LOWER BOUND ON WEIGHTS OF LARGE DEGREE THRESHOLD FUNCTIONS[*]

VLADIMIR V. PODOLSKII

Steklov Mathematical Institute, Moscow
e-mail address: podolskii@mi.ras.ru

ABSTRACT. An integer polynomial p of n variables is called a *threshold gate* for a Boolean function f of n variables if for all $x \in \{0,1\}^n$ $f(x) = 1$ if and only if $p(x) \geqslant 0$. The *weight* of a threshold gate is the sum of its absolute values.

In this paper we study how large a weight might be needed if we fix some function and some threshold degree. We prove $2^{\Omega(2^{2n/5})}$ lower bound on this value. The best previous bound was $2^{\Omega(2^{n/8})}$ (Podolskii, 2009).

In addition we present substantially simpler proof of the weaker $2^{\Omega(2^{n/4})}$ lower bound. This proof is conceptually similar to other proofs of the bounds on weights of nonlinear threshold gates, but avoids a lot of technical details arising in other proofs. We hope that this proof will help to show the ideas behind the construction used to prove these lower bounds.

1. INTRODUCTION

Let $f\colon \{0,1\}^n \to \{0,1\}$ be a Boolean function. *A threshold gate* for the Boolean function f is an integer polynomial $p(x)$ of n variables $x = (x_1, \ldots, x_n)$ such that for any $x \in \{0,1\}^n$ we have $f(x) = 1$ if and only if $p(x) \geqslant 0$. In other words, for all $x \in \{0,1\}^n$ it is true that $f(x) = \operatorname{sgn} p(x)$, where we adopt the following definition of the sign function: $\operatorname{sgn}(t) = 1$ if $t \geqslant 0$ and $\operatorname{sgn}(t) = 0$ otherwise.

Thus, threshold gates are just representations of Boolean functions as the signs of the polynomials. The formal study of such representations started in 1968 with the seminal monograph of Minsky and Papert [9]. Since then representations of this form found a lot of applications in circuit complexity, structural complexity, learning theory and communication complexity (see, for example [17, 7, 2, 20]).

Two key complexity measures of threshold gates are their degree and their weight. *The degree* $\deg p$ of a threshold gate p is just the degree of the polynomial. *The weight* $W(p)$ of a threshold gate p is the sum of absolute values of all its coefficients.

The complexity measures of a Boolean function f related to these complexity measures of threshold gates are the minimal threshold degree of a threshold gate for f which we denote by $\deg_\pm f$ and call *the threshold degree* and the minimal weight of a threshold gate for f. Both of these complexity measures play an important role in theoretical computer science (see the references above). In this paper we are interested in the minimal possible value of the weight of a threshold gate for some function f when the degree of the threshold gate is bounded. It is convenient to denote by $W(f, d)$ the minimal weight of a threshold gate of degree at most d for f. Note that this value is defined only if $d \geqslant \deg_\pm f$. It is also not hard to see that for all f we have $\deg_\pm f \leqslant n$ and $W(f, n) \leqslant 2^{O(n)}$ (just consider the polynomial p such that $p(x) = f(x)$ for all x).

The first results on the value of $W(f, d)$ were proven for $d = 1$. In [11] (see also [10] and [6]) it was proven that for all f with $\deg_\pm f = 1$ it is true that $W(f, 1) = n^{O(n)}$. For a long time only lower bounds of the form $W(f, 1) = 2^{\Omega(n)}$ were known (see [12] for one of the early results). Tight lower bound was proven in [6], that is the function f with $\deg_\pm f = 1$ was constructed such that $W(f, 1) = n^{\Omega(n)}$.

Concerning higher degree d, upper bound can be easily extended from the case $d = 1$. Namely for all f with $\deg_\pm f \leqslant d$ it is true (and easy to see) that $W(f, d) = n^{O(dn^d)}$ (see [18, 3, 15]). Note that this upper bound is much worse than for the case $d = 1$. Concerning the lower bounds, it is rather easy to see that the bound $n^{\Omega(n)}$ can be translated from the case $d = 1$ to arbitrary d (see the discussion preceding Theorem 1.2). The first lower bound improving this was given in [15] and showed that the upper bound stated above is actually tight for constant d. That is, for any constant d the function f of the threshold degree d was constructed in [15] such that $W(f, d) = n^{\Omega(n^d)}$ (constant in Ω here depends on d). It is implicit in [15] though that the argument works for nonconstant d also and the resulting lower bound (with the dependence on d) is

$$\left(\frac{n}{d}\right)^{\frac{1}{2}(\frac{n}{2d})^d - o((\frac{n}{d})^d)}. \tag{1.1}$$

For this result another proof was given in [1]. Some other results on large degree threshold gates which are not directly connected to the problem we consider have appeared in [5, 2, 19, 14, 16, 4].

Thus it turns out that the required weight grows with the growth of the degree d. In this paper we are interested in how large it might grow (note that for $d = n$ the weight is small again: $W(f, n) = 2^{O(n)}$). That is we study the value

$$W = \max_d \max_{f\colon \deg_\pm(f) \leqslant d} W(f, d).$$

The lower bound (1.1) works even for d depending linearly on n and so gives doubly exponential lower bound on this value. But it works only for $d \leqslant (n - c)/32$, where c is some constant, so the best lower bound we get from [15] is $2^{\Omega(2^{n/8})}$.

In this paper we prove the following bound.

Theorem 1.1. $W \geqslant 2^{\Omega(2^{2n/5})}$.

We note that the best upper bound known is simple $2^{O(n2^n)}$ (this can be deduced from the upper bound for the case $d = 1$ and the fact that there are at most 2^n monomials).

To prove our lower bound we adopt the strategy of [15] and provide a unified treatment of the argument of that paper. In short, the proof strategy is as follows. Starting from some function of the threshold degree 1 (with some additional properties) that requires large weight when represented by degree-1 threshold gates we construct its "d-dimensional" generalization in a very specific way. For this generalization we are able to prove a lower bound for degree-d threshold gates and due to the specific features of our generalization we can prove a strong lower bound.

In the paper [15] the construction of the function starts with the function constructed by Håstad in [6] to prove the optimal lower bound for the case $d = 1$. This helps to get n in the base of the exponent in the lower bound and thus to prove a strong lower bound for the case of constant d. On the other hand, Håstad's function is very complicated and has desired properties only for large enough number of variables (16 variables). This does not allow us to prove a lower bound for d close to n. In this paper we start with a much simpler functions having required properties starting from just 3 variables. With this function we cannot get n in the base of the exponent, but on the other hand we are able now to prove bounds for much larger d and thus to get better lower bound on W.

We start exposition of our result by giving a simpler proof of the weaker bound of $2^{\Omega(2^{n/4})}$. In this proof we are able to avoid a lot of technical complications arising in the proof of [15] and make the function for which we prove the bound much simpler (here we use as a starting function of the threshold degree 1 well known "greater than" function). We hope that this makes the proof easier to read and helps to show the ideas behind the construction which were not very clear in [15].

After that we define another starting function and explain how to change the proof to get $W \geqslant 2^{\Omega(2^{2n/5})}$ lower bound. The idea here is not only that we can prove the bound for larger d, but also that, roughly speaking, choosing the good function we can remove the constant 2 from the denominator of the term $\left(\frac{n}{2d}\right)$ in the exponent in the bound (1.1).

Besides representation of Boolean functions as $f\colon \{0,1\}^n \to \{0,1\}$, also representation of the form $f\colon \{-1,+1\}^n \to \{-1,+1\}$ turns out to be useful in complexity theory. Here -1 corresponds to "true" and 1 corresponds to "false". For this representation we can also consider threshold gates and also define corresponding measures of the functions. Note that we can switch from one representation to another one by a simple linear transform. Thus the threshold degree of the function does not depend on the representation and the threshold weight may change only by the 2^n multiplicative factor (see [8] for more information on the relations between the threshold weights in these two settings). Since this factor is very small compared to our lower bound, our result is true for both representations and in the proof we can choose the one of two presentations of Boolean functions which is more convenient to us. For the proof of the weaker bound we will use $\{0,1\}$ variables and for the stronger one — $\{-1,1\}$ variables.

We note in the conclusion that if we have the lower bound S on the minimal weight for the function of n variables and for the degree d, it is easy to translate it to exactly the same bound S for $n' = n + c$ and $d' = d + c$ for any c, even depending on n (see, for example, [16], Corollary 1). This observation allows us to deduce strong lower bounds on weights of threshold gates of degree close to n.

Theorem 1.2. *For any $\epsilon > 0$ and $d \leqslant (1 - \epsilon)n$ there is an explicit function f such that $W(f, d) = 2^{2^{\Omega(n)}}$. For any $d \leqslant n - 2(1 + \epsilon) \log n$ there is an explicit function f such that $W(f, d) = 2^{\Omega(n^{1+\epsilon})}$.*

The rest of the paper is devoted to the formulation and the proof of our results. In Sections 2 and 3 we give a simple proof for the weaker bound: in the former we construct the function for which in the latter we prove the lower bound. In Section 4 we explain how to change the proof to give the stronger bound.

2. Construction of the Function

In this section we present the construction of the function for which we prove a weaker form of our bound. Our function is the generalization of the GT function.

Definition 1. For Boolean $x, y \in \{0, 1\}^k$ let $\mathrm{GT}(x, y) = 1$ iff $x \geqslant y$, where $x = (x_1, \ldots, x_k)$ and $y = (y_1, \ldots, y_k)$ are considered as binary representations of integers with x_k and y_k being the most significant bits.

Our function will depend on $n = 2m$ variables

$$(x, y) = (x_1, \ldots, x_m, y_1, \ldots, y_m) \in \{0, 1\}^n.$$

Let us fix some $k_1, \ldots, k_d$ such that $\sum_{i=1}^{d} k_i = m$ and partition the input variables x and y in d groups of size $k_1, \ldots, k_d$, that is

$$(x, y) = (x^1, x^2, \ldots, x^d, y^1, y^2, \ldots, y^d),$$

where $x^i, y^i \in \{0, 1\}^{k_i}$ for all i.

Let us denote by $[k]$ the set $\{1, \ldots, k\}$. Let us denote by $<_1$ the following ordering of the set $[k]$: $1, 2, 3, \ldots, k - 1, k$, and by $<_0$ the reverse ordering: $k, k - 1, k - 2, \ldots, 2, 1$. We will use these orders on the sets $[k_1], \ldots, [k_d]$. It will always be clear from the context which set we consider.

Let us denote by $\mathrm{num}_i l$ the ordinal number of $l \in [k]$ w.r.t. the order $<_i$.

To define our function we need to define a specific order on the set $K = [k_1] \times \ldots \times [k_d]$. The construction below is essentially the same as in [15]. Our order will be similar to the lexicographic one, that is to compare two tuples from K we will compare their components one by one until we find the difference. But as opposed to the lexicographic order, where each component of the tuples is compared w.r.t. the same ordering, in our order of the tuples components might be compared w.r.t. different orderings. Moreover, the ordering in which we compare the current component depends not only on the ordinal number of the component but on the values of previous components of the tuples.

Formally, suppose we want to compare tuples $\alpha = (\alpha_1, \ldots, \alpha_d) \in K$ and $\beta = (\beta_1, \ldots, \beta_d) \in K$. First we compare α_1 and β_1 w.r.t. the ordering $<_1$. If they are not equal then we have already compared the tuples: the larger the first component is the larger the tuple is. If they are equal we proceed to the second components. To compare them we use the following recursive rule to choose the next order.

Assume that the order $<_{i_l}$ to compare the lth components of the tuples is already determined and it happens that $\alpha_l = \beta_l$. The order to compare $(l + 1)$st components is determined by the ordinal number of α_l (which coincides with β_l by the assumption) w.r.t. the order $<_{i_l}$. Namely,

$$i_{l+1} = \mathrm{num}_{i_l} \alpha_l \pmod 2. \tag{2.1}$$
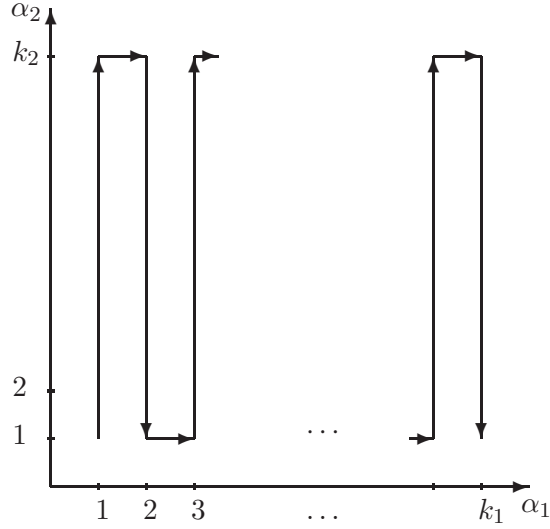
Case $d = 2$

In other words, we compare the $(l + 1)$st coordinates w.r.t. the order $<_0$ if α_l has even ordinal number w.r.t. the order $<_{i_l}$ and we compare $(l + 1)$st coordinates w.r.t. the order $<_1$ otherwise.

To say it the other way, we associate with the coordinates of any tuple $\alpha = (\alpha_1, \ldots, \alpha_d) \in K$ the orders $<_{i_1}, \ldots, <_{i_d}$ according to the rule (2.1). We use these orders to compare the coordinates of α with the coordinates of other tuples. Note that for two tuples $\alpha = (\alpha_1, \ldots, \alpha_d)$ and $\beta = (\beta_1, \ldots, \beta_d)$ the orders corresponding to their components coincides until we meet the first difference. After the first difference the orders corresponding to the components might be different in α and β but we do not need to compare the coordinates any further.

Let us denote by $\mathrm{num}_{\alpha,l}\alpha_l$ the ordinal number of the lth component of α w.r.t. the corresponding order.

It is not hard to describe the order we constructed in the case $d = 2$. Let us represent the pairs (α_1, α_2) by the points on the plane (see Figure).

The first component α_1 is associated with the horizontal axis, and the second component α_2 – with the vertical axis. The arrows indicate the direction from the smaller pairs to the larger ones.

In the case $d > 2$ the constructed order is not so easy to describe. However, as our proof goes by induction, we always consider only two consecutive coordinates of the tuple. Thus we will be in the situation that is very similar to the case $d = 2$. The only difference is that the order on the first coordinate might be different from $<_1$.

Now we can define our function.

Definition 2. For a given $(x, y) = (x^1, x^2, \ldots, x^d, y^1, y^2, \ldots, y^d)$, where $x^i = (x_1^i, \ldots, x_{k_i}^i), y^i = (y_1^i, \ldots, y_{k_i}^i) \in \{0, 1\}^{k_i}$ let $\alpha = (\alpha_1, \ldots, \alpha_d) \in K$ be the largest tuple w.r.t. the introduced order such that $\prod_{i=1}^{d}(x_{\alpha_i}^i - y_{\alpha_i}^i) \neq 0$. Then let

$$f(x^1, \ldots, x^d, y^1, \ldots, y^d) = \mathrm{sgn}\prod_{i=1}^{d}(x_{\alpha_i}^i - y_{\alpha_i}^i).$$

If there is no such α let $f(x^1, \ldots, x^d, y^1, \ldots, y^d) = 1$.

Note that if $d = 1$ our function is exactly the GT function.

3. $2^{\Omega(2^{n/4})}$ Lower Bound

First we note that our function is computable by a degree d threshold gate.

Lemma 3. $\deg_\pm(f) \leqslant d$.

Proof. Let
$$\alpha^1, \alpha^2, \ldots, \alpha^{|K|}$$
be the list of all elements of K written w.r.t. the introduced order. For any $\alpha^j = (\alpha^j_1, \ldots, \alpha^j_d)$ let t_j be the product
$$t_j = \prod_{i=1}^{d} (x^i_{\alpha^j_i} - y^i_{\alpha^j_i}).$$
Note that for any j and any input (x, y) we have $|t_j| \leqslant 1$.

Consider the polynomial
$$p(x, y) = \sum_{j=1}^{|K|} 2^j t_j.$$
We claim that it sign-represents our function.

Indeed, let us fix some input (x, y). If for this input all t_j are zeros then $f(x, y) = 1$ and $p(x, y) = 0$. Otherwise, let α^j be maximal w.r.t. the introduced order such that $t_j \neq 0$. Then all t_i for $i > j$ are zeros, and the coefficients of p are chosen in such way that the contribution of t_j is greater than the sum of contributions of all t_i for $i < j$. Thus the sign of $p(x, y)$ is the same as the sign of t_j. ∎

Now we proceed to the main result of this section.

Theorem 3.1. *Let $k_i \geqslant 2$ be even for $i < d$ and $k_d \geqslant 3$. Then*
$$\deg_\pm(f) = d$$
$$W(f, d) \geqslant 2^{(k_d - 2) \prod\limits_{i=1}^{d-1} k_i \ - d}$$

Remark. We can state an analogous theorem for arbitrary $k_i \geqslant 2$ for $i < d$ and not only for even. However, with this assumption the proof and the bound are cleaner and at the same time the theorem still gives $2^{\Omega(2^{n/4})}$ bound.

First we give some corollaries of Theorem 3.1 and then proceed to the proof.
If we let $k_i = 2$ for $i < d$ and $k_d = 3$ in Theorem 3.1 we get
$$W \geqslant W(f, \frac{n-2}{4}) \geqslant 2^{2^{\frac{n-6}{4}} - d}.$$
More generally,

Corollary 4. *For all n and all $d \leqslant \frac{n-2}{4}$ there exists (explicit) function f of n variables such that*
$$W(f, d) \geqslant 2^{(2\lfloor \frac{n-2}{4d} \rfloor)^d - (2\lfloor \frac{n-2}{4d} \rfloor)^{d-1} - d}.$$

Proof. Let $k_i = 2\lfloor \frac{n-2}{4d} \rfloor$ for $i < d$, $k_d = 2\lfloor \frac{n-2}{4d} \rfloor + 1$, consider the function from Theorem 3.1 on the first $2\sum_{i=1}^{d} k_i$ variables and leave other variables inessential. ∎

Analogously we can prove other bounds. For example we can prove the following bound with a simpler formulation

$$W(f, d) \geqslant 2^{(2\lfloor \frac{n-4}{4d} \rfloor)^d - d}.$$

3.1. **Proof of Theorem 3.1.** Let us consider an arbitrary threshold gate p for f of the degree at most d. That is, for any $x, y \in \{0, 1\}^m$ we have

$$f(x, y) = \operatorname{sgn}(p(x, y)).$$

It will be convenient for us to work in variables

$$u_j^i = x_j^i - y_j^i, \qquad v_j^i = x_j^i + y_j^i. \tag{3.1}$$

So after substituting

$$x_j^i = (u_j^i + v_j^i)/2, \qquad y_j^i = (u_j^i - v_j^i)/2$$

and multiplying the polynomial by 2^d to make the coefficients integer we obtain the polynomial p' in the variables u_j^i, v_j^i that sign-represents f. That is

$$f(x, y) = \operatorname{sgn}(p'(x - y, x + y)).$$

It is easy to see that the weight of the new polynomial is almost the same as the weight of p (compared to the value of our bound). Namely, we have the following bound.

Lemma 5. $W(p') \leqslant 2^d W(p)$.

Proof. Consider one monomial g of p of the degree $l \leqslant d$. We can think of the transformation above as of substituting each variable by the sum (or the difference) of two variables and then multiplying everything by 2^{d-l}. After opening the brackets we will have 2^l new monomials each of the weight 2^{d-l}, so the overall weight of the new monomials coming from the monomial g is 2^d times the coefficient of g. Since this happens for all monomials, the upper bound on the weight of the polynomial p' as stated follows. □

Remark. A similar bound holds in the other direction too, but we do not need it.

Now we have to prove that

$$W(p') \geqslant 2^{(k_d - 2) \prod_{i=1}^{d-1} k_i}.$$

First we will prove that we can assume that p' has a nice structure. Lemmas similar to the next one appeared in [6, 15] (see [16] for a more general version).

Lemma 6. *If we substitute by 0 all coefficients of the monomials of p' in which the variables from one of the groups $u^1, \ldots, u^d$ do not appear, the resulting polynomial q will also sign-represent f.*

Proof. Let I be the set of inputs (x, y) to f such that $x^i \neq y^i$ for all $i = 1, \ldots, d$. This condition implies that for each such input there is an $\alpha = (\alpha_1, \ldots, \alpha_d) \in K$ such that

$$\prod_{i=1}^{d} (x_{\alpha_i}^i - y_{\alpha_i}^i) \neq 0.$$

This means that the value of the function is determined by the sign of this product for the largest such α.

Consider an arbitrary input (x, y) from I and consider an arbitrary $i \in [d]$. Let us permute the variables x^i with the variables y^i. It is not hard to see that after such permutation the function f changes sign, and so p' also should change sign.

Now let us see what happens with $p'(u, v)$. Note, that after this permutation all variables u and v remains the same except u^i which changes sign. Let us denote by A the sum of all monomials of p' which contain even number of variables from u^i and by B the sum of all monomials of p' which contain odd number of variables from u^i, so

$$p'(u, v) = A(u, v) + B(u, v).$$

Note that after our permutation of variables A remains the same and B changes sign. Since p' changes sign we have that the absolute value of B is greater then the absolute value of A. We proved that this happens for any input from I, so the sign of f is determined by the sign of B for all such inputs. So we can erase all monomials from A and the resulting polynomial will still sign-represent f for all inputs from I.

Repeating this argument for all $i \in [d]$ we obtain a threshold gate q for f such that each monomial of q contains odd number of variables from each of the groups $u^1, \ldots, u^d$. But the degree of q is at most d, so it is only possible that each monomial of q contains one variable from each of the groups $u^1, \ldots, u^d$ (and no variables from $v^1, \ldots, v^d$).

We have proved that the new polynomial q sign-represents f correctly for all inputs from I. Now note, that for any input from $\{0,1\}^n \setminus I$ there is an $i \in [d]$ such that $x^i = y^i$, or equivalently, all variables from u^i are zeros. By the definition the value of the function f on such input is 1 and the value of q is 0, so q sign-represents f for these inputs also. $\square$

As a byproduct of the proof of this lemma we have the following corollary.

Corollary 7. $\deg_{\pm}(f) = d$.

Since $W(p') \geqslant W(q)$ it is enough to prove that

$$W(q) \geqslant 2^{(k_d - 2) \prod_{i=1}^{d-1} k_i}.$$

Now we need a lemma concerning degree 1 threshold gates for GT. The argument is quite standard (see [12, 13]).

Lemma 8. Let $p = \sum_{i=1}^{k} w_i u_i$ be a degree 1 threshold gate for $\mathrm{GT}(x, y)$ where $x, y \in \{0,1\}^k$. Then for $j \geqslant 2$

$$w_j \geqslant 2^{j-2} w_1 > 0 \tag{3.2}$$

and

$$w_j \geqslant w_{j-1}. \tag{3.3}$$

Proof. We will actually prove that for each j $w_j > 0$ and for each $j \geqslant 2$

$$w_j \geqslant \sum_{i=1}^{j-1} w_i. \tag{3.4}$$

The inequality (3.3) follows from this immediately and the inequality (3.2) is easy to prove by induction.

To prove the inequality (3.4) we consider the specific input

$$u = (\overbrace{-1, \ldots, -1}^{j-1}, 1, 0 \ldots, 0).$$

It is easy to see that on this input the function GT is 1, and this means that $p(u) \geqslant 0$. It is easy to see that the inequality (3.4) follows.

To prove that $w_j > 0$ just let $u_j = -1$ and $u_i = 0$ for $i \neq j$. GT is zero for such input, so $p(u) < 0$. $\qquad\square$

It will be convenient for us to consider two variants of the GT function: we denote by GT_1 the usual GT function and by GT_0 the analogous function, but now on the reversed input. That is, $GT_0(x, y) = 1$ if and only if $x \geqslant y$, where $x = (x_1, \ldots, x_k), y = (y_1, \ldots, y_k)$ are considered as binary representations of integer numbers where the most significant bits are x_1, y_1. It is easy to see that if

$$p = \sum_{i=1}^{k} w_i u_i$$

is a threshold gate for GT_0 then we have

$$w_{j-1} \geqslant w_j$$

and

$$w_{n-j+1} \geqslant 2^{j-2} w_n > 0,$$

where $j = 2, \ldots n$.

Now we can prove the main lemma.

Lemma 9. *For all $l \leqslant d$ if $\alpha \in K$ is such that $\mathrm{num}_{\alpha,i}\alpha_i = 1$ for all $i \geqslant l$ and $\beta = (\alpha_1, \ldots, k_l - \alpha_l + 1, \ldots, \alpha_d)$. Then*

$$w_\beta \geqslant w_\alpha 2^{(k_d - 2) \prod_{i=l}^{d-1} k_i}.$$

The idea is the following: we fix variables in all groups $u^1, \ldots, u^d$ except one. Then the function f becomes essentially the GT function and we can apply Lemma 8. Repeating this trick we can accumulate the large factor due to the specific construction of our order. More specifically the proof goes by the induction on the decreasing l. For the base of the induction $l = d$ we fix all variables except u^d and applying inequality (3.2) immediately obtain the desired result. For the induction step we first apply the induction hypothesis to $l + 1$ and then apply inequality (3.3) to the lth coordinate. Then we can again apply induction hypothesis to $l + 1$ and so forth. In this way we can apply induction hypothesis k_l times and obtain the desired result. We proceed to the detailed proof.

Proof. The proof goes by the induction on the decreasing l.

The base of induction $l = d$. We fix all variables u^i except u^d in the following way. For any i let $u^i_{\alpha_i} = 1$ and $u^i_j = 0$ for all $j \neq \alpha_i$.

Now we have a function in variables u^d and it is not hard to see that this function coincides with either GT_0, or GT_1. Applying the inequality (3.2) (or the corresponding inequality for GT_0) for $j = k_d$ we obtain

$$w_\beta \geqslant 2^{k_d - 2} w_\alpha.$$

Induction step. To show the lemma for $l < d$ we repeat several times the following procedure consisting of two steps. After the ith application of the procedure we will get a tuple α^i such that $\mathrm{num}_{\alpha^i,l}\alpha^i_l = i + 1$ and $\mathrm{num}_{\alpha^i,j}\alpha^i_j = 1$ for all $j > l$. To unify the notation we denote $\alpha^0 = \alpha$.

During the procedure we will not change the values of the first $l - 1$ coordinates. This means that for all tuples we consider, the order corresponding to the lth coordinate is the same. Let us assume without loss of generality that this order is $<_1$. This in particular means that $\alpha_l = 1$.

We also will not change the coordinates $\alpha_{l+2}, \ldots, \alpha_d$. Note, that this means that if $\mathrm{num}_{\alpha^i,l+1}\alpha^i_{l+1}$ is odd (as in the beginning) then all

$$\mathrm{num}_{\alpha^i,l+2}\alpha^i_{l+2}, \ldots, \mathrm{num}_{\alpha^i,d}\alpha^i_d$$

are equal to 1.

Step 1. We apply the induction hypothesis for the coordinate $l + 1$. We have that for $\widetilde{\alpha}^{i+1} = (\alpha_1, \ldots, \alpha^i_l, k_{l+1} - \alpha^i_{l+1} + 1, \ldots, \alpha_d)$

$$w_{\widetilde{\alpha}^{i+1}} \geqslant w_{\alpha^i} 2^{(k_d-2)\prod_{i=l+1}^{d-1} k_i}.$$

Note that now the ordinal number of $l + 1$st coordinate (w.r.t. the corresponding order) is k_{l+1}.

Step 2. We fix all variables u^j except u^l in the following way: for any j let $u^j_{\widetilde{\alpha}^{i+1}_j} = 1$ and $u^j_m = 0$ for all $m \neq \widetilde{\alpha}^{i+1}_j$. Now we have a function in the variables u^l and it is not hard to see that this function coincides with GT_1 (this happens because we agreed that the order corresponding to the lth component is $<_1$, if it were $<_0$ we would have GT_0 here). We apply the inequality (3.3) to the coordinate l. After that we get

$$\alpha^{i+1} = (\alpha_1, \ldots, \alpha_l + i + 1, k_{l+1} - \alpha^i_{l+1} + 1, \ldots, \alpha_d)$$

such that

$$w_{\alpha^{i+1}} \geqslant w_{\widetilde{\alpha}^{i+1}}.$$

Due to the rule (2.1) defining the order on the next component of the tuple we have that the order on the $l + 1$st component changes. This means that the ordinal number of the $l + 1$st coordinate w.r.t. the corresponding order is again 1. ¿From this we have, as we stated above, that for all $j \geqslant l + 1$ it is true that $\mathrm{num}_{\alpha^{i+1},j}\alpha^{i+1}_j = 1$. So now we are again in the position to apply Step 1.

We repeat these two steps until the lth coordinate of $\widetilde{\alpha}^i$ reaches k_l (in the end we repeat Step 1, we are unable to repeat Step 2 since the lth coordinate is already k_l and can not be increased, so in the end we get $\widetilde{\alpha}^i$ for suitable i). Since the lth coordinate increases by 1 at each iteration (on Step 2) we can repeat Step 2 $k_l - 1$ times and Step 1 k_l times. Thus in the end we get the vector $\widetilde{\alpha}^{k_l}$ and it is easy to see that

$$w_{\widetilde{\alpha}^{k_l}} \geqslant w_\alpha \prod_{j=1}^{k_l} 2^{(k_d-2)\prod_{i=l+1}^{d-1} k_i} = w_\alpha 2^{(k_d-2)\prod_{i=l}^{d-1} k_i}.$$

Let us check that $\beta = \widetilde{\alpha}^{k_l}$. It is easy to see that during this process only the coordinates l and $l + 1$ of α change. In the end of the process coordinate l has the number k_l w.r.t. the order corresponding to the lth coordinate of α. This means that $\widetilde{\alpha}^{k_l}_l = k_l - \alpha_l + 1 = \beta_l$. Note also that from this and from the evenness of k_l we have that α_l and $\widetilde{\alpha}^{k_l}_l$ define different

orders on the next coordinates. Now let us see what happens with the coordinate $l + 1$. After the process the ordinal number of $\widetilde{\alpha}_{l+1}^{k_l}$ w.r.t. the order corresponding to the $(l+1)$st component of $\widetilde{\alpha}^{k_l}$ is equal to k_{l+1}. This means that the ordinal number of $\widetilde{\alpha}_{l+1}^{k_l}$ w.r.t. the other ordering is 1. Since orders corresponding to the $(l+1)$st coordinates of α and $\widetilde{\alpha}^{k_l}$ are different we have that $\widetilde{\alpha}_{l+1}^{k_l} = \alpha_{l+1} = \beta_{l+1}$ (recall that $\mathrm{num}_{\alpha,l+1}\alpha_{l+1} = 1$). $\qquad \square$

It is easy to prove Theorem 3.1 now. Applying Lemma 9 with $l = 1$ we get

$$w_\beta \geqslant w_\alpha 2^{(k_d - 2)\prod_{i=1}^{d-1} k_i}.$$

Now, it is easy to see that $w_\alpha > 0$ (just substitute $u_{\alpha_1}^1 = -1$, $u_{\alpha_i}^i = 1$ for all $i \neq 1$ and $u_j^i = 0$ for all i and all $j \neq \alpha_i$). We conclude that $w_\alpha \geqslant 1$ and

$$w_\beta \geqslant 2^{(k_d - 2)\prod_{i=1}^{d-1} k_i}.$$

4. Improved Lower Bound

In this section we improve the argument of the previous sections to obtain the better lower bound. More precisely we prove

$$W \geqslant 2^{\Omega(2^{2n/5})}.$$

We will work with Boolean variables $\{-1, +1\}$, so we change the definition of the sgn-function: $\mathrm{sgn}(x) = 1$ if $x \geqslant 0$ and $\mathrm{sgn}(x) = -1$ otherwise.

The idea is to use another function instead of GT as a building block in our construction. Indeed, we may try to carry out the proof if we use any function $f : \{-1, +1\}^n \to \{-1, +1\}$ which can be defined in the following terms: consider uniform linear forms $L_1(x), \ldots, L_k(x)$ and for any $x \in \{-1, +1\}^n$ let $f(x)$ be the sign of $L_i(x)$, where i is the maximal index such that $L_i(x)$ is nonzero. Here uniformity of linear forms is required to make the symmetry argument of Lemma 6 work. Note also that we can use different functions on different coordinates and in fact there is sense in it since the last coordinate plays a very different role than the other coordinates. Actually, we have already used different functions in different coordinates when we choose numbers of variables k_i to be different for different i.

Note that the function GT in coordinates $1, \ldots, d - 1$ is not very economical. Indeed, we use $2k$ variables and get only k iterations in one coordinate in the proof of Lemma 9 (that is, k_l applications of Step 1 in the proof of this lemma). So we can have about $n/2k$ coordinates with k iterations each, so we have bound of approximately $2^{\Omega(k^{n/2k})}$. If with some other function we can have more iterations with less variables, we will be able to improve the bound.

We prove that for any $k \geqslant 3$ there is a function which with the use of k variables allows us to make $k - 1$ iterations.

Definition 10. For $x = (x_1, \ldots, x_k) \in \{-1, +1\}^k$ let $g(x_1, \ldots, x_k)$ be equal to $-x_k$ if the bits $x_1, \ldots, x_k$ are not all equal and let it be x_k if they are all equal.

This function can be easily represented by a linear threshold gate:

$$g(x_1, \ldots, x_k) = \mathrm{sgn}(\sum_{i=1}^{k-1} x_i - (k-2)x_k).$$

But to make g to be suitable as a building block in our combinatorial construction we have to express it in terms of linear forms $L_1(x), \ldots, L_k(x)$ (otherwise we are not able to define our ordering). Consider k linear form: $L_i(x) = x_i - x_{i+1}$ (L_i will play a role of u^i in the previous proof) for $i = 1, \ldots, k-1$ and $L_0(x) = x_1 + x_k$. The alternative (equivalent) definition of g is that $g(x)$ is equal to the sign of the last nonzero in the sequence $L_0(x), L_1(x), L_2(x), \ldots, L_{k-1}(x)$. Note that now it is more convenient for us to start the numeration of L_i from 0. The reason for this is that the actual benefit we will get only from linear forms $L_1, \ldots, L_{k-1}$. The form L_0 is needed only for technical reasons (the same role was previously played by the variables v^i).

Note that in the proof of the weaker bound we needed actually not one base function, but two of them. They were very similar though: GT_0 and GT_1. In the case of the stronger bound two functions will differ more substantially. Again, g_1 is just the function g we defined above. As for g_0 we let $g_0(x)$ be x_1 if not all bits of the input are equal and $g_0(x) = -x_1$ if all bits of input are equal. That is now we not only reverse the order of variables but also multiply the value of the function by -1. Note that $g_0(x)$ is equal to the sign of the last nonzero in the sequence $-L_0(x), L_{k-1}(x), L_{k-2}(x), \ldots, L_1(x)$.

Now we can apply the previous proof scheme with the functions g_1 and g_0 on the first $d-1$ components and with GT on the last component. We denote the new function by f again. Below we state what changes in the proof.

For the new function the construction of the ordering is the same except that we use different orderings $<_0'$ and $<_1'$ on the first $d-1$ coordinates, namely we let $<_1'$ to be $0, 1, 2, \ldots, k-2, k-1$ and $<_0'$ to be $0, k-1, k-2, \ldots, 2, 1$, that is, 0 is always the smallest element. The orderings on the last coordinate remains the same as before (as well as the rule (2.1) defining the ordering on each next coordinate).

In the Definition 2 we now have only variables $x^1, x^2, \ldots, x^d, y^d$ and we let

$$f(x) = \text{sgn}((-1)^{c_1 + \ldots + c_{d-1}} L_{\alpha_1}(x^1) L_{\alpha_2}(x^2) \ldots L_{\alpha_{d-1}}(x^{d-1})(x_{\alpha_d}^d - y_{\alpha_d}^d))$$

for the largest α for which the expression is nonzero, where $c_i = 1$ if $\alpha_i = 0$ and the order corresponding to the i-th coordinate of α is $<_0'$ and $c_i = 0$ otherwise. If there is no such α (which can happen only if $x^d = y^d$) we let $f(x)$ to be 1. Note that the number of variables $n = \sum_{i=1}^{d-1} k_i + 2k_d$ is almost twice less than before.

The theorem we prove has the following form.

Theorem 4.1. *Let $k_i \geqslant 3$ be odd for $i < d$ and $k_d \geqslant 3$. Then*

$$\deg_{\pm}(f) = d$$

$$W(f, d) \geqslant 2^{(k_d - 2) \prod_{i=1}^{d-1}(k_i - 1) - d \log n}$$

The proof of the theorem follows the same lines with minor changes. For the sake of completeness we present the details.

Let us consider an arbitrary threshold gate p for f of degree at most d. That is, for any $x^1, \ldots, x^{d-1}, x^d, y^d$ we have

$$f(x^1, \ldots, x^{d-1}, x^d, y^d) = \text{sgn}(p(x^1, \ldots, x^{d-1}, x^d, y^d)).$$

It will be convenient for us to work in the variables

$$u_j^i = L_j(x^i)$$

for $i = 1, \ldots, d-1$ and

$$u_j^d = x_j^d - y_j^d, \qquad v_j^d = x_j^d + y_j^d.$$

So after substituting

$$x_j^i = (u_0^i - u_1^i - \ldots - u_{j-1}^i + u_j^i + \ldots + u_{k-1}^i)/2$$

for $i = 1, \ldots, d-1$ and

$$x_j^d = (u_j^d + v_j^d)/2, \qquad y_j^d = (u_j^d - v_j^d)/2$$

and multiplying the polynomial by 2^d to make the coefficients integer we obtain the polynomial p' in the variables u_j^i, v_j^d that sign-represents f. That is

$$f(x^1, \ldots, x^{d-1}, x^d, y^d) = \operatorname{sgn}(p'(u^1, \ldots, u^{d-1}, u^d, v^d)).$$

It is easy to see that the weight of the new polynomial is almost the same as the weight of p (compared to the value of our bound). Namely, we have the following bound.

Lemma 11. $W(p') \leqslant n^d W(p)$.

Proof. Consider one monomial g of p of degree $l \leqslant d$. We can think of the transformation above as of substituting each variable by the sum of at most $\max_i k_i \leqslant n$ variables and then multiplying everything by 2^{d-l}. After opening the brackets we will have at most n^l new monomials each of weight 2^{d-l}, so the overall weight of the new monomials coming from the monomial g is at most n^d times the coefficient of g. Since this happens for all monomials, the upper bound on the weight of the polynomial p' as stated follows. $\qquad\square$

Next we prove that we can assume that p' has a nice structure.

Lemma 12. *If we substitute by 0 all coefficients of the monomials of p' in which the variables from one of the groups $u^1, \ldots, u^d$ do not appear, the resulting polynomial q will also sign-represent f.*

Proof. Let I be the set of inputs (x, y) to f such that $x^d \neq y^d$. This condition implies that for each such input there is an $\alpha = (\alpha_1, \ldots, \alpha_d) \in K = [k_1 - 1]_0 \times \ldots \times [k_{d-1} - 1]_0 \times [k_d]$ such that

$$u_{\alpha_1}^1 u_{\alpha_2}^2 \ldots u_{\alpha_{d-1}}^{d-1} (x_{\alpha_d}^d - y_{\alpha_d}^d) \neq 0,$$

where by $[k]_0$ we denote the set $\{0, \ldots k\}$. This means that the value of the function is determined by the sign of this product for the largest such α.

Consider an arbitrary input (x, y) from I. Let us first permute the variables x^d with the variables y^d. It is not hard to see that after such permutation the function f changes sign, and so p' should also change sign.

Now let us see what happens with $p'(u^1, \ldots, u^d, v^d)$. Note, that after this permutation all variables u and v^d remains the same except u^d which changes sign. Let us denote by A the sum of all monomials of p' which contain even number of variables from u^d and by B the sum of all monomials of p' which contain odd number of variables from u^d, so

$$p'(u, v^d) = A(u, v^d) + B(u, v^d).$$

Note that after our permutation of variables A remains the same and B changes sign. Since p' changes sign we have that the absolute value of B is greater then the absolute value of A. We proved that this happens for any input from I, so the sign of f is determined by the sign of B for all such inputs. So we can erase all monomials from A and the resulting polynomial will still sign-represent f for all inputs from I.

Now consider an arbitrary $i \in [d-1]$. Let us substitute the vector x^i by the vector $-x^i$. Again, it is not hard to see that after such substitution the function f changes sign, and so p' also should change sign.

Let us see what happens with $p'(u^1, \ldots, u^d, v^d)$. Note, that after this substitution all variables u and v^d remains the same except u^i which changes sign. Let us denote by A the sum of all monomials of p' which contain even number of variables from u^i and by B the sum of all monomials of p' which contain odd number of variables from u^i, so

$$p'(u, v^d) = A(u, v^d) + B(u, v^d).$$

After our substitution A remains the same and B changes sign. Again, we can erase all monomials from A and the resulting polynomial will still sign-represent f for all inputs from I.

Repeating this argument for all $i \in [d-1]$ we obtain a threshold gate q for f such that each monomial of q contains odd number of variables from each of the groups $u^1, \ldots, u^d$. But the degree of q is at most d, so it is only possible that each monomial of q contains one variable from each of the groups $u^1, \ldots, u^d$ (and no variables from v^d).

We have proved that the new polynomial q sign-represents f correctly for all inputs from I. Now note, that for any input from $\{-1, 1\}^n \setminus I$ we have $x^d = y^d$, or equivalently, all variables from u^d are zeros. By the definition the value of the function f on such input is 1 and the value of q is 0, so q sign-represents f for these inputs also. $\square$

As a byproduct of the proof of this lemma we have the following corollary.

Corollary 13. $\deg_{\pm}(f) = d$.

Since $W(p') \geqslant W(q)$ it is enough to prove that

$$W(q) \geqslant 2^{(k_d - 2) \prod_{i=1}^{d-1}(k_i - 1)}.$$

We present now the analog of Lemma 8.

Lemma 14. Let $p = \sum_{i=0}^{k-1} w_i u_i$ be a degree 1 threshold gate for $g_1(x)$ where $x \in \{-1, +1\}^k$. Then for $j = 0, 1, \ldots, k-1$ we have $w_j > 0$ and for $j = 2, \ldots, k-1$ we have $w_j > w_{j-1}$.

For the function g_0 analogous statement is true.

Lemma 15. Let $p = \sum_{i=0}^{k-1} w_i u_i$ be a degree 1 threshold gate for $g_0(x)$ where $x \in \{-1, +1\}^k$. Then we have $w_0 < 0$, for $j = 1, \ldots, k-1$ we have $w_j > 0$ and for $j = 2, \ldots, k-1$ we have $w_{j-1} > w_j$.

Proof of Lemma 14. To show the first part of the lemma for $j \neq 0$ let $x_1 = \ldots = x_j = -1$ and $x_{j+1} = \ldots = x_k = 1$. Then $g_1(x) = -1$ and thus $p(x) = -2w_j < 0$.

For $j = 0$ let $x_1 = \ldots = x_k = -1$. Then again $g_1(x) = -1$ and $p(x) = -2w_0 < 0$.

To show the second part let $x_j = -1$ and $x_l = 1$ for $l \neq j$. Then we have $g_1(x) = -1$ and $p(x) = 2(w_{j-1} - w_j + w_0) < 0$. And thus $w_j > w_{j-1} + w_0 > w_{j-1}$. $\square$

Proof of Lemma 15. For $j \neq 0$ letting $x_1 = \ldots = x_j = -1$, $x_{j+1} = \ldots = x_k = 1$ we have $-w_j < 0$. Letting $x_1 = \ldots = x_k = 1$ we have $w_0 < 0$ (note that we have $-L_0$ in the sequence defining g_0). For $j = 2, \ldots, k-1$ letting $x_j = 1$ and $x_l = -1$ for $l \neq j$ we have $-w_{j-1} + w_j - w_0 < 0$ and thus $w_{j-1} > w_j$ since w_0 is negative. $\square$

The analog of Lemma 9 is very similar to the previous version, but becomes a little bit clumsy since we distinguish cases of $l = d$ and $l < d$.

Lemma 16. *For all $l \leqslant d$ if $\alpha \in K$ is such that $\mathrm{num}_{\alpha,i}\alpha_i = 1$ for all $i \geqslant l$ and $\beta = (\alpha_1, \ldots, k_l - \alpha_l + \delta_{l,d}, \ldots, \alpha_d)$, where $\delta_{l,d}$ is a Kronecker delta (that is $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise) then*

$$w_\beta \geqslant w_\alpha 2^{(k_d-2)\prod_{i=l}^{d-1}(k_i-1)}.$$

Concerning the proof of the lemma, the base of the induction remains completely the same (note, that the statement is the same also). As for the induction step, it also remains the same but now we can apply the induction hypothesis $k_l - 1$ times instead of k_l times in the previous proof. For the sake of completeness we present the proof.

Proof. This proof repeats the proof of Lemma 9 almost literally.

The proof goes by induction on decreasing l.

The base of induction $l = d$. We fix all variables u^i except u^d in the following way. For any i let $u^i_{\alpha_i} = 1$ and $u^i_j = 0$ for all $j \neq \alpha_i$ (see the proof of Lemma 14 on how to do this).

Now we have a function in variables u^d and it is not hard to see that this function coincides with either GT_0, or GT_1. Applying the inequality (3.2) (or the corresponding inequality for GT_0) for $j = k_d$ we obtain

$$w_\beta \geqslant 2^{k_d-2}w_\alpha.$$

Induction step. To show the lemma for $l < d$ we repeat several times the following procedure consisting of two steps. After the ith application of the procedure we will get a tuple α^i such that $\mathrm{num}_{\alpha^i,l}\alpha^i_l = i + 1$ and $\mathrm{num}_{\alpha^i,j}\alpha^i_j = 1$ for all $j > l$. To unify the notation we denote $\alpha^0 = \alpha$.

During the procedure we will not change the values of the first $l - 1$ coordinates. This means that for all tuples we consider, the order corresponding to the lth coordinate is the same. Let us assume without loss of generality that this order is $<'_1$. This in particular means that $\alpha_l = 1$.

We also will not change the coordinates $\alpha_{l+2}, \ldots, \alpha_d$. Note, that this means that if $\mathrm{num}_{\alpha^i,l+1}\alpha^i_{l+1}$ is odd (as in the beginning) then all

$$\mathrm{num}_{\alpha^i,l+2}\alpha^i_{l+2}, \ldots, \mathrm{num}_{\alpha^i,d}\alpha^i_d$$

are equal to 1.

Step 1. We apply the induction hypothesis for the coordinate $l + 1$. We have that for $\widetilde{\alpha}^{i+1} = (\alpha_1, \ldots, \alpha^i_l, k_{l+1} - \alpha^i_{l+1} + \delta_{l+1,d}, \ldots, \alpha_d)$

$$w_{\widetilde{\alpha}^{i+1}} \geqslant w_{\alpha^i} 2^{(k_d-2)\prod_{i=l+1}^{d-1}(k_i-1)}.$$

Note that now the ordinal number of $l + 1$st coordinate (w.r.t. the corresponding order) is $k_{l+1} - 1 + \delta_{l+1,d}$.

Step 2. We fix all variables u^j except u^l in the following way: for any j let $u^j_{\widetilde{\alpha}^{i+1}_j} = 1$ and $u^j_m = 0$ for all $m \neq \widetilde{\alpha}^{i+1}_j$. Now we have a function in the variables u^l and it is not hard to see that this function coincides with g_1 (this happens because we agreed that the order corresponding to the lth component is $<'_1$, if it were $<'_0$ we would have g_0 here).

We apply the inequality from Lemma 14 to the coordinate l. After that we get $\alpha^{i+1} = (\alpha_1, \ldots, \alpha_l + i + 1, k_{l+1} - \alpha_{l+1}^i + \delta_{l+1,d}, \ldots, \alpha_d)$ such that

$$w_{\alpha^{i+1}} \geqslant w_{\widetilde{\alpha}^{i+1}}.$$

Due to the rule (2.1) defining the order on the next component of the tuple we have that the order on the $l + 1$st component changes. This means that the ordinal number of the $l + 1$st coordinate w.r.t. the corresponding order is again 1. ¿From this we have, as we stated above, that for all $j \geqslant l + 1$ it is true that $\mathrm{num}_{\alpha^{i+1},j}\alpha_j^{i+1} = 1$. So now we are again in the position to apply Step 1.

We repeat these two steps until the lth coordinate of $\widetilde{\alpha}^i$ reaches $k_l - 1$ (in the end we repeat Step 1, we are unable to repeat Step 2 since the lth coordinate is already $k_l - 1$ and can not be increased, so in the end we get $\widetilde{\alpha}^i$ for suitable i). Since the lth coordinate increases by 1 at each iteration (on Step 2) we can repeat Step 2 $k_l - 2$ times and Step 1 $k_l - 1$ times. Thus in the end we get the vector $\widetilde{\alpha}^{k_l-1}$ and it is easy to see that

$$w_{\widetilde{\alpha}^{k_l-1}} \geqslant w_\alpha \prod_{j=1}^{k_l-1} 2^{(k_d-2)\prod_{i=l+1}^{d-1}(k_i-1)} = w_\alpha 2^{(k_d-2)\prod_{i=l}^{d-1}(k_i-1)}.$$

Let us check that $\beta = \widetilde{\alpha}^{k_l-1}$. It is easy to see that during this process only the coordinates l and $l + 1$ of α changes. In the end of the process coordinate l has the number $k_l - 1$ w.r.t. the order corresponding to the lth coordinate of α. This means that $\widetilde{\alpha}_l^{k_l-1} = k_l - \alpha_l = \beta_l$. Note also that from this and from the oddness of k_l we have that α_l and $\widetilde{\alpha}_l^{k_l-1}$ define different orders on the next coordinates. Now let us see what happens with the coordinate $l + 1$. After the process the ordinal number of $\widetilde{\alpha}_{l+1}^{k_l-1}$ w.r.t. the order corresponding to the $(l + 1)$st component of $\widetilde{\alpha}^{k_l-1}$ is equal to $k_{l+1} - 1 + \delta_{l+1,d}$. This means that the ordinal number of $\widetilde{\alpha}_{l+1}^{k_l-1}$ w.r.t. the other ordering is 1. Since orders corresponding to the $(l + 1)$st coordinates of α and $\widetilde{\alpha}^{k_l-1}$ are different we have that $\widetilde{\alpha}_{l+1}^{k_l-1} = \alpha_{l+1} = \beta_{l+1}$ (recall that $\mathrm{num}_{\alpha,l+1}\alpha_{l+1} = 1$). □

To conclude the proof of our lower bound we have to choose the values of k_i to maximize the lower bound we have. It is not hard to see that the optimal way is to take $k_d = 3$ as before and to take $k_1 = k_2 = \ldots = k_{d-1}$, let us denote the value of them by k. Then the exponent of our bound will be about $(k-1)^{n/k}$. Simple analysis shows that the maximum (over integers) is attained when $k = 5$. Thus we have a lower bound of $2^{2^{2(n-6)/5}-n}$.

To prove the first part of Theorem 1.2 we can just let $m = \frac{5}{4}\epsilon n$ and consider the function from the previous paragraph with m variables. Then we have the lower bound $2^{\Omega(2^{2m/5})}$ for the degree $m/5$ threshold gates and applying the observation preceding Theorem 1.2 we get the desired bound. For the second part of the theorem let $m = \frac{5}{2}(1 + \epsilon)\log n$.

Finally we note that the result of [15] can also be reproved by the same argument and with better constants if we use Håstad's function in the last coordinate and g function in other coordinates.

References

[1] L. Babai, K. A. Hansen, V. V. Podolskii, and X. Sun. Weights of exact threshold functions. Manuscript. Preliminary version appeared in MFCS 2010, 2010.

[2] R. Beigel. Perceptrons, PP, and the polynomial hierarchy. *Computational Complexity*, 4:339–349, 1994.

[3] H. Buhrman, N. K. Vereshchagin, and R. de Wolf. On computation and communication with small bias. In *Proc. of the 22nd Conf. on Computational Complexity (CCC)*, pages 24–32, 2007.

[4] I. Diakonikolas, R. A. Servedio, L.-Y. Tan, and A. Wan. A regularity lemma, and low-weight approximators, for low-degree polynomial threshold functions. In *IEEE Conference on Computational Complexity*, pages 211–222, 2010.

[5] M. Goldmann, J. Håstad, and A. A. Razborov. Majority gates vs. general weighted threshold gates. *Computational Complexity*, 2:277–300, 1992.

[6] J. Håstad. On the size of weights for threshold gates. *SIAM J. Discret. Math.*, 7(3):484–492, 1994.

[7] A. R. Klivans and R. A. Servedio. Learning DNF in time $2^{\tilde{O}(n^{1/3})}$. *J. Comput. Syst. Sci.*, 68(2):303–318, 2004.

[8] M. Krause and P. Pudlák. Computing Boolean functions by polynomials and threshold circuits. *Comput. Complex.*, 7(4):346–370, 1998.

[9] M. L. Minsky and S. A. Papert. *Perceptrons: Expanded edition*. MIT Press, Cambridge, Mass., 1988.

[10] S. Muroga. *Threshold logic and its applications*. Wiley-Interscience, Chichester, 1971.

[11] S. Muroga, I. Toda, and S. Takasu. Theory of majority decision elements. *Journal of the Franklin Institute*, 271(5):376 – 418, 1961.

[12] J. Myhill and W. H. Kautz. On the size of weights required for linear-input switching functions. *IRE Trans. on Electronic Computers*, 10(2):288–290, 1961.

[13] I. Parberry. *Circuit complexity and neural networks*. MIT Press, Cambridge, MA, USA, 1994.

[14] V. Podolskii. Degree-uniform lower bound on the weights of polynomials with given sign function. *Proceedings of the Steklov Institute of Mathematics*, 274(1):231–246, 2011.

[15] V. V. Podolskii. Perceptrons of large weight. *Probl. Inf. Transm.*, 45:46–53, March 2009.

[16] V. V. Podolskii and A. A. Sherstov. A small decrease in the degree of a polynomial with a given sign function can exponentially increase its weight and length. *Mathematical Notes*, 87:860–873, June 2010.

[17] A. A. Razborov. On small depth threshold circuits. In *Proceedings of the Third Scandinavian Workshop on Algorithm Theory*, pages 42–52, London, UK, 1992. Springer-Verlag.

[18] M. E. Saks. Slicing the hypercube. *Surveys in Combinatorics*, pages 211–255, 1993.

[19] R. A. Servedio, L.-Y. Tan, and J. Thaler. Attribute-efficient learning and weight-degree tradeoffs for polynomial threshold functions. *Journal of Machine Learning Research - Proceedings Track*, 23:14.1–14.19, 2012.

[20] A. A. Sherstov. Communication lower bounds using dual polynomials. *Bulletin of the EATCS*, 95:59–93, 2008.