Electronic Colloquium on Computational Complexity, Report No. 20 (2002)

On one lower bound for branching programs *

E. A. Okol'nishnikova
Russia, Novosibirsk,
Sobolev Institute of Mathematics
okoln@math.nsc.ru

Abstract

The method of obtaining lower bounds on the complexity of Boolean functions for nondeterministic branching programs is proposed. A nonlinear lower bound on the complexity of characteristic functions of Reed–Muller codes for nondeterministic branching programs is obtained.

1 Introduction

The problem of finding nontrivial lower bounds on the complexity of well-defined Boolean functions is one of important problems in the theory of complexity. In the paper the computation of Boolean functions by nondeterministic branching programs is considered. A nonlinear lower bound $\Omega(n \log / \log\log n)$ on the complexity of characteristic functions of Reed–Muller codes for nondeterministic branching programs is obtained.

The best known lower bound on the complexity of Boolean functions for these programs is the bound $\Omega(\frac{n^{3/2}}{\log n})$ obtained by P. Pudlák [10] with the use of Nečiporuk's method [4]. Besides the lower bound $\Omega(n \log\log\log^* n)$ for some symmetric functions (including the Majority function) follows from the bound of A. A. Razborov [11] for the contact–rectifier circuits.

The best known lower bound on the complexity of Boolean functions for deterministic branching programs is the bound $\Omega(\frac{n^2}{\log^2 n})$ obtained by P. Pudlák [10] with the use of Nečiporuk's method [4].

For some symmetrical Boolean functions, in particular for the Majority function, P. Pudlák [9] obtained a lower bound

$$\mathrm{BP}(\mathrm{MAJ}_n) \geq \Omega(n \log\log n / \log\log\log n)$$

in the class of deterministic branching programs. This bound was subsequently improved to the bound $\Omega(n \log n / \log\log n)$ on the complexity of some symmetrical Boolean functions, including the Majority function MAJ_n and the elementary symmetrical function $E_{\lfloor n/2 \rfloor}$ in n variables [1]. E. A. Okol'nishnikova [5] obtained lower bounds on the complexity of characteristic functions of binary codes with the large number of code nodes

ISSN 1433-8092

*This research was supported by the Russian foundation for Basic Researches (Grant 00-01-00874) and Federal target program "Integration" (Grant AO–110).

and growing (on n) code distance for deterministic branching programs. In particular, the lower bound $\Omega(n \log n / \log\log n)$ for characteristic functions of BCH codes with a code distance $\log n / \log\log n$ was obtained. These codes are widely used in the code theory and its applications. Nevertheless there is some problem with the constructibility of their definition. In the present paper the computation of Boolean functions by nondeterministic and deterministic branching programs is considered. The problem of obtaining nonlinear lower bounds on the complexity of Boolean functions for branching programs is reduced to the problem of obtaining lower bounds on the complexity of covering of the set of "ones" of a Boolean function by functions of the defined type (theorem 4) or to the problem of obtaining the upper bounds on the number of "ones" of a Boolean function in i-faces of a cube of the defined dimension (theorem 5). The application of these results allows obtaining nonlinear lower bounds on the complexity of characteristic functions of Reed-Muller codes for nondeterministic programs.

The bounds were obtained by a modification of the method from [5]. This method reduces obtaining lower bounds on the complexity of Boolean functions for branching programs without restriction to obtaining lower bounds on the complexity of minorant of the considered function for branching programs with restriction on the number of occurrences of a variable in a path (read-k-times branching programs).

At present two methods [5, 2] of obtaining lower bounds on the complexity of functions for read-k-times branching programs are known. Exponential lower bounds on the complexity of Boolean functions for deterministic read-k-times branching programs ($k(n) = O(\log n / \log\log n)$) were obtained in [5]. Subsequently this method was extended to nondeterministic branching programs [6]. In [2] exponential lower bounds on the complexity of Boolean functions for nondeterministic read-k-times branching programs for $k(n) = O(\log n)$ were obtained. In the paper the method from [5] is used for obtaining lower bounds on the complexity of minorant functions of the considered function.

Results for generalized Hamming weights of linear codes [14] are essentially used for obtaining results of the present paper.

This paper is organized as follows. In section 2 we outline the general idea of the proof and give reasons for preferring a method from [5] for obtaining lower bounds on the complexity of read-k-times branching programs in this paper. Section 3 contains definitions and preliminary. In section 4 the main mathematical result (theorem 1) is proved. This theorem permits to reduce obtaining lower bounds on the complexity of Boolean functions for programs without restrictions to obtaining lower bounds on the complexity of minorant functions of the considered function for programs with restrictions (read-k-times branching programs). In section 5 a method of obtaining lower bounds on the complexity for read-k-times branching programs is considered; main theorems that permits to obtain lower bounds on the complexity of a function for programs without restrictions are given. In section 6 the result on the generalized Hamming weights [14] is used for counting the number of "ones" in i-faces of the cube for characteristic function of Reed-Muller code. It permits to receive nonlinear lower bounds on the complexity of characteristic functions of Reed–Muller codes for nondeterministic branching programs in section 7.

Main results of the paper are published in [7] (in Russian), see also [8] (in English) for a brief presentations of this paper.

2 Main idea

The idea of a method of obtaining nonlinear lower bounds for branching programs without restrictions is the same one as in [5]. Let $\mathcal{P}$ be a branching program that computes a Boolean function $f(x_1, x_2, \ldots, x_n)$. Let the number of occurrences of a variable x_i on a path from the input node to the output node exceeds $k(n)$, where $k(n) \to \infty$ as $n \to \infty$, and the number of such variables is not small. Then the size of $\mathcal{P}$ can not be small too. If the number of such variables is small, we substitute constants for these variables. It allows us to transform the program $\mathcal{P}$ to a program $\mathcal{P}'$ with restrictions on the number of occurrences of a variable in a path, i. e. to consider the computation of a minorant function of the function f by read-$k(n)$-times branching programs.

This approach allows to reduce obtaining lower bounds on the complexity of a Boolean function for branching programs without restrictions to obtaining lower bounds on the complexity of minorant functions of the considered Boolean function for programs with restrictions, namely, for read-$k(n)$-times branching programs.

Note that methods of obtaining high lower bounds on the complexity of Boolean functions for read-k-times branching programs in [5, 2, 6, 13] are similar. Let $\mathcal{P}$ be a branching program computing the Boolean function f in n variables. To each "one" of a Boolean function f we can assign a path in $\mathcal{P}$. This path is divided into "equal" parts. A separating set for these parts is a subset of nodes [5, 6] or a subset of edges [2, 13] of the branching program. The cardinality of this set depends only on beforehand selected parameters and is considerably less than the path length. The function f_i is assigned to the chosen subset of nodes or edges of the program $\mathcal{P}$. This function depends only on this subset of nodes or edges and does not depend on the paths to that these subsets belong. Thus

$$f = \vee f_i, \tag{1}$$

i. e. the functions f_i cover the set of "ones" of the function f. If the number of "ones" of each function f_i is not large, and the number of "ones" of f is large, then the number of different subsets that corresponds to units of the Boolean function is large. It allows to obtain lower bound on the number of nodes (or edges) of the branching program.

In [5] subsets of nodes of the branching program correspond to "ones" of the function. We need to transform a branching program to uniform form (the definition of this concept will be given in section 3) in this case. This transformation slightly increases the size of a program, but it makes possible to consider the generalized parts of the path. It gives the possibility to assign to a path not all nodes that were chosen as a separating set of the path, but only a part of them. Sometimes in this a way it is possible to receive bounds that are better than bounds obtained by the application of a method from [2], especially when read-k-times branching programs are used for obtaining lower bounds on the complexity for branching programs without restrictions.

In [2] subsets of edges of the branching program correspond to units of the function. On the one hand, there is no need to transform a branching program to a uniform form in this case, and on the other hand it does not allow to combine parts of the path, i. e., it is necessary to assign to a path all edges that were the separators of the parts. In order to obtain lower bounds on the complexity for branching programs by this method it is necessary to extract the root of the greater than in [5] degree from the cardinality of the obtained covering of set of "ones" of the function from (1).

3 Definitions, preliminary

We use traditional definitions of nondeterministic and deterministic branching programs (see, e.g. [12]). A branching program is called a read-k-times program if each variable does not occur more than k times in each path. By $\mathrm{NBP}(\mathcal{P})$ ($\mathrm{BP}(\mathcal{P})$) denote the size of the nondeterministic (deterministic) branching program $\mathcal{P}$. By $\mathrm{NBP}(f)$ ($\mathrm{BP}(f)$) denote the size of the minimal nondeterministic (deterministic) branching program that computes the Boolean function f. By $\mathrm{NBP}k(f)$ ($\mathrm{BP}k(f)$) denote the size of the minimal nondeterministic (deterministic) read-k-times program that computes the Boolean function f.

A nondeterministic (deterministic) read-k-times branching program is said to be *uniform* if for every node a of this program and for every i, $1 \leq i \leq n$, in each path from the input node to a node a the number of nodes labeled with the variable x_i does not depend on paths (this number can differ for different i). Besides for every i, $1 \leq i \leq n$, in every path from the input node to an output node the number of nodes labeled with a variable x_i is equal to k. By $\mathrm{UNBP}k(f)$ and $\mathrm{UBP}k(f)$ denote the complexity of a Boolean function f for nondeterministic and deterministic uniform read-k-times branching programs accordingly.

Let $\mathcal{P}$ be a uniform nondeterministic (deterministic) read-k-times branching program. A node a of the program $\mathcal{P}$ is said to be at the distance l from the input node if the number of labeled edges in any path from the input node to the node a is equal to l. By $\mathrm{UNBP}^d(\mathcal{P})$ ($\mathrm{UBP}^d(\mathcal{P})$) denote the number of nodes labeled with variables in the nondeterministic (deterministic) branching program $\mathcal{P}$ that are at the distance divisible by d from the input node.

Let's consider a branching program $\mathcal{P}$ computing a Boolean function f. A node a_i is said to *precede* a node a_j in $\mathcal{P}$ if there is a path from a_i to a_j in the program. A set of nodes $a_1, \ldots, a_t$ (not necessarily different) of the branching program $\mathcal{P}$ is said to be a *sequence of nodes* if there is a path to which the nodes $a_1, \ldots, a_t$ belong, and a_i precedes a_j or a_i coincides with a_j for $i < j$.

4 Reduction of lower bounds on complexity for programs without restrictions to bounds on complexity for read-k-times programs

Let $f(x_1, x_2, \ldots, x_n)$ be a Boolean function, $X' = \{x_{i_1}, \ldots, x_{i_m}\}$ be a subset of the set $\{x_1, x_2, \ldots, x_n\}$, and $\alpha = \{\alpha_{i_1}, \ldots, \alpha_{i_m}\}$ be a set of constants. By $f|_{X'=\alpha}$ denote the function that is obtained from f by substitution of constants from α instead of variables from X'.

The following theorem shows that it is possible to obtain lower bounds on the complexity of Boolean functions for branching programs without restrictions by the use of read-k-times branching program.

Theorem 1 *Let $g(X)$ be a Boolean function, C, $0 < C < 1$, be a constant, $\psi(n)$ be a growing function. Let X_0 be a subset of variables, i. e., $X_0 \subseteq X$, and let $|X_0| = \lfloor Cn \rfloor$. If for each X_0 there exists a substitution of constants α in the set X_0 such that $\mathrm{NBP}k(n)\left(g|_{X_0=\alpha}(X \setminus X_0)\right) \geq n\psi(n)$, then $\mathrm{NBP}(g) \geq \min\{Cnk(n), n\psi(n)\}$.*

Proof. Let $\mathcal{P}$ be a branching program that computes g. Let's show that its complexity is no less than $\min\{Cnk(n), n\psi(n)\}$. By X' denote the set of variables of the function g such that there are at least $k(n)$ nodes labeled by each variable of X' in the program $\mathcal{P}$. It is possible to consider 2 cases.

I. $|X'| > \lfloor Cn \rfloor$. Hence there are at least $\lceil Cn \rceil$ variables such that there are at least $k(n)$ nodes labeled by each of these variables in the program $\mathcal{P}$. Therefore the total number of labeled nodes in the program $\mathcal{P}$ is no less than $Cnk(n)$, and the complexity of the program $\mathcal{P}$ is not less than $\min\{Cnk(n), n\psi(n)\}$.

II. $|X'| \leq \lfloor Cn \rfloor$. If $|X'| < \lfloor Cn \rfloor$, then we complete the set X' in an arbitrary way in order to obtain the set X_0 of the cardinality $\lfloor Cn \rfloor$. By the condition of the theorem there exists a substitution of constants α in the subset of variables X_0 such that the complexity of the function $g|_{X_0=\alpha}$ for read-k-times programs is not less than $n\psi(n)$. We can substitute constants α for variables from X_0 in the program $\mathcal{P}$. By simple modification of the program $\mathcal{P}$ we shall obtain a new program $\mathcal{P}'$. The size of $\mathcal{P}'$ is no more than the size of the program $\mathcal{P}$. The program $\mathcal{P}'$ computes the function $g|_{X_0=\alpha}(X \setminus X_0)$. In the program $\mathcal{P}'$ there is no variable with which more $k(n)$ nodes are labeled. It means that $\mathcal{P}'$ is read-$k(n)$-times branching program. Therefore, by the condition of the theorem its size is not less than $n\psi(n)$.

The theorem is proved.

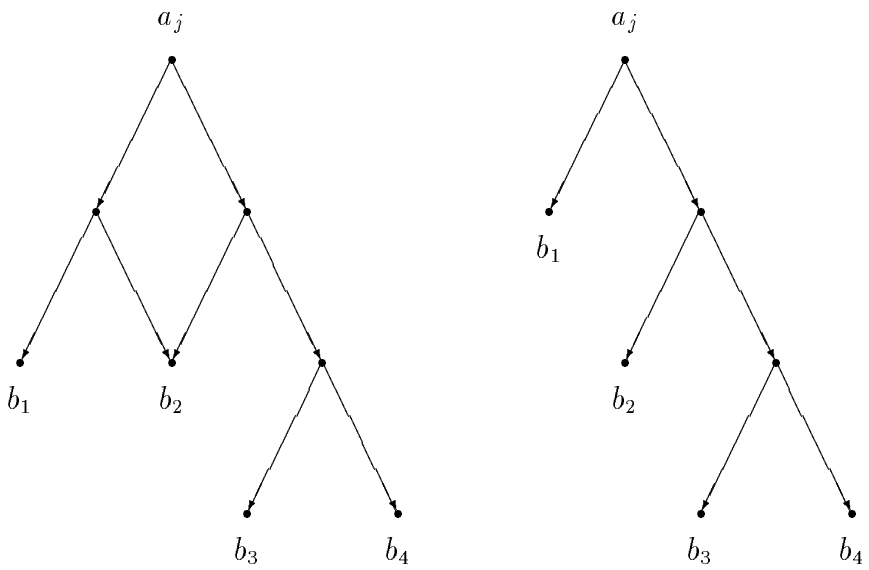
Thus, for obtaining lower bounds on the complexity of the function g for programs without restrictions it is necessary to obtain lower bounds on the complexity of minorant functions of the function g for read-k-times branching programs.

5 Lower bounds on the size of read-k-times programs

In order to estimate the increase of the complexity of a function g in going from computation of g by nondeterministic read-k-times programs to computation of g by uniform nondeterministic read-k-times programs we need to obtain an upper bound on the total number of edges (including free edges) in the nondeterministic program. It is made as well as in the proof of the Theorem 1 from [2]; but this statement was formulated for acyclic switching-and-rectifier networks in [2]. It is known that complexities of a Boolean function for acyclic switching-and-rectifier networks and nondeterministic branching programs coincide up to the order. Nevertheless, it seems convenient to formulate this statement in terms of nondeterministic branching programs.

Lemma 1 *A nondeterministic program $\mathcal{P}$ can be transformed into a nondeterministic program $\mathcal{P}'$ of the size not exceeding the size of the program $\mathcal{P}$, and the total number of edges (including free edges) in $\mathcal{P}'$ does not exceed* $16(\mathrm{NBP}(\mathcal{P}))^2 - 2$.

Proof. By $a_0, a_1, \ldots, a_L$ denote the input node and nodes to which the edges labeled with 0 or 1 enter. It is clear that $L \leq 2\mathrm{NBP}(\mathcal{P})$. Let a_j, $0 \leq j \leq L$, be a nondeterministic node, let $b_1, \ldots, b_l$ be labeled nodes or the output nodes to which the paths consisting of free edges that leave goes the node a_j ($l \leq \mathrm{NBP}(\mathcal{P}) + 2$). We can replace the set of these paths by the oriented tree with the root a_j that contains no more than $2(l-1)$ free edges; paths of this tree enter the same nodes $b_1, \ldots, b_l$ (see Fig. 1).



Program $\mathcal{P}$ Program $\mathcal{P}'$

Fig. 1

We make replacements of this kind for all nondeterministic nodes from the set $\{a_0, a_1, \ldots, a_L\}$. We receive a new program $\mathcal{P}'$ in which the number of labeled nodes is the same; the number of free edges does not exceed $(2(\mathrm{NBP}\mathcal{P}+1))(2\mathrm{NBP}(\mathcal{P})+1)$. The total number of edges in the new program does not exceed $2(\mathrm{NBP}(\mathcal{P}+1))(2\mathrm{NBP}(\mathcal{P})+1)+2\mathrm{NBP}(\mathcal{P}) \leq 16(\mathrm{NBP}(\mathcal{P}))^2-2$. The lemma is proved.

Consider the transformation of a branching program into a uniform one as it was made in [5, 6]. We were not interested in precise bounds in [6]. In this paper we estimate not only the relation between sizes of a branching program and a uniform branching program, but a relation between the size of a branching program and the number of nodes of a uniform branching program that are at the distance that is divisible by some preliminary defined number from the input node (as it was done in [5] for deterministic branching programs).

Lemma 2 *a) Each nondeterministic read-k-times branching program $\mathcal{P}$ computing a Boolean function $f(x_1, \ldots, x_n)$ can be transformed into a uniform read-k-times branching program $\mathcal{P}_0$ that computes the function f and an inequality*

$$\mathrm{UNBP}k^d(\mathcal{P}_0) \leq \lceil kn/d \rceil (4\mathrm{NBP}k(\mathcal{P}))^2$$

holds.

b) Each deterministic read-k-times branching program $\mathcal{P}$ computing a Boolean function $f(x_1, \ldots, x_n)$ can be transformed into a uniform read-k-times branching program $\mathcal{P}_0$ that computes the function f and an inequality

$$\mathrm{UBP}k^d(\mathcal{P}_0) \leq 2\lceil kn/d \rceil \cdot \mathrm{BP}k(\mathcal{P})$$

holds.

Proof of this Lemma is similar to that of Lemma 1 in [5, 6].

a) Let $\mathcal{P}$ be a nondeterministic read-k-times branching program computing the Boolean function $f(x_1, x_2, \ldots, x_n)$. By Lemma 1 a nondeterministic read-k-times branching program $\mathcal{P}$ can be transformed into a nondeterministic read-k-times branching program $\mathcal{P}'$ with the same number of labeled nodes, and with the total number of edges (including free edges) less than or equal to $16\mathrm{NBP}(\mathcal{P})^2 - 2$.

By $l_q(a)$ denote the maximal number of nodes labeled with x_q in an arbitrary path from the input node to the node a in the program $\mathcal{P}'$. It is clear that if a node a_i precedes a node a_j in $\mathcal{P}'$ then for any $q = 1, 2, \ldots, n$ an inequality

$$l_q(a_i) \leq l_q(a_j)$$

holds.

Let's consider a case when the edge (a_i, a_j) leaves a node labeled with a variable, and the case when an edge (a_i, a_j) leaves a nondeterministic node.

Case 1. Let the edge (a_i, a_j) leaves a labeled node. Without loss of generality it is possible to assume that this edge has the label 1 and leaves a node labeled with x_1. Then an inequality $l_1(a_i) < l_1(a_j)$ holds. We replace the edge (a_i, a_j) with an edge (a_i, a') and a "chain" of edges connecting nodes a' and a_j and computing fictitious testing of variables $x_1, x_2, \ldots, x_n$: $(l_1(a_j) - l_1(a_i) - 1)$ testing of the variable x_1 and $(l_q(a_j) - l_q(a_i))$, $2 \leq q \leq n$, testing of the variable x_q. There are no more than $\lceil kn/d \rceil$ nodes that are at the distance divisible by d from the input node in paths from a_i to a_j after adding fictitious testing of variables (the node a_j is not taken in the account).

Case 2. Let (a_i, a_j) be a free edge in $\mathcal{P}'$. If $l_q(a_j) = l_q(a_i)$ for each q, $1 \leq q \leq n$, then we do not make any changes. If there exists q, $1 \leq q \leq n$, such that $l_q(a_j) \neq l_q(a_i)$ we replace the edge (a_i, a_j) with an edge (a_i, a') and to a "chain" of edges connecting nodes a' and a_j and computing fictitious testing of variables $x_1, x_2, \ldots, x_n$: $(l_q(a_j) - l_q(a_i))$, $1 \leq q \leq n$, testing of a variable x_q. There are no more than $\lceil kn/d \rceil$ nodes that are at the distance divisible by d from the input node in the paths from a_i to a_j after adding fictitious testing of variables (the node a_j is not taken in the account).

Let's carry out these operations on all edges of $\mathcal{P}'$. In order to have exactly k nodes labeled with each variable in each path from the input node to the output node we insert a new output node and we connect the former output node and the new one by a "chain" of edges containing the necessary number of fictitious testing of variables $x_1, x_2, \ldots, x_n$. As a consequence we obtain a uniform read-k-timed branching program $\mathcal{P}_0$, and the inequality

$$\mathrm{UNBP}^q(\mathcal{P}_0) \leq \lceil kn/q \rceil (4\mathrm{NBP}(\mathcal{P}))^2$$

holds.

b) There are no free edges in deterministic programs; and, consequently, there is no need to transform the program as indicated in Lemma 1. Therefore, the bound from the item b) of Lemma is valid. The lemma is proved.

Let $\mathcal{P}_0$ be a uniform read-k-times program computing a Boolean function $f(X)$. Then each path in $\mathcal{P}_0$ computes either an elementary conjunction in variables from X or a conjunction that is equal to zero. Let π be a path in $\mathcal{P}_0$ computing a non-zero conjunction K. A variable x is said to *occur in an interval* (b,c) of a path π if there is an edge going out from the node labeled with x in the interval (b,c).

Let $a_1, a_2, \ldots, a_{2m}$ be a sequence of nodes of a uniform read-k-times program $\mathcal{P}_0$, and let all nodes of this sequence belong to the path π.

We introduce the following notation:

- $Q^1_\pi(a_1, a_2, \ldots, a_{2m})$ is the set of variables that occur only in the intervals $(a_1, a_2), (a_3, a_4), \ldots, (a_{2m-1}, a_{2m})$ in the path π.
- $Q^2_\pi(a_1, a_2, \ldots, a_{2m})$ is the set of variables that occur only outside the intervals above in the path π.
- $Q^0_\pi(a_1, a_2, \ldots, a_{2m})$ is the set of variables that occur in both intervals above and outside them in the path π.

A set of variables that occur in an interval (b,c) of a uniform read-k-times program $\mathcal{P}_0$ does not depend on a path to which nodes b and c belong. Therefore, the sets $Q^j_\pi(a_1, a_2, \ldots, a_{2m})$, $j = 0, 1, 2$, depend only on the sequence of nodes $a_1, a_2, \ldots, a_{2m}$ and do not depend on the path to which these nodes belong. Therefore, the index π in Q^j_π can be omitted. It is clear that the sets $Q^j(a_1, a_2, \ldots, a_{2m})$ $(j = 0, 1, 2)$ are pairwise disjoint and the union of these sets is the set of variables of the computing function.

Let k, p, and t be natural numbers such that $2 \le k \le p \le t$. Introduce the following notation:

$$n_1(n;k,p,t) = \left\lceil n \binom{t-k}{p-k} \Big/ \binom{t}{p} \right\rceil; \tag{2}$$

$$n_2(n;k,p,t) = n - p\lceil kn/t \rceil + (k-1)\left\lceil n \binom{t-k}{p-k} \Big/ \binom{t}{p} \right\rceil; \tag{3}$$

$$n_0(n;k,p,t) = n - n_1 - n_2. \tag{4}$$

Values of $n_1(n;k,p,t)$ and $n_2(n;k,p,t)$ for some parameters p and t are given in the following lemma.

Lemma 3 1. *If $p = k$, $t = k^2 + k$, and $n \ge 4(ke)^k$, then $n_1(n;k,p,t) \ge \frac{n}{2(ke)^k}$ and $n_2(n;k,p,t) \ge \frac{n}{k+1}$.*

2. *If $p = k$, $t = 2k^2$, and $n \ge (2ke)^k$, then $n_1(n;k,p,t) \ge \frac{2n}{(2ke)^k}$ and $n_2(n;k,p,t) \ge \frac{n}{2}$.*

Proof. It follows from Stirling's formula that for any b and a, $1 \le a \le b/2$, the inequality

$$\binom{b}{a} \le \frac{1}{2}\left(\frac{be}{a}\right)^a \tag{5}$$

holds.

1. Let $p = k$ and $t = k^2 + k$. From (2) and (5) it follows that $n_1 = \left\lceil n \Big/ \binom{k^2+k}{k} \right\rceil \ge \frac{2n}{((k+1)e)^k} \ge \frac{2n}{k^k e^{k+1}} \ge \frac{n}{2(ke)^k}$. From this fact and (3) it follows that $n_2 \ge \frac{n}{k+1}$.

2. Let $p = k$ and $t = 2k^2$. From (2) and (5) it follows that $n_1 = \left\lceil n \Big/ \binom{2k^2}{k} \right\rceil \ge \frac{2n}{(2ke)^k}$. From this fact and (3) it follows that $n_2 \ge \frac{n}{2}$.

The lemma is proved.

Theorem 2 *Let $\mathcal{P}_0$ be a uniform nondeterministic branching read-k-times program computing a function f; let k, p, and t be the natural numbers such that $2 \leq k \leq p \leq t$ and numbers $n_1(n;k,p,t)$, $n_2(n;k,p,t)$, $n_0(n;k,p,t)$ are positive; let $\gamma = (\gamma_1, \ldots, \gamma_n)$ be an n-tuple such that $f(\gamma) = 1$. Then there exists a sequence $\Psi(\gamma)$ containing $2p'$, $p' \leq p$, different nodes labeled with variables such that*

(a) *all nodes of the sequence $\Psi(\gamma)$ belong to a path $\pi(\gamma)$ realizing γ;*

(b) *the distance between the input node and a node of the sequence $\Psi(\gamma)$ is divisible by $\lceil kn/t \rceil$;*

(c) $|Q^1(\Psi(\gamma))| \geq n_1(n;k,p,t)$;
$|Q^2(\Psi(\gamma))| \geq n_2(n;k,p,t)$;
$|Q^0(\Psi(\gamma))| \leq n_0(n;k,p,t)$.

Proof. The proof of this lemma is conceptually identical with that of Lemma 2 in [5]. The presence of free edges changes the proof slightly.

There is a path $\pi(\gamma)$ of the length kn that realizes a "one" γ of the Boolean functions f in the uniform program $\mathcal{P}_0$. We choose nodes $a_0, a_1, \ldots, a_t$ in the path $\pi(\gamma)$, where a_0 is the first node labeled with a variable in the path π (i. e., the node that is at the distance 0 from the input node), a_1 a node labeled with a variable and is at the distance $\lceil kn/t \rceil$ from the input node, a_2 is a node labeled with a variable and is at the distance $2 \cdot \lceil kn/t \rceil$ from the input node and so on as long as the exit node labeled with 1 occurs. We choose this exit node as the next s-th node of the path. In this case the distance between the node preceding the exit node and the exit node can be less than $\lceil kn/t \rceil$. If $s < t$ then we choose the exit nodes labeled with 1 as nodes $a_{s+1}, \ldots, a_t$.

It is clear that all nodes of the chosen sequence (except for nodes that coincide with the exit node) are labeled with variables and are at the distance divisible by $\lceil kn/t \rceil$ from the input node.

Any p nodes $a_{i_1}, \ldots, a_{i_p}$, $0 \leq i_1 < i_2 < \ldots < i_p \leq t-1$, from the set $\{a_0, a_1, \ldots, a_{t-1}\}$ determine p intervals of the path $\pi(\gamma)$: (a_{i_1}, a_{i_1+1}), $(a_{i_2}, a_{i_2+1}), \ldots,$ (a_{i_p}, a_{i_p+1}). By $A_{i_1,i_2,\ldots,i_m}$ denote the set of ends of these intervals, i. e., $A_{i_1,i_2,\ldots,i_m} = \{a_{i_1}, a_{i_1+1}, a_{i_2}, a_{i_2+1}, \ldots, a_{i_m}, a_{i_m+1}\}$. It is a sequence of nodes of $\mathcal{P}_0$.

At the end of the proof of the theorem it will be shown that we can choose a sequence from the set of all sequences $A_{i_1,i_2,\ldots,i_m}$ such that its insignificant modification (a sequence B) satisfies the conditions of the theorem.

By the same way as it was done in [5, 6] we can show that there exist a sequence $A_{i_1,i_2,\ldots,i_m}$ such that

$$|Q^1(A_{i_1,i_2,\ldots,i_p})| \geq n_1; \tag{6}$$

$$|Q^2(A_{i_1,i_2,\ldots,i_m})| \geq n_2; \tag{7}$$

$$|Q^0(A_{i_1,i_2,\ldots,i_m})| \leq n_0. \tag{8}$$

It is clear that all nodes of a sequence $A_{i_1,i_2,\ldots,i_p}$ are at the distance that is divisible by $\lceil kn/t \rceil$. If a node occurs more than once in the sequence $A_{i_1,i_2,\ldots,i_p}$, then it is possible to unite intervals containing common nodes by throwning out these nodes. For example, if $i_3 = i_2 + 1$, then the node a_{i_3} occurs twice in a sequence $A_{i_1,i_2,\ldots,i_p}$. Therefore it is possible to consider a sequence that consists of the ends of intervals (a_{i_1}, a_{i_1+1}), $(a_{i_2}, a_{i_3+1}), \ldots,$ (a_{i_p}, a_{i_p+1}). We carry out this operation on all coinciding ends of intervals. By this way we transform the sequence $A_{i_1,i_2,\ldots,i_p}$ to a sequence B containing $2p'$ nodes, $p' \leq p$. It is

easy to see that the sequence B satisfies the conditions (a) and (b) of the theorem. It is also easy to see that

$$Q^j(A_{i_1,i_2,\ldots,i_p}) = Q^j(B)$$

for $j = 0, 1, 2$. Then from (2)–(4), (6),(7), and (8) it follows that the sequence B satisfy the condition (c) of the theorem. The theorem is proved.

Let $\mathcal{P}$ be a branching program computing a Boolean function f, a and b be nodes of the program $\mathcal{P}$. By $f(\mathcal{P}; a, b)$ denote a Boolean function computed by a subprogram of the program $\mathcal{P}$ in which the node a is considered as the input node, and the node b as the exit node labeled with 1. (In order to transform the obtained subprogram to a branching program it is necessary to label the node b with 1 and delete edges of the programs that leave the node b.) Let $\Psi = (a_1, a_1, \ldots, a_{2p'})$ be a sequence of nodes of a branching program $\mathcal{P}$ computing the Boolean function $f(x_1, \ldots, x_n)$. By a_0 denote the input node of the program $\mathcal{P}$, by $a_{2p'+1}$ denote the exit node labeled with 1. Put

$$\begin{aligned} f^1(\mathcal{P}; \Psi) &= \bigwedge_{j=1}^{p'} f(\mathcal{P}; a_{2j-1}, a_{2j}), \\ f^2(\mathcal{P}; \Psi) &= \bigwedge_{j=0}^{p'} f(\mathcal{P}; a_{2j}, a_{2j+1}). \end{aligned}$$

From the definition of sets $Q^j(\Psi)$, $j = 0, 1, 2$, it follows that $f^1(\mathcal{P}; \Psi)$ depends only on variables from the sets $Q^1(\Psi)$ and $Q^0(\Psi)$; the function $f^2(\mathcal{P}; \Psi)$ depends on the variables from the sets $Q^2(\Psi)$ и $Q^0(\Psi)$.

It is clear that if $f(\gamma) = 1$, then

$$f^1(\mathcal{P}; \Psi(\gamma))(\gamma) \wedge f^2(\mathcal{P}; \Psi(\gamma))(\gamma) = 1.$$

Let $\mathcal{P}_0$ be a uniform nondeterministic read-k-times branching program computing the function f; k, p, and t be natural numbers such that $k \le p \le t$ and numbers $n_1(n; k, p, t)$, $n_2(n; k, p, t)$, $n_0(n; k, p, t)$ are positive. By theorem 2 the sequence $\Psi(\gamma)$ containing $2p'$ nodes (p' is a natural number and $p' \le p$) that are at the distance divisible by $\lceil kn/t \rceil$ from the input node, and such that $|Q^1(\Psi(\gamma))| \ge n_1(n; k, p, t)$; $|Q^2(\Psi(\gamma))| \ge n_2(n; k, p, t)$; $|Q^0(\Psi(\gamma))| \le n_0(n; k, p, t)$ can be assigned to each "one" γ of a Boolean functions f

By $V(\mathcal{P})$ denote a set of $2p'$-nodes sequences of the uniform branching program $\mathcal{P}$ corresponding to "ones" of a Boolean function f that is computed by the program $\mathcal{P}$.

The following lemma is obvious.

Lemma 4 *The Boolean function f computed by the program $\mathcal{P}_0$ can be represented as*

$$f = \bigvee_{j=1}^{|V(\mathcal{P}_0)|} f^1(\mathcal{P}_0; \Psi^j)(Q^0(\Psi^j) \cup Q^1(\Psi^j)) \wedge f^2(\mathcal{P}_0; \Psi^j)(Q^0(\Psi^j) \cup Q^2(\Psi^j)).$$

Let's consider all possible representations of the function $f(Y)$, $|Y| = n$, of the form

$$f(Y) = \bigvee f^1(Y_1 \cup Y_0) \wedge f^2(Y_2 \cup Y_0), \tag{9}$$

where Y_1, Y_2, and Y_0 are nonintersecting sets; $Y = Y_1 \cup Y_2 \cup Y_0$; $|Y| = n$; $|Y_1| \ge n_1(n; k, p, t)$; $|Y_2| \ge n_2(n; k, p, t)$; $|Y_0| = n - |Y_1| - |Y_2|$.

By $R(f;n,k,p,t)$ denote the minimal number of disjunctive terms in the representation (9). It is clear that for any uniform read-k-times program $\mathcal{P}_0$ computing the function f we have

$$R(f;n,k,p,t) \le |V(\mathcal{P}_0)|. \tag{10}$$

Theorem 3 *Let f be a Boolean function essentially depending on n variables, $n \ge 16$; k, p and t, $2 \le k \le p \le t$, be natural numbers such that values $n_1(n;p,k,t)$, $n_2(n;p,k,t)$ and $n_0(n;p,k,t)$ computed from the formulas (2)-(4) are positive.*

(a) *The complexity* $\mathrm{NBPk}(f)$ *of computation of the Boolean function f with nondeterministic read-k-times programs satisfies an inequality*

$$\mathrm{NBP}k(f) \ge \max\left\{n;\ 1/4\ \sqrt{\frac{2p}{et}} \cdot (R(f;n,k,p,t))^{1/(4p)}\right\}.$$

(b) *The complexity* $\mathrm{BPk}(f)$ *of computation of the Boolean function f with deterministic read-k-times programs satisfies an inequality*

$$\mathrm{BP}k(f) \ge \max\left\{n;\ \frac{p}{et} \cdot (R(f;n,k,p,t))^{1/(2p)}\right\}.$$

Proof. Let $\mathcal{P}$ be a nondeterministic read-k-times program computing the Boolean function f. By Lemma 2 we can transform the read-k-times program $\mathcal{P}$ into a uniform nondeterministic read-k-times program $\mathcal{P}_0$ computing the function f. By L_0 denote a set of all nodes of the program $\mathcal{P}_0$ that are at the distance divisible by $\lceil kn/t \rceil$ from the input node.

By Theorem 2 all nodes of sequences corresponding to "ones" of the Boolean function f belong to the set L_0. Therefore a relation

$$V(\mathcal{P}_0)| \le \sum_{j=1}^{p} \binom{|L_0|}{2j} \tag{11}$$

is fulfilled.

As the function f essentially depends on n variables, $\mathrm{NBPk}(f) \ge n$ and $\mathrm{BPk}(f) \ge n$. Therefore it is sufficient to consider a case $R(f;n,k,p,t)^{1/2p} \ge n$. By condition of the theorem we have $n \ge 16$, therefore it is sufficient to consider a case

$$R(f;n,k,p,t) \ge n^{2p} = 2^{2p \log_2 n} \ge 2^{8p}. \tag{12}$$

Let's show that $|L_0| \ge 8p$ in this case. Assume that $|L_0| < 8p$. Then by (10) and (11) we have

$$R(f;n,k,p,t) \le V(\mathcal{P}_0)| \le \sum_{j=1}^{p} \binom{|L_0|}{2j} < 2^{|L_0|} \le 2^{8p}.$$

It contradicts the assumption (12).

Thus, $|L_0| \ge 8p$. From this fact, (10), (11) and (5) it follows that

$$R(f;n,k,p,t) \le |V(\mathcal{P}_0)| \le \sum_{j=1}^{p} \binom{|L_0|}{2j} \le 2\binom{|L_0|}{2p} \le \left(\frac{e|L_0|}{2p}\right)^{2p}.$$

Therefore,

$$|L_0| \ge \frac{2p}{e}(R(f;n,k,p,t))^{1/(2p)}. \tag{13}$$

By definition of the set L_0 and of $\mathrm{UNBP}_k^{\lceil kn/t \rceil}(\mathcal{P}_0)$ we have

$$|L_0| = \mathrm{UNBPk}^{\lceil kn/t \rceil}(\mathcal{P}_0).$$

Then, by Lemma 2, we get

$$|L_0| \le \left\lceil \frac{kn}{\lceil kn/t \rceil} \right\rceil (4\mathrm{NBPk}(\mathcal{P}))^2 \le t(4\mathrm{NBPk}(\mathcal{P}))^2.$$

From this fact and (13) it follows that

$$\mathrm{NBPk}(\mathcal{P}) \ge \frac{1}{4}\sqrt{\frac{2p}{et}}\left(R(f;n,k,p,t))^{1/(4p)}\right).$$

$\mathcal{P}$ is an arbitrary program computing the Boolean function f, therefore from the obtained inequality the statement (a) of the theorem follows.

The statement (b) of the theorem can be proved similarly.

From theorems 1 and 3 we can obtain the following theorem.

Theorem 4 *Let $g_n(X_n)$, $|X_n| = n$, be a sequence of Boolean functions, $k(n)$ be an increasing function, and C, $0 < C < 1$, be a constant. If for each subset of variables X, $X \subseteq X_n$, $|X_0| = \lfloor Cn \rfloor$, there exists a substitution of constants α in the set X, and integer-valued $p(X)$, $t(X)$, $2 \le k(n) \le p(X) \le t(X)$, such that n_1, n_2, and n_0 calculated by the formulas (2)–(4) as functions in $|X_n \setminus X|, k(n), p(X), t(X)$ are positive, then* (a) *the complexity of the function $g_n(X)$ for nondeterministic branching program (without restrictions) satisfies the inequality*

$$\mathrm{NBP}(g_n) \ge \min\left\{Cnk(n), 1/4 \cdot \sqrt{\frac{2p}{et}} \cdot (R(g_n\,|_{X=\alpha}; |X_n \setminus X|, k, p, t))^{1/(4p)}\right\};$$

(b) ([5], theorem 3) *the complexity of the function $g_n(X)$ for deterministic branching program (without restrictions) satisfies the inequality*

$$\mathrm{BP}(g_n) \ge \min\left\{Cnk(n),\ \frac{p}{et} \cdot (R(g_n\,|_{X=\alpha}; |X_n \setminus X|, k, p, t))^{1/(2p)}\right\}.$$

It is possible to propose some ways to obtain lower bounds for $R(f;n,k,p,t)$ (see [5]). We use the following one in the paper.

By $H_i(f)$ denote the maximal number of "ones" of a Boolean function f belonging to i-faces of a cube.

Lemma 5 *The value $R(f;n,k,p,t)$ satisfies an inequality*

$$R(f;n,k,p,t) \ge \frac{|f^{-1}(1)|}{2^{n_0} H_{n_1}(f) H_{n_2}(f)},$$

where $n_1 = n_1(n;k,p,t)$, $n_2 = n_2(n;k,p,t)$. $n_0 = n_0(n;k,p,t)$.

Proof. Each disjunctive term in the representation (9) realizes no more than $2^{n_0} H_{n_1}(f) H_{n_2}(f)$ "ones" of the function f. The statement of the lemma follows from that fact.

We use Lemma 5 and Theorem 4 to obtain the following theorem.

Theorem 5 *Let $g_n(X_n)$, $|X_n| = n$, be a sequence of Boolean functions; $k(n)$ be a growing function; C, $0 < C < 1$, be a constant. Let $p(n)$, $t(n)$ $(2 \le k(n) \le p(n) \le t(n))$ be integer-valued, and let values n_1, n_2, and n_0 , calculated by the formulas (2)–(4) as the functions of $\lceil (1-C)n \rceil$; $k(n)$, $p(n)$ and $t(n)$) be positive, then*

(a) *the complexity* $\mathrm{NBP}(g_n)$ *of the function $g_n(X_n)$ for nondeterministic branching programs without restrictions satisfies an inequality*

$$\mathrm{NBP}(g_n) \ge \min\left\{ Cnk(n), \frac{1}{4}\sqrt{\frac{2p}{et}} \cdot \left(\frac{|g^{-1}(1)|}{2^{n-n_1-n_2} H_{n_1}(g) H_{n_2}(g)} \right)^{1/(4p)} \right\}.$$

(b) *the complexity* $\mathrm{BP}(g_n)$ *of the function $g_n(X_n)$ for deterministic branching programs without restrictions satisfies an inequality*

$$\mathrm{BP}(g) \ge \min\left\{ Cnk(n), \frac{2p}{et} \cdot \left(\frac{|g^{-1}(1)|}{2^{n-n_1-n_2} H_{n_1}(g) H_{n_2}(g)} \right)^{1/(2p)} \right\}.$$

Proof. Let X_0 be an arbitrary $\lfloor Cn \rfloor$-element subset of the set X_n. We substitute constants in the set X_0 in such a way that an inequality

$$\left| \left(g\mid_{X_0=\alpha} \right)^{-1}(1) \right| \ge \frac{\left|(g)^{-1}(1)\right|}{2^{|X_0|}}$$

holds for the function $g\mid_{X_0=\alpha}$. From this fact and Lemma 5 it follows that

$$\begin{aligned} R\left(g\mid_{X_0=\alpha}; \lceil (1-C)n \rceil, k, p, t\right) &\ge \frac{\left| \left(g\mid_{X_0=\alpha} \right)^{-1}(1) \right|}{2^{n_0} H_{n_1}\left(g\mid_{X_0=\alpha}\right) \cdot H_{n_2}\left(g\mid_{X_0=\alpha}\right)} \\ \ge \frac{|g^{-1}(1)|}{2^{|X_0|+n_0} H_{n_1}\left(g\mid_{X_0=\alpha}\right) \cdot H_{n_2}\left(g\mid_{X_0=\alpha}\right)} &\ge \frac{|g^{-1}(1)|}{2^{\lfloor Cn \rfloor + \lceil (1-C)n \rceil - n_1 - n_2} H_{n_1}(g) \cdot H_{n_2}(g)} \\ &\ge \frac{|g^{-1}(1)|}{2^{n-n_1-n_2} H_{n_1}(g) \cdot H_{n_2}(g)}. \end{aligned}$$

From this inequality and Theorem 4 the statement of the theorem follows.

6 Estimation of the number of "ones" in i-faces for Reed–Muller codes

The concept of the generalized Hamming weights and the weight hierarchy for linear code was introduced in [14]. Let C be a $[n,k]$ linear code (i. e., a code that contains 2^k codewords of length equaled to n) and D be its subcode. The support of D (it is denoted by $\chi(D)$) is the set of not-always-zero bit position in the codewords, in each of which at least one code word of D is not equal to 0, i. e.,

$$\chi(D) = \{ i \ : \ \exists (x_1, x_2, \ldots, x_n) \in D, \ x_i \ne 0 \}.$$

In these terms a $[n,k]$ linear code is a binary linear code of rank k and with the support size $\le n$.

A one-dimensional subcode D of a code C consists of two codewords: the zero codeword and a non-zero codeword. The support of D is equal to the Hamming weight of the non-zero codeword. The size of the smallest support of the subcode of a rank r of the code C is called the rth generalized Hamming weight of the code C (denoted by $d_r(C)$), i. e.,

$$d_r(C) = \min\{\ |\chi(D)|\ :\ D \text{ is a subcode of } C \text{ with rank } r\}.$$

It is easy to see that $d_1(C)$ is equal to the traditional minimum Hamming weight of C. The weight hierarchy of a linear code C is defined to be the set of integers $\{d_1(C), \ldots, d_k(C)\}$.

The interest to the generalized Hamming weights in [14] is caused by the possibility of application of this concept in cryptography. Other application of generalized Hamming weights is t-resilient functions.

In the given work the interest to the generalized Hamming weights is caused by the fact that they allow to estimate maximum number of the code nodes belonging to i-faces.

Lemma 6 *There are no more than 2^r code nodes of a linear code C in any $d_r(C)$-face containing the zero node.*

Proof. Assume that there are more than 2^r code nodes in $d_r(C)$-face containing zero node, i. e., there are at least 2^{r+1} code nodes in it. It means that it is possible to embed a subcode with 2^{r+1} code nodes into $d_r(C)$-face containing the zero node. i. e.,

$$d_{r+1}(C) \leq d_r(C). \tag{14}$$

But, as shown in [14, Theorem 1] a sequence $d_r(C)$ is strictly monotone, namely,

$$1 \leq d_1(C) < d_2(C) < \ldots < d_k(C) \leq n.$$

Therefore, the inequality (14) is not valid. Thus, any subcode of the dimension $d_r(C)$ contains no more than 2^r code nodes. The lemma is proved.

This lemma shows that for obtaining the upper bound on the number of code nodes in faces containing zero node it is suffice to know the generalized Hamming weights of the code.

Besides in [14] the hierarchy of generalized Hamming weights for Reed–Muller codes was been researched. For these codes we shall use the notation from [3, 14]. By $\mathcal{R}(u, m)$ denote the mth Reed–Muller code of order u. $\mathcal{R}(u, m)$ is considered as a linear space composing of all Boolean polynomials of degree u or less in m variables $v_1, v_2, \ldots, v_m$.

It is known that the number of code nodes in $\mathcal{R}(u, m)$ is equal to

$$2^{1+\binom{m}{1}+\binom{m}{2}+\ldots+\binom{m}{u}}; \tag{15}$$

the length of code words is equal to

$$n = 2^m; \tag{16}$$

the minimal code distance is

$$d = 2^{m-u}.$$

It is possible to list all monomials in variables $v_1, \ldots, v_m$ in the antilexicographical order defined by $v_m \prec v_{m-1} \prec \ldots \prec v_1 \prec \Lambda$ (the empty string). For example, if $c \prec b \prec a \prec \Lambda$, $\{cba, cb, ca, c, ba, b, a, \Lambda\}$ is a list of monomials in the antilexicographic order. As it is indicated in [14] the antilexicographical order is not is not equivalent to the

reverse of the lexicographical order. Really, in the given example $\{\Lambda, a, ab, abc, ac, b, bc, c\}$ is a list in the lexicographic order. Its reverse is not in the antilexicographical order.

The complete weight hierarchy of Reed–Muller codes is described in [14] in terms of the antilexicographic order.

Theorem 6 ([14], theorem 7) *The subcode of $R(u,m)$ spanned by the antilexicographically first r monomials of degree u or less has support $d_r(R(u,m))$.*

As an example, as well as in [14], we shall consider the code $R(2,5)$. Its antilexicographically ordered base is $\{v_5v_4, v_5v_3, v_5v_2, v_5v_1, v_5, v_4v_3, v_4v_2, v_4v_1, v_4, v_3v_2, v_3v_1, v_3, v_2v_1, v_2, v_1, 1\}$. The weight hierarchy of this code is $\{8, 12, 14, 15, 16, 20, 22, 23, 24, 26, 27, 28, 29, 30, 31, 32\}$.

There is no need to know the complete weight hierarchy of Reed–Muller codes for obtaining lower bounds on the complexity for branching program. We need to know the exact values only of some values $d_r(C)$.

Theorem 7 *Let φ be a natural number not exceeding m. Then a $2^m/2^\varphi$-face of the cube contains no more than $2^{\binom{m-\varphi}{0}+\binom{m-\varphi}{1}+\ldots+\binom{m-\varphi}{u-\varphi}}$ code nodes of the code $\mathcal{R}(u,m)$.*

Proof. Let's consider the first $2^{\binom{m-\varphi}{0}+\binom{m-\varphi}{1}+\ldots+\binom{m-\varphi}{u-\varphi}}$ antilexicographically ordered monomials. All these monomials and only these ones contain a monomial $v_m \ldots v_{m-\varphi+1}$ as a factor. Therefore the support of all these monomials is contained in the support of a vector $v_m \ldots v_{m-\varphi+1}$, i. e. is equal to $2^m/2^\varphi$. Thus $2^m/2^\varphi$-face with the zero node contains no more than $2^{\binom{m-\varphi}{0}+\binom{m-\varphi}{1}+\ldots+\binom{m-\varphi}{u-\varphi}}$ code nodes.

Since Reed–Muller code is a linear code, it follows from this statement that any $2^m/2^\varphi$-face of a cube contains no more than $2^{\binom{m-\varphi}{0}+\binom{m-\varphi}{1}+\ldots+\binom{m-\varphi}{u-\varphi}}$ code nodes. The theorem is proved.

7 Lower bounds on the complexity of characteristic functions of Reed–Muller codes for branching programs

Theorems 5 and 7 will be used for obtaining lower bounds on the complexity of characterictic functions of Reed–Muller codes.

Theorem 8 *Let $\frac{m}{m-u_m} \to \infty$ as $m \to \infty$ and $m - u \geq 3$. Then*

$$\mathrm{NBP}(\mathcal{R}(\mathrm{u_m}, \mathrm{m})) = \Omega\left(2^m \frac{m}{m-u_m} \Big/ \ln \frac{m}{m-u_m}\right).$$

Proof. Since $\frac{m}{m-u_m} \to \infty$ as $m \to \infty$, we can assume that

$$\frac{m}{m-u_m} \geq 2^{16}. \tag{17}$$

We set

$$k = k(m) = \frac{1}{4}\frac{m}{m-u} \Big/ \log_2 \frac{m}{m-u}. \tag{18}$$

Put

$$C = 1/2, \quad p = k, \quad t = k^2 + k. \tag{19}$$

Then by the statement (a) of Lemma 3 we have

$$n_1(n/2, k, p, t) \geq \frac{n}{2 \cdot 2(ke)^k} = \frac{2^m}{2^{2+k\log_2(ke)}} \geq \frac{2^m}{2^{\varphi_1}},$$

where

$$\varphi_1 = \left\lfloor \frac{m}{4(m-u)} \right\rfloor. \tag{20}$$

In fact, if $\frac{m}{m-u} \geq 2^{16}$, then

$$\begin{aligned} 2 + k\log_2(ke) &\leq 2 + \frac{\frac{m}{(m-u)}}{4\log_2 \frac{m}{m-u}} \left(\log_2 \frac{m}{m-u} - \log_2 \log_2 \frac{m}{m-u} + \log_2 \frac{e}{4} \right) \\ &\leq 2 + \frac{m}{4(m-u)} - \frac{\frac{m}{m-u} \log_2 \log_2 \frac{m}{m-u}}{4\log_2 \frac{m}{m-u}} \leq \left\lfloor \frac{m}{4(m-u)} \right\rfloor. \end{aligned}$$

Furthermore,

$$n_2(n/2; k, p, t) \geq \frac{n}{2(k+1)} = \frac{2^m}{2^{1+\log_2(k+1)}} \geq \frac{2^m}{2^{\varphi_2}},$$

where

$$\varphi_2 = \left\lfloor \log_2 \frac{m}{m-u} \right\rfloor. \tag{21}$$

This statement is valid for $\frac{m}{m-u} \geq 2^{16}$.

Recall that the maximal number of "ones" of a Boolean function f that belong to i-face of the cube was denoted by $H_i(f)$. If $n' \leq n''$ then an inequality

$$H_{n''}(f) \leq 2^{n''-n'} H_{n'}(f)$$

also holds. From this fact it follows that the bound from the theorem 5 is also valid as the values n_1 and n_2 decrease. Therefore it can be considered that

$$n_1 = \frac{2^m}{2^{\varphi_1}} \text{ и } n_2 = \frac{2^m}{2^{\varphi_2}}.$$

From this fact and the theorem 7 it follows that

$$H_{n_1}(\mathcal{R}(u_m, m)) \leq 2^{\binom{m-\varphi_1}{0} + \binom{m-\varphi_1}{1} + \ldots + \binom{m-\varphi_1}{u-\varphi_1}} \tag{22}$$

and

$$H_{n_2}(\mathcal{R}(u_m, m)) \leq 2^{\binom{m-\varphi_2}{0} + \binom{m-\varphi_2}{1} + \ldots + \binom{m-\varphi_2}{u-\varphi_2}}. \tag{23}$$

We use the theorem 5 for obtaining lower bounds on NBP($\mathcal{R}(u, m)$).

Therefore it is necessary to obtain a lower bound on the value

$$R = \frac{|(\mathcal{R}(u,m))^{-1}(1)|}{2^{n-n_1-n_2} H_{n_1}(\mathcal{R}(u,m)) H_{n_2}}(\mathcal{R}(u,m)). \tag{24}$$

By (15), (22) and (23) we have

$$\begin{aligned} R &\geq \frac{2^{1+\binom{m}{1}+\binom{m}{2}+\ldots+\binom{m}{u}}}{2^{2^m-2^{m-\varphi_1}-2^{m-\varphi_2}}\cdot 2^{\binom{m-\varphi_1}{0}+\binom{m-\varphi_1}{1}+\ldots+\binom{m-\varphi_1}{u-\varphi_1}}\cdot 2^{\binom{m-\varphi_2}{0}+\binom{m-\varphi_2}{1}+\ldots+\binom{m-\varphi_2}{u-\varphi_2}}} \\ &= \frac{2^{\sum_{j=u-\varphi_1+1}^{m-\varphi_1}\binom{m-\varphi_1}{j}+\sum_{j=u-\varphi_2+1}^{m-\varphi_2}\binom{m-\varphi_2}{j}}}{2^{\sum_{j=u+1}^{m}\binom{m}{j}}} \\ &= \frac{2^{\sum_{j=0}^{m-u-1}\binom{m-\varphi_1}{j}+\sum_{j=0}^{m-u-1}\binom{m-\varphi_2}{j}}}{2^{\sum_{j=0}^{m-u-1}\binom{m}{j}}} \\ &= 2^{\sum_{j=0}^{m-u-1}\left(\binom{m-\varphi_1}{j}+\binom{m-\varphi_2}{j}-\binom{m}{j}\right)}. \end{aligned} \tag{25}$$

Therefore for obtaining a lower bound on $R(\mathcal{R}(u,m);n/2,p,t)$ it is necessary to obtain a lower bound on the value $\binom{m-\varphi}{j}/\binom{m}{j}$, where $j \leq m-u-1$. For thise purpose we use the relations (20), (21), conditions of the theorem, and the assumption (17). We have

$$m \geq 3\cdot 2^{16}; \quad 3 \leq m-u \leq m/2^{16}.$$

From this fact, Stirling's formula we obtain for $j \leq m-u-1$ and $\varphi \leq \varphi_1 \leq \frac{m}{4(m-u)}$

$$\begin{aligned} &\binom{m-\varphi}{j}\Big/\binom{m}{j} = \frac{(m-\varphi)!\,j!\,(m-j)!}{j!\,(m-\varphi-j)!\,m!} \\ \geq\ & \sqrt{\frac{(m-\varphi)(m-j)}{(m-\varphi-j)m}}\cdot\frac{(m-\varphi)^{m-\varphi}(m-j)^{m-j}}{(m-\varphi-j)^{m-\varphi-j}m^m e^{\frac{1}{12(m-\varphi-j)}+\frac{1}{12m}}} \\ \geq\ & \frac{\left(1-\frac{\varphi}{m}\right)^{m-\varphi}\left(1-\frac{j}{m}\right)^{m-j}}{\left(1-\frac{\varphi+j}{m}\right)^{m-\varphi-j}e^{\frac{1}{6(m-\varphi-j)}}} \\ \geq\ & \exp\left\{(m-\varphi)\left(-\frac{\varphi}{m}-\frac{\varphi^2}{2m^2}-\ldots\right)+(m-j)\left(-\frac{j}{m}-\frac{j^2}{2m^2}-\ldots\right)-\right. \\ & \left.-(m-\varphi-j)\left(-\frac{\varphi+j}{m}-\frac{(\varphi+j)^2}{2m^2}-\ldots\right)-\frac{1}{6(m-\frac{m}{4(m-u)}-(m-u-1))}\right\} \\ \geq\ & \exp\left\{-\varphi+\frac{\varphi^2}{2m}+\sum_{l=2}^{\infty}\frac{\varphi^{l+1}}{l(l+1)m^l}-j+\frac{j^2}{2m}+\sum_{l=2}^{\infty}\frac{j^{l+1}}{l(l+1)m^l}\right. \\ & \left.+(\varphi+j)-\frac{(\varphi+j)^2}{2m}-\sum_{l=2}^{\infty}\frac{(\varphi+j)^{l+1}}{l(l+1)m^l}-\frac{1}{5,49m}\right\} \\ \geq\ & \exp\left\{-\frac{2\varphi j}{m}+\sum_{l=2}^{\infty}\frac{\varphi^{l+1}+j^{l+1}-(\varphi+j)^{l+1}}{l(l+1)m^l}-2,8\cdot 10^{-6}\right\} \end{aligned} \tag{26}$$

We obtain the lower bound on the sum $\sum_{l=2}^{\infty}\frac{\varphi^{l+1}+j^{l+1}-(\varphi+j)^{l+1}}{l(l+1)m^l}$ for $j \leq m-u-1$ and $\varphi \leq \varphi_1 \leq \frac{m}{4(m-u)}$. Let us consider the case when $j < \varphi$ and the case when $j \geq \varphi$.

Case 1. Assume that $j < \varphi$, then

$$\sum_{l=2}^{\infty}\frac{\varphi^{l+1}+j^{l+1}-(\varphi+j)^{l+1}}{l(l+1)m^l} \geq \sum_{l=2}^{\infty}\frac{-j\left((\varphi+j)^l+(\varphi+j)^{l-1}\varphi+\ldots+\varphi^l\right)}{l(l+1)m^l}$$

$$\geq -\sum_{l=2}^{\infty} \frac{j(l+1)(2\varphi)^l}{l(l+1)m^l} \geq -\sum_{l=2}^{\infty} \frac{(m-u)\cdot 2^l m^l}{l\cdot 4^l (m-u)^l m^l} \geq -\sum_{l=2}^{\infty} \frac{1}{l\cdot 2^l (m-u)^{l-1}}$$

$$\geq -\sum_{l=2}^{\infty} \frac{1}{l\cdot 2^l (3)^{l-1}} \geq -0,047.$$

Case 2. Assume that $\varphi \leq j$, then

$$\sum_{l=2}^{\infty} \frac{\varphi^{l+1} + j^{l+1} - (\varphi+j)^{l+1}}{l(l+1)m^l} \geq -\sum_{l=2}^{\infty} \frac{\varphi\left((\varphi+j)^l + (\varphi+j)^{l-1}j + \ldots + j^l\right)}{l(l+1)m^l}$$

$$\geq -\sum_{l=2}^{\infty} \frac{\varphi(l+1)(2j)^l}{l(l+1)m^l} \geq -\sum_{l=2}^{\infty} \frac{m\cdot 2^l (m-u-1)^l}{4l(m-u)m^l} \geq -\sum_{l=2}^{\infty} \frac{2^l (m-u)^{l-1}}{4lm^{l-1}} \geq -8\cdot 10^{-6}.$$

From these bounds and (26) it follows that

$$\binom{m-\varphi_1}{j} \Big/ \binom{m}{j} \geq -\frac{2\varphi j}{m} - 0,048.$$

From these inequalities, (20), (21) and (17) we have

$$\binom{m-\varphi_1}{j} \Big/ \binom{m}{j} \geq \exp\left\{-\frac{2\varphi_1 j}{m} - 0,048\right\} \geq \exp\left\{-\frac{\frac{2m}{4(m-u)}(m-u-1)}{m} - 0,048\right\}$$

$$\geq \exp\{-0,5 - 0,048\} \geq 0,57 \tag{27}$$

and

$$\binom{m-\varphi_2}{j} \Big/ \binom{m}{j} \geq \exp\left\{-\frac{2\varphi_1 j}{m} - 0,048\right\}$$

$$\geq \exp\left\{-\frac{2(m-u-1)\log_2 \frac{m}{(m-u)}}{m} - 0,048\right\} \geq 0,95. \tag{28}$$

From these two inequality and (25) it follows that

$$\begin{aligned} R &\geq 2^{\sum_{j=0}^{m-u-1}\left(\binom{m-\varphi_1}{j}+\binom{m-\varphi_2}{j}-\binom{m}{j}\right)} \geq 2^{\sum_{j=0}^{m-u-1}(0,57+0,95-1)\binom{m}{j}} \\ &\geq 2^{0,52\binom{m}{2}} \geq 2^{0,25m^2}. \end{aligned} \tag{29}$$

From theorem 5, (19), (18), (24), (17), (29), and (16) it follows that

$$\begin{aligned} \mathrm{NBP}(\mathcal{R}(u,m)) &\geq \min\left\{\frac{n}{2}\cdot k; \frac{1}{4}\sqrt{\frac{2}{e(k+1)}}\cdot R^{1/(4k)}\right\} \\ &= \min\left\{\frac{1}{8}\cdot\frac{nm/(m-u)}{\log_2 m/(m-u)}; \frac{1}{4}\sqrt{\frac{2}{e\left(\frac{m/(m-u)}{4\log_2 m/m-u}+1\right)}}\cdot R^{\frac{m-u}{m}\log_2\frac{m}{m-u}}\right\} \\ &\geq \min\left\{\frac{1}{8}\cdot\frac{2^m m/(m-u)}{\log_2 m/(m-u)}; \frac{1}{4}\sqrt{\frac{1}{\frac{m/(m-u)}{\log_2 m/m-u}}}\cdot 2^{\frac{0,25m^2(m-u)}{m}\log_2\frac{m}{m-u}}\right\} \\ &\geq \min\left\{\frac{1}{8}\cdot\frac{2^m m/(m-u)}{\log_2 m/(m-u)}; \frac{1}{4}\sqrt{\frac{1}{\frac{m/(m-u)}{\log_2 m/m-u}}}\cdot 2^{3m}\right\}. \end{aligned}$$

From this inequality and the assumption that $m/(m-u) \geq 2^{16}$ we obtain that

$$\mathrm{NBP}(\mathcal{R}(u,m)) = \Omega\left(2^m \frac{m}{m-u} \Big/ \ln \frac{m}{m-u}\right).$$

The theorem is proved.

Corollary 1 *Let $u_m = m - C^0$, where $C^0, C^0 \geq 3$ is a constant. Then*

$$\mathrm{NBP}(\mathcal{R}(u_m,m)) = \Omega(n \log n / \log\log n),$$

where n is the number of variables of the characterictic function of Reed–Muller code $\mathcal{R}(u_m,m)$.

References

[1] **Babai L., Pudlák P., Rödl V., Szemerédi M.** (1990) Lower bounds to the complexity of symmetric Boolean functions. // Theoretical Computer Science. 1990. V. 74, N 3. P. 313–324.

[2] **Borodin A., Razborov A., Smolensky R.** (1993) On lower bounds for read-k-times branching programs // Computational Complexity. V. 3, N 1. P. 1–18.

[3] **MacWilliams F.J., Sloane N.J.F.** (1977) The theory of Error-Correcting Codes // Amsterdam: North-Holland.

[4] **Nečiporuk E.** (1966) On a Boolean function // Soviet Math. Doklady. V. 7. P. 999-1000.

[5] **Okol′nishnikova E. A.** (1991) Lower bounds on complexity for the realization of characteristic functions of binary codes by binary programs // Metody Diskret. Anal. V. 51. P. 61–83 (in Russian) (see also: Siberian Adv. Math. (1993) V. 3, No. 1, P. 152–166).

[6] **Okol'nishnikova E. A.** (1997) On the hierarchy of nondeterministic branching k-programs // Fundamentals of computation theory. 11th International symposium, FCT 97. Berlin: Springer. (Lecture Notes in Comput. Sci. V. 1279. P. 376–387).

[7] **Okol'nishnikova E. A.** (2001) On one method of obtaining lower bounds of Boolean functions for nondeterministic branching programs // Diskretn. Anal. Issled. Oper. Ser. 1. V. 8, No. 4. P. 76–102 (in Russian).

[8] **Okol'nishnikova E. A.** (2001) Lower bounds on the complexity for branching programs // Proc. of the conference devoted to 90th anniversary of A. Lyapunov (Novosibirsk, 8-11 October, 2001). P. 464–466. (http://www.ict.nsc.ru/ws/Lyap2001/2289) (in English).

[9] (1984) **Pudlák P. (1984)** A lower bound on complexity of branching programs // Mathematical foundations of computer science. Berlin: Springer–Verlag. P. 480–489. (Lecture Notes in Comput. Sci. V. 176.).

[10] **Pudlák P.** (1987) The hierarchy of Boolean circuits // Comput. Artificial Intelligence. V. 6, N 5. P. 449–468.

[11] **Razborov A. A.** (1990) Lower bounds on the complexity of symmetric Boolean functions by switching-rectifier circuits // Matemat. zametki. V. 48, No. 6. P. 79-90.

[12] **Razborov A. A.** (1991) Lower bounds for deterministic and nondeterministic branching program // Lecture Notes in Computer Science. V. 529. P. 47–61.

[13] **Thathachar J. S.** (1998) On separating the read-k-times program hierarchy // Proc. of the 30th annual ACM Symposium on theory of comptuting. (Dallas, 1998). New York: ACM Press, P. 652–662.

[14] **Wei V.K.** (1991) Generalized Hamming weights for linear codes. // IEEE Trans. on Inform. Theory. V. 37, N 5, P. 1412–1418.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'