# Superpolynomial Lower Bounds for Monotone Span Programs *

László Babai †  Anna Gál‡  Avi Wigderson§

**Abstract**

In this paper we obtain the first superpolynomial lower bounds for *monotone span programs* computing explicit functions. The best previous lower bound was $\Omega(n^{5/2})$ by Beimel, Gál, Paterson [BGP]; our proof exploits a general combinatorial lower bound criterion from that paper. Our lower bounds are based on an analysis of Paley-type bipartite graphs via Weil's character sum estimates. We prove an $n^{\Omega(\log n/\log\log n)}$ lower bound for the size of monotone span programs for the clique problem. Our results give the first superpolynomial lower bounds for linear secret sharing schemes.

We demonstrate the surprising power of monotone span programs by exhibiting a function computable in this model in linear size while requiring superpolynomial size monotone circuits and exponential size monotone formulae. We also show that the perfect matching function can be computed by polynomial size (non-monotone) span programs over arbitrary fields.

*Part of this work has been presented at the 28th ACM STOC'96. The second two authors wish to thank DIMACS for its hospitality and acknowledge its supporting agencies, the National Science Foundation under contract STC-91-19999 and the New Jersey Commission on Science and Technology.

†Department of Computer Science, University of Chicago, Chicago IL 60637-1504. E-mail: laci@cs.uchicago.edu. Supported in part by NSA grant MSPR-96G-184.

‡Dept. of Computer Sciences, The University of Texas at Austin, Austin, TX 78712. Email: panni@cs.utexas.edu. Most of this work was done while at the Institute for Advanced Study, Princeton supported by NSF Grant DMS-9304580, and DIMACS at Princeton University.

§Institute of Computer Science, Hebrew University, Jerusalem, Israel. Most of this work was done while visiting the Institute for Advanced Study, Princeton, and DIMACS at Princeton University. Research supported by the Sloan Foundation, American-Israeli BSF grant 92-00106, and a grant from the Israel Research Foundation, founded by the Israel Academy of Sciences and Humanities. Email: avi@cs.huji.ac.il .

# 1 Introduction

## 1.1 Span programs

Karchmer and Wigderson [KW] introduced span programs as a linear algebraic model for computing Boolean functions.

Let us consider a linear space $W$ over some field $K$; let $w \neq 0$ be a specified vector called the *root*. A *span program* takes a set of $n$ variables $x_1, \ldots, x_n$ and their negations, together called *literals*, and associates a subspace with each of the $2n$ literals. Such a program defines a Boolean function $f(x_1, \ldots, x_n)$ in the following way: let $U = U(\alpha_1, \ldots, \alpha_n)$ denote the span of those subspaces corresponding to TRUE literals under a given truth-assignment $x_i := \alpha_i$ ($i = 1, \ldots, n$; $\alpha_i \in \{0, 1\}$). We set $f(\alpha_1, \ldots, \alpha_n) = 1$ precisely if $w \in U$.

The *size* of a span program is the sum of the dimensions of the subspaces associated with the literals. Note that the size of the span program is not less, and at most by a factor of $2n$ greater, than the dimension of $W$ (assuming, as we may w.l.o.g., that those subspaces span all of $W$). Thus, as far as superpolynomial bounds are concerned, size and dimension are equivalent complexity measures.

A span program is called *monotone* if only the positive literals $\{x_1, \ldots, x_n\}$ are associated with subspaces (negated variables correspond to the $\{0\}$ subspace). Monotone span programs compute monotone Boolean functions, even though the computation uses non-monotone linear algebraic operations.

We denote by $\mathrm{SP}_K(f)$ (and $\mathrm{mSP}_K(f)$) the size of the smallest span program (monotone span program, resp.) over the field $K$ that computes $f$.

The class of Boolean functions with polynomial size span programs is equivalent to the class of functions with polynomial size counting branching programs [BDHM, KW]. Span program size is a lower bound on the size of symmetric branching programs [KW]. The model of symmetric branching programs is essentially the same as that of (undirected) contact schemes (for definitions, see [KW]). Lower bounds for span programs also imply lower bounds for formula size.

Span programs can be viewed as a model of *parallel computation.* Indeed, functions with polynomial size span programs over fixed finite fields belong to $NC^2$. (This is immediate from the fact that linear algebra is in $NC^2$ [Ber, BDHM, KW, Mu]). As we shall see, the monotone analog of this statement fails badly; functions admitting polynomial size monotone span programs do not necessarily have polynomial size or polylog depth monotone circuits (cf. Theorem 1.1 below).

The reduction in [KW] from symmetric branching programs to span programs preserves monotonicity, and thus lower bounds for monotone span programs imply lower bounds for monotone symmetric branching programs and for monotone formula size.

Lower bound techniques for monotone circuits and formulae are well known (e.g. Razborov [Ra1, Ra2, Ra3], Haken [Ha] for circuits, Karchmer-Wigderson [KW1], Raz-Wigderson [RW] for formulae). These techniques, however, do not appear to be adaptable to the study of monotone span programs. Beimel, Gál and Paterson [BGP] showed that monotone span programs can be strictly stronger than monotone circuits. Here we show that monotone span programs may be superpolynomially stronger than monotone circuits.

**Theorem 1.1** *There exists a family $\{f_n\}$ of monotone Boolean functions in $n$ variables such that $f_n$ can be computed by a monotone span program of size $O(n)$ (and dimension $O(\sqrt{n})$) over $GF(2)$ but requires monotone Boolean circuits of size $n^{\Omega(\log n)}$, and monotone formulae of size* $\exp(\Omega(\sqrt{n}))$ *(and consequently monotone circuits of depth $\Omega(\sqrt{n})$).*

For Boolean circuits we know that non-monotone circuits are more powerful than monotone circuits. Razborov's lower bound for the perfect matching function [Ra2] gives a superpolynomial separation between monotone and non-monotone circuits, and a result by É. Tardos [T] shows an exponential gap. No separation is known between monotone and non-monotone span programs. A natural candidate for such separation would be the perfect matching function. The function PERFECT-MATCHING$_n$ on $m = \binom{n}{2}$ variables takes as its input an undirected graph on $n$ vertices, represented by Boolean variables $x_{ij}$ for each possible edge. The value of the function is 1 if and only if the input graph contains a perfect matching. We prove that the perfect matching function can be computed by polynomial size span programs over arbitrary fields.

**Theorem 1.2** *PERFECT-MATCHING$_n$ can be computed by span programs of size polynomial in $n$ over arbitrary fields.*

It remains open to prove that this function requires large monotone span programs.

A different motivation for studying monotone span programs comes from a cryptographic tool called *"secret sharing schemes."* This connection is reviewed in detail by Beimel, Gál, and Paterson [BGP]. Without giving the definitions, we should mention that most known secret sharing schemes are "linear," and lower bounds for the total size of "shares" in linear secret sharing schemes are equivalent to lower bounds for monotone span programs. Our main result can therefore be interpreted as a *superpolynomial lower bound for linear secret sharing schemes.* For details we refer to the survey by Stinson [St] and to the extensive literature listed in [BGP].

The best known lower bound for *general* secret sharing schemes is $\Omega(n^2/\log n)$ (Csirmaz [Cs]). This immediately implies the same lower bound for monotone span programs for explicit functions. This by-product of [Cs] was improved by Beimel, Gál, and Paterson [BGP] to an $\Omega(n^{5/2})$ lower bound for monotone span programs; they prove this bound for the 6-clique function. (Here $n$ denotes the number of variables.) More importantly, [BGP] exhibits a combinatorial criterion which we shall be able to exploit to obtain superpolynomial lower bounds. We state our main result.

**Theorem 1.3** *There exists an explicit family of monotone Boolean functions $f_n$ in $n$ variables such that*

$$\mathrm{mSP}_K(f_n) = n^{\Omega(\log n/\log\log n)}$$

*over any field $K$.*

Our function family $\{f_n\}$ belongs to $NP$ but it is not known to belong to $P$, in fact the best algorithm we know for computing $f_n$ would run in essentially the same time as our superpolynomial lower bound for monotone span program complexity.

Our lower bounds for $f_n$ imply the same lower bounds for the size of monotone span programs computing the clique function. This follows from the fact that the clique problem is *monotone complete* for $NP$, i.e. there is a monotone projection reduction from any function computable by monotone nondeterministic circuits to the clique problem [SV, GSi].

To summarize, let $\mathrm{CLIQUE}_n$ be the function on $m = \binom{n}{2}$ variables taking value 1 if and only if the input graph on $n$ vertices contains a clique on $n/2$ vertices. We have the following corollary.

**Corollary 1.4**

$$\mathrm{mSP}_K(CLIQUE_n) = n^{\Omega(\log n/\log\log n)}$$

*over any field $K$.*

## 1.2 Paley-type bipartite graphs

Our construction of an explicit function family that requires superpolynomial size monotone span programs is based on Paley-type bipartite graphs.

Let $q$ be a prime power and $k|q-1$. We define the Paley-type bipartite graph $\Gamma = P(q,k)$ as follows. The two parts of the vertex set are $V_1 = V_2 = GF(q)$ (the finite field of order $q$); and two vertices $x \in V_1$ and $y \in V_2$ are adjacent if $(x+y)^{(q-1)/k} = 1$ (in $GF(q)$). These bipartite graphs are regular of degree $(q-1)/k$. (For this and other elementary facts about finite fields we refer the reader to Lidl–Niederreiter [LN].)

In order to construct the functions for which we are able to prove superpolynomial lower bounds, we need a set system for which one can control the sizes of $t$-wise intersections for $t = t(n) \to \infty$. (This function $t(n)$ will go in the exponent of the lower bound.) The first idea we used for constructing such set systems was based on the recent progress made by Kollár, Rónyai, and Szabó on the Zarankiewicz problem [KRS]. The set system was obtained from the Paley-type graphs with special parameters called "norm graphs" by [KRS] and analysed by them using the elements of commutative algebra. This approach resulted in a lower bound of $n^{\Omega(\sqrt{\log n/\log\log n})}$ on the monotone span program complexity of a family of explicit functions.

The result of [KRS] helped us a great deal in clarifying the combinatorial structure we needed, and inspired the next step, which yielded a stronger bound. This second approach uses a more general class of Paley-type graphs in the same way as we used the "norm graphs" but the analysis is done via Weil's character sum estimates in the spirit of the classical paper by Graham and Spencer [GS]. Constructions of $k$-wise nearly independent random variables have been analysed in a similar spirit (cf. [AGHP], [AMN]).

In this paper we only describe the second approach. For the details of the first solution we refer to [BGKRSW].

# 2 The power of span programs

## 2.1 Monotone span programs vs. monotone circuits and formulae - Proof of Theorem 1.1

Here we give the proof of Theorem 1.1, a result that may be interpreted as an indication why lower bounds for monotone span programs may be hard to come by. For the proof we need three results: the upper bound for monotone span programs, the lower bound for monotone circuits and the lower bound for monotone formulae.

We consider the following function ODDFACTOR$_n$ on $n = v^2$ variables: the input is a $v \times v$ (0,1)-matrix representing a bipartite graph $X$ with $v$ vertices in each part. $X$ is accepted if it has an *odd factor*, i. e., a spanning subgraph such that all vertices have odd degree in the subgraph. Note that $X$ is rejected exactly if it has a component with an odd number of vertices.

### 2.1.1 The upper bound for monotone span programs

**Theorem 2.1** $\mathrm{mSP}_{GF(2)}(ODDFACTOR_n) = n.$

**Proof.** Let $V = V_1 \cup V_2$ be the vertex set and let $W$ denote the $2v$-dimensional space over $GF(2)$ generated by the basis $\{u_i : i \in V\}$. The variables are $x_{i,j}$ ($i \in V_1, j \in V_2$). Let $x_{i,j}$ correspond to the one-dimensional subspace spanned by $u_i + u_j$. The root is the "all-ones" vector $w = \sum_{i\in V} u_i$. It should be clear that the root is the sum of a set of vectors of the form $u_i + u_j$ precisely if the corresponding edges $(i, j)$ form an odd factor. □

### 2.1.2 The lower bound for monotone circuits

Despite the simplicity of this span program (as well as the trivial sequential algorithm) for this function, it has close affinity to the perfect matching problem, which makes it as difficult for monotone Boolean models.

In [Ra2], Razborov proves an $n^{\Omega(\log n)}$ monotone circuit lower bound for the perfect matching function. We observe that his proof actually gives a stronger result, stated below.

Given a 2-coloring of a vertex set, we consider the graph consisting of all edges whose endpoints have the same color by the given 2-coloring. We refer to this graph as the *graph of the given 2-coloring*.

**Theorem 2.2** *(implicit in [Ra2] Lemma 7) Any monotone circuit that accepts all perfect matchings, and rejects any constant fraction of all the graphs of 2-colorings must have size* $n^{\Omega(\log n)}$. □

We use this to prove the following.

**Theorem 2.3** *Any monotone circuit computing ODDFACTOR$_n$ has size* $n^{\Omega(\log n)}$.

**Proof.** Every perfect matching is an odd factor, and should be accepted. On the other hand, the graph of every odd 2-coloring (in which each color occupies an odd number of vertices) has two odd components, and thus is rejected by our function. As odd 2-colorings constitute half of all 2-colorings, the above argument suffices. □

### 2.1.3 The lower bound for monotone formulae

For the formula size (equivalently circuit depth) lower bound we use a similar method to the one used by Raz and Wigderson in [RW], which is based on the communication complexity approach of Karchmer and Wigderson [KW1]. Define the disjointness function on a pair $x, y$ of $u$-bit vectors by DISJ $(x, y) = 1$ if and only if the sets represented by these vectors are disjoint.

**Theorem 2.4** *(*[KS, Ra4], cf. [BFS]*) Any* $1/3$ *error probabilistic communication protocol for* DISJ *requires* $\Omega(u)$ *communication bits.*

We prove the following.

**Theorem 2.5** *Any monotone formula computing ODDFACTOR$_n$ has size* $\exp(\Omega(\sqrt{n}))$.

**Proof.** We will reduce DISJ to the following communication problem ODDFACTOR$\,(m, c)$ on the set $V$ where $v = 4u$. The first player has a perfect matching $m$ from $V_1$ to $V_2$. The second player has an odd coloring of $V$. They have to compute an edge $e \in m$ which is 2-colored by $c$. As this is the monotone relation capturing ODDFACTOR$_n$, if this problem requires $t$ bits to solve deterministically, the monotone formula size of ODDFACTOR$_n$ is $\exp(\Omega(t))$.

Assuming a $t$-bit deterministic protocol here, we derive a probabilistic protocol of the same complexity for DISJ.

We shall use the following "gadget." Let $(a_1, a_2, a_3, a_4)$ and $(b_1, b_2, b_3, b_4)$ be ordered quadruples of distinct vertices in $V_1$ and in $V_2$, resp. Define two matchings

$$M_1 = \{(a_1, b_1), (a_2, b_2), (a_3, b_3), (a_4, b_4)\}$$

and

$$M_0 = \{(a_1, b_2), (a_2, b_1), (a_3, b_4), (a_4, b_3)\} .$$

Also define two 2-colorings (red is the complement of blue in each) by $B_1 = \{a_1, b_2, a_3, b_4\}$ and $B_0 = \{a_1, a_2, b_1, b_2\}$.

Now assume our players have the inputs $x$ and $y$, resp., for the disjointness problem, and they share a random string $r$. They interpret $r$ as a partition of $V$ into $u$ pairs of quadruples of vertices of the type described above, with a uniform distribution over all partitions and orderings. The player holding $x$ constructs the matching $m$ by taking from the $i$th part of the partition the matching $M_{x_i}$, for all $i \in [u]$. Similarly, the player holding $y$ constructs a 2-coloring $c'$ using $B_{y_i}$ for all $i \in [u]$. Since this coloring $c'$ is even, he then flips the color of a random vertex $w$ in $V$ to produce his final odd coloring $c$. On obtaining a bichromatic edge $e$ by the assumed protocol, the players answer 1 if and only if $w \in e$.

Observe that if $\mathrm{DISJ}\,(x, y) = 1$ then the coloring $c'$ makes each edge of the matching $m$ monochromatic, and there is exactly one bichromatic edge in $m$ under the coloring $c$. Therefore the players will make no error on disjoint pairs $(x, y)$.

On the other hand if $(x, y)$ intersect in $k$ places, then $(m, c')$ is uniformly distributed over all such pairs having $4k$ bichromatic edges. Since the protocol returning $e$ is deterministic, and $w$ is random, the players will now err with probability $\leq 1/(4k + 1) \leq 1/5$. □

## 2.2 Polynomial size span programs for matching - Proof of Theorem 1.2

In this section we prove Theorem 1.2.

We consider the function PERFECT-MATCHING$_n$ on $m = \binom{n}{2}$ variables. The input is an undirected graph on $n$ vertices, represented by Boolean variables $x_{ij}$ for each possible edge. The value of the function is 1 if and only if the input graph contains a perfect matching. The existence of polynomial size span programs for PERFECT-MATCHING over arbitrary fields follows from the following series of well known results.

The Tutte matrix of a graph $G$ is obtained from its adjacency matrix by substituting indeterminates $y_{ij}$ or $-y_{ij}$ (according to an arbitrary orientation) for the edges of $G$, and leaving the entries corresponding to non-edges 0. Tutte's theorem [Tu] states that a graph has a perfect matching if and only if its Tutte matrix is nonsingular.

Based on Tutte's theorem and Schwartz's lemma [Schw] about probabilistic testing of polynomial identities, Lovász [L] reduced the problem if a graph contains a perfect matching to testing if a given integer matrix is nonsingular.

**Theorem 2.6** *[L] Let $K$ be a field of order $> n \cdot 2^{\binom{n}{2}}$. Then there exists an $n \times n$ matrix $A$ with entries $a_{ij}$ from $K$, such that for an arbitrary undirected graph $G$ on $n$ vertices $G$ contains a perfect matching if and only if the matrix $A_G$ is nonsingular over $K$, where the entries of $A_G$ are $a_{ij} \cdot x_{ij}$, and the Boolean variable $x_{ij}$ takes the values 1 or 0 depending on whether or not the edge $(i, j)$ is present in $G$.* $\Box$

It follows from the arguments of Buntrock et al. [BDHM] that testing singularity of a variable matrix can be performed by polynomial size span programs. [BDHM] have not considered the model of span programs, but their model is equivalent to span programs up to polynomial increase in the size of the computation [KW]. Allender, Beals and Ogihara [ABO] showed explicitly how to construct polynomial size span programs for testing feasibility of systems of linear equations (Theorem 2.12 [ABO]). This gives the following.

**Theorem 2.7** *(implicit in [BDHM, ABO])*
*Let $K$ be an arbitrary field. Let $A_X$ be an $n \times n$ matrix with entries of the form $a_{ij} \cdot x_{ij}$, where $a_{ij} \in K$ and the $x_{ij}$ are Boolean variables. Then singularity of $A_X$ can be tested by span programs over $K$ of size polynomial in $n$.* $\Box$

From the above arguments it follows that if $K$ is a field of order $> n \cdot 2^{\binom{n}{2}}$ then PERFECT-MATCHING$_n$ can be computed by polynomial size span programs over $K$. Thus we also obtain polynomial size span programs over "large enough" finite fields of arbitrary characteristic, i.e. $GF(p^{q(n)})$, where $q(n)$ is a given polynomial in $n$ and $p$ is an arbitrary prime.

Finally, to get polynomial size span programs for PERFECT-MATCHING over arbitrary fields, we need the following result.

**Proposition 2.8** *(implicit in [KW], Theorem 12) Span programs of size $S$ over $GF(p^t)$ can be simulated by span programs of size $tS$ over $GF(p)$.*

**Proof.** A space of dimension $d$ over $GF(p^t)$ is a space of dimension $dt$ over $GF(p)$. The span of a set of subspaces remains the same subset of $W$ under both interpretations. $\Box$

Combining these results concludes the proof of Theorem 1.2. $\Box$

# 3 The lower bounds for monotone span programs - Proof of Theorem 1.3

Here we give the proof of our main result, Theorem 1.3. In section 3.1 we present the BGP lower bound method that we will use for our proof. In section 3.2 we show how to use bipartite graphs to obtain large families that satisfy the BGP lower bound condition. Finally, in section 3.3 we present the construction of such a family.

## 3.1 The BGP Lower Bound Condition

We shall make use of a technique introduced by Beimel, Gál, and Paterson [BGP], to prove lower bounds for monotone span programs. The idea is to show that if a small monotone span program accepts all the minterms of the function $f$ then it must also accept an input that does not contain any minterms, a contradiction. This approach can be viewed as an application of the "fusion method" [Ra3, Ka, Wi].

A *minterm* of a monotone Boolean function is a minimal set of variables which, if assigned the value 1, force the function to take the value 1 regardless of the values assigned to the remaining variables.

It will be convenient to use the language of *set systems* (hypergraphs) rather than Boolean functions. The correspondence is established by the bijection between $\{0,1\}^n$ and the power-set of a universe $X$ of $n$ elements. The Boolean function $f$ will then correspond to the set system $f^{-1}(1)$. Clearly, monotone Boolean functions correspond to *filters*, i. e., set systems closed under supersets. The minterms correspond to the minimal elements of this filter (with respect to inclusion).

A *Sperner family* is a family of sets none of which contains any other member of the family. The minimal elements of a filter form a Sperner family; and each Sperner family $\mathcal{F}$ uniquely defines a filter (consisting of all not necessarily proper supersets of the members of $\mathcal{F}$).

Given a Sperner family $\mathcal{F}$, we shall denote by $f_{\mathcal{F}}$ the corresponding monotone Boolean function.

Given a set system $\mathcal{F} \subset 2^X$ over the universe $X$, we say that a set $A \subseteq X$ *determines* a member $H \in \mathcal{F}$ if $H$ is the unique member of $\mathcal{F}$ containing $A$. Next we define a key concept introduced in [BGP].

**Definition 3.1** *We call a Sperner family $\mathcal{F}$* self-avoiding *if one can associate a set $D(H)$ with each $H \in \mathcal{F}$, called the* core *of $H$, such that*

*(i) $D(H)$ determines $H$ (with respect to $\mathcal{F}$); and*

*(ii) for any $H \in \mathcal{F}$ and any subset $T \subseteq D(H)$, the set*

$$S(T) := \bigcup_{G \in \mathcal{F}, G \cap T \neq \emptyset} G \setminus T,$$

*does not contain any member of $\mathcal{F}$. (We call $S(T)$ the* spread *of $T$.)*

(This definition is equivalent to a special case of "critical families" defined in [BGP, Sec. 2.2].)

The following result summarizes the BGP lower bound technique.

**Theorem 3.2 (Beimel, Gál, Paterson [BGP])** *Let $\mathcal{F}$ be a Sperner family over a universe of $n$ elements and $f_{\mathcal{F}}$ the corresponding monotone Boolean function in $n$ variables. If $\mathcal{F}$ is self-avoiding then for every field $K$ we have*

$$\mathrm{mSP}_K(f_{\mathcal{F}}) \geq |\mathcal{F}| \ .$$

For completeness we describe the proof from [BGP].

**Proof.** Consider a monotone span program computing $f_{\mathcal{F}}$. Let $W_i \subseteq W$ be the subspaces associated with the variables $x_i$ and let $r$ denote the sum of the dimensions of the subspaces, i.e. the size of the span program. Fix a basis for each subspace $W_i$ and arrange the basis vectors in an $r \times d$ matrix $M$, where $d$ is the dimension of $W$.

By the definition of $f_{\mathcal{F}}$, every $H \in \mathcal{F}$ must be accepted by the span program. This means that for every $H \in \mathcal{F}$ there is some vector $\mathbf{c}_H \in \mathcal{F}^r$ such that $\mathbf{c}_H \cdot M = \mathbf{w}$, where $\mathbf{w}$ is the root (as defined in the introduction), and $\mathbf{c}_H$ has nonzero coordinates only at rows of $M$ corresponding to basis vectors of the subspaces associated with elements of $H$. To prove the theorem, it is enough to show that all the vectors $\mathbf{c}_H$ for $H \in \mathcal{F}$ must be linearly independent.

Suppose that this is not the case, i.e. $\sum_{A\in\mathcal{F}} \alpha_A \mathbf{c}_A = \mathbf{0}$, and $\alpha_H \neq 0$ for some $H \in \mathcal{F}$. Let us consider the set $D(H)$ from Definition 3.1. Then the following must hold for every nonempty subset $T \subseteq D(H)$.

$$\sum_{A\in\mathcal{F}, A\cap T\neq\emptyset} \alpha_A = 0 \ . \tag{1}$$

To see this, let us consider the vector

$$\mathbf{c} = \sum_{A\in\mathcal{F}, A\cap T\neq\emptyset} \alpha_A \mathbf{c}_A \ .$$

Let us denote $\sum_{A\in\mathcal{F}, A\cap T\neq\emptyset} \alpha_A$ by $\delta$. We have $\mathbf{c} \cdot M = \delta \mathbf{w}$. If $\delta \neq 0$ this means that the program accepts the set of variables that are associated with the rows corresponding to nonzero coordinates of $\mathbf{c}$. From our assumption that the vectors $\mathbf{c}_A$ are not linearly independent, i.e.

$$\sum_{A\in\mathcal{F}} \alpha_A \mathbf{c}_A = \sum_{A\in\mathcal{F}, A\cap T\neq\emptyset} \alpha_A \mathbf{c}_A + \sum_{A\in\mathcal{F}, A\cap T=\emptyset} \alpha_A \mathbf{c}_A = \mathbf{0} \, ,$$

it follows that the coordinates of $\mathbf{c}$ are zero at rows associated with elements of $T$. Thus $\delta \neq 0$ would mean that the program accepts $S(T)$, which should be rejected.

From (1) together with the assumption that $\sum_{A\in\mathcal{F}} \alpha_A \mathbf{c}_A = \mathbf{0}$ and $\alpha_H \neq 0$ we get a system of linear equations in the unknowns $\alpha_A$ which has no solution. For proving this, let $|D(H)| = t$, and consider the $(2^t - 1) \times (2^t - 1)$ zero–one matrix $Q$ with its rows and columns indexed by the nonempty subsets of $D(H)$

such that $Q(Y,Z)=1$ if and only if $Y \cap Z \neq \emptyset$. The matrix $Q$ has full rank over any field $K$. However if our system had a solution, that would mean that the columns of $Q$ are not linearly independent, by taking $\beta_Z = \sum_{A \in \mathcal{F}, A \cap D(H)=Z} \alpha_A$ as a coefficient for the column $Z$ and observing that $\beta_{D(H)} = \alpha_H \neq 0$. $\square$

For a more detailed explanation of the above proof we refer to [BGP].

We note that if all the sets $D(H)$ for $H \in \mathcal{F}$ can be chosen from a proper subset of the variables $X' \subset X$ then the above proof yields a slightly stronger statement, i.e. in this case $|\mathcal{F}|$ also gives a lower bound on the sum of the dimensions of the subspaces associated with the variables from $X'$ only.

## 3.2 A sufficient condition for self-avoidance

We shall use the following general scheme to construct self-avoiding Sperner families.

Take a bipartite graph $\Gamma$ with vertex set $V_1 \cup V_2$.

**Notation 3.3** For a vertex $x$, we denote by $\Gamma(x)$ the set of neighbors of $x$; and for a subset $A$ of the vertex set, we denote by $\Gamma(A)$ the set of common neighbors of $A$, i. e., $\Gamma(A) = \cap_{x \in A} \Gamma(x)$. Let moreover $\Delta(A) = A \cup \Gamma(A)$.

**Notation 3.4** An *r-set* is a set of $r$ elements. For a set $X$ and an integer $r \geq 0$ we shall use $\binom{X}{r}$ to denote the *set* of all $r$-subsets of $X$.

Now fix an integer $t > 1$ and a subset $\mathcal{S} \subseteq \binom{V_1}{t-1}$. Set

$$\mathcal{F} = \mathcal{F}(\Gamma, \mathcal{S}) := \{\Delta(A) : \ A \in \mathcal{S}\}. \tag{2}$$

Let $m(t) = m(\Gamma, t) := \max\{|\Gamma(B)| : \ B \in \binom{V_1}{t}\}$.

**Lemma 3.5** *If $|\Gamma(A)| > (t-1) \cdot m(t)$ for all $A \in \mathcal{S}$ then $\mathcal{F}$ is self-avoiding.*

**Proof.** First, $\mathcal{F}$ is a Sperner family since all $A \in \mathcal{S}$ have the same cardinality $(t-1)$.

We define the core function as $D(\Delta(A)) = A$. This clearly satisfies item (i) in Definition 3.1. We need to verify item (ii).

Assume that $A \in \mathcal{S}$. Let $T \subseteq A$, and consider the *spread* $S(T)$ defined in Definition 3.1. Assume for a contradiction that $\Delta(A^*) \subseteq S(T)$ holds for some $A^* \in \mathcal{S}$.

By the definition of $S(T)$ and our construction, every vertex $y \in S(T) \cap V_2$ must be adjacent to some vertex $x(y) \in T$. This is true in particular for each $y \in \Gamma(A^*)$. Let $x_0$ be the most frequently occurring $x(y)$ for $y \in \Gamma(A^*)$; then $x_0$ is adjacent to more than $(t-1) \cdot m(t)/|T| \geq m(t)$ vertices in $\Gamma(A^*)$. In other words,

$$|\Gamma(A^* \cup \{x_0\})| > m(t),$$

in contradiction with the definition of $m(t)$ since $|A^* \cup \{x_0\}| = t$ (which in turn holds because $A^* \cap T = \emptyset$). $\square$

### 3.3 Construction of large self-avoiding Sperner families

The following well known lemma guarantees a tight control of the intersection sizes of vertex neighborhoods in the Paley-type graphs $\Gamma = P(q, k)$. (We use the notation given in the introduction.) For real numbers $b, c$ we use the notation $b \pm c$ to denote a quantity between $b - c$ and $b + c$.

**Lemma 3.6** *Let $a_1, \ldots, a_t$ be distinct elements of the finite field $GF(q)$ and let $k|q-1$ $(k, t \geq 2)$. Then the number of solutions $x \in GF(q)$ to the system of equations $(a_i + x)^{(q-1)/k} = 1$ $(i = 1, \ldots, t)$ is $q/k^t \pm t\sqrt{q}$.*

We state the relevant character sum estimate of A. Weil and show how to deduce Lemma 3.6 along the lines of [GS].

Let $\chi$ be a homomorphism of the multiplicative group $GF(q)^\times$ onto the group of $k^{\text{th}}$ roots of unity. We extend the domain of $\chi$ to $GF(q)$ by setting $\chi(0) = 0$. The function $\chi$ is called a *multiplicative character of order $k$* over $GF(q)$.

**Theorem 3.7 (A. Weil)** *Let $f(x)$ be a polynomial over $GF(q)$ which is not of the form $c \cdot (g(x))^k$ for any polynomial $g$ over $GF(q)$ and scalar $c \in GF(q)$. Let $t$ denote the number of distinct roots of $f$ in the algebraic closure of $GF(q)$. Let $\chi$ be a multiplicative character of order $k$ over $GF(q)$. Then*

$$\mid \sum_{x \in GF(q)} \chi(f(x)) \mid \leq (t-1)\sqrt{q}. \tag{3}$$

For a proof, see [Sch, p. 43, Theorem 2C]. □

Now we turn to the proof of Lemma 3.6. Let $\omega$ denote a primitive $k^{\text{th}}$ root of unity and let $g$ be a generator of the multiplicative group of $GF(q)$. Set $\chi(g^\ell) = \omega^\ell$. This is clearly a homomorphism onto the group of $k^{\text{th}}$ roots of unity; add $\chi(0) = 0$ to obtain a character of order $k$. (All characters of order $k$ arise this way.) It is clear that $\chi(g^\ell) = 1$ if and only if $k|\ell$, i. e., if and only if $x^{(q-1)/k} = 1$, where $x = g^\ell$.

Let $X$ denote the set of those $x \in GF(q)$ which simultaneously satisfy $\chi(a_i + x) = 1$ for $i = 1, \ldots, t$. Our aim is to estimate the number $N = |X|$.

Consider the polynomial $h(z) = 1 + z + \ldots + z^{k-1} = (z^k - 1)/(z - 1)$. Clearly $h(1) = k$, $h(\omega^j) = 0$ for $j = 1, \ldots, k-1$, and $h(0) = 1$. Let now $H(x) = \prod_{i=1}^{t} h(\chi(a_i + x))$. We observe that if $x \in X$ then $H(x) = k^t$; if $x = -a_i$ for some $i$ then $H(x) = 0$ or $H(x) = k^{t-1}$; and in the remaining cases, $H(x) = 0$.

Therefore the sum $S := \sum_{x \in GF(q)} H(x)$ satisfies

$$Nk^t \leq S \leq Nk^t + tk^{t-1}. \tag{4}$$

$H(x)$ is the product of sums of $k$ terms each. Let us expand the product to the sum of $k^t$ terms. Let $\Psi$ denote the set of the $k^t$ functions $\psi : \{1, \ldots, t\} \to \{0, 1, \ldots, k-1\}$ which will serve to index this sum. We have

$$S = \sum_{x \in GF(q)} \sum_{\psi \in \Psi} \prod_{i=1}^{t} (\chi(a_i + x))^{\psi(i)} = \sum_{x \in GF(q)} \sum_{\psi \in \Psi} \chi(f_\psi(x)), \tag{5}$$

where $f_\psi(x) := \prod_{i=1}^{t} (a_i + x)^{\psi(i)}$.

Let $\psi_0(i) := 0$ for all $i$; hence $f_{\psi_0}(x) = 1$ for all $x \in GF(q)$. After switching the order of summation in equation (5), let us separate the term corresponding to $\psi_0$; clearly, this term will be $q$. This is the "main term"; we need to estimate the "error term" $R := S - q$. Let $\Psi^* = \Psi \setminus \{\psi_0\}$.

$$|R| \le \sum_{\psi \in \Psi^*} \Big| \sum_{x \in GF(q)} \chi(f_\psi(x)) \Big|.$$

We note that all roots of $f_\psi$ belong to $GF(q)$, and $f_\psi$ has at least one and at most $t$ distinct roots, each with multiplicity $\le k - 1$. It follows that the conditions of Weil's theorem are satisfied and each inner sum has absolute value $\le (t-1)\sqrt{q}$. Consequently

$$|R| \le k^t (t-1)\sqrt{q}.$$

Combining this with equation (4), we obtain

$$\begin{aligned} N &= S/k^t \pm t/k &&= q/k^t + R/k^t \pm t/k \\ &= q/k^t \pm (t-1)\sqrt{q} \pm t/k &&= q/k^t \pm t\sqrt{q}. \end{aligned}$$

We assumed $t \le \sqrt{q}$ in the last step. Note that Lemma 3.6 holds vacuously for $t > \sqrt{q}$. □

Restated in our combinatorial setting, we obtain that for $A \in \binom{V_1}{t}$ we have $|\Gamma(A)| = q/k^t \pm t\sqrt{q}$.

It follows that as long as $q/k^t$ is much larger than $\sqrt{q}$, the $(t-1)$-wise intersections are almost uniformly $k$-times larger than the $t$-wise intersections. In view of Lemma 3.5, this suggests the ranges of the parameters in the following lemma.

**Lemma 3.8** *If $k \ge 3t$ and $q > 4t^4 k^{2t-2}$ then the system $\mathcal{F} = \{\Delta(A) : A \in \binom{V_1}{t-1}\}$ is self-avoiding.*

(In the notation of Section 3.2, we have chosen $\mathcal{S} = \binom{V_1}{t-1}$.)

Indeed, $m(t) \le q/k^t + t\sqrt{q}$ and for all $A \in \binom{V_1}{t-1}$, $|\Gamma(A)| \ge q/k^{t-1} - (t-1)\sqrt{q}$, both by Lemma 3.6. Combined with the assumption on the parameters, these inequalities guarantee that $|\Gamma(A)| > t \cdot m(t)$, hence Lemma 3.5 applies. □

To achieve the best lower bound, i. e., to maximize $|\mathcal{F}| = \binom{q}{t-1}$, given $q$, we need to maximize $t$ under the given constraints. Not all prime powers $q$ will allow this, only those belonging to specific arithmetic progressions. Let us select $t$ first, and then minimize $q$. W.l.o.g. assume $t$ is odd (otherwise add 1); set $k := 3t$; and select $q$ to be of the form $q := 2^{\ell\varphi(3t)}$ where $\varphi$ denotes Euler's totient function. Such a choice guarantees that $k|q-1$. Since $1 < \varphi(3t) \leq 2t$, one can clearly choose $\ell$ such that $q$ satisfies the inequalities $4t^4k^{2t-2} < q < 2^{2t} \cdot 4t^4k^{2t-2}$. With this choice, $t = \Theta(\log q/\log\log q)$, and the lower bound $|\mathcal{F}| = q^{\Theta(\log q/\log\log q)}$ on the monotone span program complexity of the monotone function in $2q$ variables, defined by $\mathcal{F}$, follows. $\Box$

It may seem that this lower bound applies to infinitely many, but not to all values of $n$. Note, however, that for every sufficiently large $n$, an appropriate $q$ can be found between $n/2$ and $\sqrt{n}$. Using our function augmented with $n-2q$ redundant variables, we still have a lower bound of the form $n^{\Theta(\log n/\log\log n)}$.

## 4 Open questions

It is not known if it is possible to obtain larger lower bounds by the BGP lower bound method.

**Problem 4.1** Do there exist exponential size self-avoiding families?

Two open questions naturally arise in connection with Theorem 1.1. The first question asks to increase the gap established in Theorem 1.1; the second asks to reverse the direction of the gap.

**Problem 4.2** Do there exist functions admitting polynomial size monotone span programs which require exponential size monotone circuits?

**Problem 4.3** Do there exist functions admitting polynomial size monotone circuits which require superpolynomial size monotone span programs?

It is not known how much monotone span programs are weaker than non-monotone span programs. In fact, the following question is open as well.

**Problem 4.4** Find a *polynomial time computable* family of monotone Boolean functions which does not admit polynomial size monotone span programs.

As we mentioned in the Introduction, our lower bound implies the same superpolynomial lower bound for linear secret sharing schemes.

**Problem 4.5** Find larger than $\Omega(n^2)$ lower bounds for general secret sharing schemes.

The fundamental question, of course, continues to be to find explicit functions which require superpolynomial size (non-monotone) span programs.

# References


[AB] N. Alon and R. Boppana: The monotone circuit complexity of Boolean functions. *Combinatorica* 7 (1987), 1–22.

[ABO] E. Allender, R. Beals, M. Ogihara: The complexity of matrix rank and feasible systems of linear equations. In *Proc. 28th ACM STOC*, 1996, pp. 161–167.

[AGHP] N. Alon, O. Goldreich, J. Håstad, R. Peralta: Simple constructions of almost $k$-wise independent random variables. *Random Structures and Algorithms* 3 (1992), 289–304.

[AMN] Y. Azar, R. Motwani, J. Naor: Approximating probability distributions using small sample spaces. *Combinatorica*, to appear.

[BFS] L. Babai, P. Frankl, J. Simon: Complexity classes in communication complexity theory. In: *Proc. 27th IEEE FOCS,* 1986, pp. 337–347.

[BGKRSW] L. Babai, A. Gál, J. Kollár, L. Rónyai, T. Szabó, A. Wigderson. Extremal bipartite graphs and superpolynomial lower bounds for monotone span programs. In *Proc. 28th ACM STOC*, 1996, pp. 603–611.

[BGP] A. Beimel, A. Gál and M. Paterson: Lower bounds for monotone span programs. In *Proc. 36th IEEE FOCS*, Milwaukee WI 1995, pp. 674–681, journal version to appear in *Computational Complexity*.

[Ber] S. J. Berkowitz: On computing the determinant in small parallel time using a small number of processors. *Inform. Process. Lett.* 18 (1984), 147–150.

[Bol] B. Bollobás: *Extremal Graph Theory.* Academic Press, 1978.

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

[Cs1] L. Csirmaz: The size of a share must be large. In A. De Santis, ed., *Advances in Cryptology – Eurocrypt'94, pre-proceedings*, 1994.

[Cs] L. Csirmaz: The dealer's random bits in perfect secret sharing schemes. Preprint, Mathematical Inst. Hungarian Acad. Sci., 1994.

[GS] R. L. Graham, J. H. Spencer: A constructive solution to a tournament problem. *Canad. Math. Bull.* 14 (1971), 45–48.

[GSi] M. Grigni, M. Sipser: Monotone Complexity. In *Boolean function complexity*, edited by M. Paterson, London Math. Society Lecture Note Series, 169, Cambridge University Press, 1992, pp. 57–75.

[Ha] Armin Haken: Counting bottlenecks to show monotone P$\neq$NP. In *Proc. 36th IEEE FOCS*, Milwaukee WI 1995, pp. 36–40.

[KS] B. Kalyanasundaram and G. Schnitger: The probabilistic communication complexity of set intersection. In *Proc. of Structure in Complexity Theory*, 1987, pp. 41-49.

[Ka] M. Karchmer: On proving lower bounds for circuit size. In *Proc. 8th Ann. Symp. Structure in Complexity Theory*, IEEE 1993, pp. 112–118.

[KW] M. Karchmer and A. Wigderson: On span programs. In *Proc. 8th Ann. Symp. Structure in Complexity Theory*, IEEE 1993, pp. 102–111.

[KW1] M. Karchmer and A. Wigderson: Monotone Circuits for Connectivity require Super-Logarithmic Depth. In *SIAM Journal on Discrete Mathematics*, Vol 3, No. 2, 1990, pp. 255-265.

[KRS] J. Kollár, L. Rónyai, T. Szabó: Norm-graphs and bipartite Turán numbers. *Combinatorica*, to appear.

[L] L. Lovász: On determinants, matchings and random algorithms. *Fundamentals of Computing Theory*, Akademia-Verlag, Berlin, (1979).

[LN] R. Lidl, H. Niederreiter: *Introduction to Finite Fields and their Applications*, Cambridge University Press, 1986.

[Mu] K. Mulmuley: A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica* 7 (1987), 101–104.

[RW] R. Raz and A. Wigderson: Monotone Circuits for Matching require Linear Depth. *Journal of the ACM*, Vol 39, 1992, pp. 736-744.

[Ra1] A. A. Razborov: Lower bounds for the monotone complexity of some Boolean functions. *Soviet Math. Doklady*, 31 (1985) 354–357.

[Ra2] A. A. Razborov: A lower bound on the monotone network complexity of the logical permanent. *Mat. Zametki* 37 (1985), 887-900.

[Ra3] A. A. Razborov: On the method of approximation. In *Proc. 21st ACM STOC*, 1989, pp. 167–176.

[Ra4] A. A. Razborov: On the distributional complexity of disjointness. *Theoretical Computer Science*, 106, 1992, No. 2, pp. 385-390.

[Sch] W. M. Schmidt: *Equations over Finite Fields: An Elementary Approach.* Lect. Notes in Math. vol. 536, Springer Verlag, 1976.

[Schw] J. T. Schwartz: Fast probabilistic algorithms for verification of polynomial identities JACM, 27(4), 1980, pp. 701-717.

[SV] S. Skyum, L.G. Valiant: A complexity theory based on Boolean algebra. JACM, 32(2), 1985, pp. 484–502.

[St] D. R. Stinson: An explicitation of secret sharing schemes. *Design, Codes and Cryptography* 2 (1992), 357–390.

[T] É. Tardos: The gap between monotone and non-monotone circuit complexity is exponential Combinatorica, 7(4), 1987, pp. 141-142.

[Tu] W. T. Tutte: The factorization of linear graphs *J. London Math. Soc.* 22, 1947, pp. 107-111.

[We] I. Wegener: *The Complexity of Boolean Functions.* Wiley-Teubner 1987.

[Wi] A. Wigderson: The fusion method for lower bounds in circuit complexity. In: *Combinatorics, Paul Erdős is Eighty*, (Volume 1), D. Miklós, V. T. Sós, T. Szőnyi, eds., Bolyai Society Mathematical Studies 1, Budapest 1993, pp. 453–467.