewood-Offord anti-concentration [LO43, Erd45])**.** *Let $S_1, S_2, \ldots, S_t$ be t independent random variables, where S_i is uniform over $\{a_i, b_i\}$ for $a_i \ne b_i$. Then for any integer c, $Pr[\sum_i S_i = c] \le O(1/\sqrt{t})$.*

Having described the two scenarios, we observe that each of them, taken by itself, is not sufficient. This is because the output distribution of the low-entropy function $f(u) = 0^{n/2}1^{n/2}$ has the same probability of passing the anti-concentration test T_S as the distribution D, and similarly in the other case.

We would like to use a similar approach for a generic f. The first step is to partition the input bits u of f as $u = (x, y)$ and rewrite (up to a permutation)

$$f(u) = f(x, y) = h(y) \circ g_1(x_1, y) \circ g_2(x_2, y) \circ \cdots \circ g_s(x_s, y),$$

where each function g_i depends on only the single bit x_i of x (but arbitrarily on y), and has small range: $g_i(x_i, y) \in \{0,1\}^{O(d)} = \{0,1\}^{O(\epsilon \log n)}$. A greedy approach allows for such a decomposition with $|x| = s \geq \Omega(n/d^2) = n/\text{poly}\log n$. Specifically, by an averaging argument a constant fraction of the input bits are adjacent to $\leq O(d)$ output bits. We iteratively collect such a bit x_i and move in y the $\leq O(d^2)$ other input bits adjacent to any of the input bits x_i is adjacent to.

To reduce to the previous scenarios, fix y. Two things can happen: either $\geq \sqrt{n}$ of the functions g_i are fixed, i.e., do not depend on x_i anymore, or at least $s - \sqrt{n} = n/\text{poly}\log n$ take two different values over the choice of x_i. We think of the first case as the low-entropy scenario. Indeed, for this y the output distribution of $f(x, y)$ has small support, and we can hardwire it in the test. Here we use that the input length n of f is close to the information-theoretic minimum necessary to generate D, which is $n - \Theta(\log n)$, and hence removing $\sqrt{n}$ bits of entropy yields a tiny support where D is unlikely to land.

In the second case, intuitively, we would like to use anti-concentration, since we have independent random variables $g_1(x_1, y), g_2(x_2, y), \ldots, g_s(x_s, y)$. Specifically, we let $S_i := \sum_k (g_i(x_i, y))_k$ denote the sum of the bits of g_i, and would like to apply the Littlewood-Offord inequality to argue that $f(U)$ is likely to pass the anti-concentration test T_S, which checks if the hamming weight of f is $\neq n/2$. However, the following problem arises. It may be the case that, for example,

$$g_i(0, y) = 01, \text{ and } g_i(1, y) = 10,$$

corresponding to the constant variable $S_i \equiv 1$. In this case, the value of g_i is not fixed, hence this is not a low-entropy scenario, but on the other hand it does not contribute to anti-concentration, since $S_i \equiv 1$. In fact, precisely such functions g_i arise when running this argument on the function that generates the uniform distribution over n-bit strings with an even number of ones, which can be done with locality 2 via the construction (1) in §1.

We solve this problem as follows. We add to our test the check T_0 that $\leq 2\sqrt{n}$ of the blocks of output bits corresponding to g_i are all 0. Since the blocks are small (recall $g_i \in \{0,1\}^{O(\epsilon \log n)}$), the distribution D will have $\geq n^{0.99}$ such blocks with high probability, and so will almost never pass T_0.

Consider however what happens with $f(x, y)$, for a fixed y. If $\leq 2\sqrt{n}$ functions $g_i(x_i, y)$ can output all zeros (for some $x_i \in \{0,1\}$), then $f(x, y) \in T_0$ for every x, and we are again done. Otherwise, since $\leq \sqrt{n}$ functions g_i are fixed, we have $2\sqrt{n} - \sqrt{n} = \sqrt{n}$ functions

$g_i(x_i, y)$ that take two different values over $x_i \in \{0,1\}$, and one of the two is all zero. That means that the other value is not all zero, and hence has a sum of bits $a_i > 0$. We are now in the position to apply the Littlewood-Offord anti-concentration inequality, since we have $\geq \sqrt{n}$ independent variables S_i, each uniform over $\{0, a_i\}$ for $a_i \neq 0$. The inequality guarantees that $f(x,y) \in T_S$ with probability $\geq 1 - 1/n^{\Omega(1)}$, and this concludes the overview of the proof of Theorem 1.3.

We now proceed with the formal proof. We use several times the following standard approximation of the binomial by the binary entropy function $H(x) = x\log_2(1/x) + (1-x)\log_2(1/(1-x))$:

Fact 2.2 (Lemma 17.5.1 in [CT06]). *For $0 < p < 1, q = 1-p$, and n such that np is an integer,*

$$\frac{1}{\sqrt{8npq}} \leq \binom{n}{pn} \cdot 2^{-H(p)n} \leq \frac{1}{\sqrt{\pi npq}}.$$

2.2 Proof of Theorem 1.3

We begin by bounding some parameters in a way that is convenient for the proof. First, we assume without loss of generality that $\alpha \leq 1/2$ (otherwise, complement the output of f). Next, we bound $\ell = \Theta(H(\alpha)n)$. For this, first note that if $\ell \leq \log\binom{n}{\alpha n} - \log n$ then the size of the range of f is at most a $1/n$ fraction of the support of $D_{=1/2}$, and the result follows. Hence $\ell \geq \log\binom{n}{\alpha n} - \log n$. Fact 2.2 gives $|\log_2\binom{n}{\alpha n} - H(\alpha)n| \leq O(\log n)$, for n large. Hence, $\ell = \Theta(H(\alpha)n)$.

Now consider the bipartite graph with the n output nodes on one side and the ℓ input nodes on the other, where each output node is connected to the d input nodes it is a function of. Without loss of generality, each input node has degree at least 1 (otherwise, run this proof with ℓ the number of input bits actually used by f).

Claim 2.3. *There is a set I of $s := |I| \geq \Omega(H(\alpha)^2 n/d^2)$ input bits such that (i) each input bit in I has degree at most $b = O(d/H(\alpha))$, and (ii) each output bit is adjacent to at most one input bit in I.*

Proof. The average degree of an input node is dn/ℓ. By a Markov argument, at least $\ell/2$ input nodes have degree $\leq 2dn/\ell = O(d/H(\alpha))$. Let K be the set of these nodes. We obtain $I \subseteq K$ greedily as follows: Put a $v \in K$ in I, then remove from K any other input node adjacent to one of the outputs that v is adjacent to. Repeat until $K = \emptyset$.

Since each output node has degree d, for each node put in I we remove $\leq (d-1) \cdot O(d/H(\alpha)) = O(d^2/H(\alpha))$ others. So we collect at least $(\ell/2)/(1+O(d^2/H(\alpha))) = \Omega(H(\alpha)^2 n/d^2)$. □

Let I by the set given by Claim 2.3, and without loss of generality let $I = [s] = \{1, 2, \ldots, s\}$. For an input node $u_i \in [s]$, let B_i be the set of output bits adjacent to u_i. Note $1 \leq |B_i| \leq O(d/H(\alpha))$ (the first inequality holds because input nodes have degree ≥ 1).

By dividing an input $u \in \{0,1\}^\ell$ in (x,y) where x are the first s input bits and y are the other $\ell - s$, and by permuting output bits, we rewrite f as

$$f(x,y) = h(y) \circ g_1(x_1,y) \circ g_2(x_2,y) \circ \cdots \circ g_s(x_s,y),$$

where g_i has range $\{0,1\}^{|B_i|}$.

Definition 2.4. *We say that a function g_i is y-*fixed *if $g_i(0,y) = g_i(1,y)$, i.e., after fixing y it does not depend on x_i anymore.*

For a string $z \in \{0,1\}^n$, we denote by z_{B_i} the projection of z on the bits of B_i, so that $f(x,y)_{B_2} = g_2(x_2,y)$, for example.

Definition of the statistical test. The statistical test $T \subseteq \{0,1\}^n$ which will witness the claimed statistical distance is the union of three tests:

$$\begin{aligned}
T_F :=& \{z : \exists (x,y) : f(x,y) = z \text{ and } \geq 2n^\delta \text{ functions } g_i(x_i,y) \text{ are } y\text{-fixed}, i \in [s]\},\\
T_0 :=& \{z : z_{B_i} = 0^{|B_i|} \text{ for } \leq 3n^\delta \text{ indices } i \in [s]\},\\
T_S :=& \{z : \sum_i z_i \neq \alpha n\};\\
T :=& T_F \bigcup T_0 \bigcup T_S.
\end{aligned}$$

We now prove that the output of f is likely to pass the test, while a uniform string of weight αn is not.

Claim 2.5. $\Pr_u[f(u) \in T] \geq 1 - O(1/n^{\delta/2})$.

We recall the Littlewood-Offord anti-concentration inequality.

Fact 2.1 (Littlewood-Offord anti-concentration [LO43, Erd45])**.** (Restated.) *Let $S_1, S_2, \ldots, S_t$ be t independent random variables, where S_i is uniform over $\{a_i, b_i\}$ for $a_i \neq b_i$. Then for any integer c, $Pr[\sum_i S_i = c] \leq O(1/\sqrt{t})$.*

To prove this fact, reduce to the case $a_i \leq 0, b_i > 0$. Then generate $\sum S_i$ by first permuting variables, and then setting exactly the first S of them to the smallest values of their domains, where S is binomially distributed. Since for every permutation there is at most one value of S yielding sum c, and each value has probability $\leq O(1/\sqrt{t})$, the result follows.

Proof of Claim 2.5. Write again an input u to f as $u = (x,y)$. We prove that for every y we have $\Pr_x[f(x,y) \in T] \geq 1 - O(1/n^{\delta/2})$, which implies the claimed bound. Fix any y.

If $\geq 2n^\delta$ functions $g_i(x_i,y)$ are y-fixed, then $\Pr_x[f(x,y) \in T_F] = 1$.

Also, if there are $\leq 3n^\delta$ indices $i \in [s]$ such that $g_i(x_i,y) = 0^{|B_i|}$ for some x_i, then clearly for any x the string $f(x,y)$ satisfies $f(x,y)_{B_i} = g_i(x_i,y) = 0^{|B_i|}$ for $\leq 3n^\delta$ indices i. In this case, $\Pr_x[f(x,y) \in T_0] = 1$.

Therefore, assume both that there are $\leq 2n^\delta$ functions $g_i(x_i, y)$ that are y-fixed, and that there are $\geq 3n^\delta$ indices i such that $g_i(x_i, y) = 0^{|B_i|}$ for some x_i. Consequently, there is a set $J \subseteq [s]$ of $\geq 3n^\delta - 2n^\delta = n^\delta$ indices i such that $g_i(x_i, y)$ is not y-fixed and $g_i(x_i, y) = 0^{|B_i|}$ for some $x_i \in \{0, 1\}$. The key idea is that for the other value of $x_i \in \{0, 1\}$ the value of $g_i(x_i, y)$ must have hamming weight bigger than 0, and therefore it contributes to anti-concentration.

Specifically, fix all bits in x except those in J, and denote the latter by x_J. We show that for any such fixing, the probability over the choice of the bits x_J that the output falls in T_S, i.e. $Pr_{x_J}[\sum_{k\leq n} f(x, y)_k \neq \alpha n]$, is at least $1 - O(1/n^{\delta/2})$. To see this, note that, for $i \in J$, the sum S_i of the bits in $g_i(x_i, y)$ (i.e., $S_i := \sum_{k\leq|B_i|} g_i(x_i, y)_k$) is 0 with probability 1/2 over x_i and strictly bigger than 0 with probability 1/2 (since $0^{|B_i|}$ is the only input with sum 0); moreover, the variables S_i are independent. Writing the sum of the bits in $f(x, y)$ as $a + \sum_{i\in J} S_i$ for some integer a which does not depend on x_J, we have

$$\Pr_{x_J\in\{0,1\}^{|J|}}[f(x, y) \neq \alpha n] = \Pr_{x_J\in\{0,1\}^{|J|}}[\sum_{i\in J} S_i \neq \alpha n - a] \geq 1 - O(1/n^{\delta/2}),$$

where the last inequality is by Fact 2.1. □

Claim 2.6. *Let $D = D_{=\alpha}$ be the uniform distribution over n-bit strings of hamming weight αn. Then* $\Pr_D[D \in T] \leq 1/n$.

The proof gives the stronger bound $\Pr_D[D \in T] \leq 1/2^{n^\gamma}$, for a $\gamma > 0$ depending on δ.

Proof of Claim 2.6. By a union bound,

$$\Pr_D[D \in T] \leq \Pr_D[D \in T_F] + \Pr_D[D \in T_0] + \Pr_D[D \in T_S].$$

We separately show that each term is at most $1/(3n)$.

First, $\Pr_D[D \in T_S] = 0$ by definition of D.

Also, $\Pr_D[D \in T_F] = |T_F|/\binom{n}{\alpha n}$. Note each string in T_F can be described by a string of $|y| + |x| - 2n^\delta$ bits, where the first $|y|$ are interpreted as a value for y, and the remaining $|x| - 2n^\delta$ are interpreted as values for the variables x_i corresponding to functions $g_i(x_i, y)$ that are not y-fixed. Hence,

$$|T_F| \leq 2^{|y|+|x|-2n^\delta} = 2^{\ell-2n^\delta} \leq 2^{\log\binom{n}{\alpha n}-n^\delta},$$

and

$$\Pr_D[D \in T_F] \leq 2^{-n^\delta} \leq 1/(3n),$$

for large enough n.

Finally, we bound $\Pr_D[D \in T_0]$. There are several ways of doing this; the following is self-contained. For $i \in [s]$, let N_i be the event $D_{B_i} \neq 0^{|B_i|}$, over the choice of D. Let $t := 3n^\delta$ be as in the definition of T_0. We have:

$$\begin{aligned}\Pr_D[D \in T_0] &\leq \Pr[\exists J \subseteq [s], |J| = s - t, \text{ such that } N_i \text{ holds for all } i \in J] \\ &\leq \binom{s}{t} \max_{J\subseteq[s],|J|=s-t} \Pr[N_i \text{ for all } i \in J] \leq \binom{s}{t} \max_{J\subseteq[s],|J|=n/\log^2 n} \Pr[N_i \text{ for all } i \in J], \qquad (2)\end{aligned}$$

where in the last inequality we use that $s - t = \Omega(H(\alpha)^2 n/d^2) - 3n^\delta \geq n/\log^2 n$ for $\delta < 1$, sufficiently small ϵ, and sufficiently large n, using that $d \leq \epsilon \log n$. Let

$$m := n/\log^2 n.$$

We now bound $\max_{J \subseteq [s], |J|=m} \Pr[N_i \text{ for all } i \in J]$. Without loss of generality, let the maximum be achieved for $J = \{1, 2, \ldots, m\}$. Write

$$\Pr[N_i \text{ for all } i \leq m] = \Pr[N_1] \cdot \Pr[N_2 | N_1] \cdot \cdots \cdot \Pr[N_m | N_{m-1} \wedge \ldots \wedge N_1]. \tag{3}$$

We proceed by bounding $\Pr[N_k | N_{k-1} \wedge \ldots \wedge N_1]$ for any $k \leq m$. Recall that each set B_i has size $\leq b = O(d/H(\alpha))$. So the event $N_{k-1} \wedge \ldots \wedge N_1$ depends on $\leq (k-1)b$ bits. If we condition on any value of $(k-1)b$ bits, the probability that N_k is not true, i.e. that $D_{B_k} = 0^{|B_k|}$, is at least

$$\prod_{j=0}^{b-1} \frac{(1-\alpha)n - (k-1)b - j}{n - (k-1)b - j} \geq \left(\frac{(1-\alpha)n - kb}{n}\right)^b \geq 1/3^b \geq 1/n^{O(\epsilon/H(\alpha))},$$

using our initial assumption $\alpha \leq 1/2$, and that $k \leq m = n/\log^2 n$ and $b = O(d/H(\alpha)) = O(\epsilon \log n / H(\alpha))$, so $kb = o(n)$. Hence, $\Pr[N_k | N_{k-1} \wedge \ldots \wedge N_1] \leq 1 - 1/n^{O(\epsilon/H(\alpha))}$.

Plugging this bound in Equation (3), we obtain

$$\Pr[N_i \text{ for all } i \leq m] \leq \left(1 - 1/n^{O(\epsilon/H(\alpha))}\right)^m \leq e^{-n^{1-O(\epsilon/H(\alpha))}/\log^2 n} \leq e^{-n^{(1+\delta)/2}},$$

for sufficiently small ϵ and large n (recall $\delta < 1$).

Plugging this bound back in Equation (2) we get

$$\Pr_D[D \in T_0] \leq (es/t)^t e^{-n^{(1+\delta)/2}} \leq n^{3n^\delta} e^{-n^{(1+\delta)/2}} \leq 1/(3n),$$

for large enough n. □

To conclude the proof of the theorem, note that the combination of the two claims gives $\Delta(f(U), D) \geq 1 - O(1/n^{\delta/2}) - 1/n = 1 - O(1/n^{\delta/2})$. □

This proof actually shows that for any $\tau > 0$ and $\delta < 1$, we can pick the same ϵ for any $\alpha \in (\tau, 1-\tau)$.

3 Lower bound for generating $(X, \text{majmod } X)$ locally

In this section we prove our lower bound for generating $(X, \text{majmod } X)$, restated next.

Theorem 1.4 (Lower bound for generating $(X, \text{majmod } X)$ locally)**.** (Restated.) *For any $\delta < 1$ there is $\epsilon > 0$ such that for all sufficiently large n: Let $p \in [0.25 \log n, 0.5 \log n]$ be a prime number, and let* majmod : $\{0,1\}^n \to \{0,1\}$ *be defined as*

$$\text{majmod}(x) = 1 \Leftrightarrow \sum_{i \leq n} x_i \mod p \in \{0, 1, \ldots, (p-1)/2\}.$$

Let $f : \{0,1\}^\ell \to \{0,1\}^{n+1}$ be an $(\epsilon \log n)$-local function where $\ell \leq n + n^\delta$.
Then $\Delta(f(U), (X, \text{majmod } X)) \geq 1/2 - O(1/\log n)$.

Intuition for the proof of Theorem 1.4. The proof follows closely that of the lower bound for generating the "$= \alpha n$" distribution (Theorem 1.3). The main difference is that we use anti-concentration modulo p to argue that the number of ones in the input is uniform modulo p, and thus the output is correct with probability about 1/2.

The problem in the proof of Theorem 1.3 that unfixed functions g_i can take two values with the same hamming weight translates here in the problem that g_i can take two values with the same weight modulo p. Locality is used to guarantee that the output length of g_i is smaller than p, and so if one of the two values of g_i is all zero the other one must be different modulo p.

3.1 Proof of Theorem 1.4

The beginning of the proof is like that of Theorem 1.3: we write (up to a permutation of the input and output bits):

$$f(x,y) = h(y) \circ g_1(x_1,y) \circ g_2(x_2,y) \circ \cdots \circ g_s(x_s,y),$$

where g_i has range $\{0,1\}^{|B_i|}$ (B_i denotes the output bits of g_i, so that $f(x,y)_{B_i} = g_i(x_i,y)$) for $1 \le |B_i| \le O(d)$, and $s \ge \Omega(n/d^2)$.

For notational simplicity, we assume that the last bit of f does not get permuted; so f_{n+1} is still the bit corresponding to majmod.

Definition of the statistical test. Let

$$\begin{aligned}
T_F :=& \{z \in \{0,1\}^{n+1} : \exists (x,y) : f(x,y) = z \text{ and } \ge 2n^\delta \text{ functions } g_i(x_i,y) \text{ are } y\text{-fixed}, i \in [s]\},\\
T_0 :=& \{z \in \{0,1\}^{n+1} : z_{B_i} = 0^{|B_i|} \text{ for } \le 3n^\delta \text{ indices } i \in [s]\},\\
T_S :=& \{(z',b) \in \{0,1\}^n \times \{0,1\} : \left(\sum_i z'_i \mod p \in \{0,1,\ldots,(p-1)/2\}\right) \text{ xor } (b=1)\}\\
& \text{(that is, } T_S = \text{"wrong answer")};\\
T :=& T_F \bigcup T_0 \bigcup T_S.
\end{aligned}$$

We now prove that the output of f passes the test with probability $1/2 - O(1/\log n)$, while $(X, \text{majmod}(X))$ passes the test with probability $1/n$.

Claim 3.1. $\Pr_u[f(u) \in T] \ge 1/2 - O(1/\log n)$.

The proof uses the following well-known fact, which can be thought of as an anti-concentration result for the sum of random variables modulo p.

Fact 3.2. *Let $a_1, a_2, \ldots, a_t$ be t integers not zero modulo p. The statistical distance between the distribution $\sum_{i \le t} a_i x_i \mod p$ for uniform $x \in \{0,1\}^t$ and the uniform distribution over $\{0,1,\ldots,p-1\}$ is at most $\sqrt{p} e^{-t/p^2}$.*

Proof using various results. By [BV10, Claim 33], the statistical distance is at most

$$\sqrt{p}\max_{a\neq 0}|E_{x\in\{0,1\}^t}[e(a\sum_{i\leq t}a_ix_i)] - E_{U_p}[e(aU_p)]|,$$

where $e(x) := e^{2\pi\sqrt{-1}x/p}$ and U_p is the uniform distribution over $\{0,1,\ldots,p-1\}$. Fix any $a\neq 0$. By [LRTV09, Lemma 12] $|E_{x\in\{0,1\}^t}[e(a\sum_{i\leq t}a_ix_i)] \leq e^{-t/p^2}$; also, $E_{U_p}[e(aU_p)] = 0$. □

Proof of Claim 3.1. Write again an input u to f as $u=(x,y)$. We prove that for every y we have $\Pr_x[f(x,y)\in T]\geq 1/2 - O(1/\log n)$, which implies the claimed bound. Fix any y.

If $\geq 2n^\delta$ functions $g_i(x_i,y)$ are y-fixed, then $\Pr_x[f(x,y)\in T_F]=1$.

Also, if there are $\leq 3n^\delta$ indices $i\in[s]$ such that $g_i(x_i,y)=0^{|B_i|}$ for some x_i, then clearly for any x the string $f(x,y)$ satisfies $f(x,y)_{B_i}=g_i(x_i,y)=0^{|B_i|}$ for $\leq 3n^\delta$ indices i. In this case, $\Pr_x[f(x,y)\in T_0]=1$.

Therefore, assume both that there are $\leq 2n^\delta$ functions $g_i(x_i,y)$ that are y-fixed, and that there are $\geq 3n^\delta$ indices i such that $g_i(x_i,y)=0^{|B_i|}$ for some x_i. Consequently, there is a set $J\subseteq[s]$ of $\geq 3n^\delta - 2n^\delta = n^\delta$ indices i such that $g_i(x_i,y)$ is not y-fixed and $g_i(x_i,y)=0^{|B_i|}$ for some $x_i\in\{0,1\}$. The key idea is that for the other value of $x_i\in\{0,1\}$ the value of $g_i(x_i,y)$ must have hamming weight different from 0 modulo p, and therefore it contributes to anti-concentration.

Specifically, note that g_s is the only function that may affect the output bit f_{n+1}, corresponding to majmod. If present, remove s from J. Fix all bits in x except those in J, and denote the latter by x_J. We show that for any such fixing, the probability over the choice of the bits x_J that the output falls in T_S is $\geq 1/2 - O(1/\log n)$. To see this, note that, for $i\in J$, the sum S_i of the bits in $g_i(x_i,y)$ (i.e., $S_i := \sum_{k\leq|B_i|}g_i(x_i,y)_k$) is 0 with probability 1/2 over x_i, and $a_i\neq 0 \mod p$ with probability 1/2. This is because the maximum sum is

$$|B_i| = O(d) = O(\epsilon\log n) < p$$

for sufficiently small ϵ. Moreover, the variables S_i are independent. Writing the sum of the first n bits of $f(x,y)$ as $a+\sum_{i\in J}S_i$ for some integer a which does not depend on x_J, we have by Fact 3.2 that, over the choice of x_J, the statistical distance between the sum of the first n bits of f and the uniform distribution U_p over $\{0,1,\ldots,p-1\}$ is at most

$$\sqrt{p}e^{-(n^\delta-1)/p^2}\leq 1/n,$$

since $p=O(\log n)$. Because the last bit $b:=f_{n+1}(x,y)$ is fixed (independent from x_J), and

$$\Pr_{U_p}[U_p\in\{0,1,\ldots,(p-1)/2\}] = 1/2 - 1/(2p) = 1/2 - \Theta(1/\log n),$$

we have

$$\Pr_{x_J}[f(x,y)\in T_S]\geq 1/2 - O(1/\log n) - 1/n \geq 1/2 - O(1/\log n).$$

□

Claim 3.3. *Let $D = (X, \mathrm{majmod}(X))$ for uniform $X \in \{0,1\}^n$. Then $\Pr_D[D \in T] \le 1/n$.*

The proof gives a stronger, exponential bound.

Proof of Claim 3.3. By a union bound,

$$\Pr_D[D \in T] \le \Pr_D[D \in T_F] + \Pr_D[D \in T_0] + \Pr_D[D \in T_S].$$

We separately show that each term is at most $1/(3n)$.

First, $\Pr_D[D \in T_S] = 0$ by definition of D.

Also, $\Pr_D[D \in T_F] = |T_F|/2^n$. Note each string in T_F can be described by a string of $|y| + |x| - 2n^\delta$ bits, where the first $|y|$ are interpreted as a value for y, and the remaining $|x| - 2n^\delta$ are interpreted as values for the variables x_i corresponding to functions $g_i(x_i, y)$ that are not y-fixed. Hence,

$$|T_F| \le 2^{|y|+|x|-2n^\delta} = 2^{\ell-2n^\delta} \le 2^{n-n^\delta}, \text{ and } \Pr_D[D \in T_F] \le 2^{-n^\delta} \le 1/(3n),$$

for large enough n.

Finally, we bound $\Pr_D[D \in T_0]$. For any $i \in [s]$,

$$\Pr_{X\in\{0,1\}^n}[X_{B_i}] = 0^{|B_i|} = 1/2^{|B_i|} = 1/2^{O(d)} = 1/n^{O(\epsilon)}.$$

Moreover, these events are independent for different i. Hence, recalling that $s = \Omega(n/d^2) \ge n/\log^2 n$, we have:

$$\Pr_D[D \in T_0] \le \binom{s}{3n^\delta}(1 - 1/n^{O(\epsilon)})^{s-3n^\delta} \le n^{3n^\delta} e^{-n^{1-O(\epsilon)}/\log^2 n} \le 1/(3n)$$

for a sufficiently small ϵ and large enough n. □

To conclude the proof of the theorem, note that the combination of the two claims gives $\Delta(f(U), (X, \mathrm{majmod}\ X)) \ge 1/2 - O(1/\log n) - 1/n = 1/2 - O(1/\log n)$.

4 Lower bounds for generating by decision trees

In this section we prove our lower bounds in the forest model, restated next. Recall that a function $f : \{0,1\}^\ell \to \{0,1\}^n$ is a d-forest if each bit f_i is a decision tree of depth d.

Theorem 1.6 (Lower bound for generating "$= 1/2$" or $(X, \text{majority } X)$ by forest)**.** (Restated.) *Let $f : \{0,1\}^* \to \{0,1\}^n$ be a d-forest function. Then:*

(1) $\Delta(f(U), D_{=1/2}) \ge 2^{-O(d)} - O(1/n)$, where $D_{=1/2}$ is the uniform distribution over n-bit strings with $n/2$ ones.

(2) $\Delta(f(U), (X, \text{majority } X)) \ge 2^{-O(d)} - O(1/n)$.

The proof uses anti-concentration inequalities for random variables with bounded independence (we say that $X \in \{0,1\}^n$ is k-wise independent if any k bits of X are uniformly distributed over $\{0,1\}^k$). The Pailey-Zygmund inequality would be sufficient for (1) but not immediately for (2), due to its symmetry. The next lemma is sufficient for both; it follows from the main result in [DGJ+10] and Fact 2.2.

Lemma 4.1 ([DGJ+10])**.** *There is a constant k such that for large enough n and any k-wise independent distribution $X \in \{0,1\}^n$, with probability ≥ 0.49 the variable X has strictly less than $n/2$ ones.*

Proof of Theorem 1.6, (1). Let k be the constant from Lemma 4.1. Suppose the distribution $X := f(U)$ is k-wise independent. Then by Lemma 4.1 $\Pr[\sum_i X_i < n/2] \geq 0.49$. The statistical test which checks if the output bits sum to $n/2$ proves the claim in this case.

Otherwise, there are k output bits of f that are not uniformly distributed over $\{0,1\}^k$. We claim that, for any y, the probability k output bits evaluate to y equals $A/2^{kd}$ for an integer A. To see this, note that the k output bits can be computed with a decision tree of depth dk (e.g., use the decision tree for the first bit, then use the decision tree for the second, and so on). Since the probability of outputting a value y in a decision tree is the sum over all leaves labeled with y of the probabilities of reaching that leaf, and each leaf has probability $a/2^{kd}$ for some integer a, the result follows.

Therefore, if these k bits are not uniform, there there must be an output value that has probability at least $1/2^k + 1/2^{kd}$.

But over $D_{=1/2}$, this output combination of the k bits has probability at most

$$\frac{1}{2} \cdot \frac{n/2}{n-1} \cdot \dots \cdot \frac{n/2}{n-(k-1)} \leq \frac{1}{2^k}\frac{1}{(1-k/n)^k} \leq \frac{1}{2^k}\frac{1}{(1-k^2/n)} = \frac{1}{2^k} + O(1/n).$$

So, checking if these k bits equal y we get statistical distance $\geq 1/2^{O(d)} - O(1/n)$. □

Remark 4.2. *A lower bound similar to Theorem 1.6, (1), holds also for generating the "$= \alpha$" distribution for $\alpha \neq 1/2$. This can be obtained with a similar proof but using a recent result by Gopalan et al. [GOWZ10, Theorem 1.5] which generalizes [DGJ+10] and hence Lemma 4.1 to biased distribution.*

To prove Theorem 1.6, (2), we start with the following lemma which relates the ability to generate $(X, \text{majority}(X))$ to that of generating the uniform distribution over n-bit strings with $\geq 1/2$ ones (we only need one direction for Theorem 1.6, (2)).

Lemma 4.3 (Generate $(X, \text{majority}(X)) \Leftrightarrow$ generate upper half)**.** *Let n be odd, A (for above) denote the uniform distribution over n-bit strings with $\geq n/2$ ones. Write $\oplus$ for xor and $\bar{z}$ for the bit-wise complement of z.*

(1) For any function $f : \{0,1\}^\ell \to \{0,1\}^n$ define $f' : \{0,1\}^\ell \times \{0,1\} \to \{0,1\}^{n+1}$ as

$$f'(u,b) := (f(u)_1 \oplus \bar{b}, \dots, f(u)_n \oplus \bar{b}, b).$$

Then $\Delta(f'(U,B), (X, \text{majority}(X))) \leq \Delta(f(U), A)$. (Here B is uniform in $\{0,1\}$.)

(2) For any function $f : \{0,1\}^\ell \to \{0,1\}^{n+1}$ define $f' : \{0,1\}^\ell \to \{0,1\}^n$ as

$$f'(u) := (f(u)_1 \oplus \overline{f(x)_{n+1}}, ..., f(u)_n \oplus \overline{f(x)_{n+1}}).$$

Then $\Delta(f'(U), A) \le \Delta(f(U), (X, \text{majority}(X)))$.

Proof. Think of generating the distribution $(X, \text{majority}(X))$ by first tossing a coin b, and if $b = 1$ output $(A, 1)$, and if $b = 0$ output $(\bar{A}, 0)$.

(1) Pick any test $T \subseteq \{0,1\}^{n+1}$. We have

$$\begin{aligned}
&|\Pr_{U,B}[f'(U,B) \in T] - \Pr[(X, \text{maj}(X)) \in T]| \\
&\le (1/2)(|\Pr[f'(U,1) \in T] - \Pr[(A,1) \in T]| + |\Pr[f'(U,0) \in T] - \Pr[(\bar{A},0) \in T]|) \\
&\le (1/2)(\Delta(f(U),A) + |\Pr[f'(U,1) \in T^x] - \Pr[(A,1) \in T^x]|) \\
&\le (1/2)2\Delta(f(U),A),
\end{aligned}$$

where $T^x := \{\bar{z} : z \in T\}$.

(2) Pick any test $T \subseteq \{0,1\}^n$. Let T' be the test that on input z of length $n+1$ xors the first n bits with the complement of the last bit (i.e., if the last bit is 0 then it flips the first n), and checks if the resulting string is in T. We show T' tells f from $(X, \text{maj}(X))$ equally well as T tells f' from A.

First note $\Pr[(X, \text{maj}(X)) \in T'] = (1/2)\Pr[(A,1) \in T'] + (1/2)\Pr[(\bar{A},0) \in T'] = \Pr[A \in T]$.

Also, for B the last bit of $f(U)$:

$$\begin{aligned}
\Pr[f(U) \in T'] &= \Pr[B=1]\Pr[f(U) \in T'|B=1] + \Pr[B=0]\Pr[f(U) \in T'|B=0] \\
&= \Pr[B=1]\Pr[f(U)_{1,\dots,n} \in T|B=1] + \Pr[B=0]\Pr[\overline{f(U)_{1,\dots,n}} \in T|B=0] \\
&= \Pr[f'(U) \in T].
\end{aligned}$$

Hence,

$$\Delta(f(U), (X, \text{maj}(X))) \ge |\Pr[f(U) \in T'] - \Pr[(X, \text{maj}(X)) \in T']| = |\Pr[f'(U) \in T] - \Pr[A \in T]|.$$

□

Proof sketch of Theorem 1.6, (2). By Lemma 4.3 it suffices to bound from below $\Delta(f'(U), A)$ for f' a $(2d)$-forest. For this, we follow the approach of the proof of Theorem 1.6, (1).

Let k be the constant from Lemma 4.1. Suppose the distribution $X := f'(U)$ is k-wise independent. Then by Lemma 4.1 $\Pr[\sum_i X_i < n/2] \ge 1/k$. The statistical test which checks if the output bits have sum $\ge n/2$ proves the claim in this case.

Otherwise, there are k bits that not are not uniformly distributed over $\{0,1\}^k$. A reasoning similar to that of the proof of Theorem 1.6, (1), completes the proof. □

5 Local upper bounds

The following Theorem shows that for any $t = \omega(\log n)$ there is an $O(t)$-local function whose output distribution has distance $\le 1 - O(1/\sqrt{n})$ from the uniform distribution over n-bit strings with αn ones, and the input length is only sublinearly more than the information-theoretic minimum.

Theorem 5.1 (Local & succinct generation of the "$= \alpha$" distribution)**.** *For every α there is $k \ge 1$ such that for all $n \ge k$ for which αn is an integer, and for all $t \ge k \log n$:*

There is a function $f : \{0,1\}^\ell \to \{0,1\}^n$ such that

1. *$\ell \le H(\alpha)n + nk\sqrt{(\log n)/t}$,*
2. *f has locality $\ell t/n \le H(\alpha)t + k\sqrt{t \log n}$, and*
3. *let N_α^n over $\{0,1\}^n$ be the distribution where each bit equals 1 independently with probability α.*

   Then $\Delta(f(U), N_\alpha^n) \le O(1/n)$.

In particular, $\Delta(f(U), D_{=\alpha}) \le 1 - O(1/\sqrt{n})$, where $D_{=\alpha}$ is the uniform distribution over n-bit strings with αn ones.

The proof of Theorem 5.1 uses the following lemma to "discretize" distributions.

Lemma 5.2 (Discretize distribution)**.** *For any distribution D on n elements and any $t \ge 1$ there is a function $f : \{0,1\}^{\lceil \log_2 nt \rceil} \to support(D)$ such that the statistical distance between $f(U)$ and D is $\le 1/t$.*

Proof. Partition the interval $[0,1]$ in n intervals I_i of lengths $Pr[D = i]$, $i = 1, \ldots, n$. Also partition $[0,1]$ in $\ell := 2^{\lceil \log_2 nt \rceil} \ge nt$ intervals of size $1/\ell$ each, which we call blocks. The function f interprets an input as a choice of a block b, and outputs i if $b \subseteq I_i$ and, say, outputs 1 if b is not contained in any interval.

For any i we have $|\Pr[D = i] - \Pr[f(U) = i]| \le 2/\ell$. Hence the statistical distance is $\le (1/2)\sum_i |\Pr[D = i] - \Pr[f(U) = i]| \le (1/2)n2/\ell \le 1/t$. □

We also need the following fact about the entropy function.

Fact 5.3. *For any $\alpha \in (0,1)$ and any ϵ such that $\alpha + \epsilon \in [0,1]$, we have:*

$$H(\alpha + \epsilon) \le H(\alpha) + \epsilon \log((1-\alpha)/\alpha).$$

Proof sketch of Fact 5.3. The entropy function is concave [CT06, Theorem 2.7.3], and its derivative at α is $\log((1-\alpha)/\alpha)$. □

Proof of Theorem 5.1. The "in particular" claim follows from the first claim because the probability that N_α^n contains exactly αn ones is $\Omega(1/\sqrt{n})$ (Fact 2.2).

For simplicity, we assume $\alpha \le 1/2$. Also, if $\alpha = 0$ or $\alpha = 1/2$ the theorem is easily proved (in the latter case, let $f(x) := x$). Hence, assume $\alpha \in (0, 1/2)$. Let N_α^t be the distribution on $\{0,1\}^t$ where each bit is 1 independently with probability α. By a Chernoff bound [DP09, Theorem 1.1], the probability of the event E that the number of ones is more than

$$\sqrt{ct\log n} = \alpha t\epsilon, \text{ where } \epsilon := \frac{1}{\alpha}\sqrt{\frac{c\log n}{t}},$$

away from the mean αt is

$$\exp(-\Omega(\epsilon^2\alpha t)) = \exp(-\Omega((c/\alpha)\log n)) \le 1/n^2$$

for suitable $c = O(1)$. Denoting by N' the distribution N_α^t conditioned to E, this means that

$$\Delta(N', N_\alpha^t) \le 1/n^2. \tag{4}$$

Note that N' is a distribution on

$$\text{support}(N') \le O(\sqrt{ct\log n})\binom{t}{\alpha t + \sqrt{ct\log n} = t(\alpha + \sqrt{(c\log n)/t})}$$

points, where we bounded each binomial by the largest one, using that for $\alpha < 1/2$ and $t \ge k\log n$ for a sufficiently large k depending on α, $\alpha t + \sqrt{ct\log n} \le t/2$. Hence, since $t \ge \log n$,

$$\begin{aligned}
\log_2 \text{support}(N') &\le O(\log t) + H\left(\alpha + \sqrt{(c\log n)/t}\right)t && \text{(Fact 2.2)}\\
&\le O(\log t) + H(\alpha)t + \sqrt{ct\log n}\log((1-\alpha)/\alpha) && \text{(Fact 5.3)}\\
&\le H(\alpha)t + O\left(\sqrt{t\log n}\log((1-\alpha)/\alpha)\right).
\end{aligned}$$

By Lemma 5.2, there is a function $f' : \{0,1\}^{\ell'} \to \{0,1\}^t$ on

$$\ell' = \log_2 \text{support}(N') + O(\log n) = H(\alpha)t + O\left(\sqrt{t\log n}\log((1-\alpha)/\alpha)\right)$$

bits with $\Delta(f'(U), N') \le 1/n^2$. By (4), $\Delta(f'(U), N_\alpha^t) \le 2/n^2$.

Letting $f : \{0,1\}^\ell \to \{0,1\}^n$ be n/t independent copies of f', we have an ℓ'-local function on

$$\ell = \ell' n/t = H(\alpha)n + nO\left(\sqrt{(\log n)/t}\log((1-\alpha)/\alpha)\right)$$

bits such that $\Delta(f(U), N_\alpha^n) \le n2/n^2 = O(1/n)$. $\square$

We now show how to generate $(X, \sum_i X_i \mod p)$ with locality $O((\log n)p^2\log p)$.

Theorem 5.4 (Generating $(X, \sum_i X_i \mod p)$ locally)**.** *For any n and any prime p there is an $O((\log n)p^2 \log p)$-local function $f : \{0,1\}^{O(n)} \to \{0,1\}^n$ such that $\Delta(f(U), (X, \sum_i X_i \mod p)) \leq 1/n$.*

Proof. Let $t := c(\log n)p^2 \log p$ for a $c = O(1)$ to be determined later. Divide n in $b := n/t$ blocks of t bits each. Let $R_1, R_2, \ldots, R_b$ be independent binomials over t bits, modulo p, corresponding to the sum modulo p of the t bits in each block of X. Consider the following randomized process G: on input $R_1, R_2, \ldots, R_b$, output

$$(X^1 \circ X^2 \circ \cdots \circ X^b, \sum R_i \mod p),$$

where X^i is a uniform t-bit strings with hamming weight R_i modulo p.

Note that the distribution of G, over random input $R_1, R_2, \ldots, R_b$ and random choices for the variables X^i, is the same as $(X, \sum_i X_i \mod p)$.

By Fact 3.2, for each i, letting S_i be the uniform distribution over $\{0, 1, \ldots, p-1\}$, we have

$$\Delta(R_i, S_i) \leq \sqrt{p}e^{-t/p^2} \leq o(1/n^2),$$

for a suitable $c = O(1)$. Hence,

$$\Delta((R_1, \ldots, R_b), (S_1, \ldots, S_b)) \leq o(1/n).$$

This means that if we run the randomized process G on input $S_1, \ldots, S_b$ we observe

$$\Delta(G(S_1, \ldots, S_b), (X, \sum_i X_i \mod p)) \leq o(1/n).$$

We now show how to generate $G(S_1, \ldots, S_b)$ locally. Via the telescopic trick from (1) in §1, we have (writing "$\equiv$" for "having the same distribution)

$$\begin{aligned} G(S_1, \ldots, S_b) &\equiv G(S_1, S_2 - S_1 \mod p, S_3 - S_2 \mod p, \ldots, S_b - S_{b-1} \mod p) \\ &\equiv (Z^1 \circ Z^2 \circ \cdots \circ Z^b, S_b), \end{aligned}$$

where Z^i is a uniform t-bit string with weight $S_i - S_{i-1}$ modulo p.

To generate this distribution locally, we discretize these distributions via Lemma 5.2. Specifically, first generate discretizations T_i of S_i. Since each S_i is over p values, we can generate T_i with error $o(1/n^2)$ using input length = locality $\leq \log p + O(\log n) = O(\log n)$. With input length $bO(\log n) = O(n)$, we can generate $(T_1, \ldots, T_b)$ with statistical distance $\leq o(1/n)$ from $(S_1, \ldots, S_b)$.

Then generate discretizations W^i of the variables Z^i. Each Z^i ranges on at most 2^t values, and depends on S_i and S_{i-1}. Hence we can generate W^i with statistical distance $o(1/n^2)$ from Z^i using input length = locality $\leq t + O(\log n) = O(t)$. With input length $bO(t) = O(n)$ and locality $O(t)$ we can generate $(W^1, \ldots, W^b)$ with statistical distance $\leq o(1/n)$ from $(Z^1, \ldots, Z^b)$. The total error loss is $\leq 1/n$; we output $(W^1 \circ W^2 \circ \cdots \circ W^b, T_b)$. □

A different proof yielding qualitatively similar parameters may be conceptually simpler. Think of G as a (deterministic) function that in addition to $R_1, \ldots, R_b$ takes as input bp variables $X^{i,j}$, where $X^{i,j}$ is uniform over t-bit strings with weight j modulo p, and corresponds to the i-th block. G outputs the selection of variables $X^{i,j}$ corresponding to the variables R_i. Now use the telescopic trick from (1) in §1, and then simply replace each input variable by a discretization.

6 Decision tree upper bounds

In this section we prove that various distributions can be generated by functions whose output bits are shallow decision trees. The distributions include the uniform distribution over n-bit strings with αn ones, and $(X, b(X))$ for symmetric b.

The upper bounds in this section rely on previous results on switching networks, which we now recall. See [CKKL01] for details.

Definition 6.1. *A* switching network *S of depth d with n inputs is a list of d matchings of $[n]$. The output distribution $S(x)$ of S on fixed input $x \in \{0,1\}^n$ is obtained as follows. For $i = 1, \ldots, d$: independently for any edge in the i-th matching, swap the corresponding bits of x with probability $1/2$. Output x.*

Czumaj et al. [CKKL99] prove the existence of small-depth switching networks that "shuffle" balanced strings and generate random permutations.

Theorem 6.2 ([CKKL99])**.** *(1) For any even n there is a switching network of depth $O(\log n)$ whose output distribution on input $1^{n/2}0^{n/2}$ has statistical distance $O(1/n)$ from the uniform distribution over n-bit strings with $n/2$ ones.*

(2) For any n there is a switching network of depth $O(\log^2 n)$ such that for any input $x \in \{0,1\}^n$ with k ones the output distribution on input x has statistical distance $O(1/n)$ from the uniform distribution over n-bit strings with k ones.

Remark 6.3 (Remark on Theorem 6.2)**.** *(1) is [CKKL99, Theorem 2.3]. (2) is a corollary of the stronger result in [CKKL99] that there are switching networks that generate random permutations over $[n]$. It appears (Czumaj, personal communication) that the techniques yielding (1) also establish (2) with depth $O(\log n)$ as opposed to $O(\log^2 n)$. This immediately would entail the same improvement in Theorems 6.5 and 6.6 below. Finally, we note that (1) and (2) are not explicit, but for $d = \operatorname{poly}\log n$ there are explicit such networks, see [CKKL01] and the pointers therein.*

We make the technically simple observation that decision trees can simulate switching networks. Recall from Definition 1.5 that a d-forest is a function where each output bit is a decision tree of depth d.

Lemma 6.4 (Switching network ⇒ decision trees)**.** *Let S be a switching network of depth d on n inputs, and $x \in \{0,1\}^n$ any input. There is a d-forest function $f : \{0,1\}^{dn/2} \to \{0,1\}^n$ such that the output distribution $f(U)$ equals the output distribution of S on x.*

Proof. The input to f corresponds to the random choices in the computation of S on x: one choice per edge in each of the d matchings. To compute f_i, follow the path via matchings $d, d-1, \ldots, 1$ to an input node; output that node. □

We now show how to generate the distribution D_α on $\{0,1\}^n$, which recall is the uniform distribution over n-bit strings with $n/2$ ones.

Theorem 6.5 (Generating "$= \alpha n$" distribution by decision trees)**.** *For every even n there is an $O(\log n)$-forest function $f : \{0,1\}^{O(n \log n)} \to \{0,1\}^n$ such that $\Delta(f(U), D_{=1/2}) \le O(1/n)$.*

Also, for every n and α such that αn is an integer, there is a $O(\log^2 n)$-forest function $f : \{0,1\}^{O(n \log^2 n)} \to \{0,1\}^n$ such that $\Delta(f(U), D_{=\alpha}) \le O(1/n)$.

Proof. Combine Theorem 6.2 and Lemma 6.4. □

We now turn to the task of generating distributions of the form $(X, b(X))$, where b is symmetric. The idea is to use Lemma 5.2 to "discretize" the binomial distribution $\sum_i X_i$.

Theorem 6.6 (Generating $(X, \sum_i X_i)$ by decision trees)**.** *For every n there is an $O(\log^2 n)$-forest function $f : \{0,1\}^{O(n \log^2 n)} \to \{0,1\}^n$ such that $\Delta(f(U), (X, \sum_i X_i)) \le O(1/n)$.*

In particular, $(X, b(X))$ can be generated with the same resources for any symmetric b (e.g., $b = \text{majority}, \text{majmod}$).

Proof. The "in particular" part is immediate; we now prove the first claim. We dedicate $O(\log n)$ input bits to generating $s := \sum_i X_i$ with statistical distance $O(1/n^2)$, via Lemma 5.2. Each output bit of f_i queries these bits first. Once the discretization s' of s has been determined, we use the decision trees for generating a distribution at distance $\le O(1/n)$ from the uniform distribution over n-bit strings with s' ones, given by the combination of Theorem 6.2, (2), and Lemma 6.4. □

7 Generating $(x, \sum_i x_i)$ in AC^0

In this section, we prove that small AC^0 circuits can generate $(x, f(x))$ for any symmetric function f, including for example the majority function, except for an exponentially small statistical distance. This is an immediate corollary of the following theorem, stating that one can generate $(x, \sum_i x_i) \in \{0,1\}^n \times \{0,1,\ldots,n\}$.

Theorem 7.1. *There are explicit AC^0 circuits $C : \{0,1\}^{\mathrm{poly}(n)} \to \{0,1\}^n \times \{0,1,\ldots,n\}$ of size $\mathrm{poly}(n)$ and depth $O(1)$ whose output distribution has statistical distance $\le 2^{-n}$ from the distribution $(x, \sum_i x_i) \in \{0,1\}^n \times \{0,1,\ldots,n\}$ for random $x \in \{0,1\}^n$.*

We note that the statistical distance can be made 2^{-n^c} for an arbitrary constant c at the price of having the size of the circuit be a polynomial depending on c (choose larger ℓ in §7.1).

The proof of Theorem 7.1 relies on the following result by Matias and Vishkin [MV91] and Hagerup [Hag91] about generating random permutations of $[n]$. We think of a permutation of $[n]$ as represented by an array $A[1..n] \in [n]^n$.

Lemma 7.2 ([MV91, Hag91]). *There are explicit* AC^0 *circuits* $C : \{0,1\}^{\mathrm{poly}(n)} \to [n]^n$ *of size* $\mathrm{poly}(n)$ *and depth* $O(1)$ *whose output distribution has statistical distance* $\le 2^{-n}$ *from the uniform distribution over permutations of* $[n]$.

Lemma 7.2 is obtained in [MV91] and Hagerup [Hag91, Theorem 3.9] in the context of PRAMs. Those works also achieve a nearly optimal number of processors, which appears to make the proofs somewhat technical. In §7.1 we present a proof that uses the ideas in [MV91, Hag91] but is simpler and sufficient for our purposes.

To prove Theorem 7.1 we first generate a random permutation π of $[n]$, then select $s \in \{0,1,\dots,n\}$ binomially distributed, let $x \in \{0,1\}^n$ be the string where exactly the bits at position $\pi(i)$ for $i \le s$ are set to 1, and output (x,s). The difference between this proof and that of Theorem 6.6 is that to obtain exponentially small error we use [MV91, Hag91] instead of [CKKL99] to generate random permutations, and we construct an AC^0 circuit to discretize (in fact, generate exactly) the binomial distribution instead of using Lemma 5.2.

Proof of Theorem 7.1. First, note for every $s \in \{0,\dots,n\}$ there is a circuit C_s of size $\mathrm{poly}(n)$ and depth $O(1)$ whose output distribution is exponentially close to the uniform distribution over n-bit strings of weight s. To see this, run the circuit from Lemma 7.2 to produce array $A[1..n]$ containing a random permutation. The i-th output bit of C_s is set to 1 if and only if there is $j \le s$ such that $A[j] = i$. In other words, we set to 1 exactly the first s elements of A. It is easy to see that this can be implemented using poly-size AC^0 circuits.

To generate $(x, \sum_i x_i)$, it remains to select the circuits C_s with the correct probability. To do this, recall that, given two n-bit integers a, b, we can efficiently determine if $a > b$ as follows (a_1 is the most significant digit):

$$a > b \Leftrightarrow (a_1 > b_1) \vee (a_1 = b_1 \wedge a_2 > b_2) \vee (a_1 = b_1 \wedge a_2 = b_2 \wedge a_3 > b_3) \dots .$$

Now interpret n fresh random bits as an integer $z \in \{1,\dots,2^n\}$. Let circuit $D : \{0,1\}^n \to \{0,1,\dots,n\}$ output $s \in \{0,\dots,n\}$ if and only if

$$\sum_{i=0}^{s-1} \binom{n}{i} < z \le \sum_{i=0}^{s} \binom{n}{i}.$$

To do this in parallel, we can for example construct circuits D_s each responsible of one output, and then take the OR of their outputs. By construction, the output distribution of D over uniform input equals the distribution $\sum_i X_i$ where X_i are independent uniform random bits.

The circuit claimed in the theorem first runs D to obtain s, then runs C_s to obtain $x \in \{0,1\}^n$, and outputs (x,s). □

7.1 Generating random permutations in AC^0

In this section we give a simple proof of the following lemma.

Lemma 7.2 ([MV91, Hag91])**.** (Restated.) *There are explicit* AC^0 *circuits* $C : \{0,1\}^{\mathrm{poly}(n)} \to [n]^n$ *of size* $\mathrm{poly}(n)$ *and depth* $O(1)$ *whose output distribution has statistical distance* $\leq 2^{-n}$ *from the uniform distribution over permutations of* $[n]$*.*

The main technique is known as "dart throwing:" we view the input random bits as random pointers $p_1, p_2, \ldots, p_n$ into $m \gg n$ cells. We then write i in the p_i-th cell (empty cells get "$*$"). If there are no collisions, the ordering of $[n]$ in the cells gives a random permutation of $[n]$. However, it is not clear how to explicitly write out this permutation using small depth, because to determine the image of i one needs to count how many cells before p_i are occupied, which cannot be done in small depth.

The key insight of Matias and Vishkin [MV91] (see also [Hag91]) is to view the cells as representing the permutation in a different format, one from which we can explicitly write out the permutation in small depth. The format is known as the canonical form for the cyclic notation. We now briefly review it closely following [MV91]. Just like the standard format, the alternative format represents a permutation via an array $A[1..n]$ whose entries contain all the elements $[n]$. However, rather than thinking of $A[i]$ as the image of i, we think of the entries of A as listing the cycles of the permutation in order. Each cycle is listed starting with its smallest element, and cycles are listed in decreasing order of the first element in the cycle. This format allows for computing the permutation efficiently: the image of i is the element to the right of i in A, unless the latter element is the beginning of a new cycle, in which case the image of i is the first element in the cycle containing i. Identifying the first element of a cycle is easy, because it is smaller than any element preceding it in A. One can now verify that computing the image of i can be done by circuits of size $\mathrm{poly}(n)$ and depth $O(1)$.

The benefit of this format is that it works even if the array A has $m \gg n$ cells, of which $m - n$ are empty and marked by "$*$."

To conclude the proof of the lemma, generate ℓ uniform and independent sets of pointers $p_1^i, \ldots, p_n^i$, $i = 1, \ldots, \ell$, where each pointer has range $[m]$ for m the smallest power of 2 larger than $2n^2$ (thus each pointer can be specified by $\log m$ bits).

If there exists i such that the pointers $p_1^i, \ldots, p_\ell^i$ are all distinct (i.e., there are no collisions), then run the above algorithm on the output corresponding to the first such i. This results in a random permutation.

Since the pointers are chosen independently, the probability that there is no such i is

$$\Pr[\forall i \exists j, k \leq n : p_j^i = p_k^i] = \Pr[\exists j, k \leq n : p_j^1 = p_k^1]^\ell \leq (1/2)^\ell.$$

Choosing $\ell := n$ proves the lemma.

8 Depth-efficient derandomization of AC^0

In this section we prove Theorem 1.9:

Theorem 1.9 (Depth-efficient generator against AC^0)**.** (Restated.) *The following holds for every* d*. There is a generator* $G : \{0,1\}^\ell \to \{0,1\}^n$ *such that:*

(i) each output bit of G can be written explicitly as both a DNF and a CNF of size $n^{O(1)}$,
(ii) any circuit of depth d and size n has advantage at most $o(1)$ in distinguishing the uniform distribution from $G(U)$ for random $U \in \{0,1\}^\ell$, and
(iii) $\ell \leq (\log n)^{O(\log\log n)}$.

For our generator, we need the results by Bazzi, Razborov, and Braverman that polylog-wise distributions fool small AC^0 circuits.

Theorem 8.1 ([Bra09])**.** *For every d there is c such that the following holds for every m. Let C be a boolean circuit of depth d and size m. Let D be a $(\log^c m)$-wise independent distribution. Then*

$$|Pr_U[C(U) = 1] - Pr[C(D) = 1]| \leq 1/m,$$

where U is the uniform distribution.

Overview of the proof of Theorem 1.9. The proof goes by showing how to generate $(\log^c n)$-wise independent distribution by DNFs. This relies on the expander graphs by Guruswami, Umans, and Vadhan [GUV09]. Specifically, [GUV09] gives explicit bipartite expanders with n nodes on the left, s nodes on the right, and left degree $O(\log n)$ such that any subset of left-hand nodes of size $\leq k = \log^c n$ has at least one unique neighbor. We define the i-th output bit of our $(\log^c n)$-wise independent generator as the xor of the $O(\log n)$ neighbors of the left-hand node i in this graph, which can be computed by a poly(n)-size DNF. The unique neighbor property guarantees that the xor of any $t \leq k = \log^c n$ output bits will be unbiased. And this implies that any k output bits are uniformly distributed over $\{0,1\}^k$ (as observed e.g. in [CGH$^+$, §3.1]), concluding our overview.

As noted in the Introduction, the use of unique-neighbor expanders to construct distributions with bounded independence also appears in the work by Mossel, Shpilka, and Trevisan [MST06]

We now proceed with the formal proof. We recall the result by Guruswami, Umans, and Vadhan.

Theorem 8.2 (Theorem 3.3 in [GUV09])**.** *For any integers s, m, and q, where q is a power of 2, there are explicit graphs $G : [q]^s \times [q] \to [q]^{m+1}$ such that any set $S \subseteq [q]^s$ of size at most 2^m has at least $|S| \cdot (q - (s-1)m)$ neighbors.*

The following is our efficient constructions of k-wise independent distributions.

Theorem 8.3. *For every n and $d \leq \log n$ there is an explicit circuit $C : \{0,1\}^\ell \to \{0,1\}^n$ such that:*

(i) each output bit of C can be written explicitly as both a DNF and a CNF of size $n^{O(d)}$,
(ii) the distribution $C(U)$ for random $U \in \{0,1\}^\ell$ is $(\log^d n)$-wise independent,
(iii) $\ell \leq (\log n)^{O(d\cdot\log\log n)}$.

Proof. Let c be a sufficiently large, even constant to be set later. Let q be a power of 2 such that

$$c \cdot d \cdot \log n \leq q \leq 2c \cdot d \cdot \log n.$$

Choose

$$s := \left\lceil \frac{\log n}{\log q} \right\rceil,$$

and

$$m := \left\lceil \frac{q}{4} \cdot \frac{\log q}{\log n} \right\rceil \leq \left\lceil \frac{2c \cdot d \log n}{4} \cdot \frac{\log q}{\log n} \right\rceil = \frac{c \cdot d \cdot \log q}{2}.$$

Let G be the graph given by Theorem 8.2 for the above parameters. The i-th output bit of C is the parity of the neighbors of the i-th left-hand node in G. First, note that the numbers of left-hand nodes in G is

$$q^s \geq q^{(\log n)/\log q} \geq n,$$

so the output length of C is as desired. Also, each left-hand node has $\leq 2c \cdot d \log n$ neighbors by our choice of q. The corresponding parity can be computed in brute-force by both a DNF and a CNF of size $n^{O(d)}$. This verifies (i).

The number of right-hand side nodes is the input length ℓ of C. Using our above bounds on m and q, and the assumption that $d \leq \log n$, we have

$$\ell = q^{m+1} \leq (2c \cdot d \cdot \log n)^{O(d \cdot \log q)} \leq (\log n)^{O(d \cdot \log \log n)}.$$

This verifies (iii).

To see (ii), it is enough to observe that every $S \subseteq [q]^s$ of size $|S| \leq \log^d n$ has at least one unique neighbor. This unique neighbor guarantees that the parity of the bits corresponding to S (which recall are each defined to be the parity of their neighbors) is unbiased. Therefore, as observed e.g. in [CGH$^+$, §3.1], the output distribution is $(\log^d n)$-wise independent.

Indeed, note that the expansion property of the graph holds for sets of size up to

$$\begin{aligned} 2^m &\geq \exp\left(\frac{q}{4} \cdot \frac{\log q}{\log n}\right) \geq \exp\left(\frac{c \cdot d \cdot \log n}{4} \cdot \frac{\log \log n}{\log n}\right) \geq \exp\left(\frac{c \cdot d \cdot \log \log n}{4}\right) \\ &\geq \log^d n, \end{aligned}$$

for a sufficiently large c.

Hence, by the expansion property, S has at least $|S|(q-(s-1)m)$ neighbors. To guarantee a unique neighbor, we need more than $|S|q/2$ neighbors. Equivalently, $(s-1)m < q/2$. By our choice of s and m we have

$$(s-1)m \leq \left(\frac{\log n}{\log q} + 1 - 1\right) m = \frac{\log n}{\log q} \cdot \frac{q}{4} \frac{\log q}{\log n} = \frac{q}{4} < \frac{q}{2},$$

concluding the proof. $\square$

Combining the above two theorems yields Theorem 1.9. Suitable improvements in theorems 8.2 and 8.1 would make the number of random bits in the conclusion of Theorem 1.9 match the state-of-the-art [Nis91].

We also draw a connection to the long-standing challenge of finding an algorithm that, given a DNF, computes an additive approximation to the number of satisfying assignments (see, e.g., [Tre04]). The above techniques show that the latter problem is polynomial-time reducible to the problem of computing an additive approximation to the number of satisfying assignments of a given depth-3 circuit with only $(\log n)^{O(\log \log n)}$ variables.

9 Conclusion

This paper makes a step towards a systematic study of the complexity of generating distributions. Many open problems remain. We already mentioned the problem of extending Theorems 1.3 and 1.4 to hold even when the input length is unbounded, and also the problem of generating independent biased bits with input length close to optimal. We note a few other problems: What is the complexity of generating $(x, \mathrm{majority}(x)) \in \{0,1\}^{n+1}$? We have proved the existence of AC^0 circuits with exponentially close output distribution; can they generate exactly $(x, \mathrm{majority}(x)) \in \{0,1\}^{n+1}$? We have connected this problem to generating the "upper half" of the boolean cube, and gave a lower bound for forest functions. Can we prove a stronger bound for local functions?

For simplicity we have stated Challenge 1.1 for generating a distribution exactly, but a statistical distance lower bound would be desirable. To our knowledge, that is open even for $O(\log n)$-local functions: Can we prove, say, a $\Omega(1)$ statistical distance bound for $O(\log n)$-local functions?

Acknowledgments. I am very grateful to Rajmohan Rajaraman and Ravi Sundaram for extensive collaboration on this project. I thank Artur Czumaj and Shachar Lovett for useful discussions, and the anonymous referees for helpful feedback. I also thank Yuval Ishai for a discussion on [DI06], and Oded Goldreich for useful comments on the write-up and pointers to the literature.

References

[ABI86] Noga Alon, László Babai, and Alon Itai. A fast and simple randomized algorithm for the maximal independent set problem. *Journal of Algorithms*, 7:567–583, 1986.

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. *SIAM J. Comput.*, 36(4):845–888, 2006.

[AIK08] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. On pseudorandom generators with linear stretch in NC^0. *Computational Complexity*, 17(1):38–69, 2008.

[Bab87] László Babai. Random oracles separate PSPACE from the polynomial-time hierarchy. *Inform. Process. Lett.*, 26(1):51–53, 1987.

[Bei93] Richard Beigel. The polynomial method in circuit complexity. In *8th Annual Structure in Complexity Theory Conference*, pages 82–95. IEEE, 1993.

[BL87] Ravi Boppana and Jeffrey Lagarias. One-way functions and circuit complexity. *Inform. and Comput.*, 74(3):226–240, 1987.

[BM84] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. on Computing*, 13(4):850–864, November 1984.

[BMRS02] Harry Buhrman, Peter Bro Miltersen, Jaikumar Radhakrishnan, and Venkatesh Srinivasan. Are bitvectors optimal? *SIAM J. Comput.*, 31(6):1723–1744, 2002.

[Bra09] Mark Braverman. Poly-logarithmic independence fools AC^0 circuits. In *24th Conference on Computational Complexity (CCC)*. IEEE, 2009.

[BV10] Andrej Bogdanov and Emanuele Viola. Pseudorandom bits for polynomials. *SIAM Journal on Computing*, 39(6):2464–2486, 2010. FOCS special issue.

[CG89] Benny Chor and Oded Goldreich. On the power of two-point based sampling. *Journal of Complexity*, 5(1):96–106, 1989.

[CGH$^+$] B. Chor, O. Goldreich, J. Hastad, J. Friedman, S. Rudich, and R. Smolensky. The bit extraction problem and t-resilient functions. In *26th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 396–407.

[CKKL99] Artur Czumaj, Przemyslawa Kanarek, Miroslaw Kutylowski, and Krzysztof Lorys. Delayed path coupling and generating random permutations via distributed stochastic processes. In *Symposium on Discrete Algorithms (SODA)*, pages 271–280, 1999.

[CKKL01] Artur Czumaj, Przemyslawa Kanarek, Miroslaw Kutylowski, and Krzysztof Lorys. Switching networks for generating random permutations, 2001.

[CT06] Thomas Cover and Joy Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006.

[DGJ$^+$10] Ilias Diakonikolas, Parikshit Gopalan, Ragesh Jaiswal, Rocco A. Servedio, and Emanuele Viola. Bounded independence fools halfspaces. *SIAM Journal on Computing*, 39(8):3441–3462, 2010.

[DI06] Bella Dubrov and Yuval Ishai. On the randomness complexity of efficient sampling. In *38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 711–720, 2006.

[DP09] Devdatt Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

[DPT10] Yevgeniy Dodis, Mihai Pătraşcu, and Mikkel Thorup. Changing base without losing space. In *42nd Annual Symposium on Theory of Computing (STOC)*, pages 593–602. ACM, 2010.

[Erd45] Paul Erdős. On a lemma of Littlewood and offord. *Bull. Amer. Math. Soc.*, 51:898–902, 1945.

[GGN10] Oded Goldreich, Shafi Goldwasser, and Asaf Nussboim. On the implementation of huge random objects. *SIAM J. Comput.*, 39(7):2761–2822, 2010.

[Gol08] Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.

[GOWZ10] Parikshit Gopalan, Ryan O'Donnell, Y. Wu, and David Zuckerman. Fooling functions of halfspaces under product distributions. In *25th Annual Conference on Computational Complexity (CCC)*, 2010.

[GUV09] Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *J. ACM*, 56(4), 2009.

[GV04] Dan Gutfreund and Emanuele Viola. Fooling parity tests with parity gates. In *8th International Workshop on Randomization and Computation (RANDOM)*, pages 381–392. Springer, 2004.

[Hag91] Torben Hagerup. Fast parallel generation of random permutations. In *18th Colloquium on Automata, Languages and Programming (ICALP)*, pages 405–416. Springer, 1991.

[Hås87] Johan Håstad. *Computational limitations of small-depth circuits*. MIT Press, 1987.

[IN96] Russell Impagliazzo and Moni Naor. Efficient cryptographic schemes provably as secure as subset sum. *Journal of Cryptology*, 9(4):199–216, Fall 1996.

[LO43] John Littlewood and Albert Offord. On the number of real roots of a random algebraic equation. *III. Rec. Math. [Mat. Sbornik] N.S.*, 12:277–286, 1943.

[LRTV09] Shachar Lovett, Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Pseudorandom bit generators that fool modular sums. In *13th International Workshop on Randomization and Computation (RANDOM)*, volume 5687 of *Lecture Notes in Computer Science*, pages 615–630. Springer, 2009.

[LV10] Shachar Lovett and Emanuele Viola. Bounded-depth circuits cannot sample good codes. Manuscript available at http://www.ccs.neu.edu/home/viola, 2010.

[LVW93] Michael Luby, Boban Veličković, and Avi Wigderson. Deterministic approximate counting of depth-2 circuits. In *2nd Israeli Symposium on Theoretical Computer Science (ISTCS)*, pages 18–24, 1993.

[MST06] Elchanan Mossel, Amir Shpilka, and Luca Trevisan. On epsilon-biased generators in NC^0. *Random Struct. Algorithms*, 29(1):56–81, 2006.

[MV91] Yossi Matias and Uzi Vishkin. Converting high probability into nearly-constant time-with applications to parallel hashing. In *23rd ACM Symposium on Theory of Computing (STOC)*, pages 307–316, 1991.

[Nis91] Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Computer & Systems Sciences*, 49(2):149–167, 1994.

[Pag01a] Rasmus Pagh. Low redundancy in static dictionaries with constant query time. *SIAM J. Comput.*, 31(2):353–363, 2001.

[Pag01b] Rasmus Pagh. On the cell probe complexity of membership and perfect hashing. In *33rd Annual Symposium on Theory of Computing (STOC)*, pages 425–432. ACM, 2001.

[Păt08] Mihai Pătraşcu. Succincter. In *49th Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2008.

[PV10] Mihai Pătraşcu and Emanuele Viola. Cell-probe lower bounds for succinct partial sums. In *21th Symposium on Discrete Algorithms (SODA)*, pages 117–122. ACM-SIAM, 2010.

[Tre04] Luca Trevisan. A note on approximate counting for k-dnf. In *7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (RANDOM)*, volume 3122 of *Lecture Notes in Computer Science*, pages 417–426. Springer, 2004.

[Vio] Emanuele Viola. Bit-probe lower bounds for succinct data structures. To appear in *SIAM Journal on Computing*, STOC special issue.

[Vio05] Emanuele Viola. On constructing parallel pseudorandom generators from one-way functions. In *20th Annual Conference on Computational Complexity (CCC)*, pages 183–197. IEEE, 2005.

[Vio07] Emanuele Viola. Pseudorandom bits for constant-depth circuits with few arbitrary symmetric gates. *SIAM Journal on Computing*, 36(5):1387–1403, 2007.

[Vio10] Emanuele Viola. The complexity of distributions. In *51th Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2010.

[Yao82] Andrew Yao. Theory and applications of trapdoor functions. In *23rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 80–91. IEEE, 1982.