Time-Space Tradeoffs for Branching Programs

Paul Beame*
Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350
beame@cs.washington.edu

Michael Saks†
Dept. of Mathematics
Rutgers University
New Brunswick, NJ 08903
saks@math.rutgers.edu

Jayram S. Thathachar*
Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350
jayram@cs.washington.edu

Abstract

We obtain the first non-trivial time-space tradeoff lower bound for functions $f : \{0,1\}^n \to \{0,1\}$ on general branching programs by exhibiting a Boolean function f that requires exponential size to be computed by any branching program of length $(1+\epsilon)n$, for some constant $\epsilon > 0$. We also give the first separation result between the syntactic and semantic read-k models [10] for $k > 1$ by showing that polynomial-size semantic read-twice branching programs can compute functions that require exponential size on any syntactic read-k branching program. We also show a time-space tradeoff result on the more general R-way branching program model [10]: for any k, we give a function that requires exponential size to be computed by length kn q-way branching programs, for some $q = q(k)$.

1 Introduction

One of the long-standing open questions of complexity theory is whether polynomial-time is the same as log-space. One approach to this problem has been to look at tradeoffs between time and space for natural problems in P. For example, does the addition of a restriction on the space allowed prevent one from solving problems in P within specific polynomial time bounds? Despite significant progress given by Fortnow's recent time-space tradeoff lower bounds for SAT [13], this question remains unsolved.

One natural model for studying this question is that of Boolean branching programs, which simultaneously capture time and space in a clean combinatorial manner. In this model, a program for computing a function $f(x_1, x_2, \ldots, x_n)$ is represented as a DAG with a unique start node. Each non-sink node is labeled by a variable, and the arcs out of a node correspond to the possible values of the variable. The sink nodes are each labeled by an output value. Executing the program on a given input corresponds to following a path from the start node using the values of the input variables to determine the arcs to follow. The maximum length of a path corresponds to time and the logarithm of the number of nodes corresponds to space. An algorithm running simultaneously in linear time and logarithmic space corresponds to a linear-length, polynomial-size branching program. Thus the question of finding explicit functions in P for which no such branching program exists has been of significant research interest. (In fact, finding *any* explicit function for which this is known is still open; since branching programs are a non-uniform model of computation, Fortnow's lower bound does not apply to them.)

We give results on two distinct problems for branching programs, which we summarize in the next two subsections.

1.1 Lower bounds for single-output functions

There has been much success in proving time-space tradeoff lower bounds for *multi-output* functions in FP such as sorting, pattern matching, matrix-vector product, and hashing [8, 6, 1, 2, 14]. However, for single-output functions (those whose output is one bit) the state of our knowledge is pathetic: prior to this paper, there were no lower bounds known that are better than $n+o(n)$ for any explicit n variable function. The existing techniques for multi-variate functions involve some sort of "progress measure" which quantifies how much of the output has been produced. These techniques do not seem to give any non-trivial bounds for functions with a single output bit. For example, it is not known how to relate the apparently very similar problems of sorting and element distinctness, although time-space tradeoffs for element distinctness on the structured *comparison branching program* have been shown [9, 20].

The branching program model allows the domain of the variables to be any finite set. For variables taking values in a q element set, the nodes in the program have outdegree q, corresponding to the possible values. While the case of

* Research supported by NSF grant CCR-9303017.
† Research supported by NSF grant CCR-9700239. This work was done while on sabbatical at University of Washington.

greatest interest is the case that variables are 2-valued, the general q-valued case is an interesting challenge, which can potentially provide insights into the 2-valued case. Our first result is to exhibit an explicit family of functions $\mathcal{F}_q$, where for each q, the functions in $\mathcal{F}_q$ are single output functions on q-valued variables, such that the following property holds: for any k, there is a q such that the functions in family $\mathcal{F}_q$ can not be computed in length kn and polynomial size.

This result is unsatisfying because of the dependence on q. For each k, the q required for the bound can be quite large. For q-valued variables, the number of input bits is $n \log_2 q$ and what we really want is a lower bound that is superlinear in the number of input bits. Nevertheless, we believe this result is of some interest, both in its own right, and because the proof illustrates some basic ideas which we think may prove useful in this area.

Our second lower bound pertains to the "real" model, single output functions on 2-valued variables, i.e., Boolean functions. For this model, we obtain the first non-trivial length lower bound for polynomial size branching programs for functions whose output is a single bit: we exhibit an explicit family of functions in P and show that any sub-exponential size program for it must have length at least $1.0178n$. While this is only just barely non-trivial, it is the first such result in which the length divided by the number of variables is bounded away from one. Our lower bounds also apply to the more general model of non-deterministic branching programs.

The proofs introduce some new proof techniques, some of which extend past techniques of [10, 18]. First, we show that if a function f has a small size and length branching program, then it is possible to find a small set of decision trees, each of height that is a small fraction of n, such that the AND of the functions computed by the trees accepts no 0's of f and accepts a substantial fraction of the 1's of f. So proving a size-length lower bound on branching program reduces to showing that no such set of decision trees exists.

Similar decision trees arise in analyzing the restricted branching programs considered in [10, 18] but these trees are "oblivious" and thus depend on only a small fraction of variables. The main new step uses an interesting entropy argument. Very roughly, we show that for two decision trees of height $(1/2+\epsilon)n$, either it is the case that for the vast majority of inputs, the two trees together fail to look at a positive fraction of the variables, or the trees are "approximately oblivious" in the sense that the set of variables examined by each tree does not depend very much on the input.

1.2 Semantic versus syntactic read-k branching programs

As a step towards proving super-polynomial size lower bounds for linear length branching programs, a natural restriction is to require that each input bit be read at most some fixed number of times. This led to the definition of read-k branching programs [19] in which each input can be read at most k times. Many lower bounds have been shown for several functions on read-once branching programs (for example, see [16, 17]).

Another branching program restriction that has also been considered is that of *oblivious* branching programs which test the same variable at each time-step along any path. For oblivious branching programs, linear length and read-k for some constant k are essentially the same and several size-length tradeoff lower bounds for oblivious branching programs have been shown using this connection [3, 5]. Oblivious read-once branching programs, known as OBDD's, have been very useful as representations of functions used in verification [11, 12] and so have generated significant independent interest.

Borodin, Razborov, and Smolensky [10] observed that read-k branching programs come in two flavors, *syntactic* read-k in which all paths in the branching program must satisfy the read-k restriction and the more general *semantic* read-k in which only the paths consistent with some input must satisfy the restriction. They also proved strong size lower bounds for the syntactic read-k model. However, obtaining super-polynomial size lower bounds even for semantic read-twice branching programs is an open question. (It is easy to observe that there is no distinction between syntactic read-once and semantic read-once branching programs.)

Here, we show the first separation between the syntactic and semantic read-k models for $k > 1$ by showing that polynomial-size semantic read-twice branching programs can compute functions that require exponential size for any syntactic read-k branching programs. The functions we construct are based on a class of functions that were by introduced by Thathachar [18] to separate the power of read-k and read-$(k+1)$ in the syntactic model. These functions are exponentially hard for syntactic read-k, and seem to be hard for semantic read-k. We modify these functions so as to make them semantic read-twice while still retaining hardness for syntactic read-k.

2 Notation

Throughout X denotes a set of (usually n) variables which take on values from some finite set D; usually $D = \{0,1\}$. We say X is a D-valued variable set, and additionally d-valued if $d = |D|$. An *input* σ is, as usual, a point in D^X, the set of mappings from X to D. A *Boolean function* over X is a function mapping D^X to $\{0,1\}$.

A (nondeterministic) Boolean branching program P over D^X is a directed acyclic graph having a unique source (start) vertex, with sink vertices labeled by 0 or 1, non-sink

vertices labeled by elements of X, and edges labeled by elements of D. An edge labeled by $a \in D$ that is an out-edge of a vertex labeled by $x_i \in X$ is *consistent* with input $\sigma \in D^X$ iff $\sigma(x_i) = a$; a path in P is consistent with σ if all its edges are. P accepts input σ if there is some path in P consistent with σ leading from the start node to a sink node labeled 1. We call the number of vertices in P its *size* and the length of the longest path in P its *length*. (As in [10] one can also permit unlabeled `free' edges but this does not change the size measure by more than a quadratic amount.)

P is *deterministic* if every non-sink node has precisely one out-edge labeled a for each $a \in D$. P is a *decision tree* if its graph is a rooted tree.

A branching program of length d is *leveled* if its nodes can be partitioned into d sets $V_0, V_1, \ldots, V_d$ where V_0 is the source, V_d is the set of sink nodes (one per output value) and every arc out of V_i goes to V_{i+1}, for $0 \leq i < d$. It is well known[15] that every branching program P of size s and length d, can be converted into a leveled branching program P' of length d that has at most s nodes in each of its levels and computes the same function as P (and is deterministic if P is).

3 Decision Programs

For a given Boolean function f, and a given length $d \geq n$, we want to lower bound the size of the smallest branching program of length d that computes it. This section gives a simple lemma, which shows that such lower bounds can be obtained by analyzing a different model based on decision trees.

An (r, α)-*decision program* R is an r-tuple of decision trees $(T_1, T_2, \ldots, T_r)$, each of height αn. The function computed by R, denoted as f_R, is $\bigwedge_{i \in [1,r]} f_i$, where f_i is the function computed by T_i, for $i \in [1, r]$. We say that R is *compatible* with the function f if it accepts only 1's of f.

Theorem 1. *Let f be a Boolean function. Suppose that any (r, α)-decision program R that is compatible with f, accepts at most t inputs. Then, any nondeterministic (or deterministic) branching program of length $\alpha r n$ computing f has size at least $(\left|f^{-1}(1)\right|/t)^{1/(r-1)}$.*

Proof. Let P be any branching program of size s computing f in length $d = \alpha r n$ and let P' be an equivalent leveled program of length d with at most s nodes per level. An input is accepted by P if and only if there exist nodes $v_1, v_2, \ldots, v_{r-1}$ at levels $\alpha n, 2\alpha n, \ldots, (r-1)\alpha n$ in P' such that there is a path consistent with that input of length αn from v_i to v_{i+1} for $i = 0, \ldots, r-1$ where v_0 is the start node and v_r is the unique accepting node at level $r\alpha n$. Let $T_{v_i, v_{i+1}}$ be a decision tree of height αn creating by expanding the αn levels of P' rooted at v_i into a tree and labeling a leaf 1 if and only if the corresponding path in P' starting at v_i would reach v_{i+1}. Therefore

$$f = \bigvee_{v_1, \ldots v_{r-1}} \bigwedge_{i=0}^{r-1} T_{v_i, v_{i+1}}.$$

Each conjunction in this expansion is an (r, α)-decision program, therefore it can accept at most t inputs of $f^{-1}(1)$. By construction, there are at most s^{r-1} many choices for $v_1, v_2, \ldots, v_{r-1}$, so $s^{r-1}t \geq \left|f^{-1}(1)\right|$, from which the bound on s follows. □

4 Functions based on quadratic forms

The functions for which we prove lower bounds are based on quadratic forms. (Similar functions based on bilinear forms were considered in [10].) Let $M = M_n$ be an $n \times n$ matrix over $GF(q)$. Define the function $QF_M : GF(q)^n \to \{0, 1\}$ to be true on input σ (viewed as a vector of length n) if $\sigma^T M \sigma = 0 (\mathrm{mod}\ q)$. We define the function BQF_M to be the restriction of QF_M to the domain $\{0, 1\}^n$.

To prove size-length tradeoffs for BQF_M or QF_M, we will require that the matrix M satisfy certain properties, which are stated in the next section. Explicit examples of matrices satisfying these properties are the *Sylvester matrices*. For any odd prime power q and any $n = 2^k$, the $n \times n$ Sylvester matrix N is defined over $GF(q)$ and has rows and columns indexed by binary vectors of length k. The (i, j)-th entry of N is $(-1)^{\langle i,j \rangle}$, where $\langle i, j \rangle$ denotes the inner product of i and j. We also consider the *modified Sylvester matrix*, $N^{[0]}$, obtained by setting the diagonal entries of a Sylvester matrix N to 0.

Theorem 2. *There is an $\epsilon > 0$ such that any branching program of length $(1 + \epsilon)n$ computing BQF_N, where N is the $n \times n$ Sylvester matrix over $GF(3)$, requires size $2^{\Omega(n)}$.*

Theorem 3. *For every integer k there exists a prime power q and a constant $\gamma > 0$ such that for all sufficiently large n the following holds: Let $N^{[0]}$ be an $n \times n$ modified Sylvester matrix over $GF(q)$. Then any length kn non-deterministic branching program for $QF_{N^{[0]}}$ requires size at least $q^{\gamma n}$*

The conclusion of Theorem 3 holds, more generally, whenever N is a Generalized Fourier Transform (GFT) matrix (see [10]). For any finite Abelian group G, let G^* be the set of multiplicative characters of G mapping elements of G to $GF(q)^*$, that is, $\chi(g_1 g_2) = \chi(g_1)\chi(g_2)$ for any $g_1, g_2 \in G$ and $\chi \in G^*$. Provided that q is relatively prime to $|G|$, it is known that there are $|G|$ distinct characters and that they are linearly independent when viewed as a vector space over $GF(q)$. Let $N = N_{G,G^*}$ be the matrix in which the $(g, \chi)^{\text{th}}$ element equals $\chi(g)$, for all $g \in G$ and $\chi \in G^*$. Sylvester matrices of dimension $n \times n$, where $n = 2^k$ for

some k, can be shown to be special cases of GFT matrices corresponding to the additive group of $GF(2)^k$.

5 A lower bound criterion

Our lower bounds for the functions described in the previous sections are obtained in two steps. First, we identify two parameterized combinatorial properties of functions and show that, for any function f satisfying these two properties, the branching programs for f obey certain length-size tradeoffs (depending on the parameters). Second, we show that the functions of the previous section satisfy these properties with values for the parameters that are good enough to give non-trivial tradeoffs.

Let f denote any Boolean function on a D-valued set of n variables, X. The first of the two properties, $\mathcal{P}(\theta)$, is parameterized by a real number $\theta \in (0,1)$, the second, $\mathcal{Q}(\Phi)$, is parameterized by a non-decreasing function $\Phi : [0,1] \longrightarrow [0,1]$.

$\mathcal{P}(\theta)$**:** For any partial assignment ρ to at most $(1-\theta)n$ variables of X, $f\lceil_\rho$ is a non-constant function. In particular, f has at least $|D|^{(1-\theta)n}$ satisfying inputs in D^X.

$\mathcal{Q}(\Phi)$**:** For any pair of functions g_1, g_2 such that $g = g_1 \wedge g_2$ is compatible with f, if there are two disjoint subsets $A_1, A_2 \subseteq X$, each of size at least δn, such that g_i does not depend on the variables in A_i, then g accepts at most $|D|^{(1-\Phi(\delta))n}$ inputs of $f^{-1}(1)$.

For our functions, these properties can be realized for the specific Φ and θ as stated in the lemma below. We postpone the proof of this lemma to Section 8.

Lemma 4. *Let M be a GFT matrix over $GF(q)$, where q is a prime power.*

1. *For any subset D of $GF(q)$ of size at least 2, the restrictions of the functions QF_M and $QF_{M^{[0]}}$ to D^n both satisfy $\mathcal{Q}(\Phi)$ for $\Phi(\delta) = \delta^2$.*
2. *$QF_{M^{[0]}}$ satisfies $\mathcal{P}(2/n)$.*
3. *If M is the Sylvester matrix over $GF(3)$, then BQF_M satisfies $\mathcal{P}(24 \log n/\sqrt{n})$.*

Our main general lower bounds on branching program size are expressed in terms of these two properties. The first such result is:

Theorem 5. *Let f be a boolean function on D^n, where $n \geq (k+1)2^{k+4}$. Suppose f satisfies $\mathcal{P}(\theta)$ and $\mathcal{Q}(\Phi)$ for some θ and Φ. Then, any length kn non-deterministic branching program for f requires size at least*

$$\frac{1}{2}\left(\frac{|D|^{\Phi(\beta)-\theta}}{3}\right)^{n/(k(k+1)2^{k+4})}$$

where $\beta = 1/2^{k+1}$.

The proof of this theorem is given in section 6. The theorem yields exponential size lower bounds for linear depth branching programs in the case that D is "large enough". In particular, we obtain Theorem 3: Taking f to be the function $QF_{N^{[0]}}$ for some q and n, then, by Lemma 4, the hypotheses of Theorem 5 are satisfied with $\theta = 2/n$ and $\Phi(\delta) = \delta^2$. Choosing q so that $\log\log q \geq Ck$ for some sufficiently large constant C, the conclusion of Theorem 5 implies the conclusion of Theorem 3.

Theorem 5 is of no use for the case $|D| = 2$. For this case we have the following theorem:

Theorem 6. *Let $\epsilon, \theta > 0$ with $\epsilon + \theta \leq 1/4$, let $\Phi : [0,1] \longrightarrow [0,1]$ and let n be a sufficiently large integer. If $f : \{0,1\}^n \longrightarrow \{0,1\}$ satisfies $\mathcal{P}(\theta)$ and $\mathcal{Q}(\Phi)$ then, any length $(1+\epsilon)n$ non-deterministic branching program for f requires size at least $2^{(\alpha/3-\theta)n-1}$, where α equals*

$$\Phi\left(\frac{1-\epsilon-\theta}{2}\right) - \epsilon - \theta - (1+\epsilon)H\left(\frac{2(\epsilon+\theta)}{1+\epsilon}\right) - 2H(\theta)$$

and $H(p) = -p\log_2 p - (1-p)\log_2(1-p)$ is the binary entropy function.

The proof of this theorem appears in Section 7. Using the theorem we can deduce Theorem 2 as follows. Fix n large enough and let N be the $n \times n$ Sylvester matrix over $GF(3)$. From Lemma 4, BQF_N satisfies $\mathcal{P}(\theta)$ for $\theta(n) \to 0$ and $\mathcal{Q}(\Phi)$ for $\Phi(\delta) = \delta^2$. Applying Theorem 6, we conclude that for any $\epsilon > 0$ and for sufficiently large n, if $\alpha < (1-\epsilon)^2/4 - \epsilon - (1+\epsilon)H(2\epsilon/(1+\epsilon))$ then any branching program for BQF_M of length at most $(1+\epsilon)n$ has size at least $2^{\alpha n/3}$. It can be checked that for $\epsilon \leq 0.0178$, the expression upper bounding α is strictly positive. Hence, any branching program of length at most $1.0178n$ that computes BQF_M must have exponential size.

The next three sections give the proofs of Theorems 5 and 6 and Lemma 4.

6 Lower bounds for functions over large domains

In this section, we prove Theorem 5. Let f be as hypothesized. Because f satisfies $\mathcal{P}(\theta)$, $\left|f^{-1}(1)\right| \geq D^{(1-\theta)n}$. Let $p = (k+1)2^{k+4}$ and suppose $n \geq p$. Fix any $(kp, 1/p)$-decision program $R = (T_1, T_2, \ldots, T_{kp})$ that is compatible with f. We will prove that $\left|R^{-1}(1)\right| \leq |D|^{(1-\Phi(\beta))n} 3^n 2^{kp}$. The conclusion of the Theorem then follows immediately from Theorem 1.

A generalization of a combinatorial lemma from [18] says that for $n \geq p$, given a collection of kp subsets of $[1,n]$ each of size at least $n(1-1/p)$, there exists a pair of disjoint

sets A and B of size at least βn, where $\beta = 1/2^{k+1}$, such that each set of the collection contains A or B.

For each input σ, let $S_i(\sigma)$ be the set of variables that are not read by T_i on input σ. Note that each set $S_i(\sigma)$ has size at least $n(1-1/p)$. Consider all quadruples (A_1, A_2, I_1, I_2) where $|A_1| = |A_2| = \beta n$ are disjoint sets of variables and I_1, I_2 are complementary subsets of $\{1, 2, \ldots, kp\}$. Call such a quadruple *eligible*. We say that (A_1, A_2, I_1, I_2) covers input σ if for each $i \in I_1$, $A_1 \subseteq S_i(\sigma)$ and for each $i \in I_2$, $A_2 \subseteq S_i(\sigma)$. By the combinatorial lemma, every input σ is covered by some eligible quadruple (A_1, A_2, I_1, I_2). We now upper bound the number of accepted inputs covered by a given eligible quadruple.

Given a pair (A, I) where A is a variable subset and $I \subseteq \{1, 2 \ldots, kp\}$, define $g_{A,I}$ to be the function that accepts σ if and only if, for every $i \in I$, $A \subseteq S_i(\sigma)$ and T_i accepts σ. It is easy to see that $g_{A,I}$ does not depend on the variables in A. Also for any eligible quadruple (A_1, A_2, I_1, I_2), the function $g_{A_1,A_2,I_1,I_2} = g_{A_1,I_1} \wedge g_{A_2,I_2}$ accepts exactly the set of 1's of f that are covered by the quadruple. Therefore, by property $\mathcal{Q}(\Phi)$, $\left|g^{-1}_{A_1,A_2,I_1,I_2}(1)\right| \leq |D|^{(1-\Phi(\beta))n}$. Since the number of distinct quadruples (A_1, A_2, I_1, I_2) can be crudely upper bounded by $3^n 2^{kp}$, we conclude that $\left|R^{-1}(1)\right| \leq \left|\bigcup_{(A_1,A_2,I_1,I_2)} g^{-1}_{A_1,A_2,I_1,I_2}(1)\right| \leq |D|^{(1-\Phi(\beta))n} 3^n 2^{kp}$, to complete the proof.

7 Lower bounds for Boolean branching programs

In this section, we prove Theorem 6. Let f, n, ϵ, θ and Φ be as hypothesized. We will apply Theorem 1 with $k = 2$ and $r = (1+\epsilon)/2$. Fix a $(2, (1+\epsilon)/2)$-decision program $R = (T_1, T_2)$ that is compatible with f. We will prove that $\left|R^{-1}(1)\right| \leq 2^{(1-\alpha/3)n+1}$, where α is defined as in the conclusion of the theorem. Since property $\mathcal{P}(\theta)$ implies that $\left|f^{-1}(1)\right| \geq 2^{(1-\theta)n}$, Theorem 1 yields the required conclusion.

As in the previous proof, we define $S_i(\sigma)$, for input σ and $i = 1, 2$, to be set of variables that are not read by T_i on input σ. We say that $S_i(\sigma)$ is the set of variables *missed* by σ in T_i. Note that $|S_i(\sigma)|$ is always $(1-\epsilon)/2$. For a pair of sets (S_1, S_2) each of size $(1-\epsilon)/2$, define $\mathrm{Miss}(S_1, S_2)$ to be the set of inputs σ such that $S_1(\sigma) = S_1$ and $S_2(\sigma) = S_2$ and $\mathrm{Accept}(S_1, S_2)$ to be the subset of $\mathrm{Miss}(S_1, S_2)$ accepted by R. Then $\left|R^{-1}(1)\right| = \sum_{S_1,S_2} |\mathrm{Accept}(S_1, S_2)|$, and we upper bound $\left|R^{-1}(1)\right|$ by classifying the terms in the sum, and bounding them accordingly.

We first dispense with the cases that follow easily from the properties satisfied by f. Consider those terms $\mathrm{Accept}(S_1, S_2)$ where $|S_1 \cap S_2| > \theta n$. Because f satisfies property $\mathcal{P}(\theta)$, any pair of consistent paths that miss a common set of more than θn variables must be followed by at least one falsifying input of f. Since R is compatible with f, no input following this pair of paths can be accepted since at least one of the two leaf labels of these paths must be 0. Therefore, $\mathrm{Accept}(S_1, S_2)$ is empty.

Thus we may restrict attention to the terms in the sum corresponding to (S_1, S_2) with $|S_1 \cap S_2| \leq \theta n$. For such pairs, we can choose disjoint sets $A_1 \subseteq S_1$ and $A_2 \subseteq S_2$ such that $|A_i| = \delta n$, where $\delta = (1 - \epsilon - \theta)/2$. Because f satisfies property $\mathcal{Q}(\Phi)$, applying this property with respect to A_1 and A_2, we have that $|\mathrm{Accept}(S_1, S_2)| \leq 2^{(1-\Phi(\delta))n}$. We could now naively bound the sum by multiplying by the number of possible pairs (S_1, S_2) with $|S_1 \cap S_2| \leq \theta n$, but the resulting bound is too large to be of any use. Instead we will divide the terms of the sum into two parts depending on the size of $\mathrm{Miss}(S_1, S_2)$. Let $0 \leq \gamma \leq 1$ be a constant to be fixed later. Call a pair (S_1, S_2) *common* if $\mathrm{Miss}(S_1, S_2) \geq 2^{(1-\gamma)n}$, and *rare* otherwise, and denote the sets of common and rare pairs by P_{common} and P_{rare}. Let B_{common} (resp. B_{rare}) be the (disjoint) union of $\mathrm{Accept}(S_1, S_2)$ for all $(S_1, S_2) \in P_{\mathrm{common}}$ (resp. $(S_1, S_2) \in P_{\mathrm{rare}}$). Thus $|R^{-1}(1)| = |B_{\mathrm{common}}| + |B_{\mathrm{rare}}|$; we will upper bound $|B_{\mathrm{common}}|$ and $|B_{\mathrm{rare}}|$ separately.

The number of common pairs (S_1, S_2) is clearly at most $2^{\gamma n}$, and thus $|B_{\mathrm{common}}| \leq 2^{(\gamma+1-\Phi(\delta))n}$.

To bound $|B_{\mathrm{rare}}|$ we will show that the overall number of inputs (accepting or rejecting) that correspond to rare pairs can not be too large. Let A denote the union of $\mathrm{Miss}(S_1, S_2)$ over all rare pairs. Then $B_{\mathrm{rare}} \subseteq A$. We prove:

Lemma 7. *Let $\epsilon, \theta > 0$ with $\epsilon + \theta \leq 1/4$, let $\gamma \in (0, 1)$, and let A be defined as above. Then $\log_2 |A| - n$ is at most*

$$\left(\epsilon + \theta + (1+\epsilon) H\left(\frac{2(\epsilon+\theta)}{1+\epsilon}\right) + 2H(\theta) - \gamma\right)\frac{n}{2}$$

Using this lemma, we complete the proof of Theorem 6. We have $\left|R^{-1}(1)\right| \leq |B_{\mathrm{common}}| + |A|$, and we choose γ so that the above upper bounds for $|B_{\mathrm{common}}|$ and $|A|$ are equal. For this choice of γ, $|A|, |B_{\mathrm{common}}| \leq 2^{(1-\alpha/3)n}$, where α is as defined in the statement of Theorem 6. As noted earlier, Theorem 1 now yields the desired bound.

So it remains to prove Lemma 7, which is the crux of the entire argument. The proof uses elementary information theory. We review the basic definitions and results. Let X be an arbitrary probability space. For any event A, we write $\mathbf{Prob}[A]$ for the probability of A and $\mu[A]$ for $\log_2 \mathbf{Prob}[A]$. If $\mathbf{C}$ is a random variable taking values from a finite set S, the binary entropy $H(\mathbf{C})$ is defined to be $-\sum_{s \in S} \mathbf{Prob}[\mathbf{C} = s]\, \mu[\mathbf{C} = s]$.

Proposition 8. *If $\mathbf{C}$ is a random variable taking on values from some set S then $H(\mathbf{C}) \leq \log_2 |S|$.*

If A is an arbitrary event (measurable subset) of the probability space then the conditional entropy of $\mathbf{C}$ given A is $H(\mathbf{C}|A) = -\sum_{s\in S} \mathbf{Prob}[\mathbf{C} = s|A] \log_2 \mathbf{Prob}[\mathbf{C} = s|A]$.

Proposition 9. *Let $\mathbf{C}$ be some random variable taking values on a finite set S and let A be an event.*

1. $H(\mathbf{C}|A) \le H(\mathbf{C})$.
2. *Let $S_{\mathbf{C}}(A)$ denote the set of $s \in S$ such that $\mathbf{Prob}[\mathbf{C} = s|A] > 0$. Then $H(\mathbf{C}|A) \ge \mu[A] - \max_{s\in S_{\mathbf{C}}(A)} \mu[\mathbf{C} = s]$.*

If $\mathbf{B}_1$ and $\mathbf{B}_2$ are random variables on the same probability space taking values in S_1 and S_2, respectively, $H(\mathbf{B}_1, \mathbf{B}_2)$ is the entropy of the random variable consisting of the pair $(\mathbf{B}_1, \mathbf{B}_2)$. The conditional entropy of $\mathbf{B}_1$ given $\mathbf{B}_2$ is defined by $H(\mathbf{B}_1|\mathbf{B}_2) = H(\mathbf{B}_1, \mathbf{B}_2) - H(\mathbf{B}_2) = \sum_{s\in S_2} H(\mathbf{B}_1|\mathbf{B}_2 = s) \cdot \mathbf{Prob}[\mathbf{B}_2 = s]$. The mutual information of $\mathbf{B}_1$ and $\mathbf{B}_2$ is defined to be $I(\mathbf{B}_1, \mathbf{B}_2) = H(\mathbf{B}_1) + H(\mathbf{B}_2) - H(\mathbf{B}_1, \mathbf{B}_2)$.

Proposition 10. *Let $\mathbf{B}_1, \mathbf{B}_2$ be random variables taking values on finite sets S_1 and S_2. Then*

1. $H(\mathbf{B}_1, \mathbf{B}_2) = H(\mathbf{B}_1|\mathbf{B}_2) + H(\mathbf{B}_2|\mathbf{B}_1) + I(\mathbf{B}_1, \mathbf{B}_2)$.
2. $H(\mathbf{B}_1, \mathbf{B}_2) \le H(\mathbf{B}_1) + H(\mathbf{B}_2)$.
3. $H(\mathbf{B}_1|\mathbf{B}_2) \le \max_{s\in S_2} H(\mathbf{B}_1|\mathbf{B}_2 = s)$.

Given two random variables $\mathbf{B}, \mathbf{C}$ we say that $\mathbf{B}$ *determines* $\mathbf{C}$ if $\mathbf{C}$ is a function of $\mathbf{B}$. We have:

Proposition 11. 1. *If $\mathbf{B}$ and $\mathbf{C}$ are random variables such that $\mathbf{B}$ determines $\mathbf{C}$ then (a) $H(\mathbf{C}) \le H(\mathbf{B})$ and (b) $H(\mathbf{C}|\mathbf{B}) = 0$.*

2. *If $\mathbf{B}_1, \mathbf{B}_2, \mathbf{C}_1, \mathbf{C}_2$ are random variables such that $\mathbf{B}_1$ determines $\mathbf{C}_1$ and $\mathbf{B}_2$ determines $\mathbf{C}_2$ then $I(\mathbf{C}_1, \mathbf{C}_2) \le I(\mathbf{B}_1, \mathbf{B}_2)$.*

Proposition 12. *Let $\mathbf{B}_1, \mathbf{B}_2$ and $\mathbf{C}_1, \mathbf{C}_2$ be pairs of random variables such that $\mathbf{B}_i$ determines $\mathbf{C}_i$ for $i = 1, 2$. Then*

$$\begin{aligned} &H(\mathbf{B}_1, \mathbf{B}_2) + H(\mathbf{C}_1, \mathbf{C}_2) \\ &\quad\le\quad H(\mathbf{B}_1) + H(\mathbf{B}_2) + H(\mathbf{C}_1|\mathbf{C}_2) + H(\mathbf{C}_2|\mathbf{C}_1). \end{aligned}$$

Proof. By Proposition 10(1) and Proposition 11(2):

$$\begin{aligned} &H(\mathbf{C}_1, C_2) \\ &\quad= H(\mathbf{C}_1|\mathbf{C}_2) + H(\mathbf{C}_2|\mathbf{C}_1) + I(\mathbf{C}_2, \mathbf{C}_1) \\ &\quad\le H(\mathbf{C}_1|\mathbf{C}_2) + H(\mathbf{C}_2|\mathbf{C}_1) + I(\mathbf{B}_2, \mathbf{B}_1) \\ &\quad= H(\mathbf{C}_1|\mathbf{C}_2) + H(\mathbf{C}_2|\mathbf{C}_1) \\ &\qquad\qquad + H(\mathbf{B}_1) + H(\mathbf{B}_2) - H(\mathbf{B}_1, \mathbf{B}_2). \end{aligned}$$

□

We will need a well known technical fact concerning sums of binomial coefficients (whose proof we give since we don't know a reference).

Proposition 13. *If $k \le n/2$, $\sum_{i\le k} \binom{n}{i} \le 2^{nH(k/n)}$.*

Proof. By the binomial theorem, for any $p \le 1/2$, we have $1 \ge \sum_{i\le k} p^i(1-p)^{n-i}\binom{n}{i} \ge p^k(1-p)^{n-k}\sum_{i\le k}\binom{n}{i}$. Setting $p = k/n$ and rearranging yields the desired inequality. □

Proof of Lemma 7. Consider the probability space on $\{0,1\}^X$ with the uniform distribution. We define the following random variables. For $i = 1, 2$, let $\mathbf{P}_i$ be the path taken in T_i and $\mathbf{S}_i$ be the set of variables missed in T_i by a random input. Observe that $\mathbf{S}_i$ is a function of $\mathbf{P}_i$.

The basic intuition for the proof is this. We are trying to upper bound the size of A by something like $2^{(1-\lambda)n}$ where $\lambda > 0$ and for this it suffices to upper bound $\mu[A]$ by $-\lambda n$. Note that A is defined to be the event that $(\mathbf{S}_1, \mathbf{S}_2) \in P_{\text{rare}}$, and so by Proposition 9(2), $\mu[A] \le H(\mathbf{S}_1, \mathbf{S}_2|A) + \max_{(S_1,S_2)\in P_{\text{rare}}} \mu[(\mathbf{S}_1, \mathbf{S}_2) = (S_1, S_2)]$. The second term is at most $-\gamma n$ by the definition of P_{rare}. Thus we want to upper bound $H(\mathbf{S}_1, \mathbf{S}_2|A) = H(\mathbf{S}_1|\mathbf{S}_2, A) + H(\mathbf{S}_2|\mathbf{S}_1, A) + I(\mathbf{S}_1, \mathbf{S}_2|A)$ by $\gamma' n$, where γ' is a constant less than γ. Now observe that given A, $|\mathbf{S}_1 \cap \mathbf{S}_2|$ is small, and therefore $\mathbf{S}_1$ and $\mathbf{S}_2$ are "approximately" complementary subsets of variables, and so one of them approximately determines the other. This allows us to conclude that the first two terms in the sum are small. We use Proposition 11 to upper bound the third term by $I(\mathbf{P}_1, \mathbf{P}_2|A)$. Intuitively, this represents the amount of information $\mathbf{P}_1$ reveals about $\mathbf{P}_2$ (and vice versa) given that A holds. Now A implies that $\mathbf{P}_1$ and $\mathbf{P}_2$ read very few variables in common, and this can be used to show that the mutual information is small. Although the intuition is based on mutual information, in the formal argument we use Proposition 12 and do not explicitly mention mutual information.

We now proceed with the proof by considering $H(\mathbf{P}_1, \mathbf{P}_2|A) + H(\mathbf{S}_1, \mathbf{S}_2|A)$. As noted Proposition 9(2) implies $H(\mathbf{S}_1, \mathbf{S}_2|A) \ge \mu[A] + \gamma n$. To apply the same proposition to $H(\mathbf{P}_1, \mathbf{P}_2|A)$ we note that any pair of paths (P_1, P_2) corresponding to a point in $\sigma \in A$ contains at least $(1-\theta)n$ variables and so $\mu[(\mathbf{P}_1, \mathbf{P}_2) = (P_1, P_2)] \le -(1-\theta)n$. Consequently $H(\mathbf{P}_1, \mathbf{P}_2|A) \ge \mu[A] + (1-\theta)n$ and so

$$H(\mathbf{P}_1, \mathbf{P}_2|A) + H(\mathbf{S}_1, \mathbf{S}_2|A) \ge 2\mu[A] + (1 - \theta + \gamma)n.$$

On the other hand, by Propositions 12 and 9(1),

$$\begin{aligned} &H(\mathbf{P}_1, \mathbf{P}_2|A) + H(\mathbf{S}_1, \mathbf{S}_2|A) \\ &\le H(\mathbf{P}_1|A) + H(\mathbf{P}_2|A) + H(\mathbf{S}_1|\mathbf{S}_2, A) + H(\mathbf{S}_2|\mathbf{S}_1, A) \\ &\le H(\mathbf{P}_1) + H(\mathbf{P}_2) + H(\mathbf{S}_1|\mathbf{S}_2, A) + H(\mathbf{S}_2|\mathbf{S}_1, A) \\ &\le (1+\epsilon)n + H(\mathbf{S}_1|\mathbf{S}_2, A) + H(\mathbf{S}_2|\mathbf{S}_1, A) \end{aligned}$$

where the last inequality comes from Proposition 8 and the fact that the number of paths in each tree is $2^{(1+\epsilon)n/2}$.

Since the two remaining terms are symmetric it remains to upper bound $H(\mathbf{S}_1|\mathbf{S}_2, A)$. Define the random variables $\mathbf{I} = \mathbf{S}_1 \cap \mathbf{S}_2$ and $\mathbf{J} = X - (\mathbf{S}_1 \cup \mathbf{S}_2)$ and observe that the triple $(\mathbf{S}_2, \mathbf{I}, \mathbf{J})$ determines $\mathbf{S}_1$. Hence:

$$\begin{aligned} H(\mathbf{S}_1|\mathbf{S}_2, A) &\le H(\mathbf{S}_2, \mathbf{I}, \mathbf{J}|\mathbf{S}_2, A) \\ &\le H(\mathbf{S}_2|\mathbf{S}_2, A) + H(\mathbf{I}|\mathbf{S}_2, A) + H(\mathbf{J}|\mathbf{S}_2, A) \\ &= 0 + H(\mathbf{I}|\mathbf{S}_2, A) + H(\mathbf{J}|\mathbf{S}_2, A) \\ &\le H(\mathbf{I}|A) + H(\mathbf{J}|\mathbf{S}_2, A) \end{aligned}$$

Since A implies $|\mathbf{I}| \le \theta n$, Proposition 8 and Proposition 13 imply that $H(\mathbf{I}|A) \le \sum_{i \le \theta n} \binom{n}{i} \le nH(\theta)$. Now, by Proposition 10(3), $H(\mathbf{J}|\mathbf{S}_2, A) \le \max_{S_2} H(\mathbf{J}|\mathbf{S}_2 = S_2, A)$. Given A, $|\mathbf{J}| \le (\theta + \epsilon)n$ and given $\mathbf{S}_2 = S_2$, $\mathbf{J}$ is contained in $\overline{S_2}$ which is a set of size $(1 + \epsilon)n/2$. Again, using Proposition 8 and Proposition 13 we conclude $H(\mathbf{J}|\mathbf{S}_2, A) \le \frac{1+\epsilon}{2} H\left(\frac{2(\epsilon+\theta)}{1+\epsilon}\right)$ (here we use the hypothesis that $\epsilon + \theta \le 1/4$). Thus:

$$H(\mathbf{S}_1|\mathbf{S}_2, A) \le \left[\frac{1+\epsilon}{2} H\Big(\frac{2(\epsilon+\theta)}{1+\epsilon}\Big) + H(\theta)\right] n$$

and so

$$\begin{aligned} &2\mu[A] + (1 - \theta + \gamma)n \\ &\le H(\mathbf{P}_1, \mathbf{P}_2|A) + H(\mathbf{S}_1, \mathbf{S}_2|A) \\ &\le (1+\epsilon)n + 2\left[\frac{1+\epsilon}{2} H\left(\frac{2(\epsilon+\theta)}{1+\epsilon}\right) + H(\theta)\right] n. \end{aligned}$$

Solving, we find

$$\mu[A] \le \left[\epsilon + \theta + (1+\epsilon) H\left(\frac{2(\epsilon+\theta)}{1+\epsilon}\right) + 2H(\theta) - \gamma\right] \frac{n}{2}$$

which is what we wanted to prove. □

8 Proof of Lemma 4

We being with the proof of the first part of the lemma. This will follow immediately from two additional lemmas.

For any $n \times n$ matrix M and any $0 \le \delta \le 1$, define $\Phi_M(\delta)$ to be $1/n$ times the minimum rank of any $\delta n \times \delta n$ minor of M that does not include any diagonal element of M. Trivially, $\Phi_M(\delta) = \Phi_N(\delta)$ if M and N have the same off-diagonal elements; in particular $\Phi_M(\delta) = \Phi_{M^{[0]}}(\delta)$.

Lemma 14. *Let M be a matrix over $GF(q)$ and $D \subseteq GF(q)$. The restriction of QF_M to D^n satisfies $\mathcal{Q}(\Phi)$ for $\Phi = \Phi_M$.*

Let G be a finite group. A *G-matrix* over $GF(q)$ is a matrix N whose rows are indexed by G, such that for all $g_1, g_2 \in G$ and $j \in [1, n]$, $N_{g_1 \cdot g_2, j} = N_{g_1, j} N_{g_2, j}$ (i.e., each column is a multiplicative character of G).

Lemma 15. *Let N be an $n \times n$ G-matrix of full rank, where G is a group of order n, and let M be any $n \times n$ matrix that agrees with N off the diagonal. Then $\Phi_M(\delta) = \delta^2$.*

Since a GFT matrix associated to a group G is a G-matrix of full rank, Part 1 of the Theorem follows immediately from the lemmas. So we now prove them.

Proof of Lemma 14. Let $g = g_1 \wedge g_2$ be compatible with QF_M such that for some disjoint sets of variables A_1 and A_2 of size each δn, g_i does not depend on the variables in A_i. Consider an arbitrary assignment ρ of the variables outside of $A_1 \cup A_2$. There are $|D|^{2\delta n}$ assignments that extend ρ; let Γ denote the subset of these assignments that are accepted by g. We will show that $|\Gamma| \le |D|^{(2\delta - \Phi_M(\delta))n}$; the desired bound is then obtained by summing over the $|D|^{(1-2\delta)n}$ choices of ρ.

Since g_i does not examine any variables in A_i, $\Gamma = \{\rho\} \times \Sigma_1 \times \Sigma_2$, where

$$\begin{aligned} \Sigma_1 &= \{\sigma_1 : (\rho, \sigma_1, \sigma_2) \in \Gamma \text{ for some } \sigma_2\} \\ \Sigma_2 &= \{\sigma_2 : (\rho, \sigma_1, \sigma_2) \in \Gamma \text{ for some } \sigma_1\} \end{aligned}$$

Substitute the values assigned by ρ into $X^T M X$. Because M is symmetric, we obtain a polynomial of the form $A_1^T N A_2 + F_1(A_1) + F_2(A_2)$, where N is a $\delta n \times \delta n$ matrix equal to twice the minor of M indexed by $A_1 \times A_2$ and each F_i is some polynomial function of the entries of A_i. Lemma 16 below, which is a slight generalization of results in [10, 18], implies the required bound on $|\Gamma|$.

Lemma 16. *Let N be a $t \times t$ matrix over $GF(q)$. For $i = 1, 2$, let $F_i : GF(q)^t \to GF(q)$ be arbitrary functions, and let F denote the function from $GF(q)^t \times GF(q)^t$ to $GF(q)$ given by $F(\sigma_1, \sigma_2) = \sigma_1 N \sigma_2 + F_1(\sigma_1) + F_2(\sigma_2)$. For a set $D \subseteq GF(q)$, suppose that $\Sigma_1, \Sigma_2 \subseteq D^t$ satisfy $F(\sigma_1, \sigma_2) = 0$ for every $(\sigma_1, \sigma_2) \in \Sigma_1 \times \Sigma_2$. Then $|\Sigma_1 \times \Sigma_2| \le |D|^{2t - \mathrm{rank}(N)}$.*

Proof. For $i = 1, 2$ fix some $\sigma_i^* \in \Sigma_i$. Then for any $(\sigma_1, \sigma_2) \in \Sigma_1 \times \Sigma_2$, we have $(\sigma_1 - \sigma_1^*) N (\sigma_2 - \sigma_2^*) = F(\sigma_1, \sigma_2) + F(\sigma_1^*, \sigma_2^*) - F(\sigma_1, \sigma_2^*) - F(\sigma_1^*, \sigma_2) = 0$. Defining V_i for $i = 1, 2$ to be the linear span of $\Sigma_i^* = \{\sigma - \sigma_i^* : \sigma \in \Sigma_i\}$ we have that $v_1 N v_2 = 0$ for all $v_1 \in V_1$ and $v_2 \in V_2$. This implies $\dim(V_1) + \dim(V_2) \le 2t - \mathrm{rank}(N)$. The lemma now follows since $\left|\tilde{\Sigma}\right| = \left|\tilde{\Sigma}_i\right| \le |D|^{\dim(V_i)}$. □

This completes the proof of Lemma 14. □

We now turn to the proof of Lemma 15. This lemma is an immediate consequence of the following lemma, which says, roughly, that every large minor of a GFT matrix has large rank. The lemma both simplifies and improves a bound in [10] which showed that every $u \times t$ minor of such a matrix has rank at least $ut/(\eta(n, u, t)\, n)$, where $\eta(n, u, t)$

is a function that is typically logarithmic in n. (This new bound also improves the lower bound on the size of read-k branching programs proved in that paper by shaving off a factor of k in the exponent.)

Lemma 17. *Let N be any G-matrix over $GF(q)$ where G is a group of order n. If N has full column rank, the rank of any $u \times t$ minor of N is at least ut/n.*

Proof. For $V \subseteq G$ and any set J of columns, let $N_{V,J}$ denote the submatrix of N corresponding to the rows of V and columns of J. Fix V of size u and J of size t. Since N has full column rank, the columns of $N_{G,J}$ are independent implying that it has a $t \times t$ minor $N_{W,J}$, for some $W \subseteq G$, which has full rank.

For any fixed $g^* \in V$ and a random $g \in G$, $\Pr[gg^* \in W] = \Pr[g \in W(g^*)^{-1}] = t/n$. By linearity of expectation, when g is randomly chosen from G, the expected number of $g^* \in V$ for which $gg^* \in W$ is ut/n. Therefore, for some fixed g and some $H \subseteq V$ where $|H| \geq ut/n$, we have $gH \subseteq W$. We show that $N_{H,J}$ has rank at least ut/n from which the lemma follows.

Because G is a group, $|gH| = |H| \geq ut/n$. Since $N_{gH,J}$ is a submatrix consisting of at least ut/n rows of $N_{W,J}$, it follows that $\mathrm{rank}(N_{gH,J}) \geq ut/n$. By the definition of N, for each $j \in J$, $N_{gH,j} = N_{g,j}N_{H,j}$, that is, each column of $N_{H,W}$ is multiplied by some constant to get the corresponding column in $N_{gH,W}$. Therefore $\mathrm{rank}(N_{H,J}) \geq \mathrm{rank}(N_{gH,W}) \geq ut/n$. □

This completes the proof of part 1 of Lemma 4. We now consider part 2. Let M be a GFT matrix. It suffices to show that for any partial assignment ρ that fixes all but 2 variables, z_1, z_2 of X, the restriction $QF_{M^{[0]}}\lceil_\rho$ is not the constant function. Since M is symmetric, and its off-diagonal elements are non-zero, the restriction satisfies $QF_{M^{[0]}}\lceil_\rho = 2a(z_1+b)(z_2+c)+d$ for some constants $a \neq 0$, $b, c, d \in GF(q)$. Since q is odd, setting z_1 to any value in $GF(q) - \{-b\}$, we have $2a(z_1+b) \neq 0$ and it follows that $2a(z_1+b)(z_2+c)+d$ takes on all possible values in $GF(q)$ by varying z_2. Thus, $QF_{M^{[0]}}\lceil_\rho$ is non-constant.

Finally, to prove Part 3, let M be the $n \times n$ Sylvester matrix over $GF(3)$. Notice that the above argument fails in this case, even for $M^{[0]}$, because the values of z_1, z_2 are restricted to be from $\{0, 1\}$. To prove property $\mathcal{P}(\theta)$ in this case we first need a lemma showing that in every sufficiently large principal minor of M, there exist principal minors whose entries sum to arbitrary values.

Lemma 18. *Let M be the $n \times n$ Sylvester matrix over $GF(3)$ where $n = 2^k$. Let $I \subseteq [1, n]$ be an arbitrary subset of size at least $4 \log n/\sqrt{n}$. For every $a \in GF(3)$, there exists $J \subseteq I$, with $|J| \leq 3$ such that the sum of the entries in $M_{J,J}$ is a.*

Proof. For $a = 0$, set $J = \emptyset$. So assume $a \in \{1, -1\}$. Recall that the row and column index set of the Sylvester matrix is the set of binary vectors of length k, which is identified naturally with the set of subsets of $[1, k]$ and we view I as a collection of $4 \log n/\sqrt{n}$ such subsets. For $a = 1$ and for $a = -1$, we want a subcollection J of I such that the sum of entries in $M_{J,J}$ is a. The following easily verifiable fact gives a criterion for a collection of size 3 to satisfy this.

Proposition 19. *Suppose A_0, B_1, B_2 are distinct subsets of $[1, k]$ such that $|A_0|$, $|B_1|$, and $|B_2|$ are all even or all odd and let $J = \{A_0, B_1, B_2\}$.*

1. *If $|A_0 \cap B_1|$ and $|A_0 \cap B_2|$ are both even and $|B_1 \cap B_2|$ is odd then the sum of entries in $M_{J,J}$ is -1*
2. *If $|A_0 \cap B_1|$ and $|A_0 \cap B_2|$ are both odd and $|B_1 \cap B_2|$ is even then the sum of entries in $M_{J,J}$ is +1.*

We will also need the so-called "Eventown-Oddtown" theorems (see [4]), stated as a proposition below:

Proposition 20. *Let $\mathcal{F}$ be a family of sets. We say that $\mathcal{F}$ has property $P_{i,e}$ (respectively $P_{i,o}$) if the common intersection of every collection of i distinct sets from $\mathcal{F}$ is even (respectively odd). The following table gives an upper bound on the size of any family of subsets of $[1, k]$ satisfying some of these properties:*

	$P_{1,e}$	$P_{1,o}$
$P_{2,e}$	$2^{k/2}$	k
$P_{2,o}$	k	$2^{k/2}$

Continuing the proof of Lemma 18, we now show that I contains a collection $J = \{A_0, B_1, B_2\}$ satisfying the hypothesis of the first part of Proposition 19 and thus the sum of entries in $M_{J,J}$ is -1. A similar argument handles the other case.

Since $|I| \geq 4\sqrt{n} \log n$, there is a sub-family $\mathcal{F}$ of size at least $2\sqrt{n} \log n$ such that every set in $\mathcal{F}$ has even size or every set in $\mathcal{F}$ has odd size. Consider a sub-family $\mathcal{G}$ of $\mathcal{F}$ which is maximal subject to the condition that for any distinct $A, B \in \mathcal{G}$, $|A \cap B|$ is odd. For each set $A \in \mathcal{G}$, let $\mathcal{E}(A)$ be the subfamily consisting of those sets $C \in \mathcal{F} - \mathcal{G}$ such that $|A \cap C|$ is even. By the maximality of $\mathcal{G}$, $\cup_{A \in \mathcal{G}} \mathcal{E}(A) = \mathcal{F} - \mathcal{G}$ which implies $\sum_{A \in \mathcal{G}} |\mathcal{E}(A)| \geq |\mathcal{F} - \mathcal{G}|$. Choose $A_0 \in \mathcal{G}$ for which $|\mathcal{E}(A_0)|$ is maximum and write $\mathcal{E} = \mathcal{E}(A_0)$. Then $|\mathcal{E}|\,|\mathcal{G}| \geq |\mathcal{F} - \mathcal{G}|$, or $(|\mathcal{E}| + 1)\,|\mathcal{G}| \geq |\mathcal{F}|$. To finish the proof it suffices to show that there are $B_1, B_2 \in \mathcal{E}$ whose intersection has odd size. Using Proposition 20, if every pair of distinct sets in $\mathcal{E}$ has an intersection of even size, then

$$\begin{aligned}|\mathcal{G}|\,(|\mathcal{E}| + 1) &\leq \max\{2^{k/2}(k+1), k(2^{k/2}+1)\} \\ &< 2k2^{k/2} = 2\sqrt{n}\log n \leq |\mathcal{F}|\end{aligned}$$

which contradicts $(|\mathcal{E}|+1)\,|\mathcal{G}| \geq |\mathcal{F}|$. Therefore, the claim holds. □

We now have the tools to prove Part 3 of Lemma 4. Let ρ be a partial assignment to $(1-\theta)n$ variables of X, where $\theta = 24\log n/\sqrt{n}$ and $Z \subseteq X$ be the variables unset by ρ. Then $BQF_M\lfloor_\rho = Z^T B Z + A \cdot Z + C$, where B denotes the sub-matrix of M corresponding to the rows and columns corresponding to Z, and A and C are fixed constants determined by ρ and M. It suffices to show that $q(Z) = Z^T B Z + A \cdot Z$ takes on all possible values in $GF(3)$ for the various choices of 0-1 assignments to Z.

Setting all variables to 0 makes the function 0. Fix $a \in \{-1, 1\}$. Our goal is to identify three variables such that setting them to 1 and everything else to 0 will make the function equal to a. Classify each variable x_j by the pair $(M_{j,j}, A_j) \in \{-1,1\} \times \{-1,0,1\}$. There are 6 possible values of this pair, and so there is a set of at least $4\sqrt{n}\log n$ variables $Z' \subseteq Z$ that belong to the same class. If we set any three variables in Z' to 1, and everything else to 0, $q(Z)$ evaluates to the sum of the off-diagonal entries in the 3×3 principal minor corresponding to these variables. By Lemma 18 such a minor exists whose sum evaluates to $a \pmod 3$, and Part 3 of the lemma follows.

9 Semantic versus syntactic branching programs

A path P in a branching program is a *semantic path* if it is consistent with some input. A path is not semantic if and only if there is some pair of nodes v and w on the path that have the same variable label x, such that the arcs following them have different labels. A path P is *read-k* for some integer k, if no variable appears in P more than k times. If Z is a subset of the variables of a branching program B, we say that B is *syntactic (semantic) read-k on Z* if every path (respectively every semantic path) in B is read-k. (If Z contains all the variables, we omit the qualifying phrase in which case our definition is the standard one.)

In this section we exhibit, for every k, a simple function f_k that can be computed in linear size by a semantic read-twice branching program but requires an exponential size syntactic read-k branching program. The key to defining our separating functions is the construction of functions $g_k(X,Y)$ that can be computed by linear size branching programs that are semantic read-twice on X but require exponential size on any branching program that is syntactic read-k on X. Before we give the construction of $g_k(X,Y)$, we describe the relationship between $g_k(X,Y)$ and the separating functions.

DEFINITION 21. *For a variable set Y, let $Y_1, Y_2, \ldots, Y_k$ be disjoint copies of Y. Write $y \cong y'$ if for some i, j, $y \in Y_i$ and $y' \in Y_j$ both correspond to the same variable in Y. The k^{th} extension of g on Y, denoted by $g^{(k)}(X, Y_1, \ldots, Y_k)$, is defined to be one if and only if (i) $g(X,Y_1) = 1$ and (ii) all of the blocks $Y_1, Y_2, \ldots, Y_k$ have the same setting.*

The relationship between computing $g(X,Y)$ and its k^{th} extension on Y is given by the following lemma:

Lemma 22. *Let $g(X,Y)$ be a Boolean function.*

1. *If $g(X,Y)$ can be computed by branching program P that is syntactic read-k on Y, then $g^{(k)}(X, Y_1, \ldots, Y_k)$ can be computed by a branching program Q that is syntactic read-twice on $\bigcup_i Y_i$ and satisfies $\text{size}(Q) = \text{size}(P) + O(\sum_i |Y_i|)$. Furthermore, any syntactic or semantic properties of P with respect to X also hold in Q.*

2. *If $g^{(k)}(X, Y_1, \ldots, Y_k)$ can be computed by a syntactic read-k branching program, then $g(X,Y)$ can be computed by a branching program that is syntactic read-k on X and has the same size.*

Suppose that $g(X,Y)$ can be computed efficiently by a branching program that is syntactic read-k on Y and semantic read-once on X, but requires exponential size to be computed by any branching program that is syntactic read-k on X. Lemma 22 implies that the $g^{(k)}(X, Y_1, \ldots, Y_k)$ can be computed efficiently by a semantic read-twice branching program but requires an exponential size syntactic read-k branching program. Thus $g^{(k)}(X, Y_1, \ldots, Y_k)$ witnesses the desired separation. For the rest of this section, we focus on producing such a $g(X,Y)$.

DEFINITION 23. *For a k-dimensional hypercube $[1,n]^k$ of side n, the n hyperplanes perpendicular to the d^{th} axis, $d \in [0, k-1]$ are referred to as d-planes. In other words, the i^{th} d-plane, for $i \in [1,n]$, is the set $\{v \in [1,n]^k : v_d = i\}$. We define the predicate $g_k(X,Y)$ as follows. Without loss of generality, let $k+1 = 2^r$, for some r. Let X and Y be sets of variables corresponding to a $(k+1)$-dimensional hypercube of side n. The variables of X are Boolean but we use the Fourier representation where -1 and 1 are identified with* **true** *and* **false** *respectively and also treated as elements of $GF(3)$. For each $v \in [1,n]^{k+1}$, Y contains r variables $y_v^0, y_v^1, \ldots, y_v^{r-1}$ which together determine an integer $y_v = y_v^{r-1} \ldots y_v^1 y_v^0 \in [0,k]$. For any $d \in [0,k]$, $v \in [1,n]^{k+1}$, define x_v^d to be x_v if $y_v = d$ and 1 otherwise. Define the following polynomial over $GF(3)$:*

$$H_d(X,Y) = \sum_{i \in [1,n]} \prod_{\substack{v \in [1,n]^{k+1}, \\ v_d = i}} x_v^d$$

We define the predicate $g_k(X,Y)$ to be true if $H_d(X,Y) \equiv 0 \pmod 3$ for all $d \in [0,k]$.

Informally, we can describe $g_k(X,Y)$ as follows. The variables of X and Y can each be viewed as defining two $(k+1)$-dimensional arrays, denoted x and y respectively, where the entries of x are in $\{-1,1\}$ and those in y are in $[0,k]$. We construct $k+1$ additional arrays x^d whose entry in position v is x_v if $y_v = d$ and is 1 otherwise. We say that x_v is *active in* x^d if $y_d = 1$, and let $X^d \subseteq X$ be the set of such variables. Note that the sets X^d partition X. The functions H_d are computed by considering each d-plane of x^d and summing the product of the entries and summing the products. This can be done by a branching program of size $O(|X^d| + |Y|)$ that reads each variable of $Y \cup X^d$ exactly once and no other entry of X. Thus $g_k(X,Y)$ can be computed by a branching program of size $O(|X| + (k+1)|Y|)$ that is read-$(k+1)$ on Y and read-once on X. On the other hand, we have the following hardness result.

Theorem 24. *Any non-deterministic branching program that is syntactic read-k on X requires exponential size to compute $g_k(X,Y)$.*

As noted above, Lemma 22 then implies:

Corollary 25. *Let $f_k = g_k^{(k+1)}$ be the $(k+1)$-st extension of g_k on Y. There is a simple semantic read-twice branching program of linear size computing f_k but any non-deterministic syntactic read-k branching program for f_k requires exponential size.*

The proof of Theorem 24 relies heavily on machinery developed in [10, 18], particularly the notion of planar pseudo-rectangles and the ideas for proving lower bounds for the closely related functions in [18]. We refer the reader to the fuller version of our paper [7] for details.

References

[1] K. R. Abrahamson. A time-space tradeoff for Boolean matrix multiplication. In *Proceedings 31st Annual Symposium on Foundations of Computer Science*, pages 412–419, St. Louis, MO, Oct. 1990. IEEE.

[2] K. R. Abrahamson. Time–space tradeoffs for algebraic problems on general sequential models. *Journal of Computer and System Sciences*, 43(2):269–289, Oct. 1991.

[3] N. Alon and W. Maass. Meanders and their applications in lower bounds arguments. *Journal of Computer and System Sciences*, 37:118–129, 1988.

[4] L. Babai and P. Frankl. *Linear Algebra Methods in Combinatorics with Applications to Geometry and Computer Science (Preliminary Version 2)*. University of Chicago, 1992.

[5] L. Babai, N. Nisan, and M. Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *Journal of Computer and System Sciences*, 45(2):204–232, Oct. 1992.

[6] P. W. Beame. A general time-space tradeoff for finding unique elements. *SIAM Journal on Computing*, 20(2):270–277, 1991.

[7] P. W. Beame, M. Saks, and J. S. Thathachar. Time-space tradeoffs for branching programs. Technical Report TR98-053, Electronic Colloquium in Computation Complexity, `http://www.eccc.uni-trier.de/eccc/`, 1998.

[8] A. Borodin and S. A. Cook. A time-space tradeoff for sorting on a general sequential model of computation. *SIAM Journal on Computing*, 11(2):287–297, May 1982.

[9] A. Borodin, F. E. Fich, F. Meyer auf der Heide, E. Upfal, and A. Wigderson. A time-space tradeoff for element distinctness. *SIAM Journal on Computing*, 16(1):97–99, Feb. 1987.

[10] A. Borodin, A. A. Razborov, and R. Smolensky. On lower bounds for read-k times branching programs. *Computational Complexity*, 3:1–18, Oct. 1993.

[11] R. E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, August 1986.

[12] J. Burch, E. Clarke, D. Long, K. MacMillan, and D. Dill. Symbolic model checking for sequential circuit verification. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13(4):401–424, April 1994.

[13] L. Fortnow. Nondeterministic polynomial time versus nondeterministic logarithmic space: Time-space tradeoffs for satisfiability. In *Proceedings, Twelfth Annual IEEE Conference on Computational Complexity*, pages 52–60, Ulm, Germany, 24–27 June 1997. IEEE Computer Society Press.

[14] Y. Mansour, N. Nisan, and P. Tiwari. The computational complexity of universal hashing. *Theoretical Computer Science*, 107:121–133, 1993.

[15] N. J. Pippenger. On simultaneous resource bounds. In *20th Annual Symposium on Foundations of Computer Science*, pages 307–311, San Juan, Puerto Rico, Oct. 1979. IEEE.

[16] A. A. Razborov. Lower bounds for deterministic and nondeterministic branching programs. In L. Budach, editor, *Fundamentals of Computation Theory: 8th International Conference, FCT '91*, volume 529 of *Lecture Notes in Computer Science*, pages 47–60, Gosen, Germany, Sept. 1991. Springer-Verlag.

[17] J. Simon and M. Szegedy. A new lower bound theorem for read only once branching programs and its applications. In *Advances in Computational Complexity (J. Cai, editor)*, volume 13 of *DIMACS Series in Discrete Mathematics*, pages 183–193. AMS, 1993.

[18] J. S. Thathachar. On separating the read-k-times branching program hierarchy. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, Dallas, TX, May 1998. To appear.

[19] I. Wegener. *The Complexity of Boolean Functions*. B.G. Teubner, Stuttgart, 1 edition, 1987.

[20] A. C. Yao. Near-optimal time-space tradeoff for element distinctness. In *29th Annual Symposium on Foundations of Computer Science*, pages 91–97, White Plains, NY, Oct. 1988. IEEE.