2013 IEEE Conference on Computational Complexity

The distinguishability of product distributions by read-once branching programs

John Steinberger
IIS, Tsinghua University
Beijing, China
Email: jpsteinb@gmail.com

***Abstract*—We improve the main result of Brody and Verbin [6] from FOCS 2010 on the power of constant-width branching programs to distinguish product distributions. Specifically, we show that a coin must have bias at least $\Omega(1/\log(n)^{w-2})$ to be distinguishable from a fair coin by a width w, length n read-once branching program (for each constant w), which is a tight bound. Our result introduces new techniques, in particular a novel "interwoven hybrid" technique and a "program randomization" technique, both of which play crucial roles in our proof. Using the same techniques, we also succeed in giving tight upper bounds on the maximum influence of monotone functions computable by width w read-once branching programs.**

I. INTRODUCTION

In [6] Brody and Verbin studied the question of distinguishing flips of a coin with a slight bias towards heads from those of a coin with a slight bias towards tails. More precisely, say that a coin is *ϵ-biased* if $\Pr[\text{Heads}] = \frac{1}{2} + \epsilon$. Given n flips of a coin which is either ϵ-biased or $(-\epsilon)$-biased, the question is to determine which type of bias is present. Since taking a majority vote of the tosses constitutes an optimal distinguishing strategy this question is uninteresting when the distinguisher is powerful enough to count (in which case a bias of $\epsilon = \Omega(1/\sqrt{n})$ is both necessary[1] and sufficient to distinguish with constant advantage[2]). However, the problem seems both natural and interesting for space-bounded distinguishers, and in particular for distinguishers having only a constant amount of space.

As their main result, Brody and Verbin [6] give bounds on the ability of constant width *read-once branching programs* (ROBPs) to distinguish biased coins. A read-once branching program is a model of (non-uniform) space bounded computation in which each bit of input is accessed only once, in order. (We give a formal definition of read-once branching programs in Section 2. A glance at Figure 1, however, should suffice to understand the model.) They show, among others, that ROBPs of width $w \geq 3$ can distinguish coins of suprisingly small bias: by computing a recursive tribes function, a length n ROBP of width w can distinguish an ϵ-biased coin from a $(-\epsilon)$-biased coin already for $\epsilon = 1/\log(n)^{w-2}$. (By "can distinguish" we mean, here and later, "can distinguish with constant (i.e. $\Omega(1)$) advantage".) In particular, a width 3 ROBP of length n can distinguish a $(1/\log n)$-biased coin from a $(-1/\log n)$-biased coin. (This last observation was also made, essentially, by Braverman et al. [5].)

On the lower bound side, Brody and Verbin show that, for constant w, a length n width w ROBP cannot distinguish $\pm\epsilon$-biased coins unless $\epsilon = \Omega(1/\log(n)^w)$. The lower bound is therefore off from the upper bound by a factor $\log(n)^2$, which seems substantial for programs of small width (e.g., width 3, for which the upper bound is $\epsilon = O(1/\log(n))$ and the lower bound is $\epsilon = \Omega(1/\log(n)^3)$). In this paper we give an improved lower bound that matches the upper bound of [6]. Namely, we show that, for constant[3] w, the smallest bias ϵ that can be distinguished by a width w length n ROBP is $\Omega(1/\log(n)^{w-2})$. Our analysis is also shorter than Brody and Verbin's. More interestingly than simply achieving a tight lower bound, however, is the fact that our result introduces new proof techniques that could be of independent interest for the study of ROBPs and, more generally, for the problem of derandomizing space-bounded computations. These techniques are described further below.

Note that a sequence of n independent tosses of a biased coin is a special case of a product distribution.

---

[1] A possible approach for proving necessity (since the relevant statistical distance is not so obvious to upper bound from first principles) is to use Hellinger distance. See for example [3].

[2] The *advantage* of a distinguisher D at distinguishing distributions X and Y is $|\Pr[D(X) = 1] - \Pr[D(Y) = 1]|$. See Section 2 for more precise definitions.

[3] The fact that w is constant in particular implies that $\Omega(\cdot)$- and $O(\cdot)$- notation refers exclusively to function growth with respect to n, and may hide constants that depend on w. In [6] the hidden constant is 1000^{-w}. Our hidden constant is 2^{-w}.

1093-0159/13 $26.00 © 2013 IEEE
DOI 10.1109/CCC.2013.33

IEEE computer society

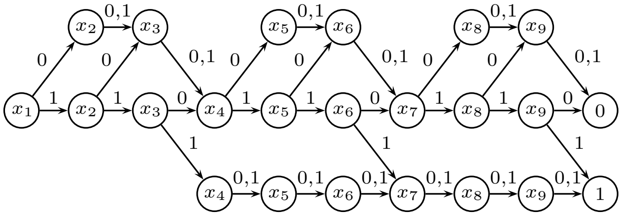

Figure 1: A width 3 read-once branching program (computing a tribes function).

(A sequence of random variables $X = (X_i)_{i=1}^n$ is a *product distribution* if and only the X_i's are (totally) independent.) One can consider, more generally, the power of a length n read-once branching program whose edges are labeled by elements of some finite alphabet Σ at distinguishing two product distributions $X = (X_i)_{i=1}^n$, $Y = (Y_i)_{i=1}^n$ where $X_i, Y_i \in \Sigma$ for all i.

Generalizing results on the distinguishability of biased coins to the distinguishability of arbitrary product distributions presupposes some kind of metric for measuring the closeness of two product distributions (i.e., requires generalizing the parameter ϵ). As explained in [6], it makes more sense, in this context, to measure closeness by probability ratios (bounding these to be near 1) rather than by probability differences (bounding these to be near 0). We say two product distributions $X = (X_i)_{i=1}^n \in \Sigma^n$, $Y = (Y_i)_{i=1}^n \in \Sigma^n$ are ϵ-*close in ratio* if for every $1 \le i \le n$ and for every $\alpha \in \Sigma$, either $\Pr[X_i = \alpha] = \Pr[Y_i = \alpha] = 0$, or else $\Pr[X_i = \alpha] \ne 0$, $\Pr[Y_i = \alpha] \ne 0$ and

$$\frac{\Pr[X_i = \alpha]}{\Pr[Y_i = \alpha]} \ge 1 - \epsilon, \qquad \frac{\Pr[Y_i = \alpha]}{\Pr[X_i = \alpha]} \ge 1 - \epsilon.$$

It is easy to see, for example, that an ϵ-biased coin is 4ϵ-close in ratio to a $(-\epsilon)$-biased coin.

Our results are most easily phrased and proved in the context of ϵ-close in ratio product distributions. Our main result is that a length n ROBP of constant width w cannot distinguish two ϵ-close in ratio product distributions unless $\epsilon = \Omega(1/\log(n)^{w-2})$. This directly implies our lower bound on the distinguishability of ϵ-biased coins, and also matches the upper bounds of Brody and Verbin (given by ϵ-biased coins). In fact, ϵ-close in ratio product distributions were already considered by Brody and Verbin themselves, who, by reducing to the case of ϵ-biased coins, proved that $\epsilon = \Omega(1/\log(|\Sigma|n)^{3w})$ is necessary to distinguish two ϵ-close in ratio product distributions over Σ^n, and that $\epsilon = \Omega(1/\log(n)^{2w})$ is necessary when $|\Sigma| = 2$. Our own lower bound shows there is essentially no difference between the cases $|\Sigma| = 2$ and $|\Sigma| > 2$: a larger alphabet size does not help the distinguisher.

The techniques used in our proof are roughly threefold. We use, firstly, the *collision lemma* of Brody and Verbin, which is a structural observation about ROBPs that are optimal distinguishers, and which we strengthen slightly for our purposes. A second component of the proof is a hybrid argument whose two endpoints are the product distributions $(X_i)_{i=1}^n$ and $(Y_i)_{i=1}^n$. Here the bits that change distribution from one hybrid to the next form each time an arithmetic progression (which is important for the argument). As these various arithmetic progressions are parallel and interleaved, we call our set of hybrids a set of *interwoven hybrids* (we are not aware of a similar set of such hybrids being used before). Our hybrid argument replaces a more standard random restriction argument by Brody and Verbin. Finally, the third main proof technique we use is *program randomization* which, in a nutshell, randomizes the distinguisher in order to compensate for certain helpful initial modifications made to the input distributions $(X_i)_{i=1}^n$ and $(Y_i)_{i=1}^n$ (see Section IV for more details). Program randomization is also an original contribution of the paper. While it is crucial to the final bounds, we consider it of secondary importance compared to the collision lemma and to the interwoven hybrid technique.

In this paper's full version [12] we give a second application of the same basic set of techniques (program randomization excluded) to the upper bounding of the maximum total influence of monotone functions computable by width w ROBPs. Our main result is that a ROBP of width $w \ge 2$ and of length $n \ge 2$ that computes a monotone function has total influence at most $4\lceil 1.5\log(n)\rceil^{w-2}$. This bound is also tight, as can be verified by considering a recursive tribes function.

II. DEFINITIONS

A branching program of width w and length n is a directed acyclic graph with n layers of w nodes each and a final layer with two nodes (accept and reject). Each non-ouptut node is labeled by a coordinate $(k, j) \in [n] \times [w]$; output nodes are labeled by coordinates $\{n + 1\} \times \{1, 2\}$, with $(n + 1, 1)$ being the accept node and $(n + 1, 2)$ being the reject node.

A node is *in layer* k, $1 \le k \le n + 1$, if its label is of the form $(k, \cdot)$. The edges of the graph are labeled by elements of the *input alphabet* Σ (a finite set). Each node in every layer $k \le n$ has one outgoing edge labeled α for each element $\alpha \in \Sigma$ whose endpoint is a node in layer $k + 1$. The branching program has a designated *start node* in the first layer, typically the node $(1,1)$. The computation of a branching program of length n on a string $x = x_1 \cdots x_n \in \Sigma^n$ is defined the natural way, by following the edge labeled x_i at step i, starting from the start node. We note the type of branching program just described is *read-once* since each character of x is examined at exactly one layer of the program.

Let f be a (read-once) branching program of length n and width w. If α is an element of the input alphabet Σ and $k \in [n]$, the *α-transition function of f at layer k* is the function $\tau_\alpha : [w] \to [w]$ such that $\tau_\alpha(i) = z$ iff the edge labeled α leaving node (k,i) has endpoint $(k+1, z)$. We say τ_α *contains a collision* if τ_α is not a permutation, i.e. if $\tau_\alpha(i) = \tau_\alpha(j)$ for some $i \neq j$.

The k-th layer of a ROBP f *equals* the j-th layer of a ROBP g if f and g have the same width w, are defined over the same input alphabet Σ, and if the α-transition function of f at layer k is identical to the α-transition function of g at layer j for every $\alpha \in \Sigma$.

The statistical distance of two random variables X, Y of same range is written $\Delta(X,Y)$. Namely, if X and Y take values in a set S, then

$$\Delta(X,Y) = \frac{1}{2} \sum_{b \in S} |\Pr[X = b] - \Pr[Y = b]|.$$

If f is a ROBP of length n over the alphabet Σ and if $X, Y \in \Sigma^n$ are two random variables, then f's *advantage* at distinguishing X and Y is defined as the statistical distance

$$\Delta(f(X), f(Y)).$$

(We note this is a statistical distance between two probability distributions on the output nodes of f.) This differs from the traditional definition of f's advantage as $|\Pr[f(X) = 1] - \Pr[f(Y) = 1]|$, but it is easy to see the two definitions are equivalent.

We write $X \sim X'$ when X, X' induce identical probability distributions over their (identical) ranges.

III. RESULTS

Our main result is an upper bound on the advantage $\Delta(f(X), f(Y))$ of a width w ROBP f at distinguishing ϵ-close product distributions $X, Y \in \Sigma^n$ for an arbitrary finite alphabet Σ. While our original interest lies with constant values of w, our main result, given by the next theorem, is slightly more general, as it also allows "small" non-constant w.

Theorem 1: There is a function[4] $\lambda(n) = o(1)$ such that for any positive integers n, w with $2 \leq w \leq \log n / \operatorname{loglog} n$, for any product distributions X, $Y \in \Sigma^n$ that are ϵ-close in ratio, and for any read-once branching program f over the alphabet Σ of width w and length n,

$$\Delta(f(X), f(Y)) \leq \epsilon(2\log(n))^{w-2}(1 + \lambda(n)). \qquad (1)$$

In particular, if w is constant, ϵ needs to be at least $\Omega(1/\log(n)^{w-2})$ in order for X and Y to be distinguishable with constant advantage, where the hidden constant[5] (depending on w but not on n) is 2^{-w}. This lower bound on ϵ is tight up to a constant factor: as shown in [6], width w, length n ROBPs can distinguish coins of bias $\pm\epsilon$ already for $\epsilon = O(1/\log(n)^{w-2})$. (For full disclosure, the hidden constant in the latter $O(\cdot)$ is 3^w; hence, there is still a gap between the upper and lower bounds as far as the constant factors are concerned.)

In the paper's full version [12] we also prove an upper bound on the maximum total influence of monotone functions computable with w ROBPs, which constitutes our second main result and reads as follows:

Theorem 2: Let $f : \{0,1\}^n \to \{0,1\}$ be a monotone boolean function computable by a ROBP of width w and length n. Then

$$\operatorname{Inf}(f) \leq 4\lceil 1.5\log(n)\rceil^{w-2}.$$

For constant w this bound is also tight up to a multiplicative factor, as can be seen using a recursive tribes function of depth $w - 1$ with the same tribe sizes as in [6]. (See also [1].)

IV. PROOF OVERVIEW

This section gives a self-contained overview of the proof of Theorem 1. For simplicity, we sketch the proof for the case $\Sigma = \{0,1\}$ (which anyway captures the full complexity of the problem). Moreover, we first sketch the proof for the case of distinguishing $\pm\epsilon$-biased coins and, later, discuss how to handle ϵ-close in ratio distributions (which, indeed, require an additional idea).

Let $X \in \{0,1\}^n$ be the product distribution of an ϵ-biased coin, and let $Y \in \{0,1\}^n$ be the product distribution of a $(-\epsilon)$-biased coin. Let X_j be the j-th bit of X.

---

[4] I.e., $\lim_{n\to\infty} \lambda(n) = 0$.

[5] In fact, for constant w, (1) can be replaced with $\Delta(f(X), f(Y)) \leq \epsilon(\delta \log(n))^{w-2} + \lambda_\delta(n)$ where $\delta > 1$ is any constant and where $\lambda_\delta(n) \to 0$ now depends on δ. Thus a sharper statement would say that the hidden constant is really "δ^w for any $\delta > 1$". We refer to the full version [12] for more details.

Let $\mathcal{F}_w$ be the set of all (binary) ROBPs of length n and width w (the parameter n is elided for simplicity). Let

$$\delta_w = \max_{f\in\mathcal{F}_w} \Delta(f(X), f(Y))$$

be the maximum distinguishing advantage. The proof bounds δ_w by establishing the recurrence

$$\delta_w = O(\log n)\delta_{w-1} + o(1) \tag{2}$$

and by showing that $\delta_2 \leq \epsilon$. In fact the $o(1)$ term is $1/\mathrm{poly}(n)$, so that recursively "unfolding" the inequality gives

$$\delta_w \leq O(\log n)^{w-2}\epsilon + o(1).$$

Tweaking the constants then yields Theorem 1. We now sketch how (2) is established.

Let

$$\mathcal{F}_w^{\max} = \{f \in \mathcal{F}_w : \Delta(f(X), f(Y)) = \delta_w\}$$

be the set of "best distinguishers". Note $\mathcal{F}_w^{\max}$ is nonempty since $\mathcal{F}_w$ is finite. A crucial observation, due to Brody and Verbin [6], is that $\mathcal{F}_w^{\max}$ contains an element f_0 in which every transition function is either the identity from $[w]$ to $[w]$, or else is not a permutation of $[w]$ at all, but contains a collision. We call an ROBP with this property a *collision ROBP*, or cROBP for short. To upper bound δ_w it thus suffices to upper bound $\Delta(f(X), f(Y))$ for an arbitrary cROBP f of length n and width w. (A nearly identical observation is called the *collision lemma* in [6]. We maintain this terminology, even while our own collision lemma is slightly different. The difference is explained in Section V.)

Let f, therefore, be a cROBP of length n and width w. By dropping layers of f at which both transition functions are the identity (these have no effect), one can assume that every layer of f has at least one transition function with a collision.

To upper bound $\Delta(f(X), f(Y))$ we use a hybrid argument over distributions $Z_0, \ldots, Z_{c\log(n)}$ on $\{0,1\}^n$, such that $Z_0 = X$ and $Z_{c\log(n)} = Y$. Here $c > 0$ is a constant we will set later (in fact, $c = 2$ will do). More precisely, assuming $c\log(n)$ is an integer (otherwise substitute $\lceil c\log(n)\rceil$ for $c\log(n)$ throughout), Z_i is the product distribution whose j-th coordinate $Z_{i,j}$ is given by

$$Z_{i,j} = \begin{cases} Y_j & \text{if } (j \bmod c\log(n)) < i, \\ X_j & \text{otherwise.} \end{cases}$$

For example, Z_1 is the distribution such that

$$Z_{1,j} = \begin{cases} Y_j & \text{if } j \equiv 0 \bmod c\log(n), \\ X_j & \text{otherwise.} \end{cases}$$

Clearly, then, $Z_0 = X$ and $Z_{c\log(n)} = Y$.

We note that Z_i and Z_{i+1} differ on a set of bits whose indices form an arithmetic progression of step size $c\log(n)$. This is the key feature of these hybrids; in fact any sequence of $c\log(n)$ hybrids with this property, starting with X and ending with Y, would do as well (there are $(c\log n)!$ possible such sequences). Let $\mathcal{Z}_i \subseteq [n]$ be the set of bits at which (the definitions of) Z_i and Z_{i+1} differ. We call $Z_0, \ldots, Z_{c\log(n)}$ a sequence of "interwoven hybrids" because $\mathcal{Z}_0, \ldots, \mathcal{Z}_{c\log(n)-1}$ are interwoven arithmetic progressions of equal step size.

By a standard argument, it suffices to bound the distance $\Delta(f(Z_i), f(Z_{i+1}))$ between two neighboring hybrids. Let $\overline{Z} \in \{0,1\}^{[n]\setminus\mathcal{Z}_i}$ be the value of Z_i, Z_{i+1} on the bits outside $\mathcal{Z}_i$. Fixing a value of $\overline{Z}$ induces (in the natural way) a width w, length $|\mathcal{Z}_i|$ ROBP $f_{\overline{Z}} : \{0,1\}^{\mathcal{Z}_i} \to \{0,1\}$ taking as input the bits in $\mathcal{Z}_i$. Let $X' \in \{0,1\}^{\mathcal{Z}_i}$ be an ϵ-biased coin, and let $Y' \in \{0,1\}^{\mathcal{Z}_i}$ be a $(-\epsilon)$-biased coin. Then $f(Z_i)$ is equidistributed to $f_{\overline{Z}}(X')$, and $f(Z_{i+1})$ is equidistributed to $f_{\overline{Z}}(Y')$, with randomness taken over $\overline{Z}, X', Y'$. By elementary properties of statistical distance, one has

$$\Delta(f(Z_i), f(Z_{i+1})) \leq \mathbb{E}_{\overline{Z}}\Delta(f_{\overline{Z}}(X'), f_{\overline{Z}}(Y')). \tag{3}$$

The crucial observation is that, in fact, $f_{\overline{Z}}$ is (equivalent to) a width $w-1$ ROBP with high probability over $\overline{Z}$. This uses the fact that f is a cROBP. Consider the transition functions τ_0, τ_1 at layer k of $f_{\overline{Z}}$. By definition of $f_{\overline{Z}}$, these transition functions depend on $c\log(n) - 1$ consecutive bits of $\overline{Z}$. Let these $c\log(n) - 1$ bits have indices $j_1, \ldots, j_{c\log(n)-1}$ in f. To picture how τ_0, τ_1 are induced by $\overline{Z}$, consider w (distinguishable) pebbles placed on the w nodes of f at layer j_1. Then for a fixed value of $\overline{Z}$, we can assign in the natural way a path to each pebble, starting at layer j_1 and ending at layer $j_{c\log(n)-1} + 1 = j_1 + c\log(n) - 1$. Then τ_0 is the composition of the 0-transition τ_0' at layer $j_1 - 1$ of f with the function from $[w]$ to $[w]$ given by the pebble paths, and likewise τ_1 is the composition of the 1-transition τ_1' at layer $j_1 - 1$ of f with the same pebble paths. Moreover, note that if two pebbles collide, they cannot separate again; thus, if two pebbles collide, τ_0 and τ_1 have at most $w - 1$ nodes in the union of their ranges.

Since f is a cROBP, there are values $b_1, \ldots, b_{c\log(n)-1} \in \{0,1\}$ such that the b_i-transition at layer j_i of f has a collision. By the above remarks, if the j_h-th bit[6] of $\overline{Z}$ is equal to b_h for any $1 \leq h \leq c\log(n) - 1$, then τ_0, τ_1 have joint range

---

[6] We index the bits of $\overline{Z}$ by their original index in Z_i, Z_{i+1}.

of size at most $w-1$. But any coordinate of $\overline{Z}$ is equal to a given binary value with probability at least $\frac{1}{2}-\epsilon$, since each coordinate of $\overline{Z}$ is distributed either according to X or according to Y; namely,

$$\Pr[\overline{Z}_{j_h}=b_h] \geq \frac{1}{2}-\epsilon \tag{4}$$

for any $1 \leq h \leq c\log(n)-1$. Thus the probability that no collisions occur among the pebbles as they travel from layer j_1 to layer $j_{c\log(n)-1}+1$ is, in the worst case, at most

$$\big(\frac{1}{2}+\epsilon\big)^{c\log(n)-1} \approx \frac{1}{n^c}.$$

(Where we use $\epsilon=o(1)$; we are being, here, a bit informal for the sake of the proof sketch.) By a union bound, the probability that *any* of the $n/c\log(n)$ pairs of transition functions of $f_{\overline{Z}}$ do not have joint range of size at most $w-1$ is at most $\approx 1/n^{c-1}\log(n)$. Thus, $f_{\overline{Z}}$ can be written as a width $w-1$ ROBP with probability at least $\approx 1-\frac{1}{n^{c-1}\log(n)}$, with the probability taken over $\overline{Z}$. This allows us to upper bound (3) by

$$\mathbb{E}_{\overline{Z}}\Delta(f_{\overline{Z}}(X'),f_{\overline{Z}}(Y')) \leq O\Big(\frac{1}{n^{c-1}\log(n)}\Big)+\delta_{w-1}.$$

(In fact, one could even replace δ_{w-1} with the advantage of the best distinguisher of length $n/c\log(n)$ and of width $w-1$, but such an optimization has little effect for constant-width ROBPs.) Finally, summing together the distances between the $c\log(n)$ pairs of neighborhing hybrids, one thus obtains that

$$\begin{aligned}\delta_w &= \Delta(f(X),f(Y))\\ &\leq c\log(n)O\Big(\frac{1}{n^{c-1}\log(n)}\Big)+c\log(n)\delta_{w-1}\\ &= O\Big(\frac{1}{n^{c-1}}\Big)+c\log(n)\delta_{w-1},\end{aligned}$$

establishing (2).

Finishing the proof also requires showing that $\delta_2 \leq \epsilon$. This is not trivial and requires the collision lemma as well as a coupling argument. We refer to Sections V for more details.

When working with arbitrary product distributions that are ϵ-close in ratio, the above analysis breaks down in one crucial place: even when ϵ is very small, there is no guarantee that $\Pr[\overline{Z}_{j_h}=b_h]$ will be near $\frac{1}{2}$, cf. (4). Instead, $\Pr[\overline{Z}_{j_h}=b_h]$ could be arbtrirarily close to 0. The probability that no collisions occur among the pebbles could thus be arbitrarily close to 1, and, therefore, $f_{\overline{Z}}$ is no longer equivalent to a width $w-1$ program with high probability.

In view of circumventing this (apparently complete) breakdown of the argument, note first that we do not care if $\Pr[\overline{Z}_{j_h}=b_h]$ is low if *both* the 0-transitions and 1-transitions at layer j_h contain collisions; in this case, indeed, we obtain width reduction with probability 1. Assume, therefore, wlog, that the 0-transition at layer j_h contains a collision, whereas the 1-transition is the identity function. Moreover assume that $\Pr[\overline{Z}_{j_h}=0]$ is low. To be concrete, say

$$\Pr[X_{j_h}=0]=\frac{1}{n}, \qquad \Pr[Y_{j_h}=0]=\frac{0.9}{n}. \tag{5}$$

Such values would be compatible with $\epsilon=0.1$, and would imply $\Pr[\overline{Z}_{j_h}=0]\leq\frac{1}{n}$.

The intuition is that in the case above, $\overline{Z}_{j_h}$ is quite likely to be equal to 1, which is an identity transition function, and therefore *it is quite likely the program does nothing at all at layer j_h*. Namely, the program is, with high probability, not reacting to input bit j_h, and layer j_h is therefore "wasted with high probability" for the program.

To leverage this intuition, let $f^\perp$ be the ROBP identical to f, but whose 0-transition function and 1-transition function at layer j_h are both the identity. Note that with high probability over the input distributions X and Y, $f^\perp$ computes the same as f (assuming (5)). We define a random ROBP f^* to be

$$f^* = \begin{cases} f^\perp & \text{w.p. } 1-\frac{1}{\gamma},\\ f & \text{w.p. } \frac{1}{\gamma}\end{cases}$$

(w.p. = with probability) where $\gamma \geq 1$ is chosen as large as possible such that the distributions X^*, Y^* defined by

$$X_k^* = \begin{cases} X_k & \text{if } k\neq j_h\\ 0 & \text{w.p. } \gamma\Pr[X_{j_h}=0] \text{ if } k=j_h\\ 1 & \text{w.p. } 1-\gamma\Pr[X_{j_h}=0] \text{ if } k=j_h\end{cases} \tag{6}$$

$$Y_k^* = \begin{cases} Y_k & \text{if } k\neq j_h\\ 0 & \text{w.p. } \gamma\Pr[Y_{j_h}=0] \text{ if } k=j_h\\ 1 & \text{w.p. } 1-\gamma\Pr[Y_{j_h}=0] \text{ if } k=j_h\end{cases} \tag{7}$$

are ϵ-close in ratio. Note that $f^*(X^*)$, $f^*(Y^*)$ are distributed identically to $f(X)$, $f(Y)$, respectively, since $\Pr[f^*=f \wedge X^*_{j_h}=0]=\Pr[X_{j_h}=0]$ and $\Pr[f^*=f\wedge Y^*_{j_h}=0]=\Pr[Y_{j_h}=0]$. (Note that $X_{j_h}=0$ exactly when the non-identity transition is used at layer j_h in the computation of f on X, and that the event $f^*=f\wedge X^*_{j_h}=0$ occurs exactly when the non-identity transition is used at layer j_h in the computation of f^* on X^*.)

In the example above, in which $\epsilon=0.1$, this means choosing γ as large as possible such that

$$\frac{1-\gamma\frac{1}{n}}{1-\gamma\frac{0.9}{n}} \geq 1-\epsilon=0.9.$$

A short computation shows the maximum value of γ is $\gamma = n/1.9$. Thus, in this case,

$$\Pr[X^*_{j_h} = 0] = \frac{1}{1.9}, \qquad \Pr[Y^*_{j_h} = 0] = \frac{0.9}{1.9}.$$

Note the difference with (5): both probabilities have moved away from 0, and are now close to $\frac{1}{2}$.

In the proof, the above operation consisting of randomizing the program at a transition (to be the original program w.p. $\frac{1}{\gamma}$, or to be the identity w.p. $1-\frac{1}{\gamma}$) and of simultaneously boosting by a factor γ in each distribution the probability of the input value giving a collision at that layer, is carried out for all layers of the program at once, with the value of γ individually computed for each layer. The resulting randomized program f^* is defined by choosing each layer independently to be either the identity or the original layer, with respect to the relevant probabilities. Since $f(X), f(Y)$ are distributed identically to $f^*(X^*), f^*(Y^*)$, with randomness taken also over the choice of f^*, we have that

$$\begin{aligned}\Delta(f(X), f(Y)) &= \Delta(f^*(X^*), f^*(Y^*)) \\ &\leq \mathbb{E}_{f^*}\Delta(f^*(X^*), f^*(Y^*)).\end{aligned}$$

In the righmost expression, the statistical distance is computed solely over the randomness induced by X^* and Y^*, for a fixed value of f^*. Because of the probability boosting, one can show that if the b-transition at the k-th layer of f^* has a collision while the $(1-b)$-transition does not (note this implies the same statement holds in f), then

$$\min(\Pr[X^*_k = b], \Pr[Y^*_k = b]) \geq \frac{1-\epsilon}{2-\epsilon}.$$

Since the latter probability is near $\frac{1}{2}$, the same hybrid method used for biased coins can be used to upper bound $\Delta(f^*(X^*), f^*(Y^*))$ for any fixed value of f^*.

We note that the (central) idea of obtaining width reduction of the program via collisions originates in [6]. There, random restrictions are used to obtain collisions and width-reduction. Our paper swaps random restrictions for a hybrid argument, which has the advantage that one can control the position of the restricted bits (these being, in the hybrid argument, the bits common to two neighboring hybrids). Having long intervals of consecutive restricted bits augments the chance of obtaining at least one collision in each of these intervals, and thus improves the chance of obtaining width-reduction. (On the other hand, longer intervals means more hybrids, implying a tradeoff.)

As another point of comparison, we note that [6] eschews program randomization in favor of a (lossy) reduction from the problem of distinguishing "well-behaved" input distributions (with probabilities near $\frac{1}{2}$) to the problem of distinguishing "troublesome" input distributions (with probabilities near 0). It is partly this reduction which causes the alphabet size $|\Sigma|$ to appear in the final bound of [6] (whereas our bounds are independent of $|\Sigma|$).

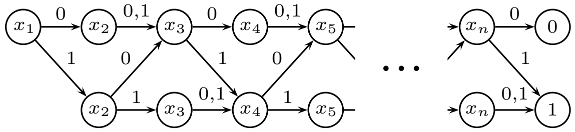

Figure 2: Width 2 branching program obtaining better than 2ϵ advantage at distinguishing a $(\frac{1}{2}+\epsilon, \frac{1}{2}-\epsilon)$-biased coin from an $(\frac{1}{2}-\epsilon, \frac{1}{2}+\epsilon)$-biased coin ($n$ odd, $n \geq 3$).

V. SOME FURTHER PROOF DETAILS: WIDTH TWO BRANCHING PROGRAMS AND THE COLLISION LEMMA

As explained, the proof of Theorem 1 relies on an inductive argument whose base case is an upper bound on the distinguishing power of width 2 branching programs. Intuitively, a ROBP of width 2 (say, when distinguishing a $\pm\epsilon$-biased coin) cannot do better than to determine acceptance based on the outcome of a single coin flip, given its limited memory—e.g., by ignoring all coin flips except for the last. Note that such a ROBP has advantage $(\frac{1}{2}+\epsilon) - (\frac{1}{2}-\epsilon) = 2\epsilon$, the statistical distance in a single coin flip.

However this intuition is incorrect. Indeed, a width 2 ROBP can distinguish a $(\pm\epsilon)$-biased coin with advantage approaching

$$2\epsilon\Big(\frac{3}{4}+\epsilon^2\Big)^{-1} \tag{8}$$

as the length n of the program goes to infinity, which is close to $\frac{4}{3}$ as large as 2ϵ for small ϵ. The program whose distinguishing advantage approaches this value is shown in Fig. 2. The program of Fig. 2 is, conjecturally, the best width 2 distinguisher of length $n = 2k+1$ for $(\pm\epsilon)$-biased coins, but we do not have a proof. (Also conjecturally, width two ROBPs of length $n = 2k+2$ do no better at distinguishing $(\pm\epsilon)$-biased coins than length $2k+1$ ROBPs.) Our own bound shows that width 2 ROBPs cannot distinguish $(\pm\epsilon)$-biased coins with advantage better than

$$1 - \frac{\frac{1}{2}-\epsilon}{\frac{1}{2}+\epsilon} = 2\epsilon\Big(\frac{1}{2}+\epsilon\Big)^{-1} \tag{9}$$

regardless of their length. For small ϵ, this is roughly 1.5 times as large as the conjectured optimal advantage (8), but this constant-factor discrepancy is unimportant for our final bound.

The general theorem which we prove on width two branching programs is the following:

Theorem 3: Let f be a width 2 ROBP of length n and let $X = (X_i)_{i=1}^n \in \Sigma^n$, $Y = (Y_i)_{i=1}^n \in \Sigma^n$ be two ϵ-close in ratio product distributions. Then $\Delta(f(X), f(Y)) \leq \epsilon$.

(We note that (9) is the direct application of Theorem 3.) The proof of Theorem 3 is in fact nontrivial and uses many ideas from the inductive proof described in Section IV, including (a strengthened version of) Brody and Verbin's collision lemma, program randomization, and a coupling argument. It would be interesting to know if an "easy" proof exists.

To state the collision lemma, which plays a key role in our results, we start by giving the formal definition of cROBPs.

Definition 1: A width w read-once branching program f is called a *collision read-once branching program* (cROBP) if every transition function τ_α of f is either the identity from $[w]$ to $[w]$ or else is not injective (i.e. is not a permutation).

Collision Lemma. (After [6].) Let $X, Y \in \Sigma^n$ be product distributions and let $w \geq 1$. Then there exists a cROBP f of length n and width w whose distinguishing advantage $\Delta(f(X), f(Y))$ is at least as great as the distinguishing advantage $\Delta(g(X), g(Y))$ of any length n width w ROBP g.

The collision lemma found in [6] states that the optimal distinguishing advantage can be achieved by a program f with the following property: at every layer of f, either all the transition functions are the identity, or else at least one of the transition functions contains a collision. This is weaker than our lemma, which implies that *all* non-identity transition functions contain a collision (i.e. are not permutations). While our version may seem much stronger at first glance, its proof only requires a minor modification of the proof of [6].

Acknowledgments. I would like to thank Kevin Matulef, Joshua Brody and Elad Verbin for helpful conversations at all stages of this work. This was supported by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61061130540, 61073174, and by NSF grant 0994380.

REFERENCES

[1] Kazuyuki Amano. Bounds on the size of small depth circuits for approximating majority. In *Proc. of the 36th International Colloquium on Automata, Languages and Programming*, 2009.

[2] Anindya De. Improved pseudorandomness for regular branching programs. In *Conference on Computational Complexity*, 2011.

[3] Boaz Barak, Ishay Haviv, Moritz Hardt, Anup Rao, Oded Regev, and David Steurer. Rounding parallel repetitions of unique games. In *Proc. of the 49th Annual ACM Symposium on the Foundations of Computer Science*, 2008.

[4] Andrej Bogdanov, Zeev Dvir, Elad Verbin, Amir Yehudahoff. Pseudorandom generators for width two branching programs. ECCC, 2009.

[5] Mark Braverman, Anup Rao, Ran Raz, Amir Yehudahoff. Pseudorandom generators for regular branching programs. In *Proc. of the 51st Annual ACM Symposium on the Foundations of Computer Science*, 2010.

[6] Joshua Brody, Elad Verbin. The coin problem and pseudorandomness for branching programs. In *Proc. of the 51st Annual ACM Symposium on the Foundations of Computer Science*, 2010.

[7] Bill Fefferman, Ronen Shaltiel, Christopher Umans, Emanuele Viola. On beating the hybrid argument. ITCS 2012.

[8] Martin Hellman, Thomas Cover. Learning with finite memory. *Ann. of Math. Stat.*, **41**, 1970.

[9] Michal Koucký, Prajata Nimbhorkar, Pavel Pudlak. Pseudorandom generators for group products. In *Proc. of the 43rd Annual ACM Symposium on the Theory of Computing*, 2011.

[10] Anup Rao, David Zuckerman. Pseudorandom generators for polynomial threshold functions. In *Proc. of the 51st Annual ACM Symposium on the Foundations of Computer Science*, 2010.

[11] Jiří Šíma, Stanislav Žák. Almost k-wise independent sets establish hitting sets for width-3, 1-branching programs. Computer Science Theory And Applications: 6th International Computer Science Symposium in Russia, St. Petersburg, 2011.

[12] John Steinberger, The indistinguishability of product distributions by read-once branching programs, (full version of this paper), `http://itcs.tsinghua.edu.cn/~john`.

[13] Emmanuele Viola. Randomness buys depth for approximate counting. Electronic Colloquium on Computational Complexity (ECCC), 17:175, 2010.