On Span Programs

M. Karchmer*

Department of Mathematics
Massachusetts Inst. of Technology
Cambridge, MA 02138

A. Wigderson

Department of Computer Science
Hebrew University
Jerusalem, Israel 91904

Abstract

We introduce a linear algebraic model of computation, the Span Program, and prove several upper and lower bounds on it. These results yield the following applications in complexity and cryptography:

- $\mathcal{SL} \subseteq \oplus\mathcal{L}$ *(a weak Logspace analogue of* $\mathcal{NP} \subseteq \oplus\mathcal{P}$*).*
- *The first super-linear size lower bounds on branching programs that count.*
- *A broader class of functions which posses information-theoretic secret sharing schemes.*

The proof of the main connection, between span programs and counting branching programs, uses a variant of Razborov's general approximation method.

1 Introduction

Giving a computational model the power to count is an old and fruitful theme in complexity theory. One such direction was to add mod m gates to unbounded fan-in circuits. This resulted in the exponential lower bounds of Razborov and Smolensky [15, 21] in the case when m is a prime, and the frustrating question of the power of $\mathcal{ACC}$, when m is composite.

Another direction was to let nondeterministic polynomial time Turing machines count the number of accepting paths. For counting mod 2, this defines ([12, 6]) the class $\oplus\mathcal{P}$. Valiant and Vazirani [23] were first to show the power of this model by giving a probabilistic Turing reduction of $\mathcal{NP}$ to $\oplus\mathcal{P}$. Toda [22] used this technique to prove a much stronger result, namely that the whole polynomial time hierarchy is probabilisticly Turing reducible to $\oplus\mathcal{P}$. Moreover, the same result can be obtained via the techniques used for the constant-depth circuits mentioned above, as shown by Allender [2].

Here we are interested in nondeterministic logspace machines that count the number of accepting paths (mod m). The analogues $\text{mod}_m\mathcal{L}$ of the polynomial counting classes $\text{mod}_m\mathcal{P}$ were defined and first studied in [5]. In [5] it was shown that most natural problems in linear algebra over $GF(p)$, such as determinant, rank and solving linear systems, are logspace complete for the class $\text{mod}_p\mathcal{L}$. Still, very little is known about the power of counting in logspace. For example, unlike in the polynomial time case, no relationship between $\mathcal{NL}$ and $\oplus\mathcal{L}$ is known. Moreover, no nontrivial lower bounds were known on the counting branching programs which capture these counting classes.

This paper makes progress on both of these questions. We show that the symmetric nondeterministic class $\mathcal{SL}$ is contained in $\text{mod}_p\mathcal{L}$ for every prime p ($\mathcal{SL}$, symmetric logspace, is the class of all problems that are reducible in logspace to undirected *st*-connectivity). Previously, it was only known that $\mathcal{SL} \subseteq \mathcal{L}/\text{poly}$ which follows from the results of [1].

We also prove the first nontrivial lower bounds on branching programs that count mod 2. The most interesting (though not the largest) is the slightly super-linear ($\Omega(n \log\log\log^* n)$) lower bound on the size of such programs that compute the Majority function. In fact, the proof characterizes those threshold functions that admit linear size programs. The same lower bound for Majority on nondeterministic branching programs was proved by Razborov [18], and indeed we use much of his machinery. In contrast, for the weaker deterministic branching programs, the best lower bound for Majority [3] is $\Omega(n \log n / \log\log n)$.

The route to both types of results goes through the same device – the *span program*. The span program (over any field K) is a linear algebraic model that computes a function f on n variables as follows. Fix a vector space W over K, a nonzero vector $w \in W$

*Partially supported by NSF grant CCR-9212184 and DARPA contract N00014-92-J-1799.

and let X_i^ϵ (with $1 \leq n$, $\epsilon \in \{0,1\}$) be $2n$ subspaces of W that correspond to the $2n$ literals of f. Any truth assignment σ to the variables makes exactly n literals 'true'. We demand that the n associated subspaces span the fixed vector w iff $f(\sigma) = 1$. The size measure for this model is the sum of dimensions of the $2n$ subspaces.

It is quite simple to see that span program size is a lower bound on the size of symmetric branching programs. The important connection we prove is that span program size is a lower bound on counting branching programs. We then prove that span programs for Majority require super-linear size, implying the afore mentioned lower bounds. We use linear algebra, and duality in particular, to develop a notion of *canonical* span programs. These are as strong as the general model, but for which lower bounds are easier to obtain. The canonical model is also useful in establishing that span program size is a natural complexity measure for Boolean functions, in that it cannot increase when applying restrictions. Note that this is not obvious from the definition.

We also study the monotone version of span programs, in which we assign subspaces only to the n positive literals. This model is interesting for several reasons. First, note that it computes only monotone functions, but that computation itself entails non-monotone operations, namely linear algebra over finite fields. Second, we obtained tight bounds for the size of monotone span programs for threshold functions, and discovered that in this model (unlike any other) all these functions (other than AND, OR) are equivalent: Majority, Threshold-2 and all the rest require size exactly $\Theta(n \log n)$. Third, we show that this model captures in a natural way information theoretic secret sharing in the sense of Shamir [20]. It enables us to extend the result of Rudich [19], and enlarge the class of functions for which such efficient secret sharing schemes exists. In the other direction, existing schemes can provide upper bounds for span programs, and indeed the $O(n \log n)$ upper bound for Majority was inspired by Shamir's construction.

Finally we describe the evolution of the idea to use span programs for lower bounds. It was inspired by the papers [18] and [9], both of which have as a common ancestor the paper [16]. In [16], Razborov introduced his generalized approximation method. He showed how to assign to every Boolean function f a set cover problem $(\Delta_M(f), \mathcal{S}_M(f))$. Here $\Delta_M(f)$ is the universe to be covered, and $\mathcal{S}_M(f)$ is a family of its subsets from which a cover should be constructed. The subscript M refers to the fact that this universe is (essentially) all *Monotone functionals* on the the zero set of f. He proved that the minimum cover number, $\delta_M(f)$, is a lower bound on the Boolean circuit size for f. Moreover, this number is tight up to polynomial factor, and can thus be used to characterize $\mathcal{P}$. While no super-linear circuit size was proved yet by this (or any other) method, Razborov successfully applied this general approximation method to branching programs. In [18] he defined a cover problem $(\Delta_M(f), \mathcal{S}\prime_M(f))$, with the same universe but with a smaller family of subsets. He showed that the cover number $\delta\prime_M(f)$ exactly equals the size of nondeterministic branching programs for f (and thus characterizes $\mathcal{NL}$). Finally he was able to prove a super-linear lower bound on $\delta\prime_M(Majority)$ which implied the lower bound mentioned above.

In [9] we proposed a variant to the approximation method in which the universe to be covered is the set of all *Linear functionals* on the set of zeros of f. We proved that the associated cover number $\delta_L(f)$ of the cover problem $(\Delta_L(f), \mathcal{S}_L(f))$, lower bounds *non- deterministic* circuit size, and can be used to characterize $\mathcal{NP}$. By combining the ideas in [18] and [9], we got a restricted cover number that lower bounds the size of Counting Branching Programs. Moreover, the restricted cover problem $(\Delta_L(f), \mathcal{S}\prime_L(f))$ simplifies, after linear algebra manipulations, to our primary model: the Span Program.

2 Background

We define all models nonuniformly. This makes the lower bounds stronger. On the other hand, all upper bounds will be easily seen to be logspace-uniform. When using asymptotic notation we think as usual of a family of functions parameterized by n.

Definition 1 *A* Branching Program *is a directed acyclic labeled graph $G(V, E, \mu)$ with two specified nodes $s, t \in V$ and a labeling $\mu : E \to \{\, x_i^\epsilon \mid i \in [n],\ \epsilon = 0, 1\} \cup \{1\}$ (where we use $x^1 = x$ and $x^0 = \bar{x}$). The* size *of G, $s(G)$, is defined as the number of edges not labeled* 1. *We say that a Branching Program is deterministic if G is restricted to have exactly two outgoing edges from every vertex (but t), labeled by complementary literals.*

For every (input) sequence $\sigma \in \{0,1\}^n$ define $G_\sigma(V, E_\sigma)$ to be the (unlabeled) subgraph of G with $e \in E_\sigma$ iff either $\mu(e) = 1$, or $\mu(e) = x_i^\epsilon$ and $\sigma_i = \epsilon$.

The table in figure 1 gives several accepting criteria, restrictions on the program, the notation for the

Accepting Criteria	*Restriction on BP*	*Program size*	*Complexity Class*
1 (mod 2)	none	$\oplus BP(f)$	$\oplus\mathcal{L}$
1 (mod m)	none	$\mathrm{mod}_m BP(f)$	$\mathrm{mod}_m\mathcal{L}$
>0	none	$NBP(f)$	$\mathcal{NL}$
>0	G undirected	$SBP(f)$	$\mathcal{SL}$
>0	deterministic	$BP(f)$	$\mathcal{L}$

Figure 1: The different complexity classes

smallest size of a Branching Program with the given criteria and restrictions and the class defined by allowing polynomial complexity. The accepting criteria of an input $\sigma \in \{0,1\}^n$ are in terms of the *number* of $s-t$ paths in G_σ.

The classes $\mathcal{NL}$, $\mathcal{SL}$ and $\mathcal{L}$ are called nondeterministic, symmetric and deterministic *logspace* respectively. It is clear that $\mathcal{L}$ is contained in all four other classes, and that $\mathcal{SL} \subseteq \mathcal{NL}$. No other nontrivial relationships were known. Later we will prove that $\mathcal{SL}$ is contained in $\oplus\mathcal{L}$. We will denote by $m\mathcal{NL}$, $m\mathcal{SL}$ and $m\mathcal{L}$ the monotone analogues of $\mathcal{NL}$, $\mathcal{SL}$ and $\mathcal{L}$ defined by allowing only positive literals to label edges of the branching programs.

There are no lower bounds known for algebraic branching programs. Nečiporuk [11] presented a method which yields lower bounds of the form $\Omega((n/\log n)^2)$ for deterministic branching programs. Pudlák [13] observed that the method yields lower bounds of the form $\Omega(n^{3/2}/\log n)$ for nondeterministic branching programs. Here we observe that Pudlák's idea carries over to the algebraic model:

Fix a partition of the variable set $[n]$ into k disjoint subsets A_i, $i \in [k]$. For every $i \in [k]$ let $c_i(f)$ be the number of distinct subfunctions of f on the variables A_i obtained by fixing the remaining variables to constants in all possible ways.

Theorem 1 With the notation of the above paragraph,

$$\oplus BP(f) \geq \frac{1}{2}\sum_{i\in[k]} \sqrt{\log c_i(f)}$$

Proof: The idea is very simple. If $G(V,E,\mu)$ is the given program (nondeterministic or $GF(2)$) computing f, any fixing of the variables outside A_i to constants results in a reduced branching program for the resulting subfunction. Let E_i be the edges of E which μ labels by literals from A_i, and let V_i be the vertices touched by these edges. Then without loss of generality the reduced program uses only the vertices V_i, on which we have the edges E_i and perhaps some extra edges labeled 1 that resulted from fixing values. But there are at most $2^{|V_i|^2}$ different possible programs, and as $|V_i| \leq 2|E_i|$ and the size of $\hat{M}$ is $\sum_{i\in[k]}|E_i|$, the bound follows. ■

Let ED_n be the function which receives n numbers in the range $\{1,\cdots,n^2\}$ and decides whether all n numbers are distinct.

Corollary 1 $\oplus BP(ED_n) = \Omega(n^{3/2}/\log n)$.

Proof: The element distinctness function ED_n is a canonical example of a function having many subfunctions (see [4]). The partition of variables is the natural one, a part for each integer ($2\log n$ bits). The number k of parts is $n/(2\log n)$, and for every part $c_i(ED_n) = 2^{\Theta(n)}$. ■

3 The basic model: Span Programs

We first need some notation. Let K be a field[1] and W a vector space over K. We implicitly fix a basis for W, and denote by $\mathbf{1}$ ($\mathbf{0}$) the vector in W all of whose entries are 1 (0). For vectors $w,z \in W$ we let $w \cdot z$ denote their inner product. The dimension of a subspace $Z \subseteq W$ is denoted $dim(Z)$, and the affine dimension of an affine subspace Z is denoted $adim(Z)$.

Let M be a matrix over K, and s,t vectors over K. Then sM and Mt denote left and right multiplication with M, where we always assume that the dimensions "match", and do not bother distinguishing between row and column vectors. The *span* of M, denoted $span(M)$ is the subspace generated by the rows of M, i.e. all vectors of the form sM.

Definition 2 *Fix a field K. A* span program *over K is a labeled matrix $\hat{M}(M,\rho)$ where M is a matrix over K and $\rho : rows(M) \to \{x_i^\epsilon \,|\, i\in[n],\, \epsilon = 0,1\}$. The size of $\hat{M}$ is the number of rows in M.*

[1] An extension to arbitrary rings is possible, but we avoid it here.

For every input sequence $\sigma \in \{0,1\}^n$ define the submatrix M_σ of M by keeping only rows r such that $\rho(r) = x_i^\epsilon$ and $\sigma_i = \epsilon$. We say that $\hat{M}$ accepts σ iff $\mathbf{1} \in span(M_\sigma)$.
REMARK: Note that the vector $\mathbf{1}$ in the definition can be replaced by any fixed nonzero vector (as will sometimes be convenient) via a change of basis.

Observe that this definition of a span program is equivalent to the one given in the introduction, and in particular we will denote by X_i^ϵ the subspace generated by the rows associated with x_i^ϵ. In this way, $s(\hat{M}) = \sum_{i,\epsilon} dim(X_i^\epsilon)$.

We denote by $SP_K(f)$ the size of the smallest span program computing f (over K) and by $\mathcal{PSP}_K$ the class of all languages for which this size measure is polynomial. When $K = GF(p)$ we abbreviate $\mathcal{PSP}_{GF(p)}$ by $\mathcal{PSP}_p$.

A span program $\hat{M}(M,\rho)$ is called *monotone* if the image of ρ is only the positive literals $\{x_1, ..., x_n\}$. It is evident that $\hat{M}$ computes a monotone function, and it is an easy exercise to see that every monotone function can be computed by a monotone span program. Define $mSP_K(f)$ to be the smallest size of a monotone span program for f, and by $m\mathcal{PSP}_K$ the complexity class of languages for which this measure is polynomial.

The connection between this model and counting branching programs can be established using the results of [5]. They show that testing linear dependence (as well as most other natural linear algebra problems like computing rank and determinant) is a complete (under logspace reductions) problem for the counting programs.

Theorem 2 For every prime p, $\mathcal{PSP}_p = \text{mod}_p\mathcal{L}$.

Proof: Follows from the arguments of [5]. ∎

Note that, as there is a polynomial loss in the simulation between the two models, it does not suffice to prove super-linear bounds. For this purpose we prove the much tighter connection for programs over $GF(2)$:

Theorem 3 For every function f, $SP_2(f) \leq 2 \oplus BP(f)$.

Proof: Let $G(V, E, \mu)$ be a branching program (over $GF(2)$), with $s, t \in V$ its source and sink respectively.

We shall be interested in all intermediate functions computed by the branching program. For every vertex $a \in V$ (resp. edge $e \in E$) we define the functions f_a (resp. f_e) computed at that vertex (resp. edge) in the natural way: for every $\sigma \in \{0,1\}^n$, $f_a(\sigma) = 1$ (resp. $f_e(\sigma) = 1$) iff there is an odd number of paths in G_σ from s ending in the vertex a (resp. the edge e). We have the following relations (denoted $(*)$ for later use) between these functions, which capture the local computation in this model.

1. If e is an edge whose tail is the vertex a, then $f_e = f_a \wedge \mu(e)$.
2. If $B \subseteq E$ is the set of all edges whose head is the vertex $b \in V$ then $f_b = \oplus_{e \in B} f_e$

Let $f = f_t$ be the function computed by G, and $U = f^{-1}(0)$ be the zero set of f. Let $2^U = \{h : U \to \{0,1\}\}$ be the set of all Boolean functions on U, which we identify with the vector space $GF(2)^U$ of all binary vectors indexed by U. The functions $\oplus, \wedge$ act on 2^U in the natural way, i.e. component-wise. In what follows, we will restrict functions to U so that two functions will be rendered as equal if they agree on U. For a function g, we will denote by $\boldsymbol{g}$ its restriction to U. We will abuse notation and look at $\boldsymbol{g}$ as both an element of 2^U and of $GF(2)^U$. Note that the local conditions $(*)$ are still satisfied by the functions $\boldsymbol{f}_a$ and $\boldsymbol{f}_e$. Also, note that $\boldsymbol{f}_s = \mathbf{1}$ and $\boldsymbol{f}_t = \mathbf{0}$.

We will work with the set R of all odd vectors in $GF(2)^U$ (having an odd number of 1's). We shall view members r of R as linear functionals on $GF(2)^U$, acting by inner product $r \cdot h$. Every such vector $r \in R$ *defines* an input $\sigma(r) \in \{0,1\}^n$ by $\sigma(r)_i = r \cdot \boldsymbol{x}_i^1$. (Note that since r is odd the complements $\overline{\sigma(r)_i}$ are consistently defined by $r \cdot \boldsymbol{x}_i^0$.) We say that $r \in R$ is *consistent* if for every edge $e = (a, b) \in E$ it satisfies the condition:

$$\text{if } \mu(e) = x_i^\epsilon \text{ then } (r \cdot \boldsymbol{f}_a) \wedge (r \cdot \boldsymbol{x}_i^\epsilon) = r \cdot (\boldsymbol{f}_a \wedge \boldsymbol{x}_i^\epsilon).$$

Claim 1 *For every $\sigma \in \{0,1\}^n$, $f(\sigma) = 0$ iff there exists a consistent $r \in R$ such that $\sigma(r) = \sigma$.*

Proof: We prove both implications.

($\Rightarrow$) Assume $f(\sigma) = 0$, thus $\sigma \in U$. Define r to be the characteristic vector of $\{\sigma\}$ (i.e. $r(u) = 1$ iff $u = \sigma$). Clearly, $r \in R$. Also, for such r and any function h we have $r \cdot \boldsymbol{h} = h(\sigma)$. Therefore, the conditions $(*)$ imply the consistency of r.

($\Leftarrow$) Assume $r \in R$ is consistent, and let $\sigma = \sigma(r)$. We shall prove that for all $a \in V$, $e \in E$ we have $f_a(\sigma) = r \cdot \boldsymbol{f}_a$ and $f_e(\sigma) = r \cdot \boldsymbol{f}_e$. This is proved by induction on the topological ordering of G. It clearly holds for the source s, as for odd r, $r \cdot \boldsymbol{f}_s = r \cdot \mathbf{1} = 1 = f_s(\sigma)$. For the induction step we have two cases, as in $(*)$.

1. If $e \in E$ has its tail in vertex a, and $\mu(e) = 1$, then because r is odd and by the inductive hypothesis we have $r \cdot (\boldsymbol{f}_a \wedge \mathbf{1}) = (r \cdot \boldsymbol{f}_a) \wedge (r \cdot \mathbf{1}) = f_a(\sigma) = f_e(\sigma)$.

2. If $e \in E$ has its tail in vertex a, and $\mu(e) = x_i^\epsilon$, then by consistency of r and the inductive hypothesis we have $r\cdot(\boldsymbol{f}_a \wedge \boldsymbol{x}_i^\epsilon) = (r\cdot\boldsymbol{f}_a)\wedge(r\cdot\boldsymbol{x}_i^\epsilon) = f_a(\sigma)\wedge\sigma_i^\epsilon = f_e(\sigma)$.

3. If $B \subseteq E$ is the subset of edges with their head in vertex b then, by linearity of the action of r and the inductive hypothesis, we have $r\cdot(\oplus_{e\in B}\boldsymbol{f}_e) = \oplus_{e\in B}(r\cdot\boldsymbol{f}_e) = \oplus_{e\in B}f_e(\sigma) = f_b(\sigma)$.

Finally, $f(\sigma) = f_t(\sigma) = r\cdot\boldsymbol{f}_t = r\cdot\mathbf{0} = 0$, which concludes the proof. ■

Next we show that testing the consistency of r on any edge e requires only two linear tests.

Claim 2 *Given $r \in R$ and $\sigma = \sigma(r)$, r is consistent on $e = (a,b)$ with $\mu(e) = x_i^\epsilon$ iff*

$$\sigma_i = 0 \rightarrow r\cdot(\boldsymbol{f}_a\wedge\boldsymbol{x}_i^1) = 0 \text{ and } \sigma_i = 1 \rightarrow r\cdot(\boldsymbol{f}_a\wedge\boldsymbol{x}_i^0) = 0$$

Proof: By inspection. ■

The above claims suggests the following construction of a span program that computes f, and has size (number of rows) which is twice as large as the size of G. Construct $\hat{M}(M,\rho)$ such that for every $e = (a,b) \in E$ with $\mu(e) = x_i^\epsilon$ there are two rows in M: one is $\boldsymbol{f}_a \wedge \boldsymbol{x}_i^1$ which ρ labels by x_i^0 and the second is $\boldsymbol{f}_a \wedge \boldsymbol{x}_i^0$ which ρ labels by x_i^1.

Finally, we have to show that the program $\hat{M}$ just defined computes f. First observe that by Claim 2, any vector $r \in R$ is consistent iff $M_{\sigma(r)}r = \mathbf{0}$. This matrix vector product is just a concise way of simultaneously checking the test of Claim 2 for every edge. Second, by Claim 1, it follows that for every $\sigma \in \{0,1\}^n$, $f(\sigma) = 0$ iff there exists an $r \in R$ with $\sigma(r) = \sigma$ and $M_\sigma r = \mathbf{0}$. Third, since r is odd, by duality the condition $M_\sigma r = \mathbf{0}$ holds iff $\mathbf{1} \notin span(M_\sigma)$. It now follows from the definition of computation by span programs that $\hat{M}$ computes f. ■

4 Symmetric vs. Counting Logspace

As stated in the introduction, no relationship is known between the classes $\mathcal{NL}$ and $\oplus\mathcal{L}$. In this subsection we will show that uniform symmetric logspace is contained in uniform $\text{mod}_p\mathcal{L}$ for all p. Note that $\mathcal{SL} \subseteq \mathcal{L}/\text{poly}$ follows from the results of [1].

Theorem 4 For every p, $\mathcal{SL} \subseteq \text{mod}_p\mathcal{L}$.

The theorem will follow from theorem 2 together with the following theorem.

Theorem 5 For every field K, $\mathcal{SL} \subseteq \mathcal{PSP}_K$. Also, $m\mathcal{SL} \subseteq m\mathcal{PSP}_K$.

Proof: The proof follows simply from the fact that a graphic matroid can be represented as a regular matroid over any field K. This will yield a small span program for any function in $\mathcal{SL}$. We briefly review the construction.

Let $G(V,E,\mu)$ be a symmetric branching program for a function f, with $s,t \in V$ its special vertices. Recall that G is an undirected graph, and that $f(\sigma) = 1$ iff there is an st-path in G_σ. Fix a field K, and a basis $\{v_a : a \in V\}$ for the vector space K^V.

The span program $\hat{M}(M,\rho)$ is constructed as follows. For every edge $e = (a,b) \in E$ add the row $v_a - v_b$ to M and label this row by $\mu(e)$. It is immediate that there is an st-path in G_σ iff M_ρ spans the vector $v_s - v_t$. Therefore $\hat{M}$ computes f, and its size is $|E|$. ■

5 Canonical Span Programs

The results in this section are stated for span programs over $GF(2)$, but can be easily extended to programs over any field K. We show that, for purposes of lower bounds, it suffices to consider span programs of a special form.

Definition 3 *Let $\hat{M}(M,\rho)$ be a span program computing f. We say that $\hat{M}$ is* canonical *if the columns of M are in $1-1$ correspondence with the zeros U of f, and for every $u \in U$, the column corresponding to u in M_u is $\mathbf{0}$.*

Note that the span program constructed in the proof of theorem 3 is canonical.

Theorem 6 Every span program can be converted to a canonical span program of the same complexity and computing the same function. Moreover, the conversion preserves monotonicity.

Proof: Given a span program $\hat{M}(M,\rho)$ computing f we construct a canonical span program $\hat{N}(N,\rho)$ for f with the same row labeling ρ (and in particular the same number of rows) as follows. Fix $u \in U$. By definition $\mathbf{1} \notin span(M_u)$ and thus by duality there exists an odd vector $r(u)$ (whose length is the same as a row in M) satisfying $M_u r(u) = \mathbf{0}$. Define the column corresponding to u in N to be $Mr(u)$. Doing this for all $u \in U$ defines $\hat{N}$ and guarantees that it rejects every u. To see that $\hat{N}$ accepts all ones of f, fix σ with $f(\sigma) = 1$, and let $w(\sigma)$ be the linear combination such that $w(\sigma)M_\sigma = \mathbf{1}$. But since $r(u)$ is odd for every $u \in U$, then $w(\sigma)M_\sigma r(u) = \mathbf{1}\cdot r(u) = 1$. We get $w(\sigma)N_\sigma = \mathbf{1}$ as required.

Note that since ρ is preserved, if $\hat{M}$ was monotone than so is $\hat{N}$. ∎

An important application of Theorem 6 is that the size measure SP_2 is monotone under restrictions. Say that g is a *restriction* of f if g is the result of giving a truth assignment to a subset of the variables of f (g will be a function of the remaining variables). It is natural to demand from any complexity measure c on Boolean functions that $c(g) \leq c(f)$ in this case. For measures like circuit size and depth this is immediate. However note that from the definition of span programs it is not clear whether such a relation holds for SP_2. The problem is that, if $\hat{M}$ computes f, there is no obvious way of getting rid of the rows corresponding to the literals that were set to "true". Still, using Theorem 6 we can prove

Theorem 7 If g is a restriction of f, then $SP_2(g) \leq SP_2(f)$. Moreover, if f is monotone, then also $mSP_2(g) \leq mSP_2(f)$.

Proof: It will clearly suffice to prove the theorem when g is obtained from f by setting the variable x_1 to 1. Let $\hat{M}(M,\rho)$ be a canonical span program for f of size $SP_2(f)$. Define the span program $\hat{N}(N,\rho\prime)$ as follows. Let U_1 be the subset of U of zeros of f whose first coordinate (corresponding to x_1) is 1. Then the matrix N is the submatrix of M on the rows ρ labels by literals other than x_1^1, x_1^0, and columns corresponding to U_1. The labeling $\rho\prime$ is the same as ρ on the rows of N.

Now we show that N computes g. The zero set of g is simply all vectors in U_1 with first coordinate removed, and clearly N rejects all of them. To see that N accepts all ones of g it suffices to note that, since $\hat{M}$ is canonical, every coordinate in M whose row was labeled x_1^1 and whose column is from U_1 is 0. If $g(\sigma) = 1$, also $f(1\sigma) = 1$ and $\mathbf{1} \in span(M_{1\sigma})$, but by this observation, in the columns U_1, the rows labeled x_1^1 were of no use for this span, and so $\mathbf{1} \in span(N_\sigma)$ as well.

Again note that this construction leaves a monotone span program monotone. ∎

6 Lower bounds on Span Programs

6.1 Affine dimension

We start by giving a lower bound on the size of a span program for a function f in terms of the affine dimension of a graph associated with f. This algebraic measure has been proposed as a source of lower bounds for formulae and Boolean branching programs, and has been studied in [14, 17].

Fix a field K and fix a partition of the variables of f into two sets A, B. Thus every sequence $\sigma \in \{0,1\}^n$ decomposes in a natural way into $\sigma^A \in \{0,1\}^A$ and $\sigma^B \in \{0,1\}^B$ such that $\sigma = \sigma^A \circ \sigma^B$ ($\circ$ denotes concatenation). A representation χ of f in a vector space W (over K) assigns to every sequence $\tau \in \{0,1\}^A \cup \{0,1\}^B$ an affine subspace $\chi(\tau)$ of W such that for every $\sigma \in \{0,1\}^n$, $f(\sigma) = 0$ iff $\chi(\sigma^A) \cap \chi(\sigma^B) = \emptyset$. Define $adim_K(f)$ to be the smallest dimension of a vector space W for which such a representation exists.

Theorem 8 For every field K and every function f, $adim_K(f) \leq SP_K(f)$

Proof: Fix a field K, and let $\hat{M}(M,\rho)$ be a span program for f. Let $[n] = A \cup B$ be a partition of the variables of f.

The underlying vector space we will use to represent f is the span of all rows of M. To any sequence $\sigma^A \in \{0,1\}^A$ assign the closure of the union of the subspaces $\{X_i^\epsilon : i \in A \text{ and } \epsilon = \sigma_i^A\}$. To every $\sigma^B \in \{0,1\}^B$ assign the affine subspace obtained by adding $\mathbf{1}$ to all vectors in the closure of the subspaces $\{X_i^\epsilon : i \in B \text{ and } \epsilon = \sigma_i^B\}$. This defines the required mapping χ. It is easy to check that for every $\sigma = \sigma^A \circ \sigma^B$, $\mathbf{1} \in span(M_\sigma)$ iff $\chi(\sigma^A) \cap \chi(\sigma^B) \neq \emptyset$. ∎

6.2 A lower bound for Majority

We now proof a lower bound for the size of any Span Program for Majority. Let MAJ_n be the function which returns 1 if strictly more than half the inputs are one.

Theorem 9 $SP_2(MAJ_n) = \Omega(n \log\log\log^* n)$.

Proof: Let $\hat{M}(M,\rho)$ be a canonical span program that computes MAJ_{2n} of size nh. An easy argument shows that at most n variables are associated with no more than h rows each. Fix the rest of the variables arbitrarily with $n-s+2d$ zeros and $s-2d$ ones where s and d are parameters to be specified later. By theorem 7, there is a span program $\hat{N}(N,\rho)$ that computes this particular restriction of Majority and it is easy to show that in $\hat{N}$ every variable is associated with at most h rows. Also, it is easy to see that $\hat{N}$ accepts all vectors with $n-s+3d$ ones but rejects all vectors with only $n-s+d$ ones.

We will show that any span program $\hat{N}$ where each positive literal is associated with h rows and which rejects all vectors with $n-s+d$ ones, rejects a vector with $n-s+3d$ ones as well. This will give us the desired lower bound. Note that the number of rows in $\hat{N}$ associated with negative literals is inmaterial for the argument.

We will use the following Ramsey–like combinatorial statement. A similar statement is implicitely proved in [18]. We will use $[n]^k$ to denote all k subsets of $[n]$. Let $q = 2^h$.

Proposition 1 Let the parameters q, d, s, n satisfy $d = (2q)!$, $4dq + d \leq s \leq 0.1 \log^* n$. Let $\chi_i : [n]^{s-d} \mapsto [q]$ be a collection of colorings, one for every $i \in [n]$. Then there exists a set $A \in [n]^s$ and three disjoint d-subsets of A, B_0, B_1, B_2, such that the three sets (in $[n]^{s-d}$) $C_l = A \backslash B_l$, $l \in Z_3 = \{0,1,2\}$ satisfy the following[2]: For every $l \in Z_3$ and for every $i \in B_l$ we have $\chi_i(C_{l+1}) = \chi_i(C_{l+2})$.

We will defer the proof of the proposition for later and finish the proof of the theorem.

Recall that $\hat{N}$ is a cannonical span program that rejects all vectors with $n-s+d$ ones. We shall restrict our attention to the columns of N associated with these vectors. Furthermore, we will associate these vectors with elements in $[n]^{s-d}$ by complementation. Let $\chi_i : [n]^{s-d} \mapsto \{0,1\}^h$ be defined as follows: $\chi_i(S)$ is the restriction of the column associated with S to the rows labelled by x_i. Clearly, χ_i can be looked at as functions whose range is $[q]$ (recall that $q = 2^h$).

Let A and B_l, C_l for $l \in Z_3$ be the sets guaranteed by proposition 1. Let u_i be the characterisitic vector of the complement of C_i for $i \in Z_3$ and $v = u_1 \oplus u_2 \oplus u_3$ (it is easy to see that $v = u_1 \vee u_2 \vee u_3$ as well). Note that each of the vectors u_i have $n-s+d$ ones and v has $n-s+3d$ ones. We will show that $\hat{N}$ rejects v. Consider the vector r which has a 1 in the positions indexed by the vectors u_i and 0 elsewhere. We will show that $N_v r = \mathbf{0}$ and, hence, by duality we have that $\mathbf{1} \notin span(N_v)$.

Recall that by definition of $\hat{N}$ the column corresponding to u_i in N_{u_i} is $\mathbf{0}$ for $i \in Z_3$. Also note that multiplication by r simply adds these three columns, but that we look only at rows whose label is set to 'true' by the assignment v. Consider a row associated with a literal y. We distinguish three cases:

- $y = x_j^0$. In this case, each u_i have a 0 in position j so that each of the three columns have a zero in this row. Hence the sum in this coordinate will also be zero.
- $y = x_j^1$ with $j \notin A$. Again, all three vectors u_i have a 1 in position j so that each of the three columns have a zero in this row. Hence the sum in this coordinate will also be zero.
- $y = x_j^1$ with $j \in B_i$ for some $i \in Z_3$. Then u_i have a 1 in position j and therefore its associated column have a 0 in position j. For the other two columns, we are guaranteed by the proposition that the value of this coordinate is the same in both. Thus also here the sum of the three columns in this coordinate is zero.

Choosing the largest possible value for s (as a function of n) in Proposition 1 and computing from it maximum values for the other parameters, we get the desired bound. ■

Proof:[of Proposition 1] The proof given here follows the ideas in [18], together with a simplification suggested to us by A. Razborov.

Define the coloring $\psi : [n]^{s-d} \mapsto [q]^{s-d}$ as follows: if $C = \{i_1, i_2, \cdots, i_{s-d}\}$ with $i_1 < \cdots < i_{s-d}$, then $\psi(C) = (\chi_{i_1}(C), \chi_{i_2}(C), \cdots, \chi_{i_{s-d}}(C))$.

It follows from Ramsey's Theorem (see e.g. [7]) that there exists a subset $A \in [n]^s$ and a vector $v \in [q]^{s-d}$ such that every subset $C \subseteq A$, $|C| = s-d$ satisfies $\psi(C) = v$.

Now assume without loss of generality that $A = [s]$. To specify the subsets B_l it suffices to give a coloring $\phi : [s] \rightarrow \{0,1,2,*\}$ so that exactly d elements from $[s]$ are mapped to each of the colors $0,1,2$ (representing the sets B_0, B_1, B_2 resp.). Think of ϕ as a vector in $\{0,1,2,*\}^{[s]}$ and let $\hat{\phi}$ be the vector that we get from ϕ by deleting all $*$'s (a vector in $\{0,1,2\}^{3d}$). Call ϕ *regular* if it satisfies $\hat{\phi} = (0^r 1^r 2^r)^{d/r}$ for some $r \leq 2q$ (note that r divides $d = (2q)!$).

For any regular ϕ, the definition of ψ and the choice of A guarantees that if $i \in B_0$ then by regularity the rank, k, of i in both C_1 and C_2 is the same (it is $|[i] \setminus B_1| = |[i] \setminus B_2|$) and thus $\chi_i(C_1) = \chi_i(C_2) = v_k$. An identical argument holds for every $i \in B_2$.

To handle the remaining case, $i \in B_1$, notice that by regularity the rank of i in C_0 and C_2 differs by r exactly. Therefore, we will choose a regular ϕ for which every $i \in B_1$ satisfies $v_{rk(i)} = v_{rk(i)+r}$, where $rk(i) = |[i] \setminus B_0|$ is the rank of i in C_0. That such a choice is possible follows immediately from the following claim.

Claim: For every vector $v \in [q]^{s-d}$ there exists $r \leq q$ and a set of positions $K \subset [s-d]$ of cardinality $|K| = (s-d)/4q$ such that for every $k \in K$, $v_k = v_{k+r}$.

Proof:[of claim] Mark each position $k \in [s-d+1]$ with an integer $r(k) \leq 2q$ which is the smallest integer

[2] We choose the indices from Z_3 as we shall perform on them addition modulo 3.

for which $v_k = v_{k+r(k)}$, or set $r(k) = BIG$ if there is no such integer. By the pigeonhole principle at most half the positions can be marked BIG, and so for some $r \leq 2q$ at least $(s-d)/4q$ of the positions (which we choose as the set K) satisfy $r(k) = r$. ■

Let us see how to use this claim to construct ϕ. Go through the elements of $A = [s]$ in order. Color the first r with color 0, then the next available r elements of K with color 1, then the next r elements with color 2, and repeat this process d/r times. Color all the skipped elements $*$. By the claim $|K| \geq (s-d)/4q$ which is larger than $3d$. Therefore, the process will terminmate succesfully. ■

Corollary 2 $\oplus BP(MAJ_n) = \Omega(n \log\log\log^* n)$.

Corollary 3 $\oplus BP(T_n^k) = O(n)$ iff either $k = O(1)$ or $n - k = O(1)$.

7 Monotone Span Programs

Monotone analogues of Boolean complexity classes are studied in [8]. We will adopt their notation of $m\mathcal{C}$ to denote the monotone analogue of the class $\mathcal{C}$. Recall that a branching program is monotone if we use only positive literals to label its edges. For the algebraic computation model of branching programs this restriction is clearly useless, as the model still computes nonmonotone functions. However, it is interesting to study monotone span programs. Recall that a span program $\hat{M}(M,\rho)$ is called *monotone* if the image of ρ is only the positive literals $\{x_1, ..., x_n\}$.

A curious thing about this model is that it uses nonmonotone operations (linear algebra over a field) to compute monotone functions. Thus, for example we don't even know if $m\mathcal{PSP}_2 \subseteq m\mathcal{P}$. In studying the complexity of threshold functions in this monotone model we reveal another curious property unique to this model: except the trivial AND and OR, all threshold functions are equivalent. For $0 \leq k \leq n$ define T_n^k to be the Boolean function which accepts vectors with at least k ones.

Theorem 10 For $k \in \{1, n\}$, $mSP_2(T_n^k) = n$. For $1 < k < n$, $mSP_2(T_n^k) = \Theta(n \log n)$.

This theorem follows from the following two theorems.

Theorem 11 $mSP_2(T_n^2) \geq n \log n$.

Proof: The proof is an algebraic variation on the $n \log n$ lower bound on the formula size for T_n^2 [10]. Let $\hat{M}(M,\rho)$ be a monotone span program for T_n^2. Let t be the number of columns in M, and R the set of odd vectors in $GF(2)^t$. Clearly $|R| = 2^{t-1}$. Let $d_i = dim(X_i^1)$. For a subspace V of $GF(2)^t$, its *orthogonal complement*, $V^\perp$, is defined as the set of vectors orthogonal to every vector in V.

For every $i \in [n]$ let $R_i = R \cap (X_i^1)^\perp$. Since $\hat{M}$ rejects vectors of weight one, $\mathbf{1} \notin X_i^1$, and so for every i, $|R_i| = 2^{t-1-d_i}$.

We now claim that $R_i \cap R_j = \emptyset$. To see this, observe that for every pair $i \neq j$ we must have that the vector $\mathbf{1}$ is in the closure of X_i^1 and X_j^1 (as $\hat{M}$ accepts the vector with ones in positions i and j). Therefore, for some $u_i \in X_i^1$ and $u_j \in X_j^1$, $u_i \oplus u_j = \mathbf{1}$. If there exist a vector $r \in R_i \cap R_j$, then $r \cdot \mathbf{1} = 1$ while $r \cdot u_i = r \cdot u_j = 0$, a contradiction.

We finish the proof by noticing that the previous two paragraphs imply that $\sum_{i \in [n]} 2^{t-1-d_i} \leq 2^{t-1}$, and by Jensen's inequality $\sum_{i \in [n]} d_i \geq n \log n$. ■

Theorem 12 $mSP_2(T_n^k) = O(n \log n)$.

Proof: The proof will be derived from a simple construction showing $mSP_K(T_n^k) = n$ whenever $|K| > n+1$, which in turn is based on the same idea behind Shamir's secret sharing scheme for threshold functions [20]. We will choose $K = GF(2^l)$ with l the smallest integer $> \log n$. It will be easy to see that a similar proof will work for any characteristic p and yield $mSP_p(MAJ_n) = O(n \log_p n)$.

Let a_i for $0 \leq i \leq n$ be $n+1$ distinct nonzero elements of $GF(2^l)$. For $0 \leq i \leq n$ define the vectors $v_i = (a_i^0, a_i^1, \cdots, a_i^{k-1})$ in $GF(2^l)^k$. Clearly every k such vectors are linearly independent over $GF(2^l)$. This suggests the following span program $\hat{M}(M,\rho)$ over $GF(2)^l$. M is an $n \times k$ matrix whose ith row is v_i and is labeled x_i. It follows that for every subset $S \subseteq [n]$, $v_0 \in span(M_S)$ iff $|S| \geq k$, and thus $\hat{M}$ computes T_n^k. A change of basis can be used to replace the spanned vector v_0 with the vector $\mathbf{1}$.

To derive from $\hat{M}$ a span program over $GF(2)$ for T_n^k, fix a representation of the elements of $GF(2^l)$ as vectors in $GF(2)^l$ in the usual way (i.e. these vectors are degree l polynomials over $GF(2)$ with addition and multiplication performed modulo a fixed irreducible polynomial). For $a \in GF(2^l)$ denote its representation by $\bar{a} = (a_0, a_1, \cdots, a_{l-1}) \in GF(2)^l$. Similarly, any vector $v \in GF(2^l)^k$ can be viewed as a vector $\bar{v} \in GF(2)^{kl}$.

The important property we use is that multiplication in this representation is a bilinear operator. Thus, there are $l \times l$ matrices A_j, $0 \leq j \leq l-1$ such that if $a, b \in GF(2^l)$ then for every j, $\overline{(ab)}_j = \bar{a} A_j \bar{b}$. Now we can "encode" every element $b \in GF(2^l)$ by an $l \times l$ matrix whose jth column is $A_j \bar{b}$. Then multiplication by b (over $GF(2^l)$) becomes vector product over $GF(2)$ by the encoding matrix.

This leads to the construction of the span program $\hat{N}(N, \rho')$ over $GF(2)$. Simply replace every element in M by its $l \times l$ encoding, which makes N of dimensions $nl \times kl$. For every row in $\hat{M}$ labeled x_i by ρ, each of the corresponding l rows in $\hat{N}$ will be labeled x_i by ρ'. It is now routine to check that N_S spans the vector $\overline{v}_0$ iff M_S spans v_0, and thus $\hat{N}$ computes T_n^k as well. The size of $\hat{M}$ is $nl \leq 2n \log n$ as required. ∎

8 Monotone Span Programs and secret sharing

It turns out that our monotone model captures in an elegant way secret sharing schemes in the information theoretic model. Informally, a secret sharing scheme for a monotone function f prescribes a way for a "sender" having a secret $s \in K$ to "break it" into n parts $s_i \in K^{d_i}$ satisfying the following: Let $T \in [n]$ be a subset of the pieces, and denote by $f(T)$ the function f evaluated on the characteristic vector of T. Then if $f(T) = 1$ the pieces $\{s_i : i \in T\}$ determine s, while if $f(T) = 0$ these pieces give no information whatsoever about s. The size of such scheme is $\sum_{i \in [n]} d_i$. Such a scheme was first described to us by Rudich [19].

Theorem 13 For every prime p, every monotone function has a secret sharing scheme (over $GF(p)$) of size $mSP_p(f)$

Proof: Fix a prime p, set $K = GF(p)$ and let $\hat{M}(M, \rho)$ be a monotone span program for a monotone function f. Let d_i be the number of rows labeled x_i by ρ, and M_i the submatrix of M consisting of these rows. Let t be the number of columns in M.

Let $s \in K$ be the secret, and let $W = \{w \in K^t : w \cdot \mathbf{1} = s\}$. Let $\boldsymbol{w} \in W$ be chosen uniformly at random, and define the "random pieces " $\boldsymbol{q} \in K^{d_i}$ for every $i \in [n]$ by $\boldsymbol{q} = M_i \boldsymbol{w}$. Further, for any subset $T \subseteq [n]$ let $\boldsymbol{q} = M_T \boldsymbol{w}$, where M_T is the matrix associated with the characteriztic vector of T. Note that $\boldsymbol{q}$ is just the concatenation of the vectors $\{\boldsymbol{q} : i \in T\}$. The theorem follows from the following claim:

Claim 3 *If $f(T) = 1$ then s can be efficiently determined from $\boldsymbol{q}$. Conversely, If $f(T) = 0$ then for every $a \in K$, $\mathbf{P}[s = a \mid \boldsymbol{q}] = 1/p$.*

To prove the first part, assume that $f(T) = 1$. Then, by definition, there is a vector v such that $vM_T = \mathbf{1}$ (and this vector can be easily computed from M). Then $s = \mathbf{1}\boldsymbol{w} = vM_T\boldsymbol{w} = v\boldsymbol{q}$.

To prove the second part, assume $f(T) = 0$. By duality, there is a vector $z \in K^t$ such that $M_T z = 0$, but $\mathbf{1} \cdot z \neq 0$. Then for any q, to any $\boldsymbol{w}$ such that $M_T \boldsymbol{w} = q$ we can associate the p vectors $w_j = \boldsymbol{w} + jz$, $j \in Z_p$. Note that $M_T w_j = q$ as well, but the values $\mathbf{1} \cdot w_j$ are all distinct and exhaust $GF(p)$. This breaks up the probability space $\{\boldsymbol{w} : M_T \boldsymbol{w} = q\}$ into p equiprobable classes, each giving s a different value, which concludes the proof of the claim. ∎

The function f is *efficiently sharable* if there is a polynomial size sharing scheme, and all computation involved in encoding and decoding is in polynomial time. We have proved:

Corollary 4 All functions in $m\mathcal{PSP}_p$ are efficiently sharable (over $GF(p)$).

This result is a generalization of a result of Rudich [19] who showed that all functions in $m\mathcal{SL}$ are efficiently sharable. Recall that by Theorem 5 (see subsection 4) $m\mathcal{SL} \subseteq m\mathcal{PSP}_K$ for every K. It is not clear how tight the upper bound in Theorem 13 is. It seems that by appropriately defining *linear* schemes by restricting all functions used in construction and decoding to be K-linear over the secret and the random strings, this theorem should become tight. In fact, the upper bound of Theorem 12 was inspired by Shamir's secret sharing scheme for threshold functions [20]. It seems that we should be able to use existing linear schemes to give complexity upper bounds. We have not tried to formalize this yet.

Acknowledgments

We are very grateful to A. Razborov for his observations which lead us to a simpler proof of proposition 1. We are also grateful to P. Pudlák for helpful comments.

References

[1] R. Aleluinas, R. M. Karp, R. J. Lipton, R. J. Lovász, and C. Rackoff. Random walks, universal sequences and the complexity of maze problems. In *Proceedings of the 20th IEEE Symposium on Foundations of Computer Science*, pages 218–223, 1979.

[2] E. Allender. A note on the power of threshold circuts. In *Proceedings of the 30th IEEE Symposium on Foundations of Computer Science*, pages 580–584, 1989.

[3] L. Babai, P. Pudlák, V. Rödl, and E. Szemeredi. Lower bounds in complexity of symmetric Boolean functions. Theoretical Computer Science, pages 313–323, 1988.

[4] R. B. Boppana and M. Sipser. The complexity of finite functions. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science, vol. A (Algorithms and Complexity)*, chapter 14, pages 757–804. Elsevier Science Publishers B.V. and The MIT Press, 1990.

[5] G. Buntrock, C. Damm, H. Hertrampf, and C. Meinel. Structure and importance of the logspace-mod class. *Math. Systems Theory*, 25:223–237, 1992.

[6] L. Goldschlager and I. Parberry. On the construction of parallel computers from various bases of boolean functions. *TCS*, 43:43–58, 1986.

[7] R. L. Graham and B. L. Rothchild and J. H. Spencer. *Ramsey Theory* Wiley-Interscience, 1980.

[8] M. Grigni and M. Sipser. Monotone complexity. In M. Paterson, editor, *Proceedings of LMS workshop on Boolean function complexity, Durham.* Cambridge University Press, 1990.

[9] M. Karchmer and A. Wigderson. Characterizing non-deterministic circuit size, To appear in STOC'93.

[10] R. E. Krichevskii. Complexity of contact circuits realizing a function of logical algebra. *Doklady of the Academy of Sciences of the USSR*, 151(4):803–806 (in Russian), 1963. English translation in *Soviet Physics Doklady* 7:4, pages 770–772 (1964).

[11] E. I. Nečiporuk. On a Boolean function. *Doklady of the Academy of Sciences of the USSR*, 169(4):765–766 (in Russian), 1966. English translation in *Soviet Mathematics Doklady* 7:4, pages 999-1000.

[12] C. Papadimitriou and S. Zachos. Two remarks on the power of counting. In *Proceedings of the 6th GI conference on Theoretical Computer Science, Lecture Notes in Computer Science,* 145, pages 269–276, Berlin, 1983. Springer-Verlag.

[13] P. Pudlák. Private communication.

[14] P. Pudlák and V. Rödl. A combinatorial approach to complexity. *Combinatorica*, 12:221–226, 1992.

[15] A. Razborov. Lower bounds on the size of bounded-depth networks over a complete basis with logical addition. *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):598–607, 1987. English translation in 41:4, pages 333-338.

[16] A. Razborov. On the method of approximation. In *Proceedings of the 21st ACM Symposium on Theory of Computing*, pages 167–176, 1989.

[17] A. Razborov. Applications of matrix methods to the theory of lower bounds in computational complexity. *Combinatorica*, 10(1):81–93, 1990.

[18] A. Razborov. Lower bounds on the size of switching-and-rectifier networks for symmetric Boolean functions. *Mathematical Notes of the Academy of Sciences of the USSR*, 48(6):79–91, 1990.

[19] S. Rudich. Private communication.

[20] A. Shamir. How to share a secret. *CACM*, 22:612–613, 1979.

[21] R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the 19th ACM Symposium on Theory of Computing*, pages 77–82, 1987.

[22] S. Toda. On the computational power of PP and $\oplus P$. In *Proceedings of the 30th IEEE Symposium on Foundations of Computer Science*, pages 514–519, 1989.

[23] L.G. Valiant and V.V. Vazirani. NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47:85–93, 1986.